Visual Interaction with Deep Learning Models through Collaborative Semantic Inference

Sebastian Gehrmann*, Hendrik Strobelt*, Robert Krüger, Hanspeter Pfister, Alexander M. Rush

Abstract—Automation of tasks can have critical consequences when humans lose agency over decision processes. Deep learning models are particularly susceptible since current black-box approaches lack explainable reasoning. We argue that both the visual interface and model structure of deep learning systems need to take into account interaction design. We propose a framework of collaborative semantic inference (CSI) for the co-design of interactions and models to enable visual collaboration between humans and algorithms. The approach exposes the intermediate reasoning process of models which allows semantic interactions with the visual metaphors of a problem, which means that a user can both understand and control parts of the model reasoning process. We demonstrate the feasibility of CSI with a co-designed case study of a document summarization system.

Index Terms—Human-Computer Collaboration, Deep Learning, Neural Networks, Interaction Design, Human-Centered Design.

1 Introduction

Deep learning has become a universal tool that is increasingly applied in automated approaches to commonplace problems. Often, improvements in performance and efficiency from deep models come at the cost of increased model complexity, which leads to difficulties in interpretation, analysis, and visualization. By relying on the outputs of these complex black-box models, users give up their agency and control over the automated process. Moreover, the users are forced to trust and rely on models that have been shown to be biased, and that can make inexplicable mistakes.

To combine the advantages of models and humans, researchers across disciplines have advocated for models as collaborative team members. Grosz argues that the development of "intelligent, problem-solving partners is an important goal in the science of AI" [29]. Horvitz' principles for mixed-initiative user interfaces [36] and the more recent guidelines for human-AI interaction [3] call for "mechanisms for efficient agent-user collaboration to refine results." Similarly, Heer points out that work on automation ignores the design space in which computational assistance augments and enriches people's intellectual work [31].

We propose a framework that can be applied to enable users to control predictive processes called *collaborative semantic inference* (CSI). CSI describes a dialogue, alternating between model predictions presented in a visual form and user feedback on internal model reasoning. This process requires exposing the model's internal process in a way that mirrors the *users mental model* of the problem and then empowering the user to influence this process. This approach is centered around the core design principles for visual analytics, which integrates visualization and analytics in a human-centered interface [42]. Endert et al. [23] define semantic interactions as those which "enable analysts to spatially interact with [such] models directly within the visual metaphor using interactions that derive from their analytic process." CSI describes *how* to connect these semantic interactions to the model inference process. The development of CSI methods and interfaces further requires a tight collaboration between the visual analytics, interaction design, and machine learning experts, which is a challenging, but promising, direction of research [24, 33, 67, 71].

- *(*) indicates equal contribution*
- *Contact: gehrmann@seas.harvard.edu, hendrik.strobelt@ibm.com*
- *S. Gehrmann and A. Rush are with the Harvard NLP group.*
- *H. Strobelt is with IBM Research Cambridge*
- *S.Gehrmann and H. Strobelt are with the MIT-IBM Watson AI Lab.*
- *R. Krüger and H. Pfister are with the Harvard Visual Computing group.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

Most deep neural networks do not expose their internal reasoning process. The CSI framework requires the development of model extensions that expose intermediate reasoning that can be associated with user-understandable choices. In this work, we present a proposal that incorporates discrete latent variables [45] into the model design. These variables act as "hooks" that can control the reasoning process and output of a model. The hooks enable what-if analyses by answering what internal choices would have led to a specific output. Crucially for CSI, the hooks also allow a user to infer the model's reasoning process by seeing how a given output was selected. This visual analysis in the backward direction, from prediction to input, is typically not possible without model modifications.

To contextualize the multi-disciplinary co-design process and assist in developing collaborative tools, we provide an overview of the visualization and interaction design space for neural network models. We describe the actions required to move towards the goal of retaining human agency through visual CSI interfaces. We further connect our categorization to the previously described user roles of architect, trainer, and end-user [74] and a classification into interpretability methods that aim to understand and shape the model or its decisions.

As proof of concept, we apply our design process to the use case of a document summarization system. When this task is handled by an automated system, the results almost always require heavy post-editing by humans. Our use-case presents a first attempt at a collaborative, deep learning-based, interface for this problem. This use-case also demonstrates the expanded design space of interactive visual interfaces for collaborative models. We further identify challenges encountered in these collaborative models: How do we develop an interface that visualizes the prediction process on a granular level for any kind of model-specific input type (e.g., text, images, spectrograms), and how can a visual interface provide an integration of human interventions as part of the prediction process.

The paper contains the following contributions. In Section 2, we define the **concept of collaborative semantic inference** and its place in the design space of integrating deep neural models with visually interactive interfaces. We then describe **how CSI can be incorporated** into models using latent variables in Section 3. We **showcase a visually interactive use-case** for applying CSI to a text summarization task in Section 4. Based on our experience gained from this use case, we describe learned lessons towards building a systematic co-design process (Section 5). We discuss the implications of CSI and its advantages and disadvantages in Section 6.

2 Design Space for Integrating a Machine-Learned Model and Interactive Visualization

While previous work has categorized and described the vast design space for visual interpretation of machine learning models [24, 33, 60,

61], the problem of co-designing models, visualizations, and interactions remains challenging. In an analysis, Crouser and Chang find that there exists no common language to describe human-computer collaboration interfaces in visual analytics and propose to reason about interfaces based on the possible interactions humans can have with a model [19].

Expanding this idea, we contextualize different co-design approaches by categorizing the design space based on three criteria: (1) the level of integration of a machine learning model and a visually interactive interface, (2) the user type, and (3) applications that aim to understand and shape the model or model-decisions. Table 1 shows a categorization of the related work using criteria (1) and (3).

We identify three broad integration approaches between models and visual interfaces (Figure 1): *passive observations*, *interactive observations*, and *interactive collaborations*. As we will discuss in detail in this section, each category comprises a class of techniques that address different challenges of an analysis pipeline.

To decide on the visualizations and interactions for a specific problem in this design space we need to additionally consider the user type. We follow Strobelt et al. [74] who describe the roles of model *architects*, *trainers*, and *end-users*. Architects are defined as users who are developing new machine learning methodologies or who adapt existing deep architectures for new domains. Trainers are those users who apply known architectures to tasks for which they are domain experts. End-users are those users who use trained models for various tasks and who may not know how a model functions. As domain experts, the end-users' main goal is to achieve and explain the results of a model.

In addition to the level of integration and user type, we divide the design space between visual interfaces to *understand and shape the model or the decisions* of the model. Model-understanding/shaping describes systems with the goal of understanding or shaping the model through its features and parameters. The methods can help to gain insights into or modify the parameters that a model is in the process of learning or has learned already. Decision-understanding/shaping systems have the goal of understanding or shaping the individual decisions of the model on specific instances. Decision-based applications aim to understand how the model arrives at a given output but does not modify the parameter of the model. Shaping the model and decisions requires a tight coupling between the model and the interface, which in our classification is only possible with interactive collaboration interfaces, whereas the other categories focus on understanding.

2.1 Passive Observation

The first stage in our design space is passive observation, in which visualizations present a user with static information about the model or its output. The information can target any user-type and range from global information about model training to heatmaps overlaid over specific inputs. Passive observation interfaces only require a loose coupling between the interface and the model (Figure 1(a)).

Model-Understanding Architects and trainers are often concerned about how well a model is training, i.e., the model *performance*. Tools can assist trainers by tracking and comparing different models and metrics, e.g., Tensorboard [84]. Moreover, it is crucial for trainers to understand whether a model learns a good representation of the data as a secondary effect of the training, and to detect potential biases or origins of errors in a model [9]. To address this issue, many model-understanding techniques aim to visualize or analyze learned global features of a model [8, 12, 63, 90].

Decision-Understanding Decision-understanding passive observation tools assist end-users in developing a mental model of the machine learning model behavior for particular examples. The most commonly applied decision-understanding techniques present overlays over images [22, 66] or texts [20, 54, 55]. These overlays often represent the activation levels of a neural network for a specific input. For example, in image captioning, this method can show a heatmap that indicates which part of an image was relevant to generate a specific word [85]. A qualitative assessment of these methods may focus on whether highlights match human intuition before and after changes to the input or model [1, 46]. These methods also commonly assist in a pedagogical context [33, 69].

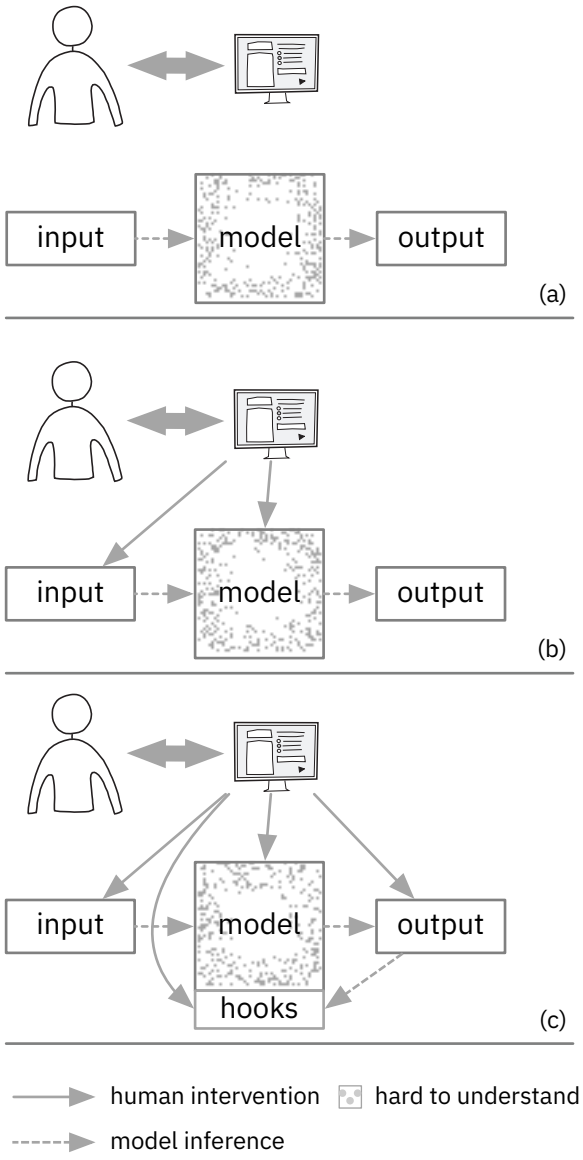

Fig. 1. We define the three levels of integration between models and visual interfaces: (a) passive observation, (b) interactive observation, and (c) interactive collaboration. Each successive stage extends the design space by adding new potential interactions. In the interactive collaboration stage, the model itself is extended with "hooks" to enable semantic interactions.

2.2 Interactive Observation

Interactive observation interfaces can receive feedback or information from the model itself (Figure 1(b)). This feedback enables the testing of multiple hypotheses about the model behavior within a single interface. We classify tools as interactive observation that allow changing inputs or extracting any model-internal representation as part of an interactive analysis. We call these *forward* interactions, analogous to how sending inputs into a model is called the forward pass. This approach can be used by trainers with domain knowledge to verify that a model has learned known structures in the data, or by end-users to gain novel insights into a problem. The development of interactive observation methods has been an active field of research, as summarized in recent review papers [33, 60, 61]. Interactive observation allows for a richer

	Model-Understanding/Shaping	Decision-Understanding/Shaping
Passive Observation [Understanding]	Activation Atlas [12], Classification Visualization [15], DeepEyes [65], Feature classifiers [8], Tensorboard [84], Weight Visualization [76]	Deconvolution [90], LIME [66], Neuron Analysis [20], Rationals [54], Saliency [55], Structured Interpretation [22], Prediction Difference [92]
Interactive Observation [Understanding]	ActiVis [41], Deep Visualization [89], Embedding Projector [70], GANViz [79], LSTMVis [74], Prospector [48], RetainVis [51], RNNVis [62], ShapeShop [32]	Instance-Level Explanations [47], Manifold [91], NLIZE [59], RNNBow [14], Semantic Image Synthesis [16], Seq2Seq-Vis [73]
Interactive Collaboration [Shaping]	Human-in-the-Loop Training [35], Statistical Active Learning [17], Tensorflow Playground [69], Explanatory Debugging [50]	**Achievable via CSI**, for example in GANPaint [7]

Table 1. A classification of some of the related work, which shows the distinct lack of collaborative interfaces for decision-understanding and shaping. Our CSI framework aims to fill this void, with a particular focus on applications built for end-users. However, different user types span all of the previous work, some aiming towards architects, trainers, end-users, or a combination of them.

space of potential interactions than passive observation tools and thus require a closer coupling between visualizations, interface, and the model [24].

Model-Understanding In an extension of visualization of learned features, interactive observational tools enable end-users and trainers to test hypotheses about global patterns that a model may have learned. One example is Prospector [48], which can be used to investigate the importance of features and learned patterns. Alternatively, counterfactual explanations can be used to investigate changes in the outcome of a model for different inputs, thereby increasing trust and interpretability [78, 81].

Decision-Understanding Interactive decision-understanding tools visualize how minor changes to an input or the internal representation influences the model prediction. Interactively building this intuition is crucial to end-users since past research has shown that statically presenting only a few instances may lead to misrepresentation of the model limitations [44] or the data that the model uses [77]. Another desired outcome of interactive observational tools is the testing of hypotheses about local patterns that a model may have learned. For example, we developed Seq2seq-Vis [73], which allows users to change inputs and constrain the inference of a translation model in order to pinpoint what part of the model is responsible for an error. This interaction is a type of non-collaborative shaping where the interactions are limited to the forward direction with the intended goal of understanding the decisions of the model. Similar debugging-focused approaches [47, 91] only visualize the necessary parts of a model to find and explain errors, instead of giving in-depth insights into hidden model states.

2.3 Interactive Collaboration

We characterize collaboration in interactive interfaces for deep learning models through the ability of end-users or trainers with domain knowledge to present feedback to the model (Figure 1(c)) with the goal to shape the model or its decisions. We call these *backward* interactions. Since each interaction direction needs to call a different shaping process within the model, the interface and model in interactive collaboration tools require a tight coupling. Only co-designing the model, visualizations, and interactions can achieve this tight integration.

Model-Shaping On the model-level, the feedback is expressed as user-provided labels which can be used to change the model parameters in an active learning setting [17, 35, 39]. In the forward direction, model performance can be visualized, and new samples for the labeling process can be selected [35]. The model parameters can be updated through backward interactions [50, 69].

Decision-Shaping Interactive collaboration for decision-shaping requires an interface in which the end-user can guide the model-internal reasoning process to generate a different output than the model would have reached on its own. Since interactive collaborative interfaces also retain the ability for forward interactions, the intervention enables to an interplay between suggestions by the model and feedback by the user. Our proposed approach to developing CSI methods, which we describe in Section 3, presents one way to design such applications.

During forward interactions, the visualization shows what the model-internal reasoning process looks like for a specific input. During backward interactions, the end-user can modify the output and observe how the model-internal reasoning process would have looked like to arrive at that specific output. The incorporation of these feedback mechanisms into visual analytics tools requires three essential components. First, the model needs to expose an interpretable hook along its internal reasoning chain, which should be transparent in derivation and understandable for non-experts [58]. Second, this interpretable hook needs to correspond to the mental model of the end-user. Most importantly, a collaborative tool needs to enable efficient interactions with the visual metaphor of the hook through semantic interaction [23].

The interpretable hooks of a model can act as explanations for rules of behavior that models learn. These explanations have been shown to improve model personalization [11] and explainability [13, 49]. Conversely, failing to provide explanations can inhibit the development of mental models in end-users [56]. However, explanations should also not overwhelm an end-user, and many previous approaches thus choose to select less complex models [52] or aim to reduce the feature space of a trained model [18, 87].

CSI systems have models and end-users collaborate on the same output. This contrasts with previous work that often treats the model as a complementary assistant, for example recommending citations for a writer [5]. Moreover, CSI argues for a design approach to collaborative interfaces where the user retains agency over the exposed parts of the model's reasoning process. Even in related approaches where the model and user both generate content, the users either do not have control over the model suggestions [30] or the model is replaced by uncontrollable crowdworkers [10]. While previous work on interactive phrase-based machine translation showed promising results towards the goal of collaborative interfaces, the same techniques are not possible with deep learning-based approaches [28]. This lack of previous work (see Table 1) can in part be attributed to the progress of deep learning methods, which have only recently reached the performance levels necessary for CSI-style interfaces.

3 Rearchitecting models to enable collaborative semantic inference

Interactive collaboration requires interpretable model hooks that enable semantic collaboration. From the machine learning side, these hooks can be implemented as discrete latent variables. During the prediction, or *inference*, process, the variables take on explicit and understandable values. To illustrate this process, consider a hook resembling a lever that directs a train towards a left or right track, as shown in Figure 2. A model is predicting where a train will end up. Without the hook, the model can predict the end position accurately, but it is not clear how it

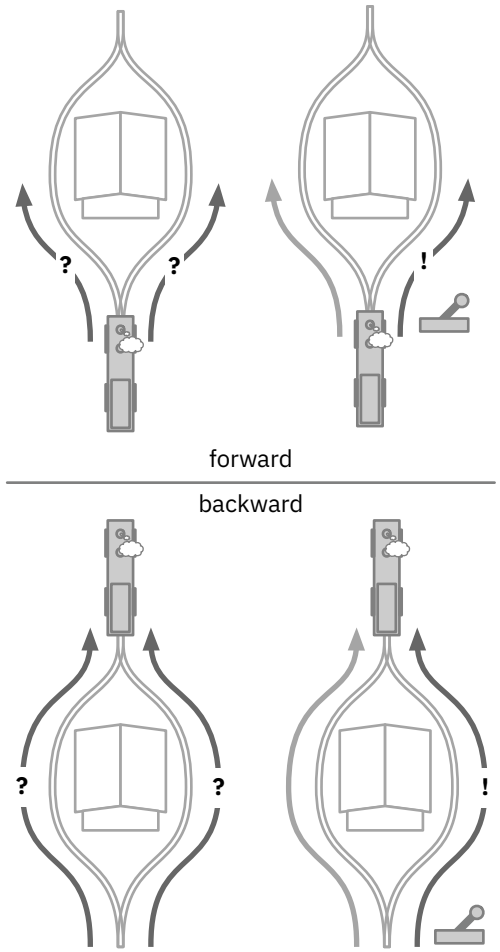

Fig. 2. The advantage of latent variables is the transparent reasoning process. CSI requires model reasoning, illustrated by a lever that dictates which turn a train is taking. In the forward inference direction, we know in both cases where the train will end up, but only the lever allows us to know what path it will take. Similarly, in the backward inference direction, we know where the train originated, but only the lever shows the track it took to get there.

will get there (top-left). Similarly, once the train reaches the top of the tracks, the model cannot explain how it got there (bottom-left). With the hook, however, the prediction explicitly exposes the lever decision. That means that it is predictable to the user whether the train will take the left or right branch (top-right). Once the train is at the top of the intersection, the user can look at the position of the lever to find the path the train has taken (bottom-right).

To formally describe hooks, we consider the problem of generating a sequence of words $y_1, \ldots, y_T$ that is conditioned on an input x using a deep model. In the exemplary problem of document summarization, x represents a long document and y the summary. A forward-only deep sequence model defines a conditional distribution to predict one word at a time, $p(y_{t+1} \mid y_{1:t}, x)$ while considering all previously generated words. In the collaborative approach, the deep sequence model is extended to expose intermediate terms as latent variables z. In the summarization example, this could be the decision what words in an input are considered important enough to be included in a summary. The architect defines $p(y_{t+1} \mid y_{1:t}, x) = \sum_z p(y_{t+1} \mid y_{1:t}, x, z) \times p(z \mid y_{1:t}, x)$ for the latent variable z. That means that the model considers all possible values of z to make a prediction. In the train-lever example, that means we can run the inference process to compute the best lever position by looking at which of the two positions would lead to a better final position as judged by the model.

This decision splits the black-box into multiple parts: a *prediction network*, $p(y_{t+1} \mid y_{1:t}, x, z)$, that predicts the next word, and a *hook network*, $p(z \mid y_{1:t}, x)$, that predicts the value of the latent variables z. Because this model is probabilistic, we can also perform posterior or *backward inference*. This gives $p(z \mid y_{1:T}, x)$, the distribution of z after taking into account the entire output, i.e., the text summary.

Another example of a collaborative interface could be for semantic image synthesis [16, 64, 80]. In this application, a user-defined input x describes high-level features, for example, the location of grass and sky, and a neural network generates the corresponding image. In this case, y is an image and not a sequence of words. The current approaches do not allow iterative refinement through interaction with an image and are limited to changing the input and generating a completely new output. A collaborative approach to the same problem is GANPaint [7], which uses a hook network to associate parts of the latent space of an image-generating model with the semantic features and exposes a modification interface to the user.

In a real-world example of a hook-network, Google Translate recently introduced a semi-collaborative approach to prevent gender-discrimination in translation systems. By treating the gender as a hook, they can present both possible options whenever the gender in a language is ambiguous, for example in Turkish. This approach allows a user to pick the translation they want.

The model hooks represent ways in which users can constrain and direct interpretable predictions in otherwise end-to-end black boxes. They are extensions of otherwise well-performing models to expose the latent variable, co-designed by experts in interaction design, visualization, and machine learning. They have to decide on the model hooks, desired interactions, and the associated visual encodings. While some guidelines have been developed for interaction design for machine learning [2, 25, 75, 86], they focus on designing machine learning methods where the model performs complementary tasks to the user, or where the user interacts with black-box models. In contrast, CSI enables the study of interaction design for models that approach the same task as the user. One important question that requires further study is how many hooks are actually useful to a user. As more latent variables are designed and incorporated into a model, the training process becomes increasingly more challenging and the model performance might degrade. Moreover, the increase in potential user interactions with additional hooks might overwhelm users. As a consequence, we focus on a model with a single hook throughout our use case and show how a single hook can already enable many powerful interactions.

We now describe our co-design approach for a CSI system for document summarization and present the lessons we learned from our CSI co-design process in Section 5.

4 Use Case: A Collaborative Summarization Model

We demonstrate an application of the CSI framework to the use case of text summarization. Text summarization systems aim to generate summaries in natural language that concisely represent information in a longer text. This problem has been targeted by the deep learning community using an approach known as sequence-to-sequence models with attention [85]. These models have three main architectural components: (1) an encoder that reads the input and represents each input word as a vector, (2) a decoder that takes into consideration all previously generated words and aims to predict the next one, and (3) an attention mechanism that represents an alignment between the current decoding step and the inputs. In summarization, the attention can be loosely interpreted as the current focus of the generation and can be visualized for each generated word [73].

Imagine an end-user called Anna, who wants to use an interface powered by a summarization model. Current deep learning models act in a forward-only manner, as described above; therefore the design space is limited to an interactive observation interface. This interface allows Anna to paste an input text and have the model infer an output summary. If Anna does not like the output summary, she can edit the suggestion to her liking, but it is not possible to reuse the model to check her changes. Moreover, if the model produced a bad or wrong output, Anna would have to write the entire summary from scratch.

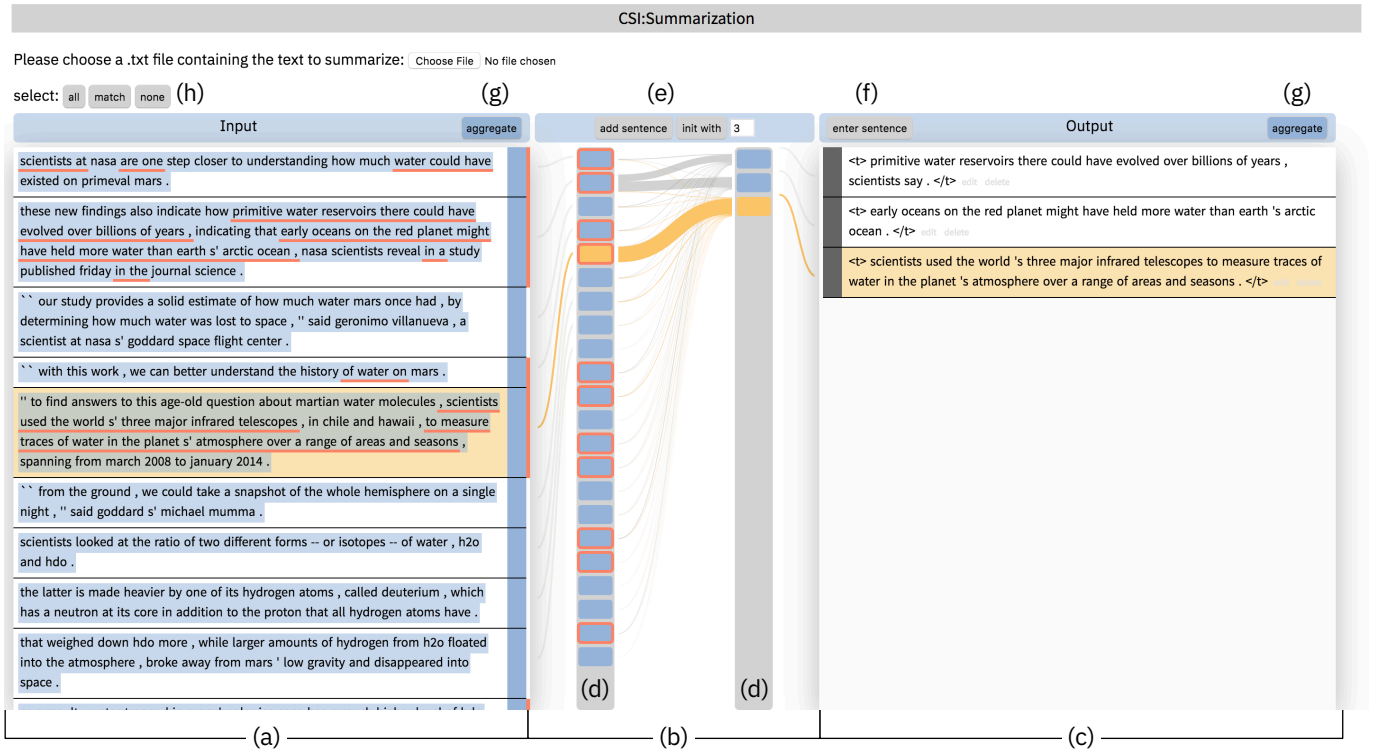

Fig. 3. Overview of the CSI:Summarization visual interface. (a) shows the input and overlays the currently selected content selection (blue) and the current content of the summary (red). (b) shows a connection between the input and the current summary through the attention, which shows explicitly where words in the summary came from. (c) shows the current summary, (d) shows proxy elements for both input and output groups. This enables an overview of a document, even when the text does not fit on one page. (e) allows the user to request suggestions from the model, (f) enters the edit mode and adds a new sentence to the summary. (g) toggles whether the text should be aggregated into sentences. (h) provides quick selections for the content selection (blue) by being able to match the red highlights, or (de-)select everything.

Applying the CSI framework by extending a well-performing summarization model with the previously defined hooks can address these issues. By tying the user's interactions to understandable reactions within the model, we can achieve three previously not possible interactions. The collaborative interface (1) **guides the model** towards important content, (2) **enables the dialogue** between human-generated and machine-generated output, and (3) allows a user to **review the decisions** the model would have made to generate a specific output, i.e., what parts of an input text the model chose to summarize.

These changes require designing a semantic model hook that can describe the content that a model considers for a summary. We base the hook on a similar model we introduced in our previous work on document summarization [27]. Commonly applied neural summarization models have the ability to either copy a word from the source document or generate a word. In contrast to users who write a summary, this approach does not have a planning phase where the model decides what content should be able to be copied from a more global perspective. We, therefore, introduce a hook for each word in a document that expresses whether the model should be able to copy the word. By exposing the hook within the interface, we can describe the semantic interaction as the user-decision what content of a document is relevant for its summary. A detailed description of the forward and backward models can be found in Appendix A.

We now describe how Anna generates a summary by using our CSI interface to collaborate semantically with the model.

4.1 Collaborative Summarization: Anna's Story

Anna intends to collaboratively write a summary of an article describing how scientists found water on Mars[1]. She begins by reading the article to assess what information is relevant and should be part of the summary. Then to begin the interaction, she selects the entire input text, shown by the blue highlights in Fig. 3a, letting the model know it is free to summarize any relevant part of the document.

She starts the collaborative writing process by requesting that the model suggest three initial sentences (Fig. 3c). This triggers a forward inference of the model and a visual update that presents the suggestions to Anna. At the same time, the system computes a backward inference to show which part of the input has been summarized. Visually, the input text may be longer than the browser window, so we introduce a proxy element for each sentence in input and output (Fig. 3d). The covered words in the input are presented with a red underline and the proxies of the sentences with at least one covered word have a red border (Fig. 3a,b). The interface also visualizes the model "attention," which shows which covered words were selected by the model at what step during the generation. The attention is visualized using grey ribbons that are aggregated across each sentence and connect the proxy elements. By hovering over one of the sentences or proxies, the interface highlights the relevant connections in yellow (Fig. 3b).

Through these visual interactions with the output summary, Anna observes that the second input sentence (*"scientists at nasa are one step closer to understanding how much water could have existed on primeval mars. these new findings also indicate how primitive water reservoirs there could have evolved over billions of years, indicating that early oceans on the red planet might have held more water than earth's arctic ocean, nasa scientists reveal [...]"*) splits into two different summary sentences (*"primitive water reservoirs there could have evolved over billions of years, scientists say."* and *"early oceans on the red planet might have held more water than earth's arctic ocean."*). It is common for summarization models to compress, merge, and split input sentences, but the user would not be aware of where the

[1]The article can be found at `https://www.cnn.com/2015/03/06/us/mars-ocean-water-study`

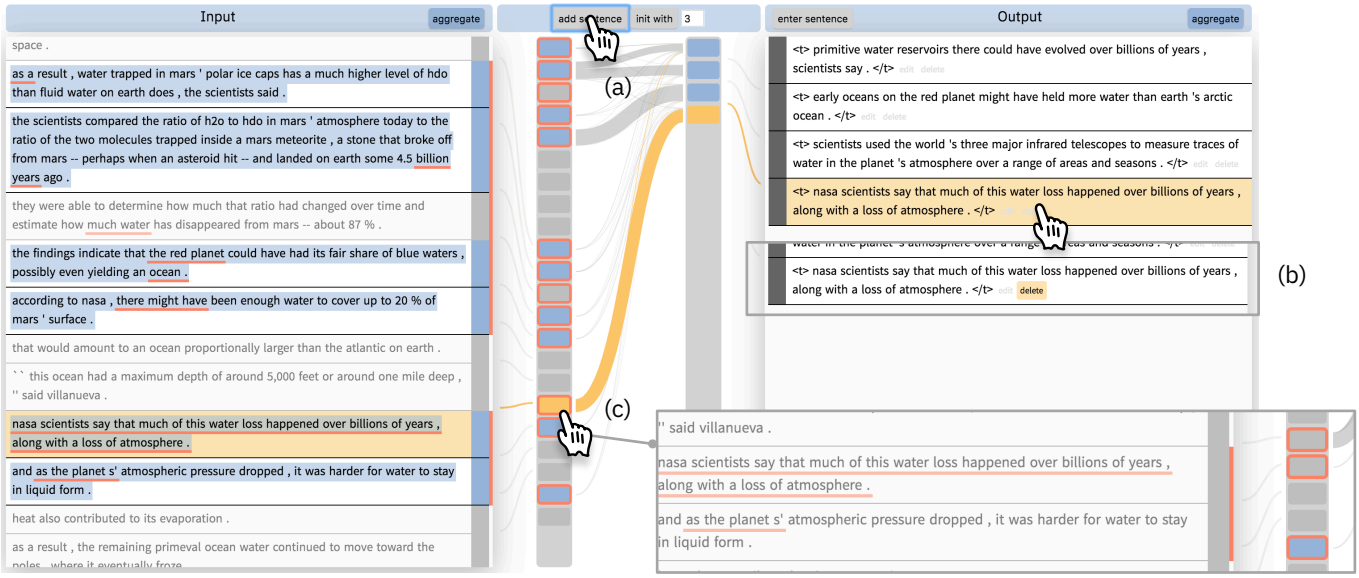

Fig. 4. After selecting the content shown on the left, Anna requests the model to generate a fourth sentence (a). She does not like the suggestion and deletes it (b). To influence the model to generate about other topics in the input, she deselects the sentences that caused the suggestion (c).

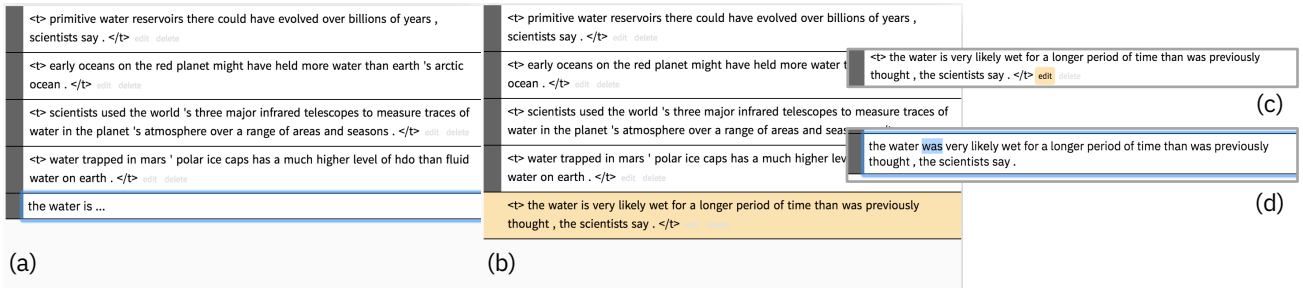

Fig. 5. Anna wants to generate a sentence about the water on Mars and starts typing *"The water is ..."* (a). This initiates the model to finish the sentence for her (b). To correct a minor mistake in the generated sentence, Anna activates the edit mode (c) and replaces the wrong word (d).

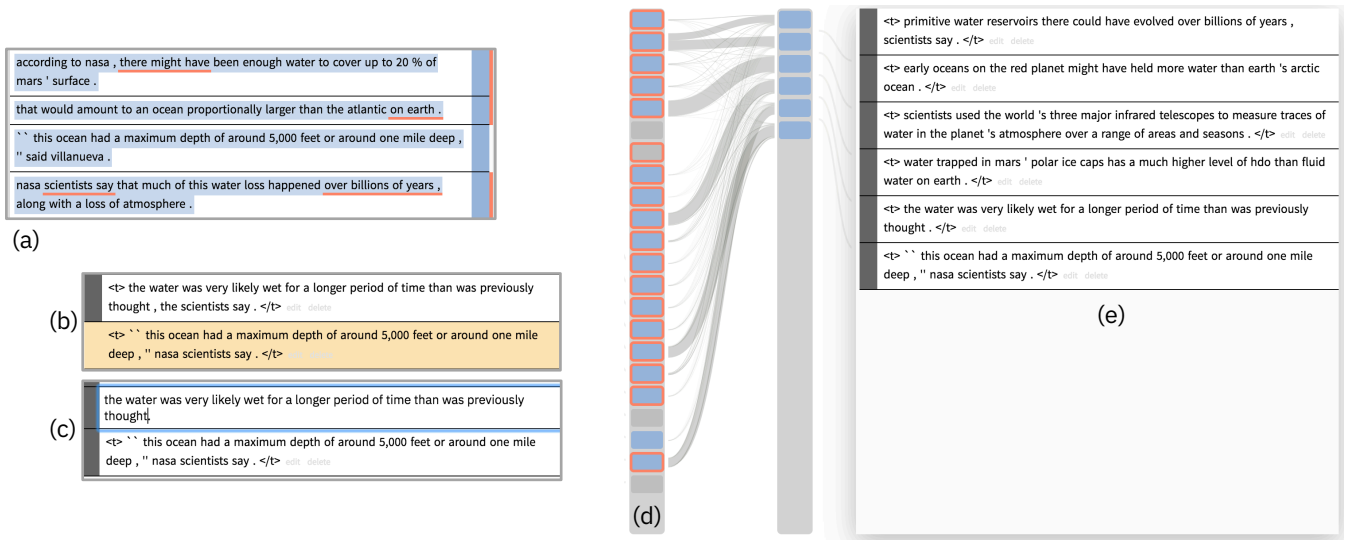

Fig. 6. Anna selects a sentence that is currently not covered by the summary, indicated by the lack of a red border on the right (a). She generates a new sentence (b) and corrects the repeated phrase "nasa scientists say" (c). With the finished draft of her summary, she can now evaluate the coverage of the input document (d) and the final summary (e).

inputs for each summary sentence originate. By exposing the internal model decisions in a CSI interface, a user can immediately discover how the input connects to the output.

From this suggested summary, Anna determines that the input focus of the system was correct, but that output text should elaborate on these sentences in more detail. She communicates this by first constraining the model to the currently focused region. She can match the content selection (blue highlights) with the result of the backward inference (red underlines) by clicking "match" in Fig. 3h. With these constraints, she triggers the generation of an additional output sentence (*"nasa scientists say that much of this water loss happened over billions of years, along with a loss of atmosphere."*) (Fig. 4a). This suggestion is the result of a forward inference with her additional constraint on the model hook.

Unfortunately, Anna is dissatisfied with the new sentence, so she removes it from the summary (Fig. 4b). To prevent the model from suggesting the same sentence again, she consults the backward inference and deselects the input sentence that had the highest influence on its generation by clicking on its proxy element (Fig. 4c). With this updated constraint, Anna generates another sentence (*"water trapped in mars' polar ice caps has a much higher level of hdo than fluid water on earth."*) that better captures her goals. (Fig. 5a).

Anna would next like to include more details to the summary, particularly about water found on Mars. The system allows her to intervene in the output text directly. She starts writing "the water is" and then adds an ellipse (...) that triggers sentence completion by the model (Fig. 5b,c). The resulting sentence (*"the water is very likely wet for a longer period of time than was previously thought, the scientists say."*) is acceptable to her, but she now spots an error with the verb (*"is"*) which should be in the past tense. She quickly corrects this output (Fig. 5d,e), which invokes a backward inference to the input document. By updating the red highlights in the input, the interface provides her with information about what content was selected was used to create this improved sentence. These interactions help her create a mental picture of the model behavior.

Anna would finally like the model to help her generate a sentence about a region of the input that is currently not included in the summary. She selects a previously unused sentence in the input (*"this ocean had a maximum depth of around 5,000 feet or around one mile deep, said villanueva."*) (Fig. 6a), requests another forward inference, and approves of the resulting suggestion (*"this ocean had a maximum depth of around 5,000 feet or around one mile deep, nasa scientists say."*). However, she dislikes the repetition of "scientist say" (Fig. 6b). After entering the edit mode on the output side, she removes one of the repeated phrases (Fig. 6c). Upon leaving the edit mode, the interface automatically triggers another backward inference that updates which parts of the inputs are covered. Anna uses this information to evaluate how much of the document is covered by her summary. By looking at the computed coverage (Fig. 6d), she can observe which sentences are covered and analyze how many of the proxies have a red border. Her six sentences (Fig. 6e) summarize the original text pretty well.

4.2 Visual and Interaction Design

We designed the text summarization prototype (*CSI: Summarization*) such that text occupies the majority of screen estate as the central carrier of information for the task. Two central panels (Fig. 3a,c) represent input text and output text. Each text box represents words that are aggregated into sentences. Text highlights in the input show information about the model hooks and relations between input and output. Neutral gray colors are used on the output side to clearly distinguish them from the blue colors that represent selections on the input (Fig. 3c).

The input and output text are connected by a bi-partite graph that indicates model attention (Fig. 3b), which expands on previous work on visualizing and normalizing attention [57, 72]. Due to the length of source documents, displaying the entire graph is not feasible or informative. Therefore, we use two design elements to observe the full graph in a de-cluttered view: aggregations and proxies. First, we allow the aggregation of words into meaningful word groups, e.g., sentences, that can be dissolved on demand (Fig. 7d) if this level of detail is required. Aggregating words implicitly requires the aggregation of the attention which simplifies the graph. Secondly, we represent sentences by a vertical arrangement of boxes that are space-filling in height and which act as proxies for the full sentences. In that way, all sentences are always visible by their proxy, even when they are outside the display area. These proxies mirror selections and highlights of their related text boxes.

CSI: Summarization offers a range of user interactions. As a general principle, buttons trigger forward (left to right) actions (Fig. 3e) because a forward inference can change the output summary itself. Unintended changes to the output could confuse the user. To let users explicitly request updates instead of automatically intervening is inspired by similar mixed-initiative writing assistants [5], where researchers found that this type of interaction is seen as least intrusive. All backward inferences are automatically triggered after exiting the edit mode by hitting enter. Since backward interactions do not change the content of the summary, the automatic invocation does not lead to accidentally overwriting important information. Users can define which sentences or words to consider for generating the next sentences by selecting them (blue color). Clicking on the bar on the right of an aggregation group selects or deselects the entire group. The same action is triggered by clicking on the proxy element of the corresponding sentence (Fig. 3d).

The interface additionally provides three selection templates (Fig. 3h) for convenience: select all sentences, select no sentence, or select only those sentences that match the selection from the backward step (match red and blue). For the forward pass, the selection can be used to either initialize a new summary with a user-selected number of sentences (`init with`) or to add a sentence to the existing summary (`add sentence`) (Fig. 3e). On the output side, sentences can be deleted or edited by clicking the `edit` and `delete` buttons at the end of each sentence.

4.3 Design Iterations

During the creation of the prototype, we explored multiple designs for model hooks, visualizations, interactions, and their integration. Overall, we found that CSI systems are more difficult to design because of the, sometimes competing, interactions between all these elements. We want to highlight one example for each of the elements.

On the model side, our initial backward inference had good accuracy but did not reveal useful information within the interface. Only after re-allocating efforts to train a different model with a much higher performance did the backward model match the human intuition. Since model hooks complicate the design of the machine learning models, it also complicates their training process. Issues, such as the one described above, are only uncovered with the visual interface.

On the visualization side, we explored multiple different designs for the input selections presented in the interface. The selections we currently show are (1) the selection that was used for the most recent forward step, (2) the selection that was returned from the most recent backward step, and (3) the sentences that the user is selecting for the next forward pass. In our first iteration, we aimed to show all of them in separate views (Fig. 7a), and additionally have a view that highlights their intersection (Fig. 7b,c). Internal tests revealed that only the combined view was useful to users to avoid having to switch forth and back. We, therefore, replaced the different views by the current more natural and coherent use of red and blue highlights within the same view. In conclusion, this design iteration provides a good example of the prevalent visualization challenge how to encode necessary information for the targeted user group. For CSI interfaces, the task is severe because of the major difference in abstraction between model internals and end-users intuitive understanding.

On the interaction side, we found that requesting the model to generate words without a constraint on the minimum or the maximum number of sentences often led to output that was unreasonable to users. The model architects on our team pointed out that the training data for summarization models rarely contains examples where the summary is longer than three sentences. Forcing a model to generate longer summaries than it was trained to generate led to the degradation in output quality. We also found that users had more control over the content of a summary if they iteratively built up a summary from a short initial suggestion instead of suggesting a lengthy summary and letting the

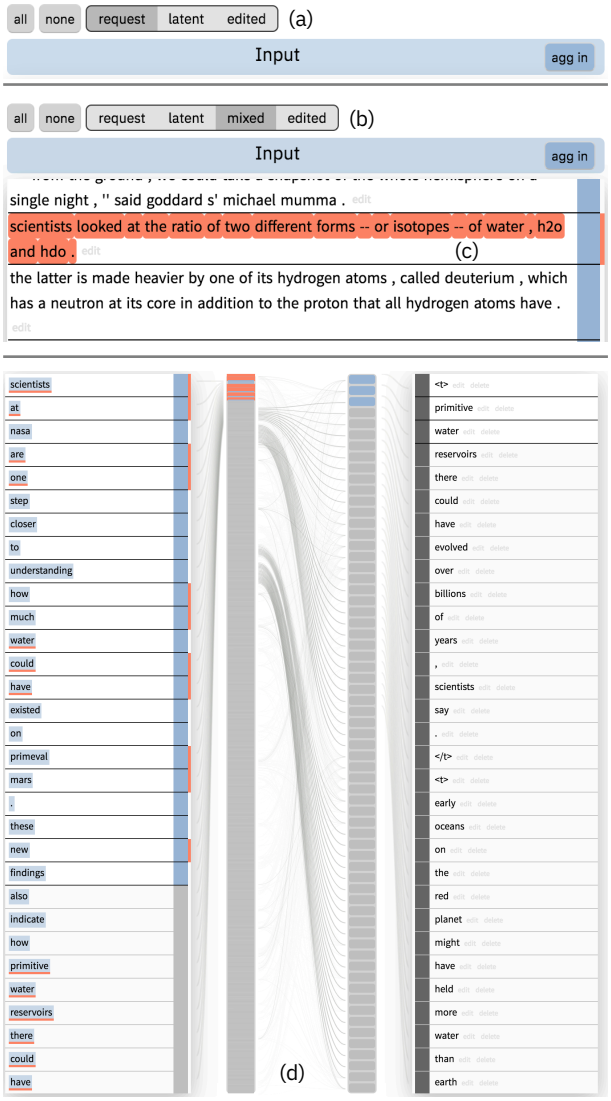

Fig. 7. Visual design iterations for *CSI:Summary*. (a) shows radio buttons for selecting a specific selection (content selection vs. backward model). In (b), we introduced a mixed mode that showed the user content selection in the final blue color and the result from the backward model with a red highlight (c). Underlines later replaced the dominant highlight. (d) shows an example of the complexity of the attention graph without any aggregation.

user change it afterward. Our current modes combine these findings by designing the interaction after discussions with our model architects and visualization experts. The first interaction initially generates a small maximum number of sentences. This both leads to better model output but also lets users explore the output space more effectively. Similarly, adding one new sentence at a time by incorporating previous sentences as prefix context and allowing users to select the content enables users to quickly generate and review new content.

5 Towards a Co-Design Process for CSI Systems

During the implementation of the *CSI:Summarization* prototype we developed an understanding of how an integrated design process for CSI systems could look like and also experienced its limitations. We discuss our insights as learned lessons.

Prioritize collaborative output. CSI systems enable joint production by model and end-user together. The resulting output should be the central element for developing visualization and interaction ideas.

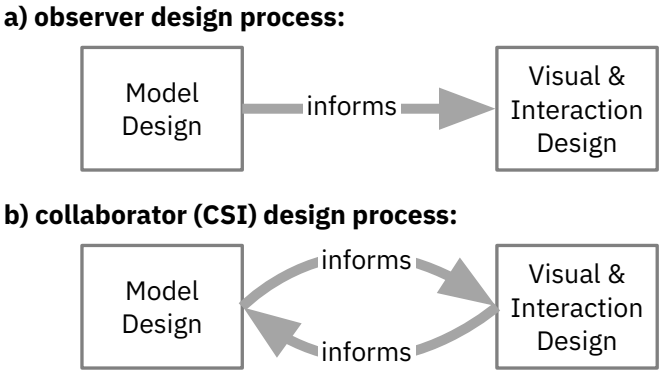

Fig. 8. CSI interfaces require a design process that spans machine learning and visual interaction design. The feedback loop is used to produce interpretable semantic representations that inform the reasoning procedure of the model while providing useful visual interaction.

It is essential to evaluate if a *CSI approach is beneficial* for the given task, e.g., a face recognition model used to unlock a cell phone does not benefit from the CSI approach as no shared output is produced. Since *CSI methods are decision-shaping*, they require human oversight and interventions and are thus not suited for processing massive data. Moreover, since CSI interfaces are targeted at end-users, the visualization can be domain-specific but should abstract model-internals in an intuitive way. Finally, *no side should dominate over the other* to allow model-human collaboration, which should be reflected in the visualization and interaction design – e.g., allowing equal easy access to triggers for human input and model suggestions.

Co-design requires continuous evaluation. In the development of the summarization system, there was a *constant negotiation between visualization requirements and model capabilities*. This process led to iteration on the question: "does this visual encoding help the end-user collaborating with the model?" and "which additional model behavior do we need to help to encode relevant information?". Fig. 8 illustrates how this continuous co-design play forms a bilateral relationship between model design and visualization design. CSI systems, like most visual analytics systems, can help to *reveal model problems* immediately. If, for example, a specific model hook is performing so poorly that it cannot facilitate the user's mental model, it will be immediately revealed. On the other hand, requirements for a *visualization might over-constrain a model* such, that it breaks. E.g., creating a system for poem writing which should suggest lines of text that rhyme, even for human entered text, might over-constrain the model. CSI systems are about to find the middle ground between what would be ideal for a user interface vs. what would be possible with the underlying ML model.

CSI may be a worthwhile investment. Since CSI is centered around a single abstraction that should reflect the mental model that an end-user has of a problem, we pose that *machine learning needs to learn about interactions*. There is currently a limited understanding of the space of easily trainable hooks and interaction strategies. Since machine learning techniques do not natively consider bidirectional interactions with end-users, the visualization, interaction design, and machine learning experts need to teach each other about desired interactions and the limits of deep models. Deep learning models with hooks thus lead to an increased development complexity for both machine learning and visualization experts. However, during the development of the summarization use case, we also experienced that *CSI has a learning curve*. While CSI systems are individualized to a problem and thus one-of-a-kind systems, most of the techniques are transferable and the team can apply the insights gained from one CSI project to the next one. Moreover, the long-term benefits of the increased control over models can justify the additional development complexity, especially considering that many applications in industry are used for many years. As shown in our summarization example, the CSI methods can even

lead to more structure and subsequent improvements in outcomes.

6 Conclusions and Future Work

In this paper, we introduced a framework for collaborative semantic inference, which describes a design process for collaborative interactions between deep learning models and end-users. Interfaces designed within this framework tightly couple the visual interface with model reasoning. We applied CSI to develop a collaborative system for document summarization that demonstrates that CSI systems can achieve powerful interactions within an interface powered by a neural model.

While previous studies have shown that explainability methods can mediate in an agency-efficiency trade-off [53, 88], none of them demonstrate a way for users to retain agency while gaining the efficiency benefits of models interaction. We believe this is due to the difficulty of engaging the user in the prediction process of a black-box model. We address this problem by designing semantic interactions as part of the model itself. While CSI does not solve problems with biased data and models or the lack of interpretability of models, it aims to expose important model decisions and facilitate collaboration between an end-user and the model to take these decisions. Further developments in interpretability research could be used in conjunction with CSI for a better overall model understanding.

The CSI framework significantly expands the interaction design space over conventional interaction strategies for many deep learning models. CSI-style approaches have the potential for application especially in scientific or safety-critical applications where explainable AI may become mandatory. Moreover, since CSI treats the model as a team member, another area of particular interest are creative applications, where models can assist users in creating stories [26], chord progressions [37], or even recipes [43]. Future work might also investigate how similar principles could support cases where more than a single end-user and a single model aim to collaborate.

Finally, it is crucial to develop ways to systematically evaluate collaborative interfaces and to investigate the implications of designing algorithmic interactions with humans [82,83]. While an interpretability-first approach could assist in highlighting fairness and bias issues in data or models [34, 38], it could also introduce unwanted biases by guiding the user towards what the model has learned [4]. It is thus insufficient to limit the evaluation of a system to measures of efficiency and accuracy. Future work needs to address these shortcomings by developing nuanced evaluation strategies that can detect undesired biases and their effect on end-users.

We provide a demo, the code, and the required models for CSI:Summarization at `www.c-s-i.ai`.

Acknowledgements

We would like to thank Michael Behrisch for constructive discussions. This work was funded in part by the NIH grant 5R01CA204585-02, an AWS Faculty Award, and a Google Faculty Research Award.

A Details on the Summarization Model Hooks

Summarization models use source words $x_1, \ldots, x_n$ as input with the goal to generate a summary $y_1, \ldots, y_m$ with $m \ll n$.The most commonly used neural approach to document summarization is the sequence-to-sequence model [6], which encodes a source document and then generates one word at a time. Sequence-to-sequence models incorporate an attention distribution $p(a_j|x, y_{1:j-1})$ for each decoding (generation) step j over all the source words in x, which is calculated as part of the neural network. The attention can also be interpreted as the current focus of the model. Since summaries often reuse words from the source document, current neural architectures incorporate a mechanism called copy-attention. This mechanism introduces a binary switch that enables the model to either generate a new word or copy a word from the source document during the generation [68]. Similar to the standard attention, the copy-attention computes a distribution over all source tokens for each decoding step. The copy-attention (shown in Fig. 3e), is directly interpretable since a high value means that the model is actually copying a specific word from the input.

While this approach yields a high performance on automatic metrics, the end-to-end approach with decisions per generated word is disconnected from human summarization approaches. There have been multiple studies that show that humans typically follow a two-step approach in which they first decide what content within a document is important, and then try to paraphrase it [40]. Following the goals of the CSI framework, we aimed to design a hook that reflects this two-step approach so that we can expose the planning stage and model decision what content is important to the end-user. Importantly, the end-user, in this case, may only have minimal or no knowledge of the underlying machine learning model, which means that every action by the end-user should be intuitive and directly connected to the hook.

To enforce the human-like process within the model, we developed a CSI add-on for the high-performing model. We introduce binary tags for each of the source words, $t_1, \ldots, t_n$. The model is allowed to copy the i-th word only if t_i is 1[2]. The *prediction network* that generates the summary $p(y_j|x, y_{1:j-1}, t)$ is thus prevented from copying the blanked out content. The additional *hook network*, which predicts the tag probabilities $p(t|x)$, decides what content is important, which leads to significant improvements in fully automatic summarization. Since the model explicitly reasons over content-importance through the hook network, we can achieve semantic interactions by letting users define a prior on $p(t|x)$. When user deselects a sentence from the input, we set the prior $p(t)$ for all its words to 0, which means that the hook network can no longer identify the words as important which means that it is prevented from copying deselected words.

The last step towards the fully integrated CSI:Summarization is the *backward inference*, i.e. the identification of what content a summary actually used, or $p(t|x, y)$. The result of this is shown with red highlights in the interface in Fig. 6. The backwards model is a separate model we specifically developed for the interface. It uses a contextualized representation of words in both input and summary that represents them as vectors of size d_{hid} [21]. Given the representation for a word x_i, the model computes an attention over y, such that each summary word y_k is assigned an attention weight a_k. We use these weights to derive a context for the word x_i which we denote c_i, by computing

$$c_i = \sum_{k=1}^{m} a_k \times y_k.$$

To arrive at a probability that x_i was used in the output, we compute

$$p(t_i|x, y) = \sigma(W_2 \tanh(W_1[x_i, c_i] + b_1) + b_2),$$

where $b_{1,2}$ are trainable bias terms and $W_1 \in \mathbb{R}^{d_{hid} \times 2d_{hid}}$ and $W_2 \in \mathbb{R}^{1 \times d_{hid}}$ trainable parameters. Since this model is independent of the forward model, it can analyze arbitrary summaries, even those that are user-written, as we show throughout our use case.

References

[1] J. Adebayo, J. Gilmer, I. Goodfellow, M. Hardt, and B. Kim. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, 2018.

[2] S. Amershi, M. Cakmak, W. B. Knox, and T. Kulesza. Power to the people: The role of humans in interactive machine learning. *AI Magazine*, 35(4):105–120, 2014.

[3] S. Amershi, D. Weld, M. Vorvoreanu, A. Fourney, B. Nushi, P. Collisson, J. Suh, S. Iqbal, P. N. Bennett, K. Inkpen, et al. Guidelines for human-ai interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, p. 3. ACM, 2019.

[4] K. Arnold, K. Chauncey, and K. Z. Gajos. Sentiment bias in predictive text recommendations results in biased writing. In *Proceedings of Graphics Interface 2018*, pp. 33 – 40. Canadian Human-Computer Communications Society / Societe canadienne du dialogue humain-machine, 2018.

[5] T. Babaian, B. J. Grosz, and S. M. Shieber. A writer's collaborative assistant. In *Proceedings of the 7th International Conference on Intelligent User Interfaces*, pp. 7–14. ACM, 2002.

[2] How to train the model is described in our previous work [27]

[6] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[7] D. Bau, J.-Y. Zhu, H. Strobelt, Z. Bolei, J. B. Tenenbaum, W. T. Freeman, and A. Torralba. Gan dissection: Visualizing and understanding generative adversarial networks. *arXiv preprint arXiv:1811.10597*, 2018.

[8] Y. Belinkov, N. Durrani, F. Dalvi, H. Sajjad, and J. Glass. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 861–872, 2017.

[9] Y. Belinkov and J. Glass. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72, 2019.

[10] M. S. Bernstein, G. Little, R. C. Miller, B. Hartmann, M. S. Ackerman, D. R. Karger, D. Crowell, and K. Panovich. Soylent: a word processor with a crowd inside. In *Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology*, pp. 313–322. ACM, 2010.

[11] S. Bostandjiev, J. O'Donovan, and T. Höllerer. TasteWeights: a visual interactive hybrid recommender system. In *Proceedings of the 6th ACM Conference on Recommender Systems*, pp. 35–42. ACM, 2012.

[12] S. Carter, Z. Armstrong, L. Schubert, I. Johnson, and C. Olah. Activation atlas. *Distill*, 2019. https://distill.pub/2019/activation-atlas.

[13] R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, and N. Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1721–1730. ACM, 2015.

[14] D. Cashman, G. Patterson, A. Mosca, and R. Chang. Rnnbow: Visualizing learning via backpropagation gradients in recurrent neural networks. In *Workshop on Visual Analytics for Deep Learning (VADL)*, 2017.

[15] J. Chae, S. Gao, A. Ramanthan, C. Steed, and G. D. Tourassi. Visualization for classification in deep neural networks. In *Workshop on Visual Analytics for Deep Learning*, 2017.

[16] Q. Chen and V. Koltun. Photographic image synthesis with cascaded refinement networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1511–1520, 2017.

[17] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

[18] M. W. Craven and J. W. Shavlik. Using neural networks for data mining. *Future Generation Computer Systems*, 13(2-3):211–229, 1997.

[19] R. J. Crouser and R. Chang. An affordance-based framework for human computation and human-computer collaboration. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2859–2868, 2012.

[20] F. Dalvi, N. Durrani, H. Sajjad, Y. Belinkov, A. Bau, and J. Glass. What is one grain of sand in the desert? analyzing individual neurons in deep nlp models. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

[21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[22] D. Dey, V. Ramakrishna, M. Hebert, and J. Andrew Bagnell. Predicting multiple structured visual interpretations. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2947–2955, 2015.

[23] A. Endert, P. Fiaux, and C. North. Semantic interaction for visual text analytics. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 473–482. ACM, 2012.

[24] A. Endert, W. Ribarsky, C. Turkay, B. W. Wong, I. Nabney, I. D. Blanco, and F. Rossi. The state of the art in integrating machine learning into visual analytics. In *Computer Graphics Forum*, vol. 36, pp. 458–486. Wiley Online Library, 2017.

[25] J. A. Fails and D. R. Olsen Jr. Interactive machine learning. In *Proceedings of the 8th International Conference on Intelligent User Interfaces*, pp. 39–45. ACM, 2003.

[26] A. Fan, M. Lewis, and Y. Dauphin. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018.

[27] S. Gehrmann, Y. Deng, and A. Rush. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4098–4109, 2018.

[28] S. Green, S. I. Wang, J. Chuang, J. Heer, S. Schuster, and C. D. Manning. Human effort and machine learnability in computer aided translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1225–1236, 2014.

[29] B. J. Grosz. Collaborative systems (aaai-94 presidential address). *AI magazine*, 17(2):67, 1996.

[30] M. J. Guzdial, J. Chen, S.-Y. Chen, and M. Riedl. A general level design editor for co-creative level design. In *Thirteenth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2017.

[31] J. Heer. Agency plus automation: Designing artificial intelligence into interactive systems. *Proceedings of the National Academy of Sciences*, 116(6):1844–1850, 2019.

[32] F. Hohman, N. Hodas, and D. H. Chau. Shapeshop: Towards understanding deep learning representations via interactive experimentation. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pp. 1694–1699. ACM, 2017.

[33] F. M. Hohman, M. Kahng, R. Pienta, and D. H. Chau. Visual analytics in deep learning: An interrogative survey for the next frontiers. *IEEE Transactions on Visualization and Computer Graphics*, 2018.

[34] K. Holstein, J. W. Vaughan, H. Daumé III, M. Dudík, and H. Wallach. Improving fairness in machine learning systems: What do industry practitioners need? *arXiv preprint arXiv:1812.05239*, 2018.

[35] A. Holzinger. Interactive machine learning for health informatics: when do we need the human-in-the-loop? *Brain Informatics*, 3(2):119–131, 2016.

[36] E. Horvitz. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 159–166. ACM, 1999.

[37] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, I. Simon, C. Hawthorne, A. Dai, M. Hoffman, M. Dinculescu, and D. Eck. Music transformer: Generating music with long-term structure. 2019.

[38] M. C. Hughes, G. Hope, L. Weiner, T. H. McCoy Jr, R. H. Perlis, E. B. Sudderth, and F. Doshi-Velez. Semi-supervised prediction-constrained topic models. In *AISTATS*, pp. 1067–1076, 2018.

[39] L. Jiang, S. Liu, and C. Chen. Recent research advances on interactive machine learning. *Journal of Visualization*, 22(2):401–417, 2019.

[40] H. Jing and K. R. McKeown. The decomposition of human-written summary sentences. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 129–136, 1999.

[41] M. Kahng, P. Y. Andrews, A. Kalro, and D. H. P. Chau. Activis: Visual exploration of industry-scale deep neural network models. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):88–97, 2018.

[42] D. Keim, G. Andrienko, J.-D. Fekete, C. Görg, J. Kohlhammer, and G. Melançon. Visual analytics: Definition, process, and challenges. In *Information Visualization*, pp. 154–175. Springer, 2008.

[43] C. Kiddon, L. Zettlemoyer, and Y. Choi. Globally coherent text generation with neural checklist models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 329–339, 2016.

[44] B. Kim, R. Khanna, and O. O. Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In *Advances in Neural Information Processing Systems*, pp. 2280–2288, 2016.

[45] Y. Kim, S. Wiseman, and A. M. Rush. A tutorial on deep latent variable models of natural language. *arXiv preprint arXiv:1812.06834*, 2018.

[46] P.-J. Kindermans, S. Hooker, J. Adebayo, M. Alber, K. T. Schütt, S. Dähne, D. Erhan, and B. Kim. The (Un)reliability of saliency methods. *NIPS workshop on Explaining and Visualizing Deep Learning*, 2017.

[47] J. Krause, A. Dasgupta, J. Swartz, Y. Aphinyanaphongs, and E. Bertini. A workflow for visual diagnostics of binary classifiers using instance-level explanations. *arXiv preprint arXiv:1705.01968*, 2017.

[48] J. Krause, A. Perer, and K. Ng. Interacting with predictions: Visual inspection of black-box machine learning models. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 5686–5697. ACM, 2016.

[49] T. Kulesza, M. Burnett, W.-K. Wong, and S. Stumpf. Principles of explanatory debugging to personalize interactive machine learning. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, pp. 126–137. ACM, 2015.

[50] T. Kulesza, S. Stumpf, M. Burnett, W.-K. Wong, Y. Riche, T. Moore, I. Oberst, A. Shinsel, and K. McIntosh. Explanatory debugging: Supporting end-user debugging of machine-learned programs. In *Visual Languages and Human-Centric Computing (VL/HCC), 2010 IEEE Symposium on*, pp. 41–48. IEEE, 2010.

[51] B. C. Kwon, M.-J. Choi, J. T. Kim, E. Choi, Y. B. Kim, S. Kwon, J. Sun, and J. Choo. RetainVis: Visual analytics with interpretable and interactive recurrent neural networks on electronic medical records. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):299–309,

2019.

[52] C. Lacave and F. J. Díez. A review of explanation methods for bayesian networks. *The Knowledge Engineering Review*, 17(2):107–127, 2002.

[53] V. Lai and C. Tan. On human predictions with explanations and predictions of machine learning models: A case study on deception detection. *arXiv preprint arXiv:1811.07901*, 2018.

[54] T. Lei, R. Barzilay, and T. Jaakkola. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*, 2016.

[55] J. Li, X. Chen, E. Hovy, and D. Jurafsky. Visualizing and understanding neural models in nlp. *arXiv preprint arXiv:1506.01066*, 2015.

[56] B. Y. Lim, A. K. Dey, and D. Avrahami. Why and why not explanations improve the intelligibility of context-aware intelligent systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2119–2128. ACM, 2009.

[57] H. Lin, T. Wu, K. Wongsuphasawat, Y. Choi, and J. Heer. Visualizing attention in sequence-to-sequence summarization models.

[58] Z. C. Lipton. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490*, 2016.

[59] S. Liu, Z. Li, T. Li, V. Srikumar, V. Pascucci, and P.-T. Bremer. NLIZE: A perturbation-driven visual interrogation tool for analyzing and interpreting natural language inference models. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):651–660, 2019.

[60] S. Liu, X. Wang, M. Liu, and J. Zhu. Towards better analysis of machine learning models: A visual analytics perspective. *Visual Informatics*, 1(1):48–56, 2017.

[61] Y. Lu, R. Garcia, B. Hansen, M. Gleicher, and R. Maciejewski. The state-of-the-art in predictive visual analytics. In *Computer Graphics Forum*, vol. 36, pp. 539–562. Wiley Online Library, 2017.

[62] Y. Ming, S. Cao, R. Zhang, Z. Li, Y. Chen, Y. Song, and H. Qu. Understanding hidden memories of recurrent neural networks. In *2017 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 13–24. IEEE, 2017.

[63] A. Odena, V. Dumoulin, and C. Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016.

[64] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2337–2346, 2019.

[65] N. Pezzotti, T. Höllt, J. Van Gemert, B. P. Lelieveldt, E. Eisemann, and A. Vilanova. Deepeyes: Progressive visual analytics for designing deep neural networks. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):98–108, 2018.

[66] M. T. Ribeiro, S. Singh, and C. Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144. ACM, 2016.

[67] D. Sacha, M. Kraus, D. A. Keim, and M. Chen. Vis4ml: An ontology for visual analytics assisted machine learning. *IEEE transactions on visualization and computer graphics*, 25(1):385–395, 2019.

[68] A. See, P. J. Liu, and C. D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 1073–1083, 2017.

[69] D. Smilkov, S. Carter, D. Sculley, F. B. Viégas, and M. Wattenberg. Direct-manipulation visualization of deep networks. *arXiv preprint arXiv:1708.03788*, 2017.

[70] D. Smilkov, N. Thorat, C. Nicholson, E. Reif, F. B. Viégas, and M. Wattenberg. Embedding projector: Interactive visualization and interpretation of embeddings. *arXiv preprint arXiv:1611.05469*, 2016.

[71] C. D. Stolper, A. Perer, and D. Gotz. Progressive visual analytics: User-driven visual exploration of in-progress analytics. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1653–1662, 2014.

[72] H. Strobelt, S. Gehrmann, M. Behrisch, A. Perer, H. Pfister, and A. Rush. Debugging sequence-to-sequence models with Seq2Seq-Vis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 368–370, 2018.

[73] H. Strobelt, S. Gehrmann, M. Behrisch, A. Perer, H. Pfister, and A. M. Rush. Seq2Seq-Vis: A visual debugging tool for sequence-to-sequence models. *IEEE transactions on visualization and computer graphics*, 25(1):353–363, 2019.

[74] H. Strobelt, S. Gehrmann, H. Pfister, and A. M. Rush. LSTMVis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE transactions on visualization and computer graphics*, 24(1):667–676, 2018.

[75] S. Stumpf, V. Rajaram, L. Li, W.-K. Wong, M. Burnett, T. Dietterich, E. Sullivan, and J. Herlocker. Interacting meaningfully with machine learning systems: Three experiments. *International Journal of Human-Computer Studies*, 67(8):639–662, 2009.

[76] F.-Y. Tzeng and K.-L. Ma. Opening the black box-data driven visualization of neural networks. In *VIS 05. IEEE Visualization, 2005.*, pp. 383–390. IEEE, 2005.

[77] A. Vellido, J. D. Martín-Guerrero, and P. J. Lisboa. Making machine learning models interpretable. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, vol. 12, pp. 163–172. Citeseer, 2012.

[78] S. Wachter, B. Mittelstadt, and C. Russell. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *arXiv preprint arXiv:1711.00399*, 2017.

[79] J. Wang, L. Gou, H. Yang, and H.-W. Shen. Ganviz: A visual analytics approach to understand the adversarial game. *IEEE transactions on visualization and computer graphics*, 24(6):1905–1917, 2018.

[80] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8798–8807, 2018.

[81] What-If Tool. `https://pair-code.github.io/what-if-tool`, 2017. [Online; accessed: 2018-01-09].

[82] Y. Wilks. Close engagements with artificial companion, 2010.

[83] T. Williams. Toward ethical natural language generation for human-robot interaction. In *Companion of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 281–282. ACM, 2018.

[84] K. Wongsuphasawat, D. Smilkov, J. Wexler, J. Wilson, D. Mané, D. Fritz, D. Krishnan, F. B. Viégas, and M. Wattenberg. Visualizing dataflow graphs of deep learning models in tensorflow. *IEEE transactions on visualization and computer graphics*, 24(1):1–12, 2018.

[85] K. Xu, J. Ba, R. Kiros, K. Cho, A. C. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the International Conference on Machine Learning*, pp. 2048–2057, 2015.

[86] Q. Yang. Machine learning as a ux design material: How can we imagine beyond automation, recommenders, and reminders? In *2018 AAAI Spring Symposium Series*, 2018.

[87] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the International Conference on Machine Learning*, vol. 97, pp. 412–420, 1997.

[88] M. Yin, J. Wortman Vaughan, and H. Wallach. Understanding the effect of accuracy on trust in machine learning models. ACM Press, April 2019.

[89] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

[90] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pp. 818–833. Springer, 2014.

[91] J. Zhang, Y. Wang, P. Molino, L. Li, and D. S. Ebert. Manifold: A model-agnostic framework for interpretation and diagnosis of machine learning models. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):364–373, 2019.

[92] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling. Visualizing deep neural network decisions: Prediction difference analysis. In *Conference Track Proceedings of the 5th International Conference on Learning Representations*, 2017.