Contextual Evaluation of AI: a New Gold Standard

FINALE DOSHI-VELEZ* and ELENA L. GLASSMAN*, John A. Paulson School of Engineering & Applied Sciences, Harvard University, USA

Foundation models, such as large language models, all share two qualities that make them particularly difficult to evaluate: (1) a large surface of their inputs and outputs and (2) their applicability in settings where personal context, goals, preferences, values, and risk tolerances 'at task time' dominant a person's experience of the model's usability and utility. As AI and HCI researchers respectively, we believe that this calls for a fundamental shift in *both* communities about how we evaluate such systems. Specifically, we believe, in personal-context-dominating settings, for this type of model, the gold standard evaluation method should be task-time evaluation by users, made as safe as possible, not benchmarks (as is common in AI) nor user studies in which participants are asked to perform assigned tasks. Like the method of contextual inquiry reveals unanticipated needs, we refer to this evaluation strategy as contextual evaluation.

CCS Concepts: • **Human-centered computing** → **Field studies**; **User studies**.

Additional Key Words and Phrases: AI system design, contextual evaluation

ACM Reference Format:

Finale Doshi-Velez and Elena L. Glassman. 2018. Contextual Evaluation of AI: a New Gold Standard. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 9 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

For decades, the standard way to validate AI systems has been to test the system on some held-out test data. These data could be a portion of the training data, or, to facilitate comparison across different AI systems, a public benchmark.[1] The logic is the following: if an AI system performs well on the benchmark, then it will likely perform well in real settings. Indeed, rigorous statistical theories—such as those on empirical risk minimization—speak to the expected generalization error that an AI system may accrue given its benchmark performance.

However, LLMs and other foundation models have ushered a new era for vetting machine learning models; we believe that one of two key reasons for this is because of the large surface of their inputs and outputs. A task like classifying images may seem large, because of all the images that are possible. However, there are still sensible ways to describe what are the types of images likely to be encountered in a particular setting, setting up ways to flag images that do not match that setting, and using interpretability techniques, e.g., [12], to determine if appropriate features are being used—all in advance of deployment.

Of course, we have always known that benchmarks were imperfect: Since there have been AI benchmarks, there have been concerns about AI systems overfitting to them. Specifically, we recognize that if the scenario in which the AI system is deployed differs from the benchmark, the results may not generalize. However, in many settings, one

*Both authors contributed equally to this research.

[1]Early speech recognition researchers used benchmarks that took the form of *tapes* that they received *in the mail*.

Manuscript submitted to ACM

could still rely on test data from that new scenario. For example, if deploying a digit recognition system in a new country, one could test its performance on some digit data in that country. We could also inspect the model using modern interpretability techniques—or require an inherently interpretable model—to determine whether the features that the model was using were sensible. These evaluations *prior* to deployment can tell us whether the system is likely to perform well if put into use.

Moreover, as AI systems have advanced, so have the benchmarks. For example, when VizWiz [1] was published in 2010, crowds of humans were able to answer questions about the contents of pictures in near real-time, which was far outside the capability of computer vision (CV) algorithms at the time. The benchmark of questions, images, and answers was of no use to CV researchers at the time because it was too hard. Now, the algorithms are sophisticated enough that the VizWiz benchmark dataset is an invaluable asset. There are now benchmarks in many fields of AI that contain much broader collections of data than early benchmarks.

In contrast, we do not believe it is possible to describe all the types of documents an LLM may be asked to summarize, or all the kinds of ideas it may be asked to generate, in a way that meaningfully would connect to generalization. Even if it were possible to unambiguously, objectively label the quality of an input-output pair, the large surface of input-output pairs for even relatively specific tasks, such as summarization, means we cannot sufficiently cover the space of likely inputs. These models are also too big to interpret for global qualities (e.g., overall relying on the right features). Thus, our toolkit of approaches to vet machine learning models prior to deployment, including those that involve human inspection of models and data, fall short in these regimes.

And of course, all of the above was in the case of being able to perfectly assess the quality of an input-output pair. We believe the second of two key reasons we are entering a new era for vetting machine learning models is the expansion of tasks that these foundation models can support. We are now in situations where the values, perspectives, contexts, goals, and preferences of the *specific* user may define a significant portion of what *that* user considers correct. Just because one user believes that an input-output pair is of high quality, that does not mean another user will agree. The fact that correctness of an AI system output may be dominated by user and task specific considerations further questions how one might construct a procedure to meaningfully validate such an AI system in advance.

These challenges require a fundamental shift in how we approach validation in LLMs and other foundation models. While *prior* testing of these models will continue to be an element—for example, one might test to see if an LLM produces reasonable summaries of medical notes on a few patients—these tests in advance of deployment can only be used to flag problems. That is, if a model does poorly on those initial tests, then we should doubt whether it will perform well if deployed; however, if it performs well on those initial tests, we cannot be confident in its performance once deployed.

How then do we use these AI systems with confidence? We argue that we need ways to vet these systems *at task-time*, where users are working in their on contexts on the tasks they personally care about or need to complete. A small group of engineers and domain experts can no longer vet the system for most errors in advance. Instead, workflows and interfaces for contextual evaluation must be designed and built that empower users to efficiently and accurately determine whether the output created in response to *their specific input* is in accordance with their needs and values. That is, significant validation labor will need to be done by the user for each of their tasks—not for the system designer but for themselves.[2] At the same time, significant labor will need to be done on the part of the system and evaluation designers to ensure that the system (1) has minimal negative impact on users when it makes choices that do not match

[2]Perhaps but not necessarily captured for the training of a personalized version of the system.

their partially observable contexts, perferences, and goals and (2) supports users' accurate mental modeling of its performance for their context.

As AI and HCI researchers, respectively, we believe that designing systems, workflows, interfaces, principles, and evaluation techniques that safely support the labor of *contextual evaluation* will require developing methods and best practices that are new to both the HCI and AI. And recognition of the necessity of task-time evaluation within the AI community may create the meaningfully robust bridge between the AI and HCI communities that has struggled to be built in the past.

2 HCI HAS SUPPORTED OTHER TASK-TIME VALIDATION

The idea that a system cannot be perfectly vetted in advance, and thus requires ways to facilitate human inspection at task-time is not new to large AI systems. For example, consider the ways in which we handle the identification of suspicious emails, text messages, or phone calls. While some messages may automatically go to a spam folder, others are tagged as potentially spam. The tag encourages the user to pay more attention to the validity of that specific message or call—a form of *task-time* facilitation—but leaves the final decision of how to treat the message up to the user. The approach of tagging suspicious messages acknowledges the fact that spam-detection systems cannot be fully-vetted in advance; some determinations of spam or not need to happen in the moment.

Similarly, current spelling and grammar checking systems do not change potentially incorrect text for you; they highlight regions of potential error and posit suggestions. Again, this form of feedback acknowledges that these spelling and grammar correction systems will never fully understand the full context and intent of the user such that one could certify that all proposed changes will be correct. Thus, the user must check the system recommendation again at *task-time*.

Another form of spelling correction, handled very differently but still in the spirit of task-time validation, is in the context of internet search. Here, when searching for a misspelled query, the results will instead be shown for a corrected version of the query. However, a flag will indicate explicitly the corrected version used to make the query, and the user will also be provided the option of searching for the original, supposedly misspelled query text. In this case, the system is making a decision on behalf of the user—the spelling correction—but is still allowing the user to correct the system at task-time.

In all of these cases, there is an acknowledgement that the task is such that the AI system cannot be certified to be of sufficiently high quality prior to use that its outputs can be trusted to be the right ones. In some cases, the system does not take action but provides information to suggest a possible alternative (e.g. this message may be spam). In other cases, the system takes an action (e.g. correcting the spelling in a search query), but ensures that the action is sufficiently transparent that the user can decide to discard it.

3 THE SETTING: VALIDATION OF LLM SUMMARIZATION AND IDEATION: WHAT IS MISSING?

We now move into the case of validation for the outputs of large surface models such as modern LLMs, emphasizing the need to develop methods for task-time validation. In the remainder of this document, we will focus on two use-cases for LLMs: summarization and ideation. We choose these two use-cases because they are common applications for LLMs and present interesting opportunities for task-time validation. (Other common use cases, such as querying for information, have more established forms of validation e.g. providing reference links.) That said, we expect the ideas here to be relevant to other use cases as well.

3.1 Summarization.

A very common application of LLMs is summarization. In this setting, the goal of the LLM is to distill key points from a larger text or texts. The process of summarization is inherently a lossy one: the entire point of the exercise is to highlight the most salient points and remove what is redundant and irrelevant. However, notions of relevant or salient involve some form of judgement. Certain information may be useful for a certain downsteam task but not another; certain information may elevate certain perspectives while other information may elevate others.

Examples of uses of LLMs in this summarization context include:

- Judges reviewing court documents, in which local, regional, and/or country-level laws may intersect in particular ways for the given type of case and the judge may also have personal values and preferences over what details are particularly relevant or irrelevant, e.g., CaseText's CoCounsel.
- Professors interested in major themes in course feedback, or, even during a course, interested in real-time summarization of student inputs as they enter thoughts or responses, e.g., Mudslide [6].
- All kinds of administrators automating the process of distilling key points into meeting minutes based on the meeting or a transcript of the meeting, e.g., MeetScript [3].
- Social scientists with large amounts of qualitative data (e.g. narratives) from which they want to identify themes, e.g., PaTAT [4] and Cody [11].
- Government officials needing to process large numbers of public comments or other feedback into the main types of suggestions, e.g., CommunityPulse [8].
- Clinicians wanting summaries of their patient distilled from all the patient's prior lab results and clinical notes, e.g., MedKnowts [10].

While using machine learning for summarization has been an area of natural language processing for some time, the main difference between those works and LLM-based summarization is that prior work tended to focus on much more specific settings. For example, the goal might be to identify the key points from a news articles. There existed many examples of summarization—that is, human-generated bullet points or taglines associated with each news article—providing a large training set. One could apply standard test-train splits to test how well a summarization tool trained on some portion of that data performs on new articles, as measured by match to the human-generated summaries; if the system did well by that metric, one could imagine it would likely do well on other, similarly written news articles. While imperfect—there are many works on summarization metrics—one could do signficant validation in advance.

However, the ease with which LLMs summarize many different kinds of documents—as seen by the use-cases above—means that LLMs are being applied to many more settings than in previous summarization work. In the setting of interest, we may not have large amounts of gold standard, human-generated summaries. Indeed, as the number of settings in which LLM-based summarization may get applied increases, it is highly unlikely that we will be able to keep up in terms of being able to validate the quality of that summarization in advance. Thus, we need that paradigm shift: while we should always check as much as we can about a system in advance, we must be prepare for reality where users of LLM-based summarization will need to validate the summaries at *task-time*, for their specific set of inputs.

3.2 Ideation.

The second use-case we consider is ideation. Here, the LLM is used to produce some ideas for the user to select from. As with summarization, the process of ideation inherently involves some kind of judgement: an idea might be a good

one for one set of goals, but not another. We focus on situations in which the LLM is used to generate a collection of ideas from which the user would select one of interest—or get inspired for something that is even better for their goals.

Examples of LLM uses for ideation include:

- Getting ideas for a birthday party celebration
- Getting ideas to propose for a participatory budget period in which citizens suggest how dollars should be spent
- Getting ideas for ways to make a company or organization more inclusive

While there is work on AI-assisted creativity, e.g., SOLVENT [2], there is complementary work in the machine learning community related to producing diverse alternatives for human inspection. For example, rather than output a single treatment option, a machine learning system may output many treatment alternatives and list their advantages and disadvantages. Rather than providing just one route, planners for driving directions will output multiple routes for the driver to choose from.

Again, the main difference between previous forms of AI-assisted ideation and now is the number of possible settings. One can imagine validating a system that produces treatment alternatives or driving routes in advance. But LLMs are being asked to generate ideas for a very large number of settings, we cannot expect that the LLM will be validated to produce reasonable ideas in all of them. Instead, again, we must provide methods for the user to perform validation of those ideas at test time.

4 RECOGNIZING WHAT'S MISSING

While they may seem quite different, both summarization and ideation tasks have several similarities from the perspective of validation. Unlike a question-answering application, in both these cases, the user has some larger partially observable context that shapes the concrete task. For example, one user may want to use a summary of a patient's history in order to identify any chronic conditions, while another user may be wanting to use a summary of that same patient's history to identify any concerns for adverse effects to new treatments. The type of public works ideas that someone finds interesting and valuable may differ depending on whether that person is a cyclist, a parent of school-age children, or a long-distance commuter.

Also, in both cases, the system's output is lossy; indeed, that is the whole point. The goal of AI-assisted summarization is to distill key themes or information from a larger set. The goal of AI-assisted ideation is to create a manageable list of reasonable ideas, not somehow cover the space of all possible ideas. Together, the facts that not all information is being provided, and that the goal of the user is not fully specified—they may not fully understand their own goal fully yet, and their understanding may evolve over time as they refine their mental models of their goal, the system, the data, etc. [5]—which creates room for the system to make choices that do not serve the user well. In general, irrelevant information or poorly aligned ideas that are surfaced by the LLM might be an annoyance but are easily disregarded. However, what is not surfaced by the LLM can cause much larger issues. If the user uses a summary to quickly check for concerns about drug interaction, and the summary does not include all the relevant information, then that may put a patient at risk. Automated meeting minutes or course evaluation summaries may leave out important minority viewpoints, and then those viewpoints will be lost to everyone who only looks at the summary.

We need ways of identifying the missing at *task-time*. In the following, we lay out some more specific ideas of how to go about this for the specific contexts of summarization and ideation tasks, and then pose a broader question of how one can know what voices are being included and excluded. And regardless of the specific approaches instantiated in a given scenario, the interfaces and workflows must minimize the impact of AI choices that are misaligned with the

user's needs [7] so as to minimize the inconvenience or even harm that could come to users contextually evaluating new AI tools instead of continuing to use existing systems.

4.1 Missing Information in Summarization

We begin with the case of summarization. In the summarization context, we do have a precise notion of the complete information: it is all of the documents or sources that the user has provided to the LLM to summarize. Thus, defining what has been left out is relatively clear: it is all the information that is in the complete set of documents that is not included in the summary. In the case of extractive summaries, where the summary is literally made of pieces of the original documents, this is very straightforward; for other types of summaries, this is more challenging but still something we can attempt.

The question is, of all the things that we know have been left out by the summary, what may have been left out inappropriately? Only the human during task time can fully answer that question, given their context and goals, and yet the full information is too large for someone to go through and check.

One approach is to summarize what has been left out, with the goal of helping the user efficiently identify information that they might have wanted but the original summary did not include. Given that the system could be confidently wrong in what it chooses to include in the original summary as well as the summary of what was left out, the designer must consider how to help the user notice and recover from when the system is confidently wrong [7].

Another approach to handling the information not used in the summary could be to apply ways to at least organize and render it so users are more likely to notice and discern the latent invariants and dimensions of variation present within the left-out data. The Variation Theory [9] of human concept learning suggests that this can help a human develop robust accurate mental models of the object of learning, i.e., what has been left out of the system's summary.

Alternatively, rather than as unclustered items organized and rendered by latent dimensions of variation, one could cluster the data, so that the user can review left out data cluster by cluster. But note that both the dimensions of variation approach and the clustering approach may anchor on clusters or latent dimensions of variation that privilege aspects of the data that are not the most relevant for the user performing the task in their context. AI recommendations can help prioritize the information that the user is more likely to determine as important missing information, and down-weight what is more likely to be irrelevant, which may help the user as long as the AI is not confidently wrong.

Additionally, we can allow the user to slice the missing by various computational criteria: the most common missing information (e.g., the largest clusters), the missing information least correlated with information in the summary (based on various information criteria), and the most rare missing information (the end of the long tail). We can provide views based on the type of language or other features as well. If we have a sense of common tasks that the summaries are often used for, we can use those tasks as proxies to elevate missing information most relevant to those tasks, in hopes that they might also be the missing information that the user is most keen to check.

4.2 Missing Information in Ideation

Both summarization and ideation can output lists: lists of the most relevant information and lists of the most relevant ideas for some imperfectly specified goal, respectively. However, the key difference is that in the summarization case, we have a clear sense of what is being left out, while in ideation, it is less clear how to imagine what ideas are being included and what ideas are being excluded.

Despite this challenge, we still believe there are opportunities here to highlight to the user what kinds of ideas may be included or excluded by a particular LLM. In particular, we can still imagine grouping ideas and trying to

categorize them by types. While each ideation task will be unique—and thus, require validation at task-time—there may be invariants and emergent dimensions of variation that, if explicitly called out, could help the user (1) recognize additional missing points along the existing dimensions of variation as well as (2) imagine dimensions of variation that do not exist yet by trying to come up with alternatives for what has been invariant so far in the generated ideas.

From a more technical perspective, we can recall also that the outputs of the LLM—all of the ideas—can be described by embeddings. One form of missingness might be to consider embeddings that lie in the span of the generated ideas but were not themselves included by the LLM in the list. It would be interesting to explore ways in which the embeddings themselves can provide ways to identify what is missing. Given that these embeddings which may or may not reflect what the user cares about given their context, allowing for these embeddings to be user-steerable at task time is likely a key component of developing AI systems that are indeed found useful during contextual evaluation.

Alternatively, system outputs could be clustered by notions of risk, cost, fun, or value to certain constituencies to reveal what types of ideas are more or less supported as common by the LLM. By categorizing the ideas that are produced, and by suggesting some types of ideas that are not produced, the system may be able to provide an anchor for the user to identify valuable ideas that the LLM may have left out. The utility, again would be a function of how well these notions are captured (or can be captured, with user feedback during task time) by the LLM and made to reflect the user's notions of them.

4.3 Missing Voices

So far, we have focused on what is missing from a summary or set of ideas mostly with an eye toward content, i.e., what categories of content are not included in a summary and what categories of content are left out of a generated set of ideas. But an important category of missing is that of missing voices. This is especially important in the context of leaving out information or opportunities that are relevant to marginalized communities, and we also increasingly understand that different settings will have different notions of how voices may group themselves.

For certain common types of categories, such as culture or political leaning, one may be able to use other text to at least classify the ideas and information. In doing so, one could highlight that perhaps the summary includes voices from a certain group and not others, or that the ideas all share certain similarities. To some extent, categorizing the style of the writing may serve as a proxy for certain types of groups.

That said, an approach like the above will be imperfect. Even when subgroups of interest are reasonably clearly defined, such as by gender or race, it may not be possible to accurately make those determinations based on just the text provided. Not everyone from one community writes in a particular way, nor can we always accurately identify whether a certain fact will be relevant to a certain community or another. Moreover, the relevant communities may vary significantly between settings. For example, in a classroom context, if a tool is summarizing student inputs in real-time, we may want to know whether that tool is privileging those within the major over those from other majors, perhaps intersected with some other characteristic.

5 CONCLUSION

The need for task-time human evaluation (and possible corrective feedback or tuning in the moment) has always existed with AI systems, but with prior AI systems, significant evaluation could be done in advance. The advent of large-surface AIs with applicability to user-context-dominated tasks has created a need for interactive, *task-time* evaluation of AI outputs. In this work, we focused on several specific applications: using LLMs for summarization and for ideation. In

both of these cases, the tasks are not completely specified (e.g., what exactly is the summary or set of ideas for?) nor are the user's particular context, preferences, values etc. fully observable (nor will they ever be).

For very specific use cases, such as using LLMs to produce summaries of clinical notes for a particular hospital department, we can imagine that with sufficient testing and design iteration, one could become confident that the outputs of the LLM summaries can be trusted. However, for the very many situations in which LLMs are being used, will be used that do not have such a clear, repetitive nature—even different public comments may have very different types of text—it is highly unlikely that we will be able to certify an LLM as being a "good" summarizer or idea generator in advance.

This observation motivated our call for AI and HCI researchers to develop best practices for a (responsible) contextual evaluation that can become a new gold standard for evaluating these foundation models in both fields: one that presumes that the system will be imperfect, and provides the user the tools to vet the quality of the AI system's outputs at task-time, that is, in the context of their specific task and leverage that AI—with a better understanding of what it is underrepresenting and what it is missing—to still get farther towards what they want as a final outcome than they could have on their own. In the context of summarization and ideation, we expanded on how this user assistance would involve helping the user efficiently and effectively understand what information and ideas have been included and what has been excluded. Other uses of large surface models may have different qualities, but will share this quality of needing tools to help the user evaluate the output in the context of their specific task.

REFERENCES

[1] Jeffrey P. Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C. Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samual White, and Tom Yeh. 2010. VizWiz: nearly real-time answers to visual questions. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology* (New York, New York, USA) *(UIST '10)*. ACM, New York, NY, USA, 333–342. https://doi.org/10.1145/1866029.1866080

[2] Joel Chan, Joseph Chee Chang, Tom Hope, Dafna Shahaf, and Aniket Kittur. 2018. SOLVENT: A Mixed Initiative System for Finding Analogies between Research Papers. *Proc. ACM Hum.-Comput. Interact.* 2, CSCW, Article 31 (nov 2018), 21 pages. https://doi.org/10.1145/3274300

[3] Xinyue Chen, Shuo Li, Shipeng Liu, Robin Fowler, and Xu Wang. 2023. MeetScript: Designing Transcript-Based Interactions to Support Active Participation in Group Video Meetings. *Proc. ACM Hum.-Comput. Interact.* 7, CSCW2, Article 347 (oct 2023), 32 pages. https://doi.org/10.1145/3610196

[4] Simret Araya Gebreegziabher, Zheng Zhang, Xiaohang Tang, Yihao Meng, Elena L. Glassman, and Toby Jia-Jun Li. 2023. PaTAT: Human-AI Collaborative Qualitative Coding with Explainable Interactive Rule Synthesis. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (<conf-loc>, <city>Hamburg</city>, <country>Germany</country>, </conf-loc>) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 362, 19 pages. https://doi.org/10.1145/3544548.3581352

[5] Elena L Glassman. 2023. Designing Interfaces for Human-Computer Communication: An On-Going Collection of Considerations. *arXiv preprint arXiv:2309.02257* (2023).

[6] Elena L. Glassman, Juho Kim, Andrés Monroy-Hernández, and Meredith Ringel Morris. 2015. Mudslide: A Spatially Anchored Census of Student Confusion for Online Lecture Videos. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 1555–1564. https://doi.org/10.1145/2702123.2702304

[7] Elena L Glassman and Jonathan K Kummerfeld. 2023. AI-Resilient Interfaces: Improving AI Safety and Utility by Making AI's Choices Easier to Notice, Judge, and Recover From. *in submission to alt.CHI* (2023).

[8] Mahmood Jasim, Enamul Hoque, Ali Sarvghad, and Narges Mahyar. 2021. CommunityPulse: Facilitating Community Input Analysis by Surfacing Hidden Insights, Reflections, and Priorities. In *Proceedings of the 2021 ACM Designing Interactive Systems Conference* (Virtual Event, USA) *(DIS '21)*. Association for Computing Machinery, New York, NY, USA, 846–863. https://doi.org/10.1145/3461778.3462132

[9] Ference Marton. 2014. *Necessary conditions of learning*. Routledge.

[10] Luke Murray, Divya Gopinath, Monica Agrawal, Steven Horng, David Sontag, and David R Karger. 2021. MedKnowts: Unified Documentation and Information Retrieval for Electronic Health Records. In *The 34th Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '21)*. Association for Computing Machinery, New York, NY, USA, 1169–1183. https://doi.org/10.1145/3472749.3474814

[11] Tim Rietz and Alexander Maedche. 2021. Cody: An AI-Based System to Semi-Automate Coding for Qualitative Research. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (<conf-loc>, <city>Yokohama</city>, <country>Japan</country>, </conf-loc>) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 394, 14 pages. https://doi.org/10.1145/3411764.3445591

[12] Andrew Ross, Nina Chen, Elisa Zhao Hang, Elena L Glassman, and Finale Doshi-Velez. 2021. Evaluating the interpretability of generative models by interactive reconstruction. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–15.