LLMCheckup: Conversational Examination of Large Language Models via Interpretability Tools and Self-Explanations

Qianli Wang[1,2] **Tatiana Anikina**[*1,3] **Nils Feldhus**[*1]
Josef van Genabith[1,3] **Leonhard Hennig**[1] **Sebastian Möller**[1,2]
[1]German Research Center for Artificial Intelligence (DFKI)
[2]Technische Universität Berlin, Germany
[3]Saarland Informatics Campus, Saarbrücken, Germany
`{firstname.lastname}@dfki.de`

Abstract

Interpretability tools that offer explanations in the form of a dialogue have demonstrated their efficacy in enhancing users' understanding (Slack et al., 2023; Shen et al., 2023), as one-off explanations may fall short in providing sufficient information to the user. Current solutions for dialogue-based explanations, however, often require external tools and modules and are not easily transferable to tasks they were not designed for. With LLMCHECKUP[1], we present an easily accessible tool that allows users to chat with any state-of-the-art large language model (LLM) about its behavior. We enable LLMs to generate explanations and perform user intent recognition without fine-tuning, by connecting them with a broad spectrum of Explainable AI (XAI) methods, including white-box explainability tools such as feature attributions, and self-explanations (e.g., for rationale generation). LLM-based (self-)explanations are presented as an interactive dialogue that supports follow-up questions and generates suggestions. LLMCHECKUP provides tutorials for operations available in the system, catering to individuals with varying levels of expertise in XAI and supporting multiple input modalities. We introduce a new parsing strategy that substantially enhances the user intent recognition accuracy of the LLM. Finally, we showcase LLMCHECKUP for the tasks of fact checking and commonsense question answering.

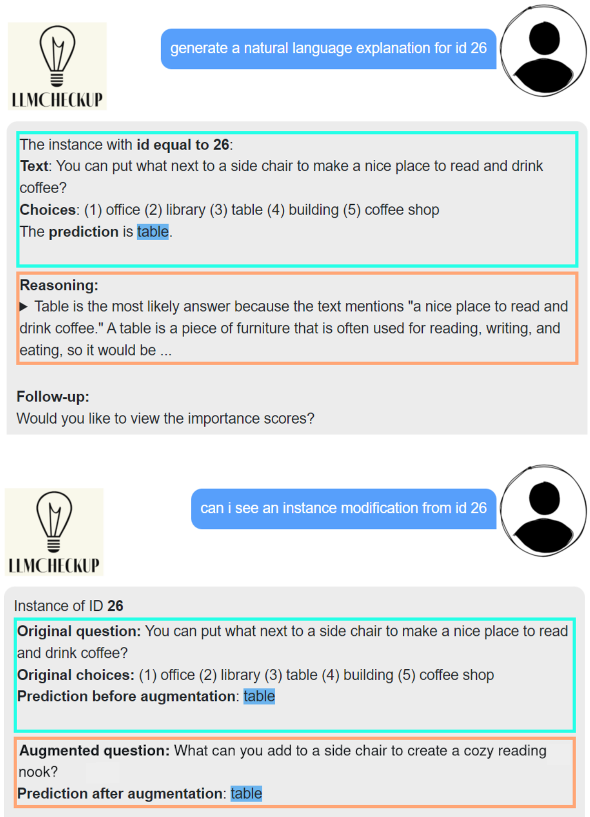

Figure 1: LLMCHECKUP dialogue with data augmentation and rationalization operations on a commonsense question answering task (ECQA). Boxes (not part of the actual UI) indicate the original instance from the dataset as well as its prediction (cyan) and the explanation requested by the user (orange).

1 Introduction

To unravel the black box nature of deep learning models for natural language processing, a diverse range of explainability methods have been developed (Ribeiro et al., 2016; Madsen et al., 2022; Wiegreffe et al., 2022). Nevertheless, practitioners often face difficulties in effectively utilizing explainability methods, as they may not be aware of which techniques are available or how to interpret results provided. There has been a consensus within the research community that moving beyond one-off explanations and embracing conversations to provide explanations is more effective for model understanding (Lakkaraju et al., 2022; Feldhus et al., 2023; Zhang et al., 2023) and helps mitigate the limitations associated with the effective usage of explainability methods to some extent

*Equal contribution
[1]https://github.com/DFKI-NLP/LLMCheckup

Proceedings of the Third Workshop on Bridging Human–Computer Interaction and Natural Language Processing (3rd HCI+NLP Workshop), pages 89–104
June 21, 2024

(Ferreira and Monteiro, 2020; Slack et al., 2023).

In the field of NLP, two dialogue-based interpretability tools, InterroLang (Feldhus et al., 2023) and ConvXAI (Shen et al., 2023), have been introduced. Both tools employ multiple, separately fine-tuned LMs to parse user intents and dedicated external LMs to provide explanations.

By contrast, our framework, LLMCheckup, only requires a single LLM and puts it on "quadruple duty": (1) Analyzing users' (explanation) requests (§2.1, §5.1), (2) performing downstream tasks (§4), (3) providing explanations for its outputs (§3), and (4) responding to the users in natural language (§2.3). Instead of using many different LMs to explain the behavior of another LLM, LLMCheckup allows us to directly employ the same LLM used for user intent recognition to self-explain its own behavior. The advantage of a single-model approach is that it simplifies the engineering aspect of building an XAI system without multiple external modules and separately fine-tuned models. At the same time, we consistently achieve good performance even with a single model, as modern LLMs are very powerful and can handle a wide range of tasks including user intent recognition and explanation generation. Thus, LLMCheckup provides a unified and compact framework that is useful for future user studies in the context of human-computer interaction and explainability.

2 LLMCheckup

LLMCheckup is an interface for chatting with any LLM about its behavior. We connect several white-box and black-box interpretability tools (§3), s.t. LLMCheckup takes into account model internals, datasets and documentation for generating self-explanations. User requests for explanations are recognized via a text-to-SQL task performed by the LLM under investigation (§2.1-2.2).

We showcase a short dialogue between the user and LLMCheckup in Figure 1 and a longer dialogue featuring different operations in Appendix B. LLMCheckup can answer various questions related to the data as well as the model's behavior. For example, in Figure 1 the user is interested in the rationale for a specific prediction and the model generates an explanation to justify the assigned label. LLMCheckup also suggests to have a look at another related operation (token-level importance scores) that can help explain model's behavior (§2.4), but the user asks for a modified (augmented) version of the same instance instead. As a result, the model paraphrases the original question which can be then treated as a new sample and the user can further examine it by using the custom input functionality of LLMCheckup (§2.4).

2.1 System architecture

Figure 2 illustrates the interaction flow of LLMCheckup. When a user asks a question, it will be parsed as an SQL-like query by the LLM. E.g., the first user question in Figure 1 will be parsed as `filter id 26 and rationalize`. The corresponding parsed operation (i.e., `filter` and `rationalize` in our example, see Table 1 for the full list of operations) will then be matched and executed. For response generation, the explanation provided by the underlying interpretability method is converted into a natural language output using a template-based approach (Slack et al., 2023; Feldhus et al., 2023) and is then displayed to the user.

2.2 Parsing

To recognize users' intents, the deployed LLM transforms a user utterance into a SQL-like query. The SQL-based approach is needed to formally represent the available operations (see Table 1) and their "semantics" including all necessary attributes. For user intent recognition, we employ two methods: Guided Decoding and Multi-prompt Parsing.

2.2.1 Guided Decoding

Guided Decoding (GD) ensures that the generated output adheres to predefined grammatical rules and constraints (Shin et al., 2021) and that parses of the user requests align with predefined operation sets (Slack et al., 2023). GD is generally more suitable for smaller LMs, since in-context learning may encounter instability attributed to the fluctuations in the order of provided demonstrations, and the formats of prompts (Ma et al., 2023).

2.2.2 Multi-prompt Parsing

As an alternative to GD, we propose and implement a novel Multi-prompt Parsing (MP) approach. While GD pre-selects prompts based on the embedding similarity with user input, the model does not see all the available operations at once and the pre-selection may not include the examples for the actual operation required. With MP, we test whether showing all possible operations in a simplified format (i.e., without any attributes such as instance ID or number of samples) and then additionally

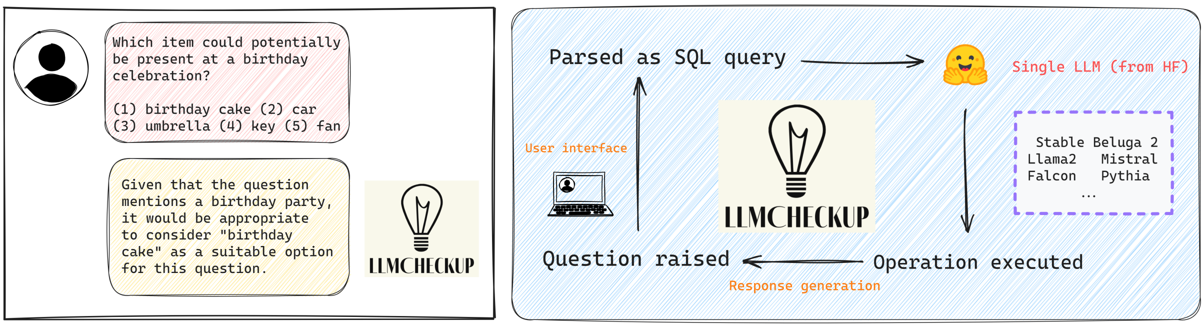

Figure 2: On the left, a dialogue example asking for explanation in natural language about a ECQA-like customized question. The workflow of LLMCHECKUP is shown on the right side.

Filter	`filter(id)` `includes(token)`	Access single instance by its ID Filter instances by token occurrence

Prediction	`predict(instance)*`
	`randompredict(number)`
	`mistakes`
	`show|count(subset)`
	`score(subset, metric)`
		Get the prediction for the given instance
		Precompute a subset of instances at random
		Count or show incorrectly predicted instances
		Determine the relation between predictions and labels

Data	`show(list)`
	`countdata(list)`
	`label(dataset)`
		Showcase a list of instances
		Count number of instances in the dataset
		Describe the label distribution across the dataset

Meta	`data()` `model()`	Information related to the dataset Metadata of the model

About	`function()`
	`self()`
	`qatutorial(op_name)`
		Inform about the functionality of the system
		Self-introduction of LLMCHECKUP
		Provide explanation for the supported operations (tutorial)

Explain	`nlpattribute(inst., topk, method_name)*` `rationalize(instance)*`	Provide feature attribution scores Explain the output in natural language
NLU	`keywords()` `similarity(instance, number)*`	Show common keywords in the data Output top k similar instances in the dataset
Pert.	`cfe(instance)*` `augment(instance)*`	Generate counterfactuals Augment the input text
Logic	`and(op1, op2)` `or(op1, op2)`	Concatenation of multiple operations Selection of multiple filters

Table 1: All operations (mappings between a partial SQL-type query and a function) facilitated by LLMCHECKUP, including all explainability methods mentioned in §3 and other supplementary operations. Operations marked with (*) support the use of custom inputs (see more details in App. A).

prompting the model to fill in more fine-grained attributes can improve performance.

As a first step, MP queries the model about the main operation (see list of operations in Table 1). Next, depending on the chosen operation, MP selects the operation-specific prompts with 2-7 demonstrations[2] (user query and parsed outputs examples) to generate the full parses that may include several attributes. E.g., for the user input *"What are the feature attributions for ID 42 based on the integrated gradients?"*, we start by generating `nlpattribute` and then augment the parse with the second prompt and transform it into `filter id 42 and nlpattribute integrated_gradient`.

Since the output of the model is not constrained, unlike in GD, in the MP setting we need to check whether the model's output matches any of the available operations and if there is no exact match we employ SBERT[3] to find the best match based on the embedding similarity. We also implement checks to avoid possible hallucinations, e.g., if the model outputs an ID that is not present in the input we remove it from the parser output. §5.1 evaluates the performance of both parsing approaches.

2.3 Interface

LLMCHECKUP provides a user interface (Figure 3) including a chat window to enter questions and settings on the right panel, including XAI expertise level selection, custom inputs, prompt editor and export functionality for the chat history. It is implemented in Flask and can be run as a Docker container. LLMCHECKUP provides a chat window (Slack et al., 2023), a dataset viewer (Feldhus et al., 2023), a custom input history viewer and question suggestions for different operations. Together, these UI elements facilitate dataset exploration and provide sample questions for all available operations to inspire users to come up with their own questions.

On the right side of the window, there is a Prompt Editor with different options for prompt modification (§3.2). The icons associated with each strategy describe them in detail, including the corresponding prompts that can be appended after the default

[2] The number of demonstrations depends on the difficulty of operation, e.g., how many attributes it may have.

[3] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

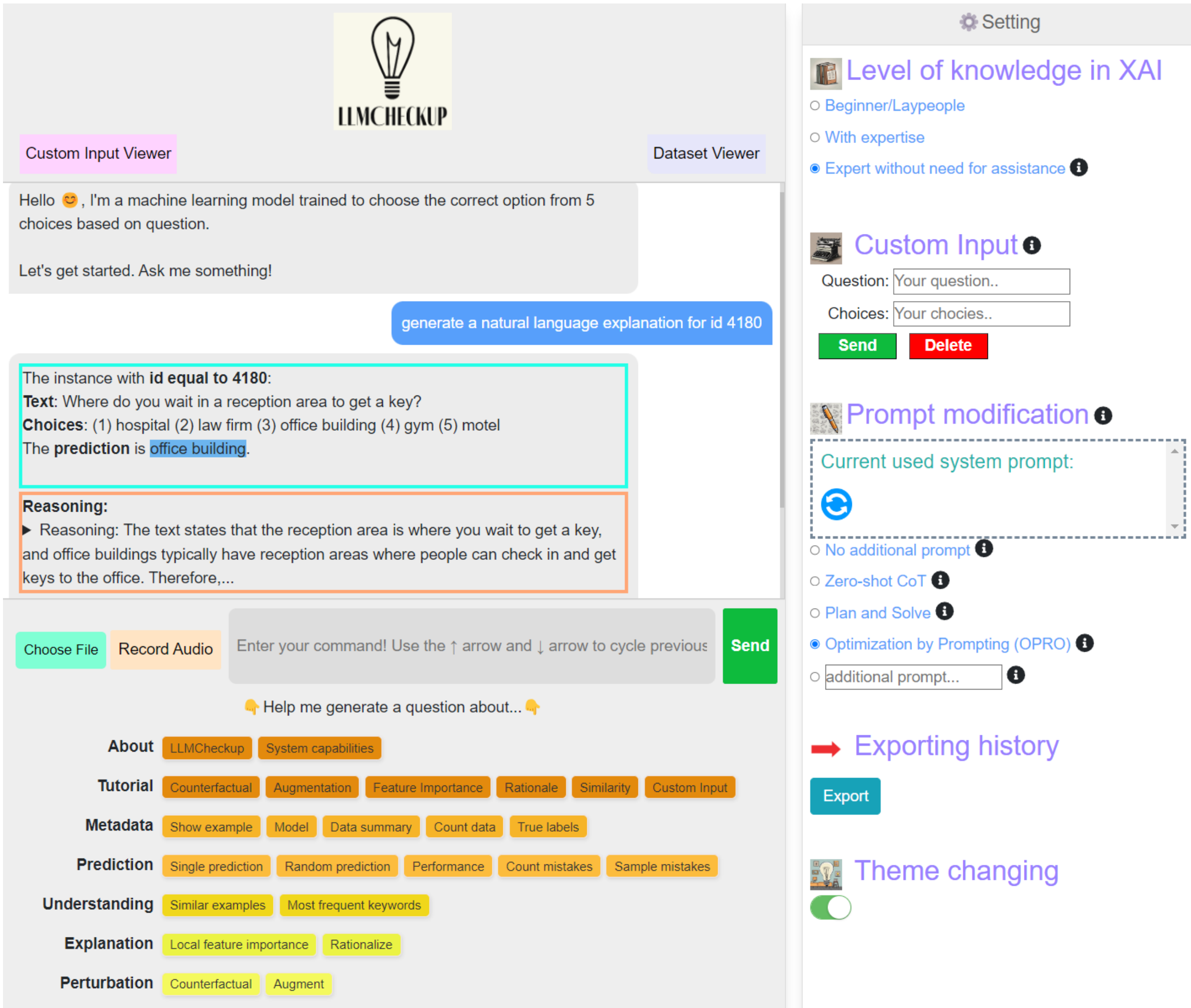

Figure 3: LLMCHECKUP interface with welcome message, free-text rationale and sample generator buttons. Expert XAI level and *OPRO* strategy are selected. For example multi-turn dialogues, see Table 5 and Table 6.

system prompt.

2.4 Key features

Supported NLP models Out of the box, we include five auto-regressive LLMs representative of the current state-of-the-art in open-source NLP (as indicated in the left column of Table 2) available through Hugging Face TRANSFORMERS (Wolf et al., 2020). The diverse choice of models demonstrates that our framework is generalizable and supports various Transformer-type models. While `Falcon-1B` (Penedo et al., 2023) and `Pythia-2.8B` (Biderman et al., 2023) are available for users with limited hardware resources (RAM/GPU), it is generally not recommended to use them due to their small model size, which may negatively affect performance and user perception. `Llama2-7B` (Touvron et al., 2023) and `Mistral-7B` (Jiang et al., 2023) are both mid-sized with 7B parameters, while `Stable Beluga 2` (Mukherjee et al., 2023) is a fine-tuned version of `Llama2-70B`. To facilitate the deployment of large models in a local environment, LLMCHECKUP offers support for various forms of LLMs. This includes **quantized models** through GPTQ (Frantar et al., 2023), loading models in 4-bits with the assistance of BITSANDBYTES (Dettmers et al., 2022), and the implementation of a **peer-to-peer** solution using PETALS (Borzunov et al., 2023), enabling efficient deployment on a custom-level GPU.

Tutorial To help non-experts get background knowledge in XAI, we introduce a tutorial functionality. It is based on prompting with different roles corresponding to levels of expertise in XAI (Figure 3) and enables us to provide tailored meta-explanations of supported operations to individuals. For example, at the beginner level, we add a system prompt hinting at the expertise: *"As a NLP beginner, could you explain what data augmentation is?"*

(Figure 4). In such a way, all users can receive meta-explanations according to their expertise.

Customized inputs & prompts In comparison to TalkToModel (Slack et al., 2023), which was limited to three datasets, LLMCheckup offers users the freedom to enter custom inputs (e.g. modified original samples or even completely new data points, see the Custom Input box on the right panel in Figure 3), going beyond just querying instances from specific provided datasets. In addition, inspired by PromptSource (Bach et al., 2022), a Prompt Editor (see Prompt modification section on the right panel in Figure 3) supports inserting both pre-defined and fully customized prompts, allowing the users to control how downstream tasks and rationalization (§3.2) are performed. All custom inputs are saved and can be inspected and reused later via a dedicated custom input history viewer.

Suggestion of follow-up questions To guide the user through the conversation, we implemented a suggestions mode. The user receives suggestions for related operations that LLMCheckup can perform based on the dialogue context, e.g., if the user asks about the top k attributed tokens for a specific sample, they will receive a suggestion to have a look at the generated rationales since both operations belong to the "Explanation" category also displayed in the user interface. The suggestions are grouped into several categories as specified in Table 1 (see Appendix F for more detail).

2.5 Add-on features

External information retrieval Since LLMs may sometimes generate incorrect responses (Welleck et al., 2020), LLMCheckup allows users to access information by conducting search through external knowledge bases, promoted by the integration of Google Search[4] (Figure 5). In particular, it provides an external link that contains information relevant to the input sample(s). Users can cross-reference the retrieved information with the provided explanations, thereby achieving a more comprehensive understanding.

Multi-modal input format Motivated by Malandri et al. (2023), LLMCheckup not only accepts text input from users but also provides support for other modalities such as images and audio. To facilitate this, we integrate packages and models tailored to each modality. For optical character recognition (OCR), we use EasyOCR[5]. For audio recognition, we employ a lightweight `fairseq S2T`[6] model (Wang et al., 2020) trained on Automatic Speech Recognition (ASR).

Dialogue sharing LLMCheckup offers the functionality to export the dialogue history between the user and the deployed LLM as a JSON file that contains the user's questions and the corresponding generated responses. This simplifies data collection and sharing of conversation logs between users.

3 NLP explainability tools

While we introduce each explainability method individually, these methods can be interconnected through follow-up questions from users or suggestions provided by LLMCheckup to preserve context. Table 5 and Table 6 show examples of explanations for each supported explainability method by LLMCheckup.

3.1 White-box

Feature attribution Feature attribution methods quantify the contribution of each input token towards the final outcome. In LLMCheckup, we deploy various auto-regressive models (§2.4), for which Inseq (Sarti et al., 2023) is used to determine attribution scores. We support representative methods from Inseq, including *Input x Gradient* (Simonyan et al., 2014), *Attention* (Bahdanau et al., 2015), *LIME* (Ribeiro et al., 2016), and *Integrated Gradients* (Sundararajan et al., 2017)[7].

Embedding analysis By calculating the cosine similarity between the sentence embeddings of the instances in datasets, we can retrieve relevant examples (Cer et al., 2017; Reimers and Gurevych, 2019) and present them for contextualizing the model behavior on the input in question.

3.2 Black-box

Data augmentation Augmentation involves synthesizing new instances by replacing text spans of the input while preserving the semantic meaning and predicted outcomes (Ross et al., 2022). Data augmentation can be achieved by LLM prompting with or without providing a few demonstrations (Dai et al., 2023). Alternatively, NLPAug[8] can be

[4] `https://github.com/Nv7-GitHub/googlesearch`

[5] `https://github.com/JaidedAI/EasyOCR`

[6] `https://huggingface.co/facebook/s2t-small-librispeech-asr`

[7] Details on the Inseq integration are described in App. C.

[8] `https://github.com/makcedward/nlpaug`

used to substitute input words with synonyms from WORDNET (Miller, 1995). Augmented texts can offer valuable insights into model behavior on perturbation tasks and prediction differences between them and their original texts.

Counterfactual generation Unlike data augmentation, counterfactuals manifest as input edits causing the predicted outcome to be different (Wu et al., 2021; Chen et al., 2023). Counterfactuals are generated by prompting LLMs with manually crafted demonstrations.

Rationalization Rationalization aims to provide free-text explanations that elucidate the prediction made by the model (Camburu et al., 2018; Wiegreffe et al., 2022) (an example is shown in Figure 1). The use of *Chain-of-Thought* (CoT) prompting enhances the reasoning capabilities of LLMs by encouraging the generation of intermediate reasoning steps that lead to a final answer (Wei et al., 2022; Wang et al., 2023b). Different CoT strategies can be applied depending on users' preferences, including *Zero-CoT* (Kojima et al., 2022), *Plan-and-Solve* (Wang et al., 2023a), and *Optimization by PROmpting* (OPRO) (Yang et al., 2023) (Figure 3).

4 Use cases

In this paper, we demonstrate the workflow of LLMCHECKUP on two typical NLP tasks: Fact checking and commonsense question answering. Figure 1 and Appendix B show sample dialogues where user asks questions regarding rationalization, data augmentation and other operations based on the ECQA data (Aggarwal et al., 2021) for commonsense question answering. The LLMCHECKUP repository includes all the necessary configuration files for different LMs and our use cases. They can be easily adopted to many other downstream tasks, data and Transformer-type models, demonstrated in a tutorial which will be available with the camera-ready version of our repository.

4.1 Fact checking

The importance of fact checking has grown significantly due to the rapid dissemination of both accurate information and misinformation within the modern media ecosystem (Guo et al., 2022). COVID-Fact (Saakyan et al., 2021) is a fact-checking dataset that encompasses various claims, supporting evidence for those claims, and contradictory claims that have been debunked by the presented evidence.

Model	Size	Strategy	Accuracy
Nearest Neighbor	-	-	42.24
`Falcon`	1B	GD	47.41
`Pythia`	2.8B	GD	51.72
`Llama2`	7B	GD	64.71
`Mistral`	7B	GD	55.88
`Stable Beluga 2`	70B	GD	67.23
`Falcon`	1B	MP	64.15
`Pythia`	2.8B	MP	75.91
`Llama2`	7B	MP	82.35
`Mistral`	7B	MP	84.87
`Stable Beluga 2`	70B	MP	88.24

Table 2: Exact match parsing accuracy (in %) for different models. GD = Guided Decoding prompted by 20-shots; MP = Multi-Prompt parsing.

4.2 Commonsense question answering

Unlike question answering, commonsense question answering (CQA) involves the utilization of background knowledge that may not be explicitly provided in the given context (Ostermann et al., 2018). The challenge lies in effectively integrating a system's comprehension of commonsense knowledge and leveraging it to provide accurate responses to questions. ECQA (Aggarwal et al., 2021) is a dataset designed for CQA. Each instance in ECQA consists of a question, multiple answer choices, and a range of explanations. Positive explanations aim to provide support for the correct choice, while negative ones serve to refute incorrect choices. Additionally, free-text explanations are included as general natural language justifications.

5 Evaluation

We conducted evaluations for parsing and data augmentation with LLMs using automated evaluation metrics[9]. Among all the supported methods presented in Table 1, we chose data augmentation as a representative operation to evaluate the performance of different LLMs.

5.1 Parsing

To assess the ability of interpreting user intents by LLMs, we quantify the performance of each deployed model by calculating the exact match parsing accuracy (Talmor et al., 2017; Yu et al.,

[9]Note that our evaluation does not involve any user study, as that aspect is considered as future work and falls outside the scope of our initial focus on engineering.

Model	#max_new_tokens	Accuracy
Falcon	10	64.15
Falcon	20	64.15
Pythia	10	**75.91**
Pythia	20	63.03
Llama2	10	74.79
Llama2	20	82.35
Llama2-GPTQ	10	82.63
Llama2-GPTQ	20	**87.40**
Mistral	10	**84.87**
Mistral	20	71.43
Mistral-GPTQ	10	78.71
Mistral-GPTQ	20	68.91
Stable Beluga 2	10	**88.24**
Stable Beluga 2	20	86.55

Table 3: Parsing accuracy (in %) using MP with different number of maximum new tokens. Note that for the `Llama2-7b` and `Mistral-7b` models, we offer various options for quantization. In this case, we have chosen GPTQ as the representative method.

2018) on a manually created test set, which consists of a total of 119 pairs of user questions and corresponding SQL-like queries. As an additional baseline, we employ the nearest neighbor approach that relies on comparing semantic similarity.

We assess parsing accuracy of our two approaches, GD and MP (§2.2). Table 2 shows that, as model size increases, the parsing accuracy tends to increase and MP demonstrates a notable improvement over GD. Despite `Stable Beluga 2` having a larger size compared to 7B models, its parsing performance only marginally surpasses that of `Mistral` and `Llama2`. This can be partially attributed to the difficulty of the parsing task[10] and the number of demonstrations, as larger models may require a greater number of demonstrations to fully comprehend the context (Li et al., 2023b).

Table 3 summarizes our parsing evaluation results for different models with different number of *'max_new_tokens'* for generation. `Llama`-based models showed better performance with more tokens to generate compared to the rest of the models. After looking at some generated outputs we realized that `Falcon-1B` and `Pythia-2.8B` are not good at extracting ids and often can only recognize the main LLMCHECKUP operation. Hence, for these two models we have an additional step that extracts a potential ID from the user input and adds it to the parsed operation. As expected, larger models tend to perform better than the ones with fewer parameters. However, we also found that the quantized `Llama` model outperforms its full (non-quantized) version on the parsing task.

5.2 Data augmentation

We assess the quality of the generated augmented output based on two key aspects: (1) **consistency:** the metric represents the proportion of instances where the augmentation process does not lead to a change in the label before and after the augmentation (Li et al., 2023a; Dai et al., 2023); (2) **fluency:** assesses how well the augmented output aligns with the original data in terms of semantic similarity (Ross et al., 2021) measured by `SBERT`. Table 4 indicates that `Mistral` and `Llama2` exhibit comparable performance, while `Stable Beluga 2` displays substantially higher consistency scores on two tasks, although it may exhibit lower fluency in certain cases. The overall performance on ECQA is relatively low compared to COVID-Fact. This difference in performance can be attributed to the increased complexity of the ECQA task. Our primary focus is to compare the performance of different LMs (Table 4), rather than aiming for state-of-the-art results on both downstream tasks or demonstrating perfect fluency and consistency[11].

6 Discussion

In contrast to previous dialogue-based XAI frameworks CONVXAI (Shen et al., 2023) and INTERROLANG (Feldhus et al., 2023), which require a fine-tuned model for each specific use case, LLMs used in LLMCHECKUP possess remarkable zero-/few-shot capabilities (Brown et al., 2020) for effectively handling many tasks without requiring fine-tuning. Although the quality of an explanation could be enhanced with further fine-tuning, LLMCHECKUP uses model outputs out of the box.

Our empirical results underline the feasibility of conversational interpretability and the usefulness of LLMCHECKUP for future studies, especially human evaluation. We focus on the ground work in terms of engineering, implementation and user interface, for connecting the human with the model. This provides user studies (Wang et al., 2019; Feldhus et al., 2023; Zhang et al., 2023) in the future with a head start, s.t. they can spend more time

[10] We have a total of 21 LLMCHECKUP operations displayed in Table 1 (excluding the logic operations), and many of these offer multiple options. For instance, *score* operation supports F_1, *precision*, *recall* and *accuracy* matrices.

[11] Creating gold data is out of scope for this work, because it involves costly human annotations. For the lack of gold data, we have intentionally omitted providing a baseline.

Model	Dataset Size	COVID-Fact *Consistency*	*Claim Fluency*	*Evidence Fluency*	ECQA *Consistency*	*Question Fluency*
`Mistral`	7B	0.66	0.88	0.96	0.50	0.76
`Llama2`	7B	0.65	0.88	0.94	0.50	0.76
`Stable Beluga 2`	70B	1.00	0.85	0.96	1.00	0.73

Table 4: Consistency and fluency scores of data augmentation from three models. `falcon` and `pythia` are not considered due to poor performance because of small model size.

on conducting their study. We see evaluation measures for differences between users' mental models and model behavior and objective metrics beyond simulatability as the most important gaps to fill.

7 Related work

Interfaces for interactive explanations LIT (Tenney et al., 2020) is a GUI-based tool available for analyzing model behaviors across entire datasets. However, LIT has less functionalities in terms of prompting and lower accessibility, e.g. no tutorial and a lower level of integration with HUGGINGFACE. CROSSCHECK (Arendt et al., 2021) exhibits the capability to facilitate quick cross-model comparison and error analysis across various data types, but adapting it for other use cases needs substantial code modification and customization. XMD's (Lee et al., 2023) primary purpose is model debugging, but it shares similarities in the focus on feature attributions, visualization of single instances and user feedback options. It is, however, limited to feature attribution explanations and smaller, efficiently retrainable models. IFAN (Mosca et al., 2023) enables real-time explanation-based interaction with NLP models, but is limited to the sequence-to-class format, restricting its applicability to other tasks and it offers only a limited set of explainability methods.

Dialogue-based systems for interpretability Carneiro et al. (2021) point out that conversational interfaces have the potential to greatly enhance the transparency and the level of trust that human decision-makers place in them. According to Zhang et al.'s (2023) user studies, delivering explanations in a conversational manner can improve users' understanding, satisfaction, and acceptance. Jacovi et al. (2023) emphasizes the necessity of interactive interrogation in order to build understandable explanation narratives. CONVXAI (Shen et al., 2023), TALKTOMODEL (Slack et al., 2023), INTERROLANG (Feldhus et al., 2023) and Brachman et al. (2023) share some similarities with our framework, but are more complex in their setup and consider fewer explainability methods. Additionally, they might overrely on external LMs to explain the deployed LM's behavior, whereas LLMCHECKUP places a strong emphasis on self-explanation, which is crucial for faithfulness. Finally, LLMCHECKUP uses auto-regressive models, as they have become increasingly dominant in various NLP applications nowadays. In ISEE (Wijekoon et al., 2023), a chatbot adapts explanations to the user's persona, but they do not consider LLMs.

8 Conclusion

We present the interpretability tool LLMCHECKUP, designed as a dialogue-based system. LLMCHECKUP can provide explanations in a conversation with the user facilitated by any auto-regressive LLM. By consolidating parsing, downstream task prediction, explanation generation and response generation within a unified framework, LLMCHECKUP streamlines the interpretability process without switching between different LMs, modules or libraries and serves as a baseline for future investigation.

Future work includes exploring RAG models (Lewis et al., 2020) combined with explainability, as currently LLMCHECKUP relies on search engines for external information retrieval. We also want to add multi-modal models, so that converting image or audio input to texts would no longer be necessary, but the current state of interpretability on such models lags behind unimodal approaches (Liang et al., 2023). Integrating our framework into HUGGINGCHAT[12] would further increase the visibility and accessibility through the web.

Limitations

In LLMCHECKUP, we do not focus on dataset analysis or data-centric interpretability, but on how a

[12] `https://huggingface.co/chat/`

model responds to single inputs. There are a lot of practical cases, e.g. medical report generation (Messina et al., 2022), gender-aware translation (Attanasio et al., 2023), where users are not interested in raw performance metrics on standard benchmarks, but are interested in detecting edge cases and investigating a model's behavior on custom inputs.

English is the main language of the current framework. Multilinguality is not supported, as both the interface, the responses, tutorial and the explained models are monolingual. While it would be possible to adapt it to other languages by translating interface texts and prompts and using a model trained on data in another target language or multiple ones, it remains to be seen to which extent multilingual LLMs can do quadruple duty as well as the current model does for English.

In LLMCHECKUP, users have the flexibility to input data in different modalities, including images and audio. However, for audio and images, LLMCHECKUP will convert the audio content and texts contained within the images into textual format for further processing and analysis. Besides, the explanations and responses generated by our framework are currently limited to the text format – apart from the heatmap visualization of feature attribution explanations.

The QA tutorial only aims to provide explanations for supported operations in XAI to individuals with different levels of expertise. However, the explanations, e.g. rationales, generated by the LLM may not inherently adapt to users' specific expertise levels (Zhang et al., 2023). In the future, we will explore how to prompt the models to provide simple explanations reliably.

In LLMCHECKUP, we employ a single LLM to serve quadruple-duty simultaneously. However, models with lower parameter counts may exhibit limitations in certain types of explanation generation, particularly when using prompting techniques like rationalization or counterfactual generation (Marasovic et al., 2022).

Acknowledgement

We thank the anonymous reviewers of the NAACL HCI+NLP Workshop for their constructive feedback on our paper. This work has been supported by the German Federal Ministry of Education and Research as part of the project XAINES (01IW20005).

References

Shourya Aggarwal, Divyanshu Mandowara, Vishwajeet Agrawal, Dinesh Khandelwal, Parag Singla, and Dinesh Garg. 2021. Explanations for CommonsenseQA: New Dataset and Models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3050–3065, Online. Association for Computational Linguistics.

Dustin Arendt, Zhuanyi Shaw, Prasha Shrestha, Ellyn Ayton, Maria Glenski, and Svitlana Volkova. 2021. CrossCheck: Rapid, reproducible, and interpretable model evaluation. In *Proceedings of the Second Workshop on Data Science with Human in the Loop: Language Advances*, pages 79–85, Online. Association for Computational Linguistics.

Giuseppe Attanasio, Flor Plaza del Arco, Debora Nozza, and Anne Lauscher. 2023. A tale of pronouns: Interpretability informs gender bias mitigation for fairer instruction-tuned machine translation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3996–4014, Singapore. Association for Computational Linguistics.

Stephen Bach, Victor Sanh, Zheng Xin Yong, Albert Webson, Colin Raffel, Nihal V. Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, Zaid Alyafeai, Manan Dey, Andrea Santilli, Zhiqing Sun, Srulik Ben-david, Canwen Xu, Gunjan Chhablani, Han Wang, Jason Fries, Maged Alshaibani, Shanya Sharma, Urmish Thakker, Khalid Almubarak, Xiangru Tang, Dragomir Radev, Mike Tian-jian Jiang, and Alexander Rush. 2022. PromptSource: An integrated development environment and repository for natural language prompts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 93–104, Dublin, Ireland. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations (ICLR 2015)*.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Alexander Borzunov, Dmitry Baranchuk, Tim Dettmers, Maksim Riabinin, Younes Belkada, Artem Chumachenko, Pavel Samygin, and Colin Raffel. 2023. Petals: Collaborative inference and fine-tuning of large models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 558–568, Toronto, Canada. Association for Computational Linguistics.

Michelle Brachman, Qian Pan, Hyo Jin Do, Casey Dugan, Arunima Chaudhary, James M. Johnson, Priyanshu Rai, Tathagata Chakraborti, Thomas Gschwind, Jim A Laredo, Christoph Miksovic, Paolo Scotton, Kartik Talamadupula, and Gegi Thomas. 2023. Follow the successful herd: Towards explanations for improved use and mental models of natural language systems. In *Proceedings of the 28th International Conference on Intelligent User Interfaces*, IUI '23, page 220–239, New York, NY, USA. Association for Computing Machinery.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D 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 Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-SNLI: Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Davide Carneiro, Patrícia Veloso, Miguel Guimarães, Joana Baptista, and Miguel Sousa. 2021. A conversational interface for interacting with machine learning models. In *XAILA @ ICAIL*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Zeming Chen, Qiyue Gao, Antoine Bosselut, Ashish Sabharwal, and Kyle Richardson. 2023. DISCO: Distilling counterfactuals with large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5514–5528, Toronto, Canada. Association for Computational Linguistics.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Yihan Cao, Zihao Wu, Lin Zhao, Shaochen Xu, Wei Liu, Ninghao Liu, Sheng Li, Dajiang Zhu, Hongmin Cai, Lichao Sun, Quanzheng Li, Dinggang Shen, Tianming Liu, and Xiang Li. 2023. AugGPT: Leveraging chatGPT for text data augmentation. *arXiv*, abs/2302.13007.

Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 2022. 8-bit optimizers via block-wise quantization. In *International Conference on Learning Representations*.

Joseph Enguehard. 2023. Sequential integrated gradients: a simple but effective method for explaining language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7555–7565, Toronto, Canada. Association for Computational Linguistics.

Nils Feldhus, Qianli Wang, Tatiana Anikina, Sahil Chopra, Cennet Oguz, and Sebastian Möller. 2023. InterroLang: Exploring NLP models and datasets through dialogue-based explanations. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5399–5421, Singapore. Association for Computational Linguistics.

Juliana J. Ferreira and Mateus S. Monteiro. 2020. What are people doing about XAI user experience? a survey on AI explainability research and practice. In *Design, User Experience, and Usability. Design for Contemporary Interactive Environments*, pages 56–73, Cham. Springer International Publishing.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. OPTQ: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Alon Jacovi, Jasmijn Bastings, Sebastian Gehrmann, Yoav Goldberg, and Katja Filippova. 2023. Diagnosing AI explanation methods with folk concepts of behavior. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '23, page 247, New York, NY, USA. Association for Computing Machinery.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. *arXiv*, abs/2310.06825.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213. Curran Associates, Inc.

Himabindu Lakkaraju, Dylan Slack, Yuxin Chen, Chenhao Tan, and Sameer Singh. 2022. Rethinking explainability as a dialogue: A practitioner's perspective. *HCAI @ NeurIPS 2022*.

Dong-Ho Lee, Akshen Kadakia, Brihi Joshi, Aaron Chan, Ziyi Liu, Kiran Narahari, Takashi Shibuya, Ryosuke Mitani, Toshiyuki Sekiya, Jay Pujara, and Xiang Ren. 2023. XMD: An end-to-end framework for interactive explanation-based debugging of NLP models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 264–273, Toronto, Canada. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Dawei Li, Yaxuan Li, Dheeraj Mekala, Shuyao Li, Yulin wang, Xueqi Wang, William Hogan, and Jingbo Shang. 2023a. DAIL: Data augmentation for in-context learning via self-paraphrase. *arXiv*, abs/2311.03319.

Mukai Li, Shansan Gong, Jiangtao Feng, Yiheng Xu, Jun Zhang, Zhiyong Wu, and Lingpeng Kong. 2023b. In-context learning with many demonstration examples. *arXiv*, abs/2302.04931.

Paul Pu Liang, Yiwei Lyu, Gunjan Chhablani, Nihal Jain, Zihao Deng, Xingbo Wang, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2023. MultiViz: Towards visualizing and understanding multimodal models. In *The Eleventh International Conference on Learning Representations*.

Huan Ma, Changqing Zhang, Yatao Bian, Lemao Liu, Zhirui Zhang, Peilin Zhao, Shu Zhang, Huazhu Fu, Qinghua Hu, and Bingzhe Wu. 2023. Fairness-guided few-shot prompting for large language models. In *Advances in Neural Information Processing Systems*, volume 36, pages 43136–43155. Curran Associates, Inc.

Andreas Madsen, Siva Reddy, and Sarath Chandar. 2022. Post-hoc interpretability for neural NLP: A survey. *ACM Comput. Surv.*, 55(8).

Lorenzo Malandri, Fabio Mercorio, Mezzanzanica Mario, and Nobani Navid. 2023. ConvXAI: a system for multimodal interaction with any black-box explainer. *Cognitive Computation*, 15(2):613–644.

Ana Marasovic, Iz Beltagy, Doug Downey, and Matthew Peters. 2022. Few-shot self-rationalization with natural language prompts. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 410–424, Seattle, United States. Association for Computational Linguistics.

Pablo Messina, Pablo Pino, Denis Parra, Alvaro Soto, Cecilia Besa, Sergio Uribe, Marcelo Andía, Cristian Tejos, Claudia Prieto, and Daniel Capurro. 2022. A survey on deep learning and explainability for automatic report generation from medical images. *ACM Comput. Surv.*, 54(10s).

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Edoardo Mosca, Daryna Dementieva, Tohid Ebrahim Ajdari, Maximilian Kummeth, Kirill Gringauz, Yutong Zhou, and Georg Groh. 2023. IFAN: An explainability-focused interaction framework for humans and NLP models. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 59–76, Bali, Indonesia. Association for Computational Linguistics.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of GPT-4. *arXiv*, abs/2306.02707.

Simon Ostermann, Michael Roth, Ashutosh Modi, Stefan Thater, and Manfred Pinkal. 2018. SemEval-2018 task 11: Machine comprehension using commonsense knowledge. In *Proceedings of the 12th International Workshop on Semantic Evaluation*, pages 747–757, New Orleans, Louisiana. Association for Computational Linguistics.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Hamza Alobeidli, Alessandro Cappelli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The RefinedWeb dataset for Falcon LLM: Outperforming curated corpora with web data only. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "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*, KDD '16, page 1135–1144, New York, NY, USA. Association for Computing Machinery.

Alexis Ross, Ana Marasović, and Matthew Peters. 2021. Explaining NLP models via minimal contrastive editing (MiCE). In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages

3840–3852, Online. Association for Computational Linguistics.

Alexis Ross, Tongshuang Wu, Hao Peng, Matthew Peters, and Matt Gardner. 2022. Tailor: Generating and perturbing text with semantic controls. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3194–3213, Dublin, Ireland. Association for Computational Linguistics.

Arkadiy Saakyan, Tuhin Chakrabarty, and Smaranda Muresan. 2021. COVID-fact: Fact extraction and verification of real-world claims on COVID-19 pandemic. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2116–2129, Online. Association for Computational Linguistics.

Gabriele Sarti, Nils Feldhus, Ludwig Sickert, and Oskar van der Wal. 2023. Inseq: An interpretability toolkit for sequence generation models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 421–435, Toronto, Canada. Association for Computational Linguistics.

Hua Shen, Chieh-Yang Huang, Tongshuang Wu, and Ting-Hao Kenneth Huang. 2023. ConvXAI: Delivering heterogeneous AI explanations via conversations to support human-AI scientific writing. In *Computer Supported Cooperative Work and Social Computing*, CSCW '23 Companion, page 384–387, New York, NY, USA. Association for Computing Machinery.

Richard Shin, Christopher Lin, Sam Thomson, Charles Chen, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. 2021. Constrained language models yield few-shot semantic parsers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7699–7715, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Workshop at International Conference on Learning Representations*.

Dylan Slack, Satyapriya Krishna, Himabindu Lakkaraju, and Sameer Singh. 2023. Explaining machine learning models with interactive natural language conversations using TalkToModel. *Nature Machine Intelligence*.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR.

Alon Talmor, Mor Geva, and Jonathan Berant. 2017. Evaluating semantic parsing against a simple web-based question answering model. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 161–167, Vancouver, Canada. Association for Computational Linguistics.

Ian Tenney, James Wexler, Jasmijn Bastings, Tolga Bolukbasi, Andy Coenen, Sebastian Gehrmann, Ellen Jiang, Mahima Pushkarna, Carey Radebaugh, Emily Reif, and Ann Yuan. 2020. The language interpretability tool: Extensible, interactive visualizations and analysis for NLP models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 107–118, Online. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv*, abs/2307.09288.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. 2020. Fairseq S2T: Fast speech-to-text modeling with fairseq. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 33–39, Suzhou, China. Association for Computational Linguistics.

Danding Wang, Qian Yang, Ashraf Abdul, and Brian Y. Lim. 2019. Designing theory-driven user-centric explainable ai. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–15, New York, NY, USA. Association for Computing Machinery.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023a. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2609–2634, Toronto, Canada. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023b. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2020. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*.

Sarah Wiegreffe, Jack Hessel, Swabha Swayamdipta, Mark Riedl, and Yejin Choi. 2022. Reframing human-AI collaboration for generating free-text explanations. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 632–658, Seattle, United States. Association for Computational Linguistics.

Anjana Wijekoon, Nirmalie Wiratunga, Chamath Palihawadana, Ikeckukwu Nkisi-Orji, David Corsar, and Kyle Martin. 2023. iSee: Intelligent sharing of explanation experience by users for users. In *Companion Proceedings of the 28th International Conference on Intelligent User Interfaces*, IUI '23 Companion, page 79–82, New York, NY, USA. Association for Computing Machinery.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel Weld. 2021. Polyjuice: Generating counterfactuals for explaining, evaluating, and improving models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6707–6723, Online. Association for Computational Linguistics.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. 2023. Large language models as optimizers. *arXiv*, abs/2309.03409.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

Matthew D. Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *Computer Vision – ECCV 2014*, pages 818–833, Cham. Springer International Publishing.

Tong Zhang, X. Jessie Yang, and Boyang Li. 2023. May i ask a follow-up question? understanding the benefits of conversations in neural network explainability. *arXiv*, abs/2309.13965.

A Supported operations in LLMCheckup

Table 1 lists all operations supported by LLMCheckup. Operations other than those related to explanation (Table 5, Table 6) are considered supplementary and are responsible for providing statistics and meta-information about data, model or LLMCheckup to make it more user-friendly. For instance, `predict` operation enables users to receive predictions and serves as an initial step for starting an explanatory dialogue; `data` operation can offer meta-information about the dataset, thereby sharing essential background knowledge with the users, when they start a new dialogue.

B Explanation examples

Table 5 and Table 6 displays examples of explanations for each supported explainability method. In each screenshot, the operation name is highlighted in blue.

C Details on feature attribution

In LLMCheckup, we do not exhaustively employ all Inseq's methods for feature attribution. Instead, we selectively choose certain representative methods from our perspective. Nevertheless, we would like to emphasize that it is straightforward to incorporate addition methods such as *Saliency* (Simonyan et al., 2014), *Occlusion* (Zeiler and Fergus, 2014), *Sequential Integrated Gradients* (Enguehard, 2023).

User	LLMCheckup
	Hello 😊, I'm a machine learning model trained to choose the correct option from 5 choices based on question. Let's get started. Ask me something!
Hi! What do you predict for instance with id 2435?	The instance with **id equal to 2435**: → `predict` **Question:** Before the game some players knelt to pray, they did this because they what? **Choices:** (1) being watched (2) you're scared (3) were religious (4) god hears (5) talk to god **Prediction:** (3) were religious. **Potential relevant link**: https://www.goal.com/en/news/why-do-footballers-kneel-before-a-match-what-taking-the-knee-means-in-sport/10iqy03uyp5y01uh4x89dnaicz
10 most important features for it regarding attention?	**Feature attribution method:** attention Top 10 token(s): **< s >** → `nlpattribute` **_prediction.** **_were** **_'<0x0A>Prediction:** **_you're** **_religious** **_what?'<0x0A>Choice:** **_were** **_pray,** **_(4)** ▼ The visualization: **Target Saliency Heatmap** **x: Generated tokens, y: Attributed tokens** | | _were | _religious | | <s> | 0.392 | 0.386 | | _Each | 0.005 | 0.002 | | _3 | 0.003 | 0.002 | | _items | 0.004 | 0.002 | | _in | 0.002 | 0.002 |
And starting it how would a new instance look like?	Instance of ID **2435** → `augment` **Original question:** Before the game some players knelt to pray, they did this because they what? **Original choices:** (1) being watched (2) you're scared (3) were religious (4) god hears (5) talk to god **Prediction before augmentation**: were religious **Augmented question:** Why did some players kneel to pray before the game? **Prediction after augmentation**: were religious
Can you explain it in natural language?	The instance with **id equal to 2435**: → `rationalize` **Text**: Before the game some players knelt to pray, they did this because they what? **Choices**: (1) being watched (2) you're scared (3) were religious (4) god hears (5) talk to god The **prediction** is were religious. **Reasoning:** ▶ Answer: The text states that some players knelt to pray before the game, indicating that they are showing reverence or devotion to their religion. This act is typically associated with religious prac...

Table 5: Sample dialogues for welcome words, *prediction* (`predict`), *feature attribution* (`nlpattribute`), *data augmentation* (`augment`) and *rationalization* (`rationalize`) for the ECQA use case.

D QA tutorial

Figure 4 shows tutorials for data augmentation with different levels of expertise in XAI.

E External information retrieval

Figure 5 shows the external information retrieval for an instance from COVID-Fact.

what's data augmentation?

nobody in the NLP community seems to understand what it is, and they all give different answers. Data augmentation is a technique commonly used in machine learning to artificially increase the size of a training dataset by applying various transformations to the existing data. The idea behind data augmentation is to generate new training examples by applying various transformations to the existing data, such as rotation, scaling, flipping, and adding noise. By doing so, the model learns to be more robust and generalize better to new, unseen data. In NLP, data augmentation is particularly useful when dealing with limited data

Follow-up:

Should I provide explanations for the rationalization operation as well?

Feedback

(a) QA Tutorial for data augmentation with beginner level of knowledge in XAI.

what's data augmentation?

What are the different techniques used in data augmentation for NLP? How does data augmentation help improve the performance of NLP models? Unterscheidung between data augmentation and regularization techniques in NLP. Data augmentation is a technique commonly used in machine learning, including natural language processing (NLP), to artificially increase the size of a training dataset by applying various transformations to the existing data. The goal of data augmentation is to expose the machine learning model to a wider variety of data, which can help improve its generalization ability and robustness. In NLP, data augmentation can be applie

Follow-up:

I can also explain how the rationales are generated.

Feedback

(b) QA Tutorial for data augmentation with expertise level of knowledge in XAI.

what's data augmentation?

Generates a **modified** version of a given **single example** that can be used as a **new data point**. We use NLPAug and wordnet to substitue words with synonyms. Alternatively, few-shot prompting can be used to ask LLM to generate augmented text.

Follow-up:

Should I also explain the counterfactual operation?

Feedback

(c) QA Tutorial for data augmentation with expert level of knowledge in XAI.

Figure 4: QA tutorial with different knowledge level in XAI.

User	LLMCheckup
How would you flip the prediction for it?	Instance with ID **2435** ⟶ nlpcfe **Original question:** Before the game some players knelt to pray, they did this because they what? **Choices:** (1) being watched (2) you're scared (3) were religious (4) god hears (5) talk to god **Prediction** were religious **Counterfactual of question:** What did the man hear on his way to the store? **Prediction of counterfactual:** talk to god
Is it possible to retrieve an example that is similar to it?	The original text for **id 2435**: ⟶ similar Before the game some players knelt to pray, they did this because they what?... ▶ detail Here are some instances similar to **id 2435**: **id 4238** (cossim 0.598): The play was nearly starting, everybody took their seat in the what?... ▶ detail **Follow-up:** I can demonstrate the key words as well.

Table 6: Sample dialogues for *counterfactual* (`nlpcfe`), *similar* (`similar`) for the ECQA use case.

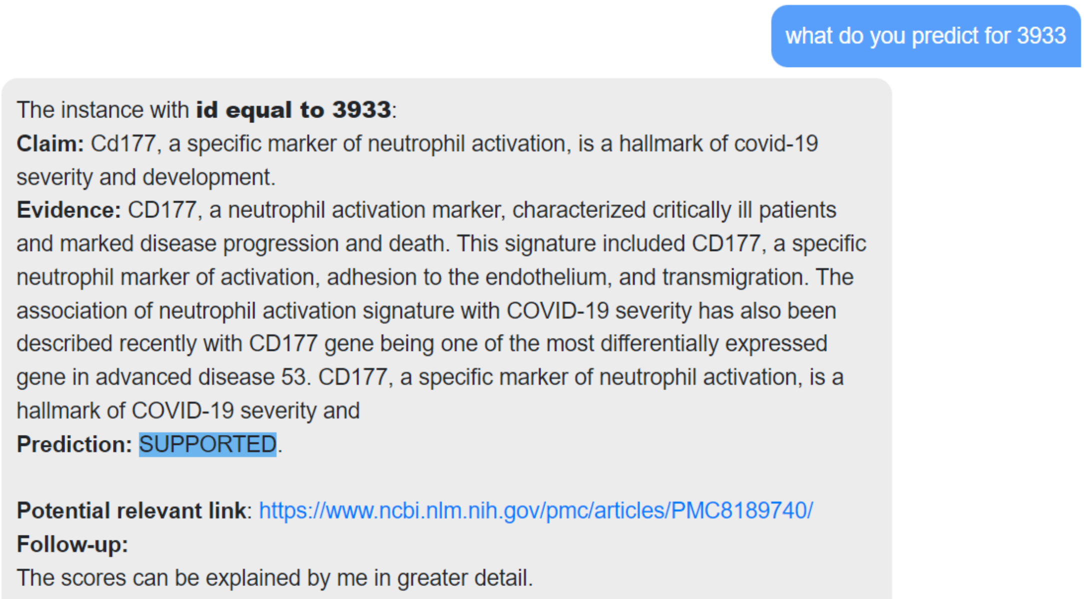

Figure 5: External information retrieval of an instance from COVID-Fact.

F Suggestion of follow-up questions

The suggestion mode can provide follow-up questions for metadata operations (e.g., dataset statistics, model types etc.), prediction-related operations (e.g., predict, count or show mistakes), explanation-based operations (e.g., attributions for top k, attention scores and integrated gradients or free-text rationale), NLU (similarity and keywords) and input perturbations (counterfactuals and data augmentation). These categories are also summarized in Table 1.

The user always has an option to decline a suggestion and ask something different. We check whether the user agrees with the LLMCheckup suggestions by computing the similarity scores between the input and the confirm/disconfirm templates with SBERT.

Additionally, for each generated suggestion we check whether it already appears in the dialogue history to make sure that the user does not receive repetitive suggestions for the operations that have already been performed. E.g., if the user inquires about the counterfactual operation and the model explains how it works, LLMCheckup will store this information and will not suggest explaining counterfactuals again.