ChatHF: Collecting Rich Human Feedback from Real-time Conversations

Andrew Li, Zhenduo Wang, Ethan Mendes, Duong Minh Le, Wei Xu, Alan Ritter
Georgia Institute of Technology
{ali403, zwang926, emendes3, dminh6}@gatech.edu; {wei.xu, alan.ritter}@cc.gatech.edu

Abstract

We introduce ChatHF, an interactive annotation framework for chatbot evaluation, which integrates configurable annotation within a chat interface. ChatHF can be flexibly configured to accommodate various chatbot evaluation tasks, for example detecting offensive content, identifying incorrect or misleading information in chatbot responses, and chatbot responses that might compromise privacy. It supports post-editing of chatbot outputs and supports visual inputs, in addition to an optional voice interface. ChatHF is suitable for collection and annotation of NLP datasets, and Human-Computer Interaction studies, as demonstrated in case studies on image geolocation and assisting older adults with daily activities. ChatHF is publicly accessible at https://chat-hf.com.

1 Introduction

Advances in large language models and vision-language models have led to surprisingly effective chatbots such as GPT-4V, Llama-3, Gemini, and many more. While these chatbots display interesting and useful emergent capabilities, they can also exhibit some undesirable behaviors. How to evaluate LLM-based chatbots remains a challenge. Some studies make use of automated GPT-based evaluations (Liu et al., 2023), but human evaluation is still needed to measure the effectiveness of these automatic metrics on new tasks. Other recent works, such as Chatbot Arena (Chiang et al., 2024), make use of human evaluators, but present only holistic evaluations of which model produces "better" outputs (i.e., preference).

In this paper we present an interactive framework, **ChatHF** (§3), for evaluation and analysis of chatbots that supports fine-grained error detection and collecting human feedback simultaneously (§5). Rather than the common setup where researchers first collect LLM-generated responses then evaluate (or annotate) as an afterthought, we envision an approach where the human annotators seamlessly interleave annotation with conversation. That is, human evaluators directly chat with LLMs on specific topics relevant to the phenomenon to be studied (see Figure 1). This not only saves the annotator's time and energy to accomplish two tasks in a single pass, but also encourages annotators to engage in more interesting and complex conversations — as we show in two case studies: cooking chatbot (Le et al., 2023) and multimodal privacy QA (Mendes et al., 2024).

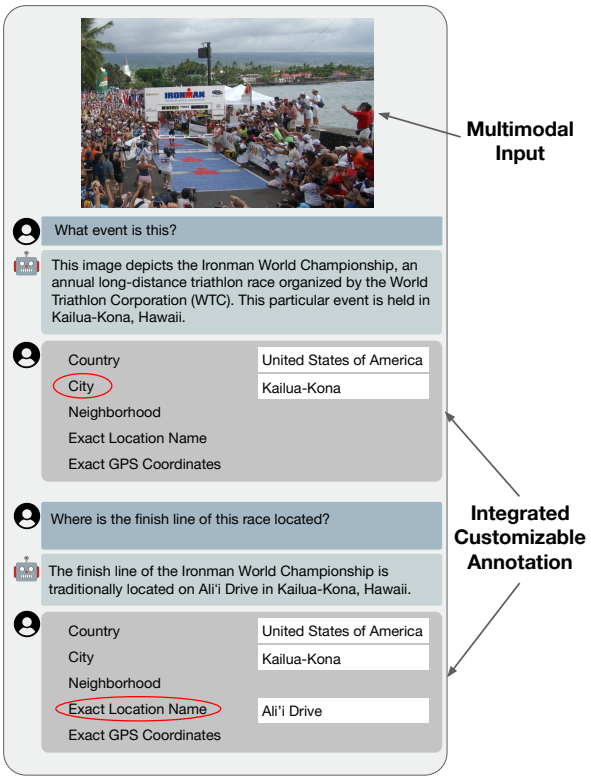

Figure 1: ChatHF incorporates integrated multimodal dialogue annotation. This concept figure shows an example for privacy-preserving moderation in conversational geolocation QA (Mendes et al., 2024).

ChatHF is flexible and can be configured for many annotation tasks, such as offensive outputs (Baheti et al., 2021), misinformation (Musi et al.,

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing
System Demonstrations, pages 270–279

2023), or compromised privacy (Zhang et al., 2024), enabling the creation of curated conversational datasets and the study of emergent behaviors in LLM-based chatbots. Its unique features include flexible configuration, post-editing of chatbot outputs, and multimodal inputs with images and voice interaction (speech-to-text and text-to-speech). ChatHF supports both standard NLP data collection and annotation, as well as interactive Human-Computer Interaction (HCI) studies involving chatbots. In the two case studies (§6 and §7), we used ChatHF to (1) collect a dataset of image geolocation conversations that are labeled with the granularity of location information revealed at each step of the conversation, and (2) as an interface, to support an HCI user study on older adults using chatbots to assist with activities of daily living.

2 Related Work

The field of text annotation tools has seen iterative advancements in the past decade. This section gives a high-level overview of previous text annotation tools from two perspectives: conversational texts evaluation and human feedback management.

ChatBot Evaluation STAV (Stenetorp et al., 2011) and BRAT (Stenetorp et al., 2012) are examples of early text annotation tools. BRAT supports manual curation of the annotation and is optimized for rich structured annotation tasks and annotator productivity. It also provides high-quality annotation visualization. More recent tools like POTATO (Pei et al., 2022) support higher degrees of configuration and customization and provide even better quality control and productivity enhancement. However, most of them are mostly useful for annotation tasks within one sentence or one paragraph rather than multi-turn conversations.

Within the field of conversational text annotation tools, there has been only a limited amount of available open-source tools. LIDA (Collins et al., 2019) was the first tool designed specifically for annotating multi-turn conversational text data (Liu et al., 2020). Its later evolution MATILDA (Cucurnia et al., 2021) improved it by facilitating multilingual and multi-annotator annotations. However, these tools have no web interfaces and require some technical knowledge for model integration and configuration, which inhibits their accessibility. EZCAT (Guibon et al., 2022) can be used directly on their web application to both configure text labels, on a message or conversation level, and go through the annotation process. However, EZCAT does not have the option to collect multiple labels per turn. In this work, we aim to supply this field with a flexible multi-purpose annotation tool with a configurable and easy-to-use interface.

Human Feedback It is increasingly important to audit and evaluate LLMs and VLMs by human, and in turn, learn from rich and diverse human feedback (see the excellent survey by Pan et al. (2024)) to improve the model's performance. However, in addition to their restricted accessibility, existing annotation tools are also limited to only utilizing human feedback at the end of each conversation as an afterthought (Heeman et al., 2002; Garg et al., 2022; Klie et al., 2018). For example, INCEpTION (Klie et al., 2018) and GATE (Cunningham et al., 2002) provide large feature sets, but cannot display conversation data as turns (Cucurnia et al., 2021). LIDA and MATILDA fully support conversational text annotation tasks such as task-oriented dialogue systems. However, their frameworks can only be used to annotate static recorded dialogues. Such an annotation scheme fails to address human feedback during the conversation, which leads to systemic productivity loss.

In contrast, we present a customizable annotation tool capable of managing real-time human feedback during conversations. Annotators are allowed to edit model-generated utterances and to reverse and modify chat history to reflect their feedback. We track all these edits and reversals, as well as the reasons why these changes are made as free-text and/or multi-choice annotations.

3 Chatbot Infrastructure

ChatHF supports various models and configuration options for easy prompt engineering and experimentation. Our public web demo supports testing OpenAI, Anthropic, Google Gemini, and Mistral models directly through their respective APIs. For security, all configuration settings like API keys are stored client-side, and can be downloaded and loaded as a YAML file for easy sharing.

Run locally or self-hosted, ChatHF can be used with Ollama[1] and Huggingface[2] models. Additionally, API keys can be hidden in an environment file. For more complex generation schemes, sample code is provided to set up a custom arbitrary generate function.

[1] https://ollama.com/
[2] https://huggingface.co/

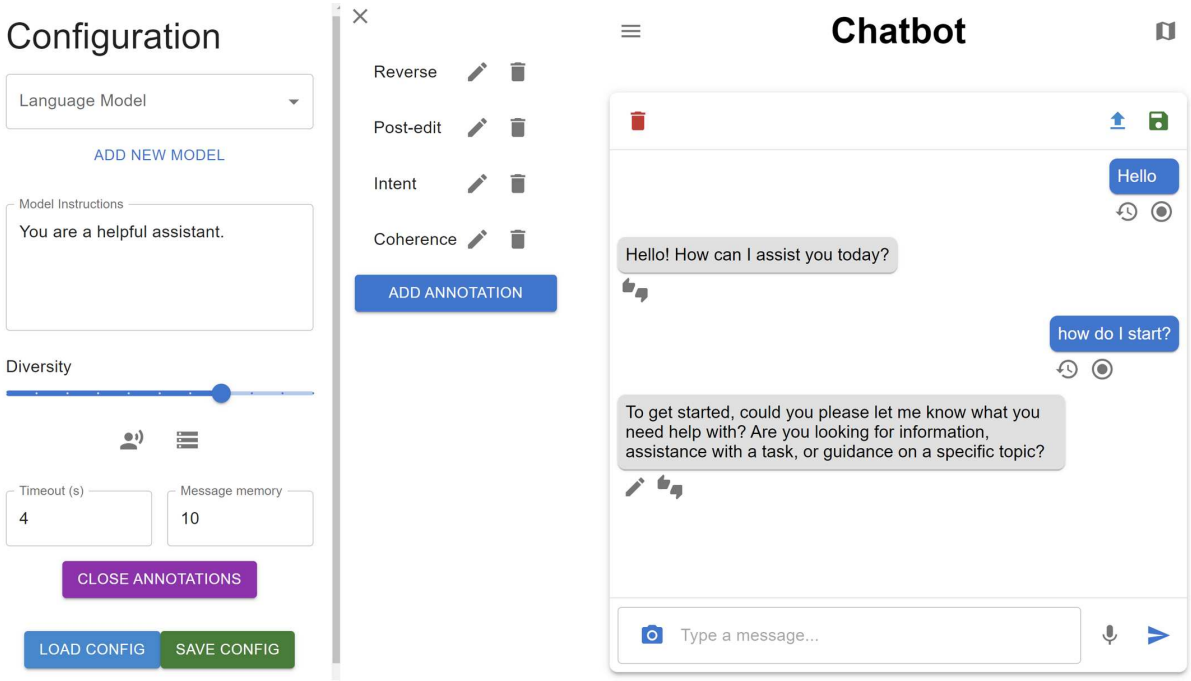

Figure 2: Screenshot of the main ChatHF interface. Configuration options can be modified on the left panel, with changes automatically reflected on the chat interface on the right. See more screenshots of included features in Appendix A.

ChatHF also offers several configuration options to experiment with model settings, such as the system prompt, temperature, timeout limit, and conversation history memory length. Any changes are automatically reflected in the chat window. At each turn of the conversation, the model is passed the conversation history truncated to the memory length with the system prompt inserted at the start, and the model generates a response with the set temperature, timing out if the processing time exceeds the timeout limit.

Multimodality To support voice chatbot applications, ChatHF integrates the option for text-to-speech on model outputs and speech-to-text with microphone input. Features such as press-to-talk, continuous listening, and text-to-speech are customizable, allowing ChatHF to cater to different needs from accessibility to hands-free operations.

Interfacing with Vision-Language models are also possible as ChatHF allows for image input to the chatbox, which are simply saved as Base64 images in the chat history to be sent to the model.

User Interface ChatHF is built on a Flask backend and a React frontend, with a publicly available codebase released under an Apache 2.0 license. We include a Flask backend written in Python to allow for easier integration of custom models or generation schemes into our chatbot interface. Text-to-speech and and speech-to-text are implemented via Azure AI Speech[3], using their proprietary models.

Chat History All messages in the chat history are saved into a JSON log file, timestamped with the date and time. User feedback is saved with each message with the user-specified name and value. In the case of a reversal, the old chat history is not overwritten, and instead, an additional chat history created with all messages until the reversal point.

ChatHF supports downloading the log file locally or to a database such as Google Firebase[4], as well as uploading a log file to view the chat history or edit the evaluation later. The user also has the option to clear the chat history to start a new conversation.

4 Customizable Annotation Configuration

In addition to the chatbot interface, ChatHF enables integrated on-the-fly human evaluation of the generated conversation and allows users to customize the annotation formats according to their needs. During a conversation, the user can annotate

[3]https://azure.microsoft.com/en-us/products/ai-services/ai-speech

[4]https://firebase.google.com/

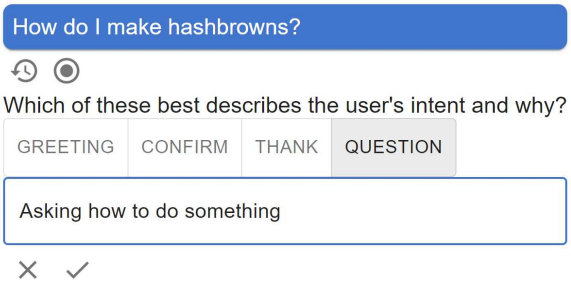

Figure 3: Demonstration of a multiple choice annotation for intent labeling of an AI cooking application with the option to give an explanation.

user messages, the generated model responses, or both. These messages can be annotated in various formats including binary, Likert-scale, multiple-choice, multiple-select, and free-text inputs. All annotation types can have a custom question and the option to require the annotator to provide an explanation through an additional textbox. Furthermore, the labels for binary, Likert-scale, multiple-choice, and multiple-select annotations are all customizable, and annotations can be specific to user messages, model responses, or both.

The full control of the annotation format and customizable labels is implemented as an annotator's configuration panel in our tool located in the upper left corner. The panel settings can be saved and uploaded for reuse later. If needed, custom annotations can also be edited and deleted.

In the chatbot interface, if the annotation feature is turned on, icons representing each annotation type appear below each user message or model response (See Figure 2). Users can click on an annotation icon to reveal its prompt and input the specified response. This process is quick and responsive to facilitate real-time fine-grained data collection.

To demonstrate the efficacy of ChatHF's customizable evaluation, we describe and release sample configuration files for our two example use cases.

5 Rich Human Feedback

Along with the more traditional formats for human feedback, ChatHF includes two unique annotation types to collect real-time post-editing and reversal data for richer human feedback.

Post-editing Post-editing can be useful when only a portion of the model response is incorrect

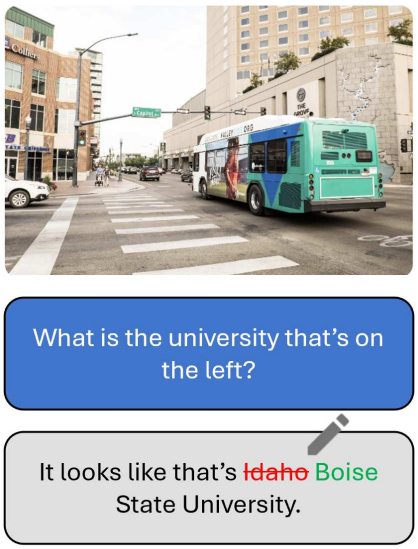

Figure 4: In this visual question-answering task, the model is unable to fully identify the university in the picture. The user uses a post-edit to correct the mistake.

and requires changing or deleting, or if the output could be improved with just a minor addition. For instance, hallucinations and toxic language can be edited out and the offending spans can be easily extracted by comparing the post-edited and original text. Post-editing is also helpful when the model is partially correct, such as Figure 4, allowing for fine-grained corrections.

Crucially, post-editing corrects the conversation history, so that errors cannot propagate. This creates a more seamless chat experience and reduces the need to restart or reverse the conversation, which can be especially valuable in time, effort, or resource sensitive situations such as human studies in real world settings. (§7).

With post-editing selected in the configuration, users can directly edit the LLM-generated response. Similarly to the other annotations, users may be required to provide an explanation for the edit. Upon confirming the post-edit, the previous conversation history before the edit is added to the conversation log as a record of an unsuccessful termination.

Furthermore, each message stores its post-edit, with the most recent edit and original model output saved to the conversation log file To ensure there is a fair evaluation only the most recent bot-message are editable. A list of the edits made will automatically be generated and saved as well.

Reversal In other cases, the model may have made an error that was not caught earlier in the

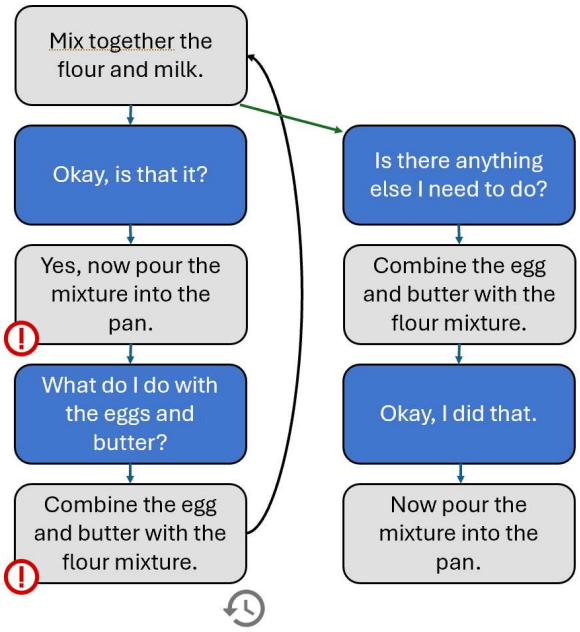

Figure 5: In this cooking assistance dialogue task, the model gives the incorrect order of steps without the user immediately realizing. The user then reverses to previous turn to try again, with the model giving the correct order of steps the second time.

conversation or had errors build up until the conversation was no longer salvageable. For instance, in instructional tasks where the order of instructions is crucial such as cooking, errors cannot be corrected by continuing the conversation, such as in the example in Figure 5. The choice to reverse may even be more subtle, perhaps due to uninteresting or stagnant dialog. Either way, it would be helpful to identify at which turn the conversation was recognized to be unrecoverable, and the point where the direction of the conversation shifted.

ChatHF's reversal option allows for this rich feedback, saving both the reversed chat in the JSON log as well as either an optional annotator-provided reversal explanation or a simple indication of the success of the final dialogue. By default, when saving the conversation log, the current, most recent conversation is considered successful.

Multi-branch Conversation Employing the post-editing and reversal features, ChatHF can be used to explore a branching dialogue with multiple potentially successful continuations or completions. The set of branching conversations created by post-editing and reversing can be represented with a tree structure. At the simplest, a single continuous conversation is represented as one node. Once a branch is made, the conversation truncated at the branching point is set as the parent node, and the messages after the branching point both in the previous conversation and in the new conversation are each a child node.

This tree of interactions over a single overarching conversation topic can be viewed and each node can be selected to jump to a certain conversation.

6 Example Use Case #1: Leveraging ChatHF to Collect Richly Annotated Geolocation Dialogues

We build on ChatHF to construct GPTGeoChat (Mendes et al., 2024), a benchmark for granular privacy controls to moderate image geolocation dialogues, i.e. a human having multi-turn dialogues with a model about the location of an image provided in context. This work showcases the *multi-modal model integration* of ChatHF (see §3). The goal of this task was to train moderation agents to determine whether or not to withhold a vision language model (VLM) response based on whether or not the response violated the granular system privacy configurations:

$$[\texttt{Granularity Config}, \texttt{Image}, \texttt{Dialogue}] \xrightarrow{\text{Agent}} [\texttt{Y}, \texttt{N}]$$

For the studied geolocation task, these granular configurations were location granularities e.g., the *city*, *neighborhood*, or *exact-gps-coordinates* indicating the level of geolocation should be allowed during a conversation.

Data Collection To train and evaluate geolocation moderation agents, 1000 GPT4V-human dialogues are collected towards image geolocation, which form GPTGeoChat (Mendes et al., 2024). In-house annotators conversed with GPT-4v about the location of the image provided in context using ChatHF. During the conversation, each model response was annotated for (1) the finest granularity (*country*, *city*, *neighborhood*, *exact-location-name*, *exact-gps-coordinates*) of the location information revealed so far in the dialogue (2) the corresponding revealed location information e.g. *{'country':'United Kingdom', 'city':'London'}*. For the finest granularity, they represent each of the five granularities along with a *none* option using ChatHF's multiple-choice annotation input. Similarly, they use multiple ChatHF-supported free-form text input fields for the corresponding location information.

Agent	Country	City	Neighborhood	Exact Location Name	Exact GPS Coordinates
LLaVA-13B (prompted)	0.56	0.55	0.52	0.41	0.48
IDEFICS-80B-instruct (prompted)	0.80	0.74	0.67	0.62	0.28
GPT-4v (prompted)	0.86	**0.89**	**0.84**	0.73	0.76
LLaVA-13B (finetuned on GPTGeoChat)	**0.87**	**0.89**	**0.84**	**0.79**	**0.96**

Table 1: Performance (F1-score) on the geolocation moderation task as evaluated on the GPTGeoChat test set (Mendes et al., 2024). The results from the best-performing moderation agent at each granularity are **bolded**.

Figure 6: A pilot HCI user study using ChatHF configured to support a voice assistant cooking chatbot (§7).

Task Evaluation As shown in Table 1, finetuning a smaller model on a small high-quality training set of 400 dialogues from GPTGeoChat yields superior performance on the geolocation dialogue moderation task compared to prompting much larger models.

7 Example Use Case #2: Supporting an HCI User Study for AI Cooking Assistance with Older Adults

We have deployed ChatHF to support the HCI user study on how a cooking chatbot can assist older adults to cook, an important activity of daily living, in coordination with the NSF AI Caring Institute.[5] In our pilot study (Figure 6), we configure ChatHF to work in a real kitchen environment, where the system interacts with users via a voice interface (i.e., speech-to-text and text-to-speech modules) and help him/her to prepare meals. Particularly, we add a "press to talk" button to support the study condition, and reduce the speed of the text-to-speech module. In addition, we conduct prompt engineering to instruct the GPT-4o-mini to provide step-by-step and easy-to-follow guidance to users.[6] Our next plan is to have users from the target population to interact with ChatHF to identify specific challenges that older adults might face when using this technology.

ChatHF is also used to support the human analysis of the responses from different cooking chatbots. In this study, we investigate the outputs of ChatGPT and different fine-tuned versions of GPT-J models (Wang and Komatsuzaki, 2021): the base GPT-J model, GPT-J model incorporated with user intent information (*GPT-J+int*), GPT-J model incorporated with the instruction state information (*GPT-J+cut* and *GPT-J+ctr*), and GPT-J model incorporated with both types of information (*GPT-J+ctr+int*). In each conversation, each model response is annotated as correct or having one of the following errors: wrong order, irrelevant, lack of information, or wrong information. Table 2 demonstrates the error analysis of responses of the models on a subset of the test set of the Chattychef dataset (Le et al., 2023).

Models	Order	Irrelevant	Lack info.	Wrong info.
GPT-J	22.9	10.7	8.4	8.4
GPT-J+int	18.3	8.4	11.5	6.1
GPT-J+cut	20.6	6.9	10.7	6.1
GPT-J+ctr	23.7	3.8	11.5	4.6
GPT-J+ctr+int	22.9	5.3	9.9	7.6
ChatGPT	6.1	0.0	1.5	3.1

Table 2: Percentage of responses from models having each type of error. The evaluation in conducted on 10 multi-turn conversations (131 generated responses) in the test set of the ChattyChef dataset ("Order": wrong order, "Lack info.": lack of information, "Wrong info.": wrong information).

8 Conclusion

We present ChatHF, an interactive, customizable, and open-source tool for evaluating LLM-based multimodal chatbots with rich human feedback and annotation. It supports *real-time* conversation and manual annotation (or human evaluation) at the same time. For example, the users may directly revise LLM-generated response or request the LLM to regenerate another response when they are not satisfied with the LLM-generated response, then continue on the conversation, etc.

[5] https://www.ai-caring.org/

[6] The configured ChatHF for cooking chatbots is available at: https://tinyurl.com/chattychef2

Acknowledgments

We would like to thank Jeongrok Yu for assistance developing the voice interface, in addition to Connor Rosenberg, Kala Jordan, Maribeth Coleman, Vicky Wang, and Jeongrok Yu for conducting the pilot user study. We would also like to thank Azure's Accelerate Foundation Models Research Program for graciously providing access to API-based `GPT-4v`. This research is supported in part by the NSF (IIS-2052498, IIS-2144493 and IIS-2112633), and the Ford Motor Company. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of NSF or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Ashutosh Baheti, Maarten Sap, Alan Ritter, and Mark Riedl. 2021. Just say no: Analyzing the stance of neural dialogue generation in offensive contexts. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4846–4862, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, et al. 2024. Chatbot Arena: An open platform for evaluating llms by human preference. *arXiv preprint arXiv:2403.04132*.

Edward Collins, Nikolai Rozanov, and Bingbing Zhang. 2019. LIDA: Lightweight interactive dialogue annotator. 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): System Demonstrations*, pages 121–126, Hong Kong, China. Association for Computational Linguistics.

Davide Cucurnia, Nikolai Rozanov, Irene Sucameli, Augusto Ciuffoletti, and Maria Simi. 2021. MATILDA - multi-AnnoTator multi-language InteractiveLightweight dialogue annotator. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 32–39, Online. Association for Computational Linguistics.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. GATE: an architecture for development of robust HLT applications. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 168–175, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Muskan Garg, Chandni Saxena, Sriparna Saha, Veena Krishnan, Ruchi Joshi, and Vijay Mago. 2022. CAMS: An annotated corpus for causal analysis of mental health issues in social media posts. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 6387–6396, Marseille, France. European Language Resources Association.

Gaël Guibon, Luce Lefeuvre, Matthieu Labeau, and Chloé Clavel. 2022. EZCAT: an easy conversation annotation tool. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 1788–1797, Marseille, France. European Language Resources Association.

Peter A. Heeman, Fan Yang, and Susan E. Strayer. 2002. DialogueView - an annotation tool for dialogue. In *Proceedings of the Third SIGdial Workshop on Discourse and Dialogue*, pages 50–59, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. 2018. The INCEpTION platform: Machine-assisted and knowledge-oriented interactive annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9, Santa Fe, New Mexico. Association for Computational Linguistics.

Duong Le, Ruohao Guo, Wei Xu, and Alan Ritter. 2023. Improved instruction ordering in recipe-grounded conversation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10086–10104, Toronto, Canada. Association for Computational Linguistics.

Ximing Liu, Wei Xue, Qi Su, Weiran Nie, and Wei Peng. 2020. metaCAT: A metadata-based task-oriented chatbot annotation tool. 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 20–25, Suzhou, China. Association for Computational Linguistics.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: Nlg evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522.

Ethan Mendes, Yang Chen, James Hays, Sauvik Das, Wei Xu, and Alan Ritter. 2024. Granular privacy control for geolocation with vision language models. *Preprint*, arXiv:2407.04952.

Elena Musi, Elinor Carmi, Chris Reed, Simeon Yates, and Kay O'Halloran. 2023. Developing misinformation immunity: How to reason-check fallacious news in a human–computer interaction environment. *Social Media + Society*, 9(1):20563051221150407.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2024. Automatically correcting large language models: Surveying the landscape of diverse automated correction strategies. *Transactions of the Association for Computational Linguistics*, 12:484–506.

Jiaxin Pei, Aparna Ananthasubramaniam, Xingyao Wang, Naitian Zhou, Apostolos Dedeloudis, Jackson Sargent, and David Jurgens. 2022. POTATO: The portable text annotation tool. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 327–337, Abu Dhabi, UAE. Association for Computational Linguistics.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, Avignon, France. Association for Computational Linguistics.

Pontus Stenetorp, Goran Topić, Sampo Pyysalo, Tomoko Ohta, Jin-Dong Kim, and Jun'ichi Tsujii. 2011. Bionlp shared task 2011: Supporting resources. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 112–120, Portland, Oregon, USA. Association for Computational Linguistics.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Zhiping Zhang, Michelle Jia, Hao-Ping (Hank) Lee, Bingsheng Yao, Sauvik Das, Ada Lerner, Dakuo Wang, and Tianshi Li. 2024. "it's a fair game", or is it? examining how users navigate disclosure risks and benefits when using llm-based conversational agents. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '24, New York, NY, USA. Association for Computing Machinery.

A Appendix

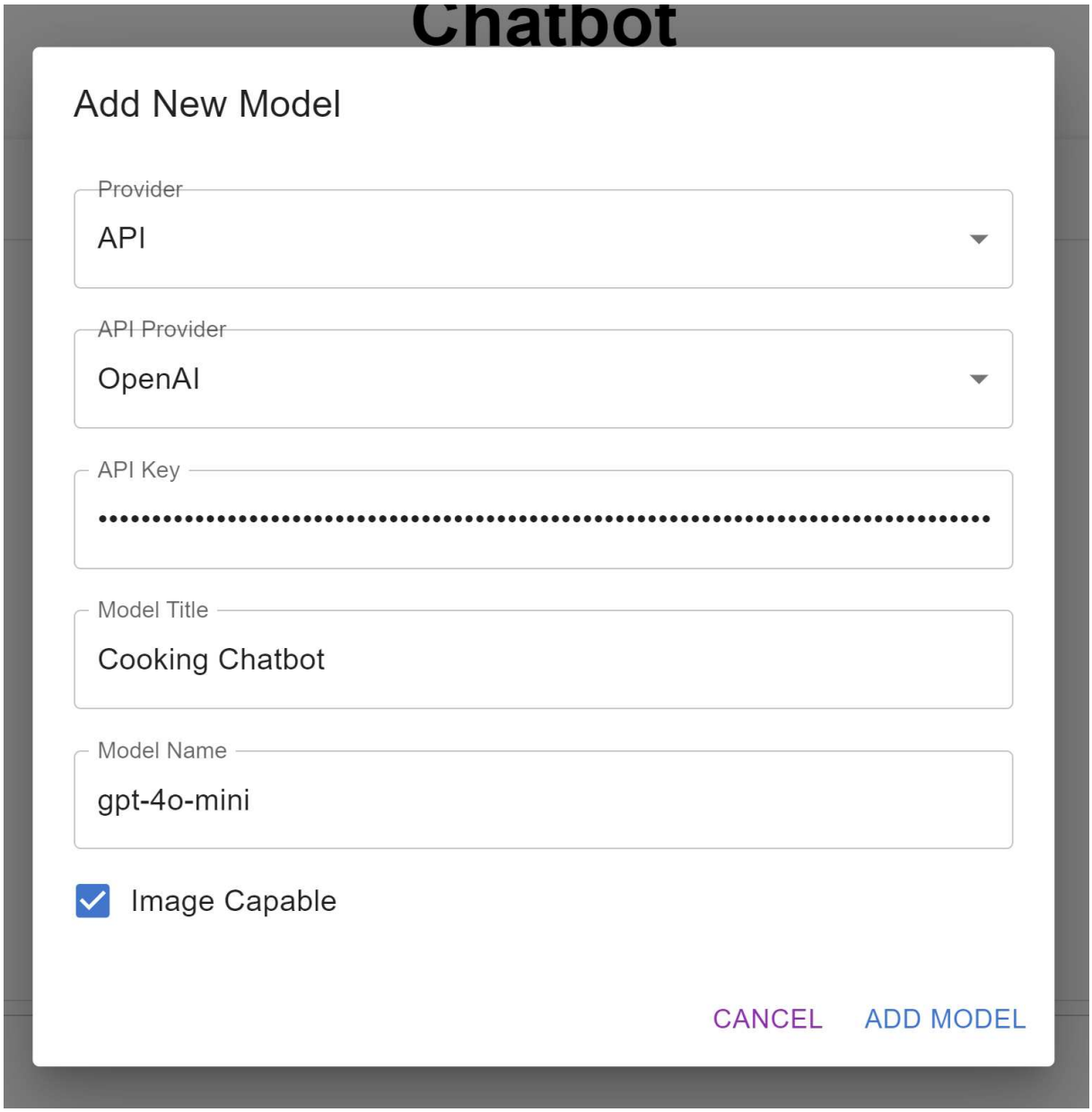

Figure 7: The screen to add a new model to the list.

Figure 8: The screen to create a custom annotation.