DOI: 10.1111/cgf.14840

Eurographics Conference on Visualization (EuroVis) 2023
D. Archambault, R. Bujack, and T. Schreck
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 42 (2023), Number 3

LINGO : Visually Debiasing Natural Language Instructions to Support Task Diversity

A. Arunkumar[1], S. Sharma[1], R. Agrawal[1], S. Chandrasekaran[1] and C. Bryan[1]

[1]Arizona State University, Tempe, United States

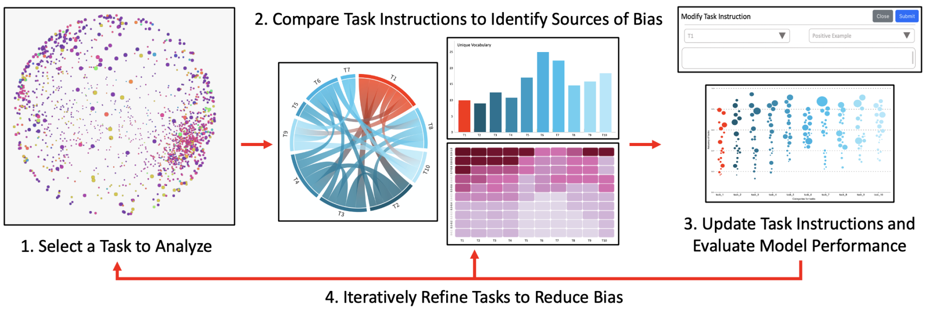

1. Select a Task to Analyze
2. Compare Task Instructions to Identify Sources of Bias
3. Update Task Instructions and Evaluate Model Performance
4. Iteratively Refine Tasks to Reduce Bias
Modify Task Instruction
Close
Submit
Unique Vocabulary

Figure 1: *LINGO workflow: (1) Select a task to analyze based on the sentence embedding space of task instructions. (2) Next, compare the linguistic characteristics of nearest neighbor task instructions to identify features that lead to instruction bias. (3) Modify or create task instructions, and evaluate model performance on the updated task. (4) Tasks can be iteratively modified till bias is sufficiently reduced.*

Abstract

Cross-task generalization is a significant outcome that defines mastery in natural language understanding. Humans show a remarkable aptitude for this, and can solve many different types of tasks, given definitions in the form of textual instructions and a small set of examples. Recent work with pre-trained language models mimics this learning style: users can define and exemplify a task for the model to attempt as a series of natural language prompts or instructions. While prompting approaches have led to higher cross-task generalization compared to traditional supervised learning, analyzing 'bias' in the task instructions given to the model is a difficult problem, and has thus been relatively unexplored. For instance, are we truly modeling a task, or are we modeling a user's instructions? To help investigate this, we develop LINGO, a novel visual analytics interface that supports an effective, task-driven workflow to (1) help identify bias in natural language task instructions, (2) alter (or create) task instructions to reduce bias, and (3) evaluate pre-trained model performance on debiased task instructions. To robustly evaluate LINGO, we conduct a user study with both novice and expert instruction creators, over a dataset of 1,616 linguistic tasks and their natural language instructions, spanning 55 different languages. For both user groups, LINGO promotes the creation of more difficult tasks for pre-trained models, that contain higher linguistic diversity and lower instruction bias. We additionally discuss how the insights learned in developing and evaluating LINGO can aid in the design of future dashboards that aim to minimize the effort involved in prompt creation across multiple domains.

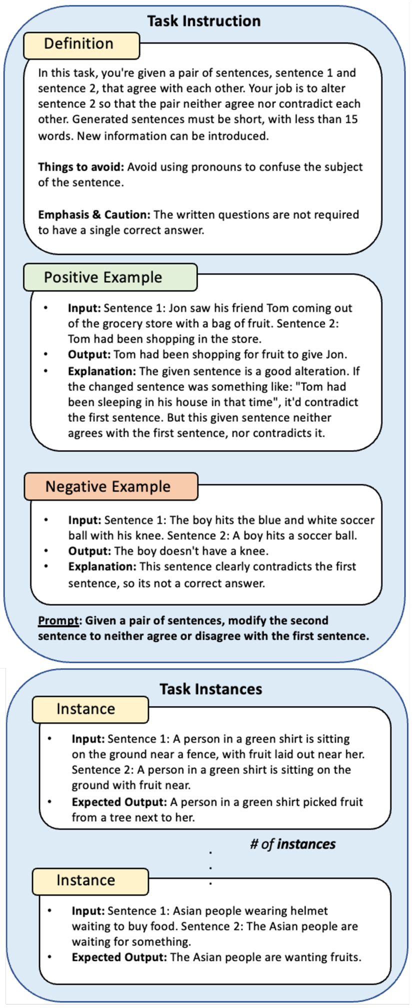
CCS Concepts

• Human-centered computing** → **Visual analytics; Text input; • Computing methodologies** → **Natural language processing;

1. Introduction

Benchmark datasets play a key role in driving progress in Natural Language Processing (NLP) [RGA23, WMA*22]. Pre-trained language models (PLMs) have achieved state-of-the- art performance on many benchmark tasks [PNI*18, BMR*20] and have shown promising generalization capabilities [KMK*20, AGS*21]. However, cross-task generalization, or generalization of a PML to unseen tasks, remains a comparatively hard challenge [MKBH21].

In contrast, humans are adept at such generalization. For instance, NLP benchmarks are commonly created through crowdsourcing, wherein crowdworkers create instances for a task, following instructions from dataset creators [EL20, SVN*21]. Recently, the NLP community has made great progress in building models for generalization to unseen tasks via in-context instructions [BMR*20, CND*22]. These instructions comprise a natural language prompt that defines and exemplifies a task for a PLM, similar to crowdsourcing instructions. Figure 2 represents the organization of a natural language instruction (composed of a *task description* and *examples*) for a text modification task, along with task instances for model evaluation.



Task Instruction

Definition

In this task, you're given a pair of sentences, sentence 1 and sentence 2, that agree with each other. Your job is to alter sentence 2 so that the pair neither agree nor contradict each other. Generated sentences must be short, with less than 15 words. New information can be introduced.

Things to avoid: Avoid using pronouns to confuse the subject of the sentence.

Emphasis & Caution: The written questions are not required to have a single correct answer.

Positive Example

- **Input:** Sentence 1: Jon saw his friend Tom coming out of the grocery store with a bag of fruit. Sentence 2: Tom had been shopping in the store.
- **Output:** Tom had been shopping for fruit to give Jon.
- **Explanation:** The given sentence is a good alteration. If the changed sentence was something like: "Tom had been sleeping in his house in that time", it'd contradict the first sentence. But this given sentence neither agrees with the first sentence, nor contradicts it.

Negative Example

- **Input:** Sentence 1: The boy hits the blue and white soccer ball with his knee. Sentence 2: A boy hits a soccer ball.
- **Output:** The boy doesn't have a knee.
- **Explanation:** This sentence clearly contradicts the first sentence, so its not a correct answer.

Prompt: Given a pair of sentences, modify the second sentence to neither agree or disagree with the first sentence.

Task Instances

Instance

- **Input:** Sentence 1: A person in a green shirt is sitting on the ground near a fence, with fruit laid out near her. Sentence 2: A person in a green shirt is sitting on the ground with fruit near.
- **Expected Output:** A person in a green shirt picked fruit from a tree next to her.

# of instances

Instance

- **Input:** Sentence 1: Asian people wearing helmet waiting to buy food. Sentence 2: The Asian people are waiting for something.
- **Expected Output:** The Asian people are wanting fruits.

Figure 2: *Schema used for representing natural language instructions. The task is created from the SNLI benchmark [BAPM15]*

Unfortunately, prior work in crowdsourcing has shown that task instructions provided by dataset creators often influence crowdworkers to follow specific patterns during instance creation. This leads to collection of biased data, which inflates model performance [GGB19, PMGB22, HSG*21]. This is particularly critical in high-risk domains such as healthcare [MA21a], where incorrect answers to a task can prove fatal. Therefore, despite the success of instructions in PLMs, the NLP community is faced with the question of: *are we truly modeling a task or a user's instructions*? To investigate this, we developed *LINGO*, an end-to-end visual analysis and authoring tool for natural language instructions. LINGO is designed to address real-world challenges faced in NLP benchmark creation, by helping authors identify/compare ***instruction bias*** pertaining to different types of tasks; LINGO also facilitates real-time task instruction creation/modification and evaluation with PLMs, as shown in Figure 1. ***To our knowledge, LINGO represents the first visual analytics system that enables realtime feedback and reconciliation opportunities to data creators for instruction bias.***

In this paper, we describe the process of developing LINGO, based on a pre-study with expert instructional prompt creators to identify salient challenges involved in the identification and reconciliation of instruction bias. We robustly evaluate LINGO via a real-world usage scenario, and a user study with both novice and expert instruction authors over a dataset of 1,616 linguistic tasks and their natural language definitions, spanning 55 different languages. (We note that, while LINGO is language-independent, linguistic knowledge is required for users to analyze multi-lingual tasks.) The results indicate that LINGO promotes the creation of more difficult tasks for pre-trained models, that are defined with higher linguistic diversity and exhibit lower instruction bias. In following a design study methodology, and based on the process of creating and evaluating LINGO, feedback from our formative interviews and the user study, we additionally discuss how tools like LINGO can benefit the NLP and AI communities in using prompting to create less biased benchmark datasets and evaluate PLMs, by reducing the effort involved in diverse prompt creation across multiple domains and languages, and helping standardize the quantification and elimination of bias in text data.

2. Related Work

2.1. Learning from Instructions

The success of PLMs [BMR*20, CND*22] has powered the development of various 'prompting' techniques [LYF*23] in NLP. Typically, prompts are extremely short and may not include a complete definition of complex tasks [SS21, RM21]. During prompting, a text snippet is added to unlabelled data; this allows the downstream tasks to be represented in the format as pre-training objectives, and requires no new parameters, enabling model learning without having to finetune for each task. Prompt construction is an active research area; we focus on a technique that extends prompting to better define benchmark tasks, called 'natural language instructions'. This approach stems from NLP benchmark creation via crowdsourcing, where instructions are given to crowdworkers for sample creation pertaining to a particular task [GGB19, EL20]. Natural language instructions are a versatile means of defining goals; they have previously been studied in applications such as SQL translation [KSHL20] and interaction within a visual environment [STG*20]. Instructions describe tasks in natural language [WLGP20], and guide PLMs to generalize to unseen tasks [MKBH21, WBZ*] without requiring task-specific training (see Figure 2). Meta-datasets (i.e., datasets of datasets) [TZD*] of human-authored instructions (as shown in Table 1) have been used as benchmarks for cross-task generalization in PLMs.

Meta-Dataset	SUP-NATINST$^{+,\bigcirc}$ [WMA*22]	NATINST$^{\bigcirc}$ [MKBH21]	PROMPT-SOURCE [BSY*22]	FLAN^{+} [WBZ*]
Number of Tasks	1616	61	176	62
Number of Instructions	1616	61	2052	620
Number of Task Categories	76	6	13	12
Average Task Definition Length (in words)	56.6	134.4	24.8	8.2

Table 1: *Notable open-source meta-datasets of natural language instructions.* $\bigcirc$ *: indicates the presence of negative examples in the task instruction.* $+$ *: indicates the presence of non-english tasks.*

2.2. Bias in Instructions

Prior work has shown that crowdsourced natural language benchmarks exhibit various spurious biases (i.e., unintended correlations between input and output), that lead to overestimation of PLM performance [SSK*17,PNH*18,GSL*18,LBSB*20]. Several techniques have been proposed to handle such bias post-creation, including improving linguistic diversity of samples [YZFBC*20, LML*19, SYH20] and augmenting data with adversarial samples intended to fool the model [WRF*19,KBN*21,TYLB*]. Similarly, there is evidence that natural language instructions provided by dataset creators during crowdsourcing influences crowdworkers to follow specific patterns during sample creation [GGB19,PMGB22, HSG*21]. These patterns, termed as 'instruction bias,' propagate to the dataset and are subsequently over-represented in the collected data. For instance, the crowdsourced DROP dataset [DWD*19] defines a reading comprehension task where discrete reasoning (e.g., addition, sorting, or counting) must be done based on a paragraph of text. The given instructions contain examples for reference during sample creation; 70% of these examples begin with `"How many [field goals | yards | points | touchdowns]"` Subsequently, ~ 62% of the crowdsourced data samples in both the train and test splits begin with the same pattern of phrasing [DWD*19].

2.3. Visual Analysis of Natural Language Benchmarks

The majority of prior approaches for handling data bias in NLP benchmarks focus on the adjustment of hyperparameters for machine learning models, or on artificially rebalancing the biased datasets post-creation [LV19, LLV18, KHL, GAB*20, NWD*20]. For instance, users can be encouraged to rephrase highlighted portions of text that are important for a model to make a prediction to promote adversarial sample creation during benchmark construction [WRF*19, KBN*21, TYLB*, AIP15, Van19]. Visual analysis tools that afford NLP leaderboard probing through metric customization have also been proposed as a way to isolate model biases [MA21b]. The visualization of bias detection algorithm results [LMT*20, LMTT] has been leveraged to inform non-expert data creators on what constitutes data bias, although the specific use of visualization for analyzing NLP benchmarks is relatively underdeveloped. Considering the problem of biased data more generally across machine learning and deep learning, related visualization techniques and tools have been developed to analyze dataset biases, vulnerabilities, predictive fairness [CEH*19, MXLM19, WXC*20, CMJ*20], although these tools are designed to address different tasks and goals than LINGO. More specifically, LINGO examines a wide array of linguistic features to identify and resolve instruction bias. Analytical capabilities are balanced with visualization techniques catering to both expert and non-expert prompt authors.

3. Design Process and Goals

To help motivate our system design, we surveyed five NLP practitioners (with 4–6 years of experience in investigating data bias and prompt authoring). Our motivating question was: *what are the salient challenges in the identification and reconciliation of instruction bias, where visual analytics can be a key approach for providing insights/solutions?* Table 2 summarizes several challenges that were explicitly discussed by the practitioners. All participants noted that bias detection in NLP benchmarks is mainly performed as a post-hoc analysis using backend code and algorithms, however there are no standardized practices or metrics for isolating bias. If implemented, bias reconciliation primarily involves either data augmentation or model restriction. Importantly, visualization-based approaches for bias detection and reconciliation in both NLP instructions and datasets are relatively unexplored.

	Challenges for Analyzing Instruction Bias		
(5)	Wide range of instruction features that can cause bias	*	**DR1,DR2**
(4)	Difficult to juxtapose samples to identify bias sources	*	**DR3**
(4)	Different instruction types will exhibit different biases	*	**DR1,DR5**
(3)	PLM performance depends on instruction quality	*	**DR4**
(3)	Instruction examples dictate created sample diversity	*	**DR4,DR5**
(3)	Instructions optimal to a model may not be optimal for humans		

Table 2: *Our survey with NLP experts identified several items as salient challenges for analyzing instruction bias. We note the number of users who referenced each challenge in parentheses. Items addressed by our current tool are noted with an asterisk, and are mapped to derived design requirements.*

Based on the challenges listed in Table 2, we derived a set of five high-level design requirements (**DRs**) that a visual analytics system can employ for analyzing and reconciling instruction bias.

DR1: Compare instructions in a categorized manner. Writing style is dictated by the task category, domain (reasoning type), and dataset for which the instruction is created; this must be accounted for when comparing instructions within or between categories.

DR2: Allow granular analysis of different parts of an instruction. Instructions consist of the task definition and examples (positive and negative) with explanation. We consider the entire instruction for analysis as an overview comparison against constituent data samples used for PLM evaluation. However the biases exhibited within and between different parts of the instruction may vary, so multi-granular analysis can be particularly beneficial.

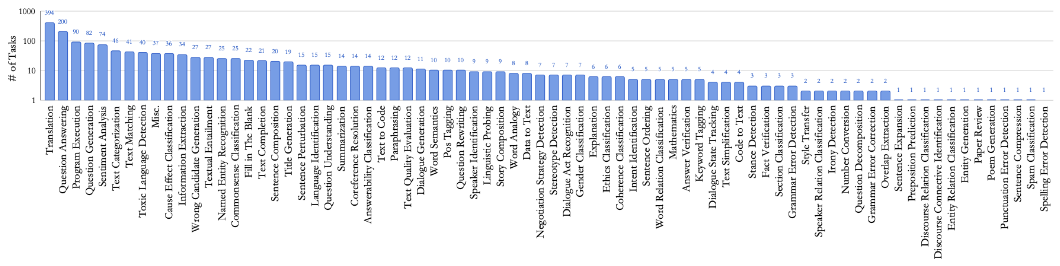


Figure 3: *Task categories in* SUP-NATINST.

DR3: Support juxtaposition of instruction examples and data samples. The modification of instructions to reduce instruction bias is contingent on the relationship between instruction examples and data samples. Examples must have sufficient inductive bias to inform a PLM/crowdworker of the task, but must also exhibit sufficient diversity so sample creation isn't artificially constrained.

DR4: Evaluate PLM on modified instructions in situ. Instruction effectiveness must be gauged in situ in terms of PLM performance. This ensures that instruction modification to reduce instruction bias still retains sufficient inductive bias for the model to learn.

DR5: Provide multiple perspectives for analysing bias features. Instruction bias can arise from a wide variety of linguistic features. Task-agnostic features must be selectable to facilitate analysis of different instruction categories.

Notably, a significant constraint in successfully addressing **DR1–DR5** is that the target user base (prompt authors) likely lacks visualization expertise. Therefore, in keeping with a design study methodology [SMM12], when implementing LINGO we adhered to the standard of a "well-justified combination of [primarily] existing techniques." This allowed us to balance the analytical insight and power provided by LINGO, while not overwhelming them with overly complex or esoteric visual encodings, or requiring a steep learning curve. We also note that LINGO is designed to potentially benefit both expert and non-expert prompt authors during instruction creation for PLM evaluation; additionally, instructions for crowdsourced data collection can be improved, based on insights gleaned by dataset creators.

4. Dataset Schema

To demonstrate LINGO, we use a state-of-the-art meta-dataset of task instructions, called Super-Natural Instructions (SUP-NATINST) [WMA*22]. This benchmark consists of 1,616 NLP tasks along with their natural language instructions; 76 task categories span 55 different languages (576 non-english tasks) are present (Figure 3 shows the diversity and sizes of the 76 task categories). These tasks were contributed by 88 NLP practitioners in a crowdsourced manner. SUP-NATINST is considered state of the art for model evaluation on task instructions.

Instructions are represented via a unified schema, as shown in Figure 2. Broadly, schema ingredients are defined as follows:

- **Definition** defines a given task in natural language. This is a complete definition of how an input text (e.g., a sentence or a document) is expected to be mapped to an output text.
- **Positive Examples** are samples of inputs and their *correct* outputs, along with a short explanation for each.
- **Negative Examples** are samples of inputs and their *incorrect* or *invalid* outputs, along with a short explanation for each.

Task instances (i.e., data samples to be solved by the model given the instruction) are organized as textual input and a list of acceptable textual outputs. The number of instances is limited to 6.5K per task to avoid instance imbalance, as done for model evaluation in SUP-NATINST [WMA*22].

A single dataset can be used to generate multiple subtasks; for instance, the SNLI dataset used for the task in Figure 2 is used to generate 12 tasks in total for SUP-NATINST pertaining to the task categories of answer generation, wrong answer generation, text modification, and classification.

Bias Category	Features Evaluated
Diversity	Sample Length [WRF*19], Unique Vocabulary (new words introduced) [YZFBC*20, LZM*20]
Similarity	N-gram/POS-tag (part-of-speech tags: nouns, verbs, adjectives, adverbs): Frequency, Overlap, Jaccard Similarity [WRF*19, YZFBC*20, LML*19, PNH*18, GSL*18]
Component Bias$^{\triangle}$	[ENT] × [ENT] Overlap [LML*19], [ENT] × [ENT] Correlation [SYH20, WRF*19]

Table 3: *LINGO supports three bias categories, each composed of several available metrics.* $^{\triangle}$ *[ENT] represents* `{task instruction | definition | example (positive/example/both) | task instance}`

5. Instruction Bias Measures

LINGO supports three broad categories of instruction bias measures, based on existing bias identification measures for NLP benchmark analysis. Table 3 defines the categories of instruction bias supported in the system, along with the evaluation metrics curated from prior research for each category. (LINGO is also extensible to accommodate additional measures.)

Broadly, **diversity** examines an individual task instruction, while **similarity** juxtaposes multiple task instructions; both utilize the full instruction in calculation. **Component bias** compares different parts of instructions and task instances; this can be done (i) within or between tasks (e.g., a positive example from Task X can be compared to a task instance either from Task X or any other task), and

(ii) at multiple granularities (sentence/word/part-of-speech tag/n-gram). Metrics are calculated using the NLTK library [LB02] with a Python API; text pre-processing involves stop-word removal, tokenization, POS tagging, and lemmatization.

6. LINGO

We now describe LINGO's system design. The system is a full-stack application, with a backend server for data storage, query, and NLP-based computation, and a frontend interface for visualization and interaction. For examples of how LINGO can be used to analyze and reduce instruction bias, see the usage scenario in §7 and the demo video include in the supplemental materials.

6.1. Backend

The backend server is built using Python and Flask, and acts as the storage and service layer between the instruction dataset and the frontend user interface. A MongoDB [MON] database stores the tasks as JSON objects, supporting quick retrieval.

Based on a user's choice of (i) task instruction(s) to analyze, (ii) components of the instruction to include in the analysis, and (iii) bias metric(s) to be used, we retrieve instruction(s) from the database (this supports **DR2**) and perform necessary metric computations (see §5) to populate the front-end panels (thus supporting **DR5**). We evaluate a subset of task instances using a Python script to call on the GPT-3 [BMR*20] Open API (davinci-instruct-beta-v3), on load/modification of task instructions (**DR4**). We follow [MKBH21] and treat the tasks as text-generation problems, using ROUGE-L [ROU04] to automatically evaluate the generated outputs and visualize the results.

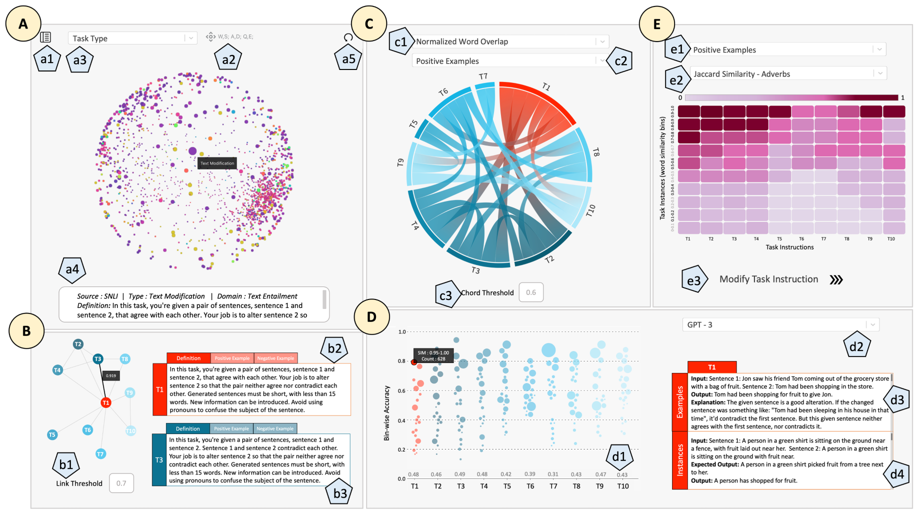
6.2. Frontend

Figure 4 shows an overview of LINGO's frontend interface. It consists of five panels **(A)**—**(E)** which are linked via coordinated interactions (e.g., hovering on or selecting a task instruction in one view highlights or selects the instruction in other views) and designed to support the five **DRs** described in §5. We use fixed color scales that accommodate users with color-impaired vision using the R colorspace toolbox.

(A) Overview Panel. The overview panel consists of a t-SNE projection, where each point is a task instruction. The pairwise distances between instructions is based on sentence embedding distances, calculated over the complete task instruction (i.e., `definition+examples`). We use t-SNE here as it ably preserves geometry for nested clusters of varying density, though LINGO is easily extensible to support MDS, PCA, etc. This projection can be rendered both on a traditional 2D plane, or as a rotatable 3D sphere; we model the embedding sphere after embedding projectors used in TensorFlow. Regarding this latter encoding option, in particular we found during early testing that a 2D projection was at times overly cluttered, which made it difficult to identify and select individual points (in particular, translation tasks, which accounted for ∼24% of the tasks in the dataset used, often contained very similar phrasing). Toggling to a 3D projection could provide a better separation of points and identification of clusters. However, filtering is sometimes necessary to enhance scalability and prevent clutter for larger or more closely distributed datasets. Point color represents the categorization of the instructions based on task type/domain/source dataset **(DR1)**. **(a1)** A color legend can be viewed by clicking on a toggle icon. **(a2)** allows interacting with the sphere projection; rotation along x/y/z axes is done via arrow keys. **(a3)** The basis for categorizing task instructions can be changed with the dropdown; the default categorization is based on task type (i.e., the nature of the task). Mousing over of a point highlights all task instructions belonging to the same category; a tooltip details the category of the task and **(a4)** a text area shows the source dataset, domain, and definition of the task; users can also search for a specific task by typing in this text area. Clicking on a point selects the corresponding "root instruction" for further analysis, along with the k-most highly correlated instructions with respect to the root instruction (k default: 9). For the selection, the root node is colored red, and the additional k nodes are assigned blue shades based on their correlation amount. Additional instructions in the same category as the root instruction remain colored with decreased opacity; all other points are toggled to grayscale with decreased opacity. **(a5)** Clicking a refresh icon clears the selection, resets the sphere to the default view, and clears other panels in the display.

(B) Correlation Panel. This panel displays the selected root and k instructions (from **(A)**), with a node-link diagram. Each node represents an instruction, colored and labelled in order of correlation (sentence similarity). (T1 represents the root instruction and is colored red, T2 represents the most correlated node, and so on.) Links between nodes indicate a similarity score over a threshold (adjustable using **(b1)**, default: 0.5). Mousing over (or clicking) a node displays its similarity score in a tooltip, as well as the text of the corresponding root and task instructions (in **(b2)** and **(b3)**). Mousing over (or clicking) a link displays the instructions corresponding to the end nodes in **(b2)** and **(b3)** and also the similarity between the nodes in a tooltip. The tabs in **(b2)**/**(b3)** allow viewing either the full instruction/definition/positive/negative examples for comparison **(DR2)**. We preserve positional information from the 3D embedding space through the use of these network links, which indicate the strength of the relationship between two tasks at different granularities. The color scale assigns more similar samples with darker colors. Also, the similarity thresholds to draw links and the k-value can be customized by the user; therefore the range of similarity considered varies. (B) hence streamlines the selection of a subset of tasks from A that implicitly provide contextual information for the root task instruction. This is particularly important when using models such as ChatGPT which can remember several previously entered instruction prompts.

(C) Instruction Decomposition Panel. This panel shows inter-task relationships between the selected root and k instructions using a chord diagram. Multi-granular comparison can be done, based on sub-components of the task instruction chosen for examination. Specifically, the inter-task relationship is examined in terms of normalized length/normalized word overlap correlation (chosen from **(c1)**). Sub-components that can be used for analysis are: `{task instruction | definition only | example only (positive/example/both)}`**(DR2)**, selected using **(c2)**. The task labels and color scheme follow (B);



Source : SNLI | *Type : Text Modification* | *Domain : Text Entailment*
Definition: In this task, you're given a pair of sentences, sentence 1 and sentence 2, that agree with each other. Your job is to alter sentence 2 so

Definition | Positive Example | Negative Example

T1: In this task, you're given a pair of sentences, sentence 1 and sentence 2, that agree with each other. Your job is to alter sentence 2 so that the pair neither agree nor contradict each other. Generated sentences must be short, with less than 15 words. New information can be introduced. Avoid using pronouns to confuse the subject of the sentence.

T3: In this task, you're given a pair of sentences, sentence 1 and sentence 2. Sentence 1 and sentence 2 contradict each other. Your job is to alter sentence 2 so that the pair neither agree nor contradict each other. Generated sentences must be short, with less than 15 words. New information can be introduced. Avoid using pronouns to confuse the subject of the sentence.

T1

Examples
Input: Sentence 1: Jon saw his friend Tom coming out of the grocery store with a bag of fruit. Sentence 2: Tom had been shopping in the store.
Output: Tom had been shopping for fruit to give Jon.
Explanation: The given sentence is a good alteration. If the changed sentence was something like: "Tom had been sleeping in his house in that time", it'd contradict the first sentence. But this given sentence neither agrees with the first sentence, nor contradicts it.

Instances
Input: Sentence 1: A person in a green shirt is sitting on the ground near a fence, with fruit laid out near her. Sentence 2: A person in a green shirt is sitting on the ground with fruit near.
Expected Output: A person in a green shirt picked fruit from a tree next to her.
Output: A person has shopped for fruit.

Figure 4: *LINGO consists of 5 panels to support (A) overview and selection of instructions, (B) examination of instruction correlation, (C) instruction decomposition and comparison, (D) model evaluation with instructions, and (E) inspection and resolution of instruction bias.*

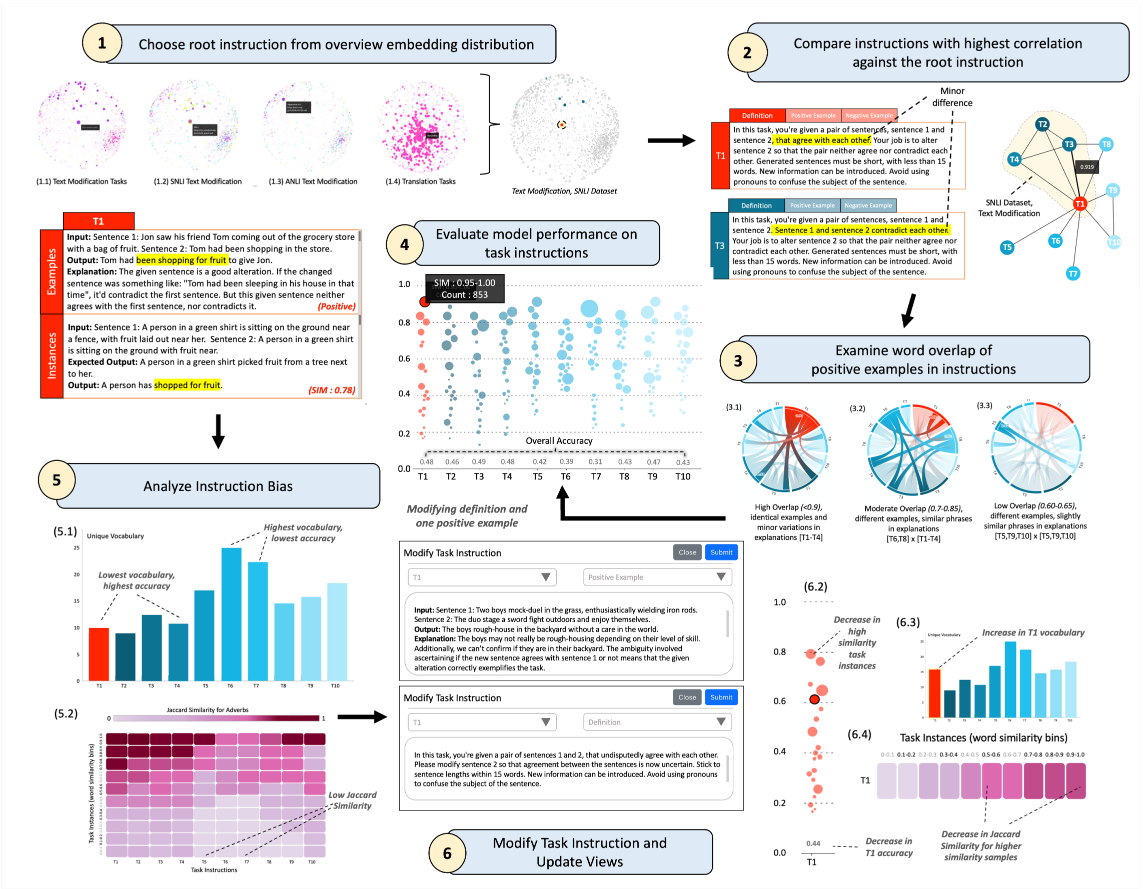
chord ribbons are drawn between tasks with relational values over a threshold (see **(c3)**, default: 0.5). Color corresponds to task color, and thickness represents the strength of the relation. Mousing over (or clicking) a chord highlights its corresponding tasks with labels displaying the similarity values. Additionally, the corresponding tasks are highlighted in (**B**), and **(b2)** and **(b3)** are populated. (C) explicitly highlights which sub-components of the task instruction that contribute the highest bias. By tracking the linguistic diversity used for low-bias tasks connected to the root instruction with chords, users can understand the scale and nature of instruction modification required.

(D) Model Results Panel. This panel displays the model evaluation results for task instances corresponding to the subset of instructions in (B); **(d2)** is used to select the model for evaluation. As noted in §6.1, we use GPT-3 **(DR4)**. Each beeswarm plot represents a task (T1–T10); ordered and colored by its correlation to the root instruction. The y-axis plots the accuracy of GPT-3 on the task instances. Each task instruction is evaluated by testing the model on 6.5K task instances; **(d1)** denotes the GPT-3 accuracies for all tasks. To reduce clutter, we perform an aggregation step: the task instances are grouped into 20 equidistant bins based on their word similarity to the positive and negative examples given in the task instruction as $[0.00-0.05], [0.05-0.10], \ldots, [0.95-1.00]$ **(DR3)**. The average accuracy over instances belonging to each bin is plotted as the y-axis value (meaning each beeswarm displays up to 20 points). The size of the point represents the number of task instances present in that bin (if a bin is empty, it is not plotted as a point). Mousing over (or clicking) a points displays: (i) a tooltip with the number of instances, and similarity bin limits, (ii) a set of positive and negative examples for the corresponding task in **(d3)**, and (iii) a set of three task instances from the corresponding similarity bin, which have the highest word overlap with the examples in **(d4)**. To summarize, this panel examines the influence of instance-example similarity bias on model performance, i.e., are the examples given in the task instruction sufficiently diverse to enable to model to generalize to unseen instances?

(E) Bias Metrics Panel. This final panel can be used to asses instruction bias using metrics that gauge diversity and similarity of instructions (see Table 3). **(e1)** The user can can select the subcomponents of the instruction to analyze (DR2), **(e2)** and can also select desired bias metrics **(DR5)**. Frequency based metrics (for n-grams and POS Tags), sample length, and unique vocabulary count are represented using bar charts, where each bar corresponds to a task (Figure 5**(6.3)** shows an example). These metrics are only applied to the task instruction, and measure the diversity of the task. Overlap (normalized) and Jaccard Similarity for n-grams and POS Tags is plotted with a heatmap, where increasingly dark pink shades represent values closer to 1. These metrics compare the similarity of the task instruction to the task instances. The task instances are grouped into 10 equal-spaced bins as $[0.0-0.1]\ldots[0.9-1.0]$, based on normalized word similarity to instruction examples). Clicking **(e3)** opens a dialog window where the definitions and/or examples of any one task instruction (from T1-T10) can be modified (see demo video, Figure 5**(6.1)**). On submitting the new instruction, panels **(A)**–**(E)** update accordingly, where the modified task instruction now serves as the new *root instruction*. We use either heatmaps or bar charts to represent cross-task bias statistics. This is done to differentiate the nature of the quantities being compared, i.e., as normalized ratios or absolute counts. Thus, users are less likely to conflate bias-metrics used to identify which specific modification patterns will optimally reduce instruction bias.

7. Usage Scenario

To help illustrate how LINGO can be used to analyze instruction bias, we now present a use case scenario with Owen, a prompt author who is analyzing instructions created for Text Modification (TM) tasks (see supplementary material for additional case studies). Figure 5 shows his workflow.


1 Choose root instruction from overview embedding distribution
(1.1) Text Modification Tasks
(1.2) SNLI Text Modification
(1.3) ANLI Text Modification
(1.4) Translation Tasks
Text Modification, SNLI Dataset
2 Compare instructions with highest correlation against the root instruction
Minor difference
Definition
Positive Example
Negative Example
SNLI Dataset, Text Modification
0.919
3 Examine word overlap of positive examples in instructions
(3.1) High Overlap (<0.9), identical examples and minor variations in explanations [T1-T4]
(3.2) Moderate Overlap (0.7-0.85), different examples, similar phrases in explanations [T6,T8] x [T1-T4]
(3.3) Low Overlap (0.60-0.65), different examples, slightly similar phrases in explanations [T5,T9,T10] x [T5,T9,T10]
4 Evaluate model performance on task instructions
SIM : 0.95-1.00
Count : 853
Overall Accuracy
0.48 0.46 0.49 0.48 0.42 0.39 0.31 0.43 0.47 0.43
T1 T2 T3 T4 T5 T6 T7 T8 T9 T10
Modifying definition and one positive example
5 Analyze Instruction Bias
(5.1) Unique Vocabulary
Lowest vocabulary, highest accuracy
Highest vocabulary, lowest accuracy
(5.2) Jaccard Similarity for Adverbs
Task Instances (word similarity bins)
Task Instructions
Low Jaccard Similarity
6 Modify Task Instruction and Update Views
(6.2) Decrease in high similarity task instances
0.44
Decrease in T1 accuracy
(6.3) Unique Vocabulary
Increase in T1 vocabulary
(6.4) Task Instances (word similarity bins)
0-0.1 0.1-0.2 0.2-0.3 0.3-0.4 0.4-0.5 0.5-0.6 0.6-0.7 0.7-0.8 0.8-0.9 0.9-1.0
Decrease in Jaccard Similarity for higher similarity samples

T1

Examples

Input: Sentence 1: Jon saw his friend Tom coming out of the grocery store with a bag of fruit. Sentence 2: Tom had been shopping in the store.
Output: Tom had been shopping for fruit to give Jon.
Explanation: The given sentence is a good alteration. If the changed sentence was something like: "Tom had been sleeping in his house in that time", it'd contradict the first sentence. But this given sentence neither agrees with the first sentence, nor contradicts it. *(Positive)*

Instances

Input: Sentence 1: A person in a green shirt is sitting on the ground near a fence, with fruit laid out near her. Sentence 2: A person in a green shirt is sitting on the ground with fruit near.
Expected Output: A person in a green shirt picked fruit from a tree next to her.
Output: A person has shopped for fruit. *(SIM : 0.78)*

T1 Definition: In this task, you're given a pair of sentences, sentence 1 and sentence 2, that agree with each other. Your job is to alter sentence 2 so that the pair neither agree nor contradict each other. Generated sentences must be short, with less than 15 words. New information can be introduced. Avoid using pronouns to confuse the subject of the sentence.

T3 Definition: In this task, you're given a pair of sentences, sentence 1 and sentence 2. Sentence 1 and sentence 2 contradict each other. Your job is to alter sentence 2 so that the pair neither agree nor contradict each other. Generated sentences must be short, with less than 15 words. New information can be introduced. Avoid using pronouns to confuse the subject of the sentence.

Modify Task Instruction — T1 — Positive Example

Input: Sentence 1: Two boys mock-duel in the grass, enthusiastically wielding iron rods. Sentence 2: The duo stage a sword fight outdoors and enjoy themselves.
Output: The boys rough-house in the backyard without a care in the world.
Explanation: The boys may not really be rough-housing depending on their level of skill. Additionally, we can't confirm if they are in their backyard. The ambiguity involved ascertaining if the new sentence agrees with sentence 1 or not means that the given alteration correctly exemplifies the task.

Modify Task Instruction — T1 — Definition

In this task, you're given a pair of sentences 1 and 2, that undisputedly agree with each other. Please modify sentence 2 so that agreement between the sentences is now uncertain. Stick to sentence lengths within 15 words. New information can be introduced. Avoid using pronouns to confuse the subject of the sentence.

Figure 5: *The interactions taken during the Usage Scenario described in §7.*

(1) Owen first searches for instructions corresponding to TM using the overview panel. He notices that TM instructions show large embedding distances **(1.1)**, with the formation of local clusters if instructions use the same source dataset or domain **(1.2, 1.3)**. This is in contrast to other tasks like translation, where a majority of points are clustered in the same region across source datasets **(1.4)**. Owen selects a TM root instruction, originally from the SNLI source dataset [BAPM15] for further analysis. The root instruction is colored red (T1), and the 9 most highly correlated instructions (T2-T10) are colored in shades of blue.

(2) In the Correlation View, Owen sets the threshold for links to 0.7. He notes the variation in correlated instructions' task domains, types, and sources (see Table 4). On further examination, Owen realizes that all the TM instructions use the same SNLI source dataset as T1. Another interesting instruction is T8. It has links to T1 and T3, and belongs to the same task domain: Text Entailment. T2–T4 use nearly verbatim definitions in task instructions when compared mutually, as well as against T1, as highlighted in Figure 5.

(3) Owen decides to compare the word overlap between the positive examples of T1–T10 (chord threshold: 0.6), in order to understand why different task types are highly correlated. He finds that between T1–T4, positive examples show high word overlap **(3.1)**; on examining the text, they are seen to only marginally differ in explanation. T6 and T8 use similar explanation phrasing to T1–T4 for positive examples **(3.2)**. Similarly, word overlaps between the T5, T9 and T10 examples arise in the explanations of the positive examples **(3.3)**.

Task ID	Source Dataset	Task Type	Task Domain	Correlation [0, 1]
T1	**SNLI**	**Text Modification**	**Text Entailment**	**1.000**
T2	SNLI	Text Modification	Text Entailment	0.920
T3	SNLI	Text Modification	Text Entailment	0.919
T4	SNLI	Text Modification	Text Entailment	0.911
T5	ROC STORIES	Sentence Ordering	Deductive	0.858
T6	XCOPA$^+$	Classification	Causal	0.850
T7	ASSET	Text Simplification	Commonsense	0.842
T8	SHERLIIC	Text Entailment	Text Entailment	0.838
T9	ROC STORIES	Text Completion	Commonsense	0.836
T10	ROC STORIES	Text Completion	Commonsense	0.836

Table 4: *Tasks with most highly correlated instructions, with respect to a selected root instruction (T1). + : non-english input.*

(4) On observing the GPT-3 results for task instances from T1–T10, Owen notes a general trend across all the datasets: task instances in bins with higher word similarity (> 0.7) to the instruction examples are more accurately solved. Owen confirms that T1–T4 and T7 show similar performance distributions, with most of the high-performing task instances being in high similarity bins. T5 and T8–T10 have a greater proportion of task instances located in bins ranging from [0.55-0.75]; these bins achieve ∼20-35% accuracy. T6 is an outlier with a more uniform distribution of task instances in similarity bins, all ranging from ∼20-30% accuracy. Owen partially attributes this to T6 containing non-English input. Owen then juxtaposes top ranked task instances belonging to bins [0.85-0.90] against instruction examples for T1-T4. He notes that these task instances follow similar topics to the examples, and are solved using similar output patterns, as highlighted in Figure 5(**5**).

(5) Owen checks the unique vocabulary contributed by examples in each task instruction **(5.1)**. He finds that T6 and T7 contribute the highest unique vocabulary, while T1–T4 contribute the lowest. T6 and T7 are also the hardest for the model to solve (as previously seen in **(4)**). Next, Owen examines the Jaccard Similarity of adverbs across full task instructions compared against their respective task instances **(5.2)**. Here, T5–T7 show the lowest values, which indicates higher variation in the data. Owen can partially attribute this to non-repetition of phrases from the definition in the example explanations.

(6) Finally, Owen modifies the task instruction for T1, in order to (i) reduce the similarity bias between instruction examples and task instances, and (ii) reduce overlap between the instruction definition and instruction example explanations. He updates the definition and replaces a positive example, so that the instruction now contributes a higher proportion of vocabulary and adverbs **(6.1)**. Owen notices that the beeswarm updates **(6.2)** for T1 to show decreased overall performance, with fewer samples in similarity bins > 0.9 as well as fewer samples crossing 80% accuracy. Additionally, T1 now exhibits lower Jaccard similarity **(6.3)** and higher unique vocabulary **(6.4)**. Overall, Owen's modifications achieve a decrease in accuracy, Jaccard similarity for words, POS-tags, and n-grams, and an increase in unique vocabulary. Hence by iteratively changing the task instruction (for instance, replacing more examples) Owen can further reduce instruction bias for T1, and create a more difficult task instruction for the model to solve.

8. User Study

To further evaluate LINGO from an empirical perspective, we conducted an in-person qualitative usability study composed of both novice and expert prompt instruction creators, using the SUPNATINST dataset. The study utilized think-aloud protocol, as well as a post-study questionnaire and feedback session about the experience of participants using LINGO. This allowed us to robustly understand how the insights afforded by LINGO (as well as its overall usability) compared both for novice users and domain experts.

Participants began the study by completing a short survey to collect demographic information and background knowledge. After this, three stages were run:

Training. The participants were first given an overview and usage scenario (with text modification tasks) of LINGO's functionality, and could ask questions and play with the system until they feel confident to proceed. Study participants were familiarized with using the visual idioms, interactions, and color mappings (since explicit legends are not provided in the interface).

Exploration. Participants were then allowed to freely explore the system for up to 30 minutes. To constrain the study design, we limited participants to analysis of question-answering task instructions. During this stage, participants use think aloud protocol to verbalize their thought processes and actions [FKG93].

Review. Participants then completed a short feedback survey about the system's affordances and user experience using Likert scale ratings (1 – strongly disagree, 7 – strongly agree). Participants also had the opportunity to provide freeform comments, suggestions, and criticisms about the interface and their experience.

Participants. We recruit 16 participants (age μ = 25.46 year, σ = 3.04, 10 males and 6 females) studying Computer Science from <Anonymous University>. All participants were proficient in English and had normal (corrected if necessary) eyesight. Participants reported familiarity with instruction authoring based on a 7-point Likert scale: μ = 4.26 (σ = 2.14). All study participants reported relative non-expertise with data visualization (familiarity: μ = 2.33, σ = 1.68). LINGO was displayed in Google Chrome on a 24-inch monitor (3840 × 2160 resolution) with keyboard and mouse, connected to a MacBook Pro running macOS Monterey. QuickTime Player recorded session screencasts and audio. We classified users as novices (familiarity: 1-3, 7 users: n1–n7, μ = 2.38, σ = 0.61) or experts (familiarity: 4-7, 9 users: e1–e9, , μ = 5.41, σ = 1.72) for the purpose of post hoc analysis. None of the novice users had previously taken a graduate course in NLP or authored instructions; two had previously participated in NLP benchmark crowdsourcing. On average, novice users spent 19.37 minutes (σ = 3.44 minutes) and expert users spent 25.21 minutes (σ = 1.92 minutes) using LINGO.

8.1. System Ratings from Survey Response

Figure 6 summarizes study results from the quantitative survey. (Note: All results are found to have $p \leq 0.02$ based on running a t-test.) This figure aggregates both novice and expert responses, as general trends remained consistent for both user groups. Overall system ratings (Q1—Q11) were generally positive, including that it is easy to learn, use, and comprehend (Q10–Q3), it supports meaningful analysis (Q5-–Q9), and encourages participants to think about instruction bias during prompt authoring and crowdsourcing (Q10—Q11). Responses about specific interfaces features (Q12—Q21) indicate that they are also generally well received.

8.2. Support for User Insights

Based on the recorded think-aloud utterances by our seven novice and expert participants (n1–n7 and e1–9), as well as freeform feedback collected during the review stage, we performed an open coding technique based on grounded theory [WM06] to understand the types of support, insights, and user experiences that LINGO specifically promoted to our study participants. We summarize four key takeaways below.

For novices, the interface was paradoxically both easy to use and too complex. We received several comments from novice participants about LINGO being *"easy to use"*(n6,n5) and *"cohesive and easily navigable pipeline"*(n1,n7). *"I like how you can easily focus on different sub-components of the instruction and that changes how its relationships look... [it] helps me streamline my analysis"* (n7). Paradoxically, other users sometimes felt overwhelmed by the system's available features, such as the number

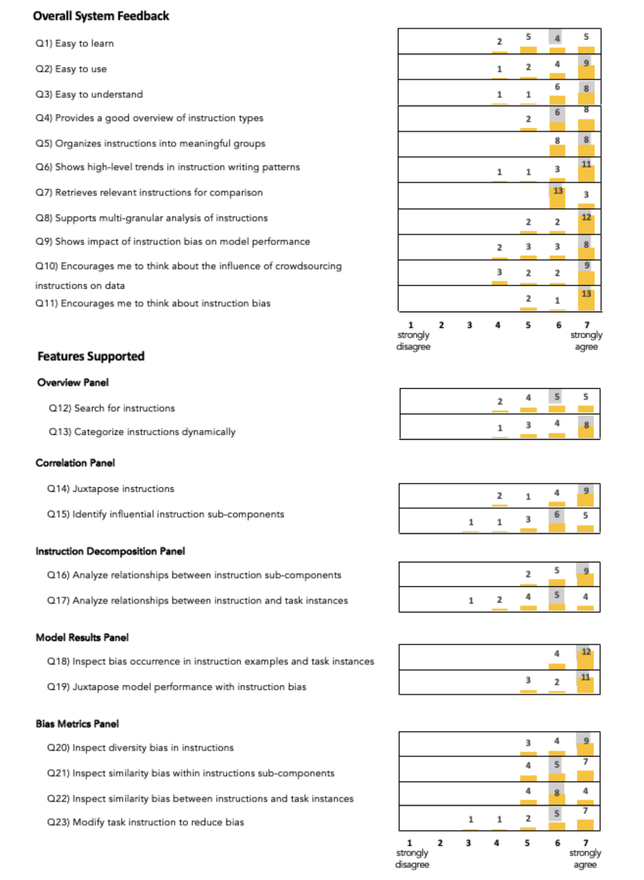


Figure 6: *Overall ratings from the post-study survey about various system aspects. Median ratings are indicated in grey.*

of bias metric choices that were available. *"I'm keeping up until the model results, it's difficult to juxtapose them with the bias metrics"* (n3). *"The chord diagram was new to me, though I got the hang of it after a couple of minutes"* (n1). Multiple users thought that follow-up sessions or helper widgets would lead to more efficient analysis: *"I'd like some more time to go through the tasks, so I understand how to use the bias metrics to change the instruction"* (n3,n2). *"[S]omething like a widget that tells you in plain text what the high-level impact of this bias metric is... helps clarify how to translate bias identification to instruction modification"* (n3,n4). This mixed feedback is likely due to a lack of domain knowledge by some novices, which hampered their user experience.

Novices echoed experts when defining instruction bias post-study. Many of the novice and expert participants agreed in conjecturing that instruction bias likely arises when: (i) *"the examples are all too similar amongst themselves"* (n1,n2,e4), (ii) *"the example explanations reiterate the definition exactly, without new contextual information"* (n3,e9), (iii) *"the beeswarm is top-heavy, meaning lots of high similarity instances exist and are correctly solved"* (n7,e2), and (iv) *"if one person creates multiple instructions for a dataset, they are probably following the same patterns throughout...they use the same words, phrases, and topics for examples"* (n3, e8). LINGO is therefore able to support effective and accurate analysis by both novel and expert users.

The beeswarms were effective for recognizing how instruction bias inflates model performance. Several participants appreciate the beeswarm representation of task instances as being *"... an intuitive way to represent a large amount of data"*(e1,e3,n2,n5). *"I like that you can drill down and see representative task instances... compare them with the examples"*(n4,e6). Participants also found it useful to use the beeswarm and chord diagrams to filter out which bias metric and tasks to focus on. *"I decided that these [four tasks] have a lot of mutual connections and the beeswarms are top-heavy, so I'm going to check the Jaccard similarities for those tasks"* (e2). *"See how Task 6 has low accuracy but Task 3 has high accuracy? Let's check the unique vocabulary contribution... I was right [Task 6] has it higher* (e8). *"If you look at sentence lengths, you can see longer instructions with shorter example explanations seem to have better performance"* (n1). Several experts also found in situ model results useful while modifying instructions–*"... [results] ground how I'm changing the instruction. I can see what does and does not work to reduce instruction bias, and further ensure I still have sufficient inductive bias for the model."* (e2) and *"I can see this being applicable to analyze bias any sort of benchmark data. We have the human, model, and metrics all simultaneously involved"* (e4).

Participants are able to identify potential caveats to reducing instruction bias. Several participants pointed out that reducing instruction bias requires a learning curve in identifying what constitutes a positive change to reduce model performance. *"I thought putting lots of synonyms in would help...I guess I need to restructure the definition completely."* (n4,n5). However, participants also recognized that the nature of changes might lead to unnatural language. *"If I keep paraphrasing [this way] to reduce model performance, I've got a shortcut to doing the bias reduction... but the instruction looks strange now so this wouldn't happen in the real-world"* (n1,e2). A couple of participants also questioned the overall effect of the changed instruction on dataset bias: *"I could be making a only a local optima... my highest correlated instruction group will change."* (e3,n4).

9. Discussion

Based on our conducted usage scenario (§7) and user study (§8), LINGO robustly supports **DR1–DR5** described in §3. Here, we discuss several takeaways from the design, implementation, and evaluation of LINGO, to synthesize generalizable actionable insights for visualization and NLP communities.

Novice users recognize broad sources of instruction bias; experts can see research possibilities. While both of our user study groups could use LINGO to analyze task instructions, one significant takeaway from the study was that the types of insights differed based on the user's experience. Novice users could identify instances were instruction bias exists, but generally could only make broad conclusions about bias. Experts, already familiar with instruction bias and its impact, could leverage LINGO to explore deeper nuances about bias. Subsequently, their instruction modifications were generally more effective in reducing bias. As the pre-study in §3 did not explicitly consider various levels of user expertise, such a result is not wholly unexpected. Future analysis systems for novices can be tailored to accommodate their lack of domain expertise.

Explicitly flagging bias in-text while authoring. Several novices noted that instruction modification proved challenging, given the potential that a modification could introduce new types of instruction bias; several requested that bias be explicitly highlighted in text. While this is possible, there is a risk that ex ante labeling could render such users dependent on the classification outputs of a model or annotator, again constraining the patterns they might follow during instruction authoring and data creation. A possible strategy might be to allow users to define and highlight patterns of interest during analysis, that will be subsequently flagged when they modify an instruction, but we leave this as future work.

The need for visualization tools in NLP Expert participants in our study were long-time researchers (+ 4 years) in NLP, but *none had previously encountered or used visual analytics tools focusing on NLP text bias*. While many backend algorithms and text-highlighting driven interfaces catering to bias identification and resolution have previously been published in the NLP community (e.g., [KHL]), there is a lack of standardization and adoption. In view of this, tools like LINGO can simplify the overhead involved in bias identification post-creation, by providing an in situ pipeline to procedurally target bias during text authoring.

Designing for non-expert visualization users. Even though our user study participants were non-experts in visualization, they were still able to use the system to probe task instructions and glean insights to reduce instruction bias. LINGO's focus on selecting and utilizing familiar visual techniques therefore provides a cohesive pipeline for instruction analysis, that facilitates a lower learning curve. Particularly, the provision of visually distinct panels to analyze different bias aspects reduces the potential for users to conflate/become overwhelmed by different choices for bias analysis, e.g.: users shortlist which aspect of bias to drill down on for instruction modification, despite experiencing visual novelty, by using panels in tandem. LINGO further motivates the necessity to standardize bias quantification and elimination in NLP. This is a non-trivial problem, but we believe visualization can likely provide an important step in the process, by helping researchers identify where the principal issues are present.

System Limitations and Future Work In particular, the user study helped us to identify several areas where LINGO (or tools like LINGO) could be improved in or expanded to the future. In particular, we note three here:

(1) Recommendations for Instruction Modification: Instruction modification was at times difficult for novice authors, given that instructions must retain sufficient inductive bias for the model to learn after modification. Particularly, for tasks like custom text generation, specialized contextual information must be provided in the instruction. For instance, in order to create instructions for a model to write the bio of a user, the user must provide sufficient personal information. The authoring/modification of such task instructions is therefore a complex process. Approaches like [MN22] involve prompting models to ask relevant questions to reduce the cognitive load on the user, in identifying key information specific to a particular task type. Such approaches can be extrapolated to provide recommendations during instruction modification that are tailored to reducing the level of bias present in an instruction. Such bias reduction techniques could therefore be less likely to introduce new types of bias. Another relevant approach to author instructions with richer contextual information is to provide the language model with multiple variants of the same instruction at once, which define the task from different perspectives [PMPB22]. This has proven useful for improving model performance, particularly on low-data tasks.

(2) Expanding Evaluation: While LINGO successfully incorporates human, model, and metric-based feedback in our analysis paradigm, the model and metric-based feedback is restricted by the chosen domain. Instruction bias examination necessitates the use of PLMs capable of accepting instructions as input, along with a limited number of task-agnostic metrics to gauge bias. Future studies can broaden the scope of LINGO's analysis to quantitatively identify textual data bias at scale, for both instructions and NLP benchmarks, across a wider array of language models and with customized bias metrics [MABB20, MA21a]; LINGO is easily extensible to integrating new metrics in either the front/backend to better support expert users. By quantitatively evaluating "data quality" in this manner, it may be possible to establish more standardized procedures for bias quantification and resolution, leveraging visualization for analysis and verification.

(3) Extensibility to Prompting for Multimedia Data LINGO focuses on text-based task instructions, covering diverse applications, that do not involve images and are purely based on text data. A key challenge in extending LINGO to analyze prompts for multimedia data is the potential lack of human-interpretable bias features. For instance, "concepts" (a group of pixels that express a meaningful notion in an image) are an explanation technique for image classification [KWG*18]. These could potentially be useful in reconciling a text prompt with a produced image. However, additional explainability techniques produce their own complexities. For example, image generation models might heavily rely on bias between n-grams in an input prompt phrase and a highly abstract concept in a produced image, and such bias might not be easily understandable to humans. Recent papers (e.g., [HMKB22]) explore visualization-drive strategies for analyzing these sorts of algorithms, and their techniques could be incorporated into tools like LINGO that expand to additional multimedia-based prompting.

10. Conclusion

We present LINGO, a novel visual analytics interface that supports an effective, task-driven workflow to (1) help identify bias in natural language task instructions, (2) alter (or create) task instructions to reduce bias, and (3) evaluate pre-trained model performance on debiased task instructions. Our evaluations demonstrate how LINGO promotes the creation of more difficult tasks for pre-trained language models that contain higher linguistic diversity and lower instruction bias. We also discuss how visual analytics tools like LINGO can reduce the effort involved in prompt creation, including with constraints like increasing task diversity and lowering instructional bias, across additional domains.

11. Acknowledgements

This research was supported in part by the U.S. National Science Foundation through grant DUE-2216452.

References

[AGS*21] Aghajanyan A., Gupta A., Shrivastava A., Chen X., Zettlemoyer L., Gupta S.: Muppet: Massive multi-task representations with pre-finetuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (2021), pp. 5799–5811. 1

[AIP15] Attenberg J., Ipeirotis P., Provost F.: Beat the machine: Challenging humans to find a predictive model's "unknown unknowns". *Journal of Data and Information Quality (JDIQ) 6*, 1 (2015), 1–17. 3

[BAPM15] Bowman S. R., Angeli G., Potts C., Manning C. D.: The snli corpus. 2, 7

[BMR*20] Brown T., Mann B., Ryder N., Subbiah M., Kaplan J. D., Dhariwal P., Neelakantan A., Shyam P., Sastry G., Askell A., et al.: Language models are few-shot learners. *Advances in neural information processing systems 33* (2020), 1877–1901. 1, 2, 5

[BSY*22] Bach S., Sanh V., Yong Z. X., Webson A., Raffel C., Nayak N. V., Sharma A., Kim T., Bari M. S., Févry T., et al.: Promptsource: An integrated development environment and repository for natural language prompts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (2022), pp. 93–104. 3

[CEH*19] Cabrera Á. A., Epperson W., Hohman F., Kahng M., Morgenstern J., Chau D. H.: Fairvis: Visual analytics for discovering intersectional bias in machine learning. In *2019 IEEE Conference on Visual Analytics Science and Technology (VAST)* (2019), IEEE, pp. 46–56. 3

[CMJ*20] Chatzimparmpas A., Martins R. M., Jusufi I., Kucher K., Rossi F., Kerren A.: The state of the art in enhancing trust in machine learning models with the use of visualizations. In *Computer Graphics Forum* (2020), vol. 39, Wiley Online Library, pp. 713–756. 3

[CND*22] Chowdhery A., Narang S., Devlin J., Bosma M., Mishra G., Roberts A., Barham P., Chung H. W., Sutton C., Gehrmann S., et al.: Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311* (2022). 2

[DWD*19] Dua D., Wang Y., Dasigi P., Stanovsky G., Singh S., Gardner M.: DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (Minneapolis, Minnesota, June 2019), Association for Computational Linguistics, pp. 2368–2378. URL: https://aclanthology.org/N19-1246, doi:10.18653/v1/N19-1246. 3

[EL20] Efrat A., Levy O.: The turking test: Can language models understand instructions? *arXiv preprint arXiv:2010.11982* (2020). 2

[FKG93] Fonteyn M. E., Kuipers B., Grobe S. J.: A description of think aloud method and protocol analysis. *Qualitative health research 3*, 4 (1993), 430–441. 8

[GAB*20] Gardner M., Artzi Y., Basmov V., Berant J., Bogin B., Chen S., Dasigi P., Dua D., Elazar Y., Gottumukkala A., et al.: Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020* (2020), pp. 1307–1323. 3

[GGB19] Geva M., Goldberg Y., Berant J.: Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (Hong Kong, China, Nov. 2019), Association for Computational Linguistics, pp. 1161–1166. URL: https://aclanthology.org/D19-1107, doi:10.18653/v1/D19-1107. 2, 3

[GSL*18] Gururangan S., Swayamdipta S., Levy O., Schwartz R., Bowman S., Smith N. A.: Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)* (New Orleans, Louisiana, June 2018), Association for Computational Linguistics, pp. 107–112. URL: https://aclanthology.org/N18-2017, doi:10.18653/v1/N18-2017. 3, 4

[HMKB22] Huang J., Mishra A., Kwon B. C., Bryan C.: Conceptexplainer: Interactive explanation for deep neural networks from a concept perspective. *IEEE Transactions on Visualization and Computer Graphics* (2022). 10

[HSG*21] Hettiachchi D., Sanderson M., Goncalves J., Hosio S., Kazai G., Lease M., Schaekermann M., Yilmaz E.: Investigating and mitigating biases in crowdsourced data. In *Companion Publication of the 2021 Conference on Computer Supported Cooperative Work and Social Computing* (2021), pp. 331–334. 2, 3

[KBN*21] Kiela D., Bartolo M., Nie Y., Kaushik D., Geiger A., Wu Z., Vidgen B., Prasad G., Singh A., Ringshia P., et al.: Dynabench: Rethinking benchmarking in nlp. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2021), pp. 4110–4124. 3

[KHL] Kaushik D., Hovy E., Lipton Z.: Learning the difference that makes a difference with counterfactually-augmented data. In *International Conference on Learning Representations*. 3, 10

[KMK*20] Khashabi D., Min S., Khot T., Sabharwal A., Tafjord O., Clark P., Hajishirzi H.: Unifiedqa: Crossing format boundaries with a single qa system. In *Findings of the Association for Computational Linguistics: EMNLP 2020* (2020), pp. 1896–1907. 1

[KSHL20] Kim H., So B.-H., Han W.-S., Lee H.: Natural language to sql: where are we today? *Proceedings of the VLDB Endowment 13*, 10 (2020), 1737–1750. 2

[KWG*18] Kim B., Wattenberg M., Gilmer J., Cai C., Wexler J., Viegas F., et al.: Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning* (2018), PMLR, pp. 2668–2677. 10

[LB02] Loper E., Bird S.: Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics* (2002), pp. 63–70. 5

[LBSB*20] Le Bras R., Swayamdipta S., Bhagavatula C., Zellers R., Peters M., Sabharwal A., Choi Y.: Adversarial filters of dataset biases. In *International Conference on Machine Learning* (2020), PMLR, pp. 1078–1088. 3

[LLV18] Li Y., Li Y., Vasconcelos N.: Resound: Towards action recognition without representation bias. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2018), pp. 513–528. 3

[LML*19] Larson S., Mahendran A., Lee A., Kummerfeld J. K., Hill P., Laurenzano M. A., Hauswald J., Tang L., Mars J.: Outlier detection for improved data quality and diversity in dialog systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (2019), pp. 517–527. 3, 4

[LMT*20] Lavalle A., Maté A., Trujillo J., et al.: An approach to automatically detect and visualize bias in data analytics. 3

[LMTT] Lavalle A., Maté A., Trujillo J., Teruel M. A.: A methodology, based on user requirements and visualization techniques, to evaluate the impact of bias in data analytics and artificial intelligence algorithms. *Based on User Requirements and Visualization Techniques, to Evaluate the Impact of Bias in Data Analytics and Artificial Intelligence Algorithms*. 3

[LV19] Li Y., Vasconcelos N.: Repair: Removing representation bias by dataset resampling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2019), pp. 9572–9581. 3

[LYF*23] Liu P., Yuan W., Fu J., Jiang Z., Hayashi H., Neubig

G.: Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys 55*, 9 (2023), 1–35. 2

[LZM*20] LARSON S., ZHENG A., MAHENDRAN A., TEKRIWAL R., CHEUNG A., GULDAN E., LEACH K., KUMMERFELD J. K.: Iterative feature mining for constraint-based data collection to increase data diversity and model robustness. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2020), pp. 8097–8106. 4

[MA21a] MISHRA S., ARUNKUMAR A.: How robust are model rankings : A leaderboard customization approach for equitable evaluation. *Proceedings of the AAAI Conference on Artificial Intelligence 35*, 15 (May 2021), 13561–13569. URL: https://ojs.aaai.org/index.php/AAAI/article/view/17599, doi:10.1609/aaai.v35i15.17599. 2, 10

[MA21b] MISHRA S., ARUNKUMAR A.: How robust are model rankings: A leaderboard customization approach for equitable evaluation. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2021), vol. 35, pp. 13561–13569. 3

[MABB20] MISHRA S., ARUNKUMAR A., BRYAN C., BARAL C.: Our evaluation metric needs an update to encourage generalization. *ArXiv abs/2007.06898* (2020). 10

[MKBH21] MISHRA S., KHASHABI D., BARAL C., HAJISHIRZI H.: Cross-task generalization via natural language crowdsourcing instructions. *arXiv preprint arXiv:2104.08773* (2021). 1, 2, 3, 5

[MN22] MISHRA S., NOURI E.: Help me think: A simple prompting strategy for non-experts to create customized content with models. *ArXiv abs/2208.08232* (2022). 10

[MON] MONGODB databases and collections. https://www.mongodb.com/docs/manual/core/databases-and-collections/. 5

[MXLM19] MA Y., XIE T., LI J., MACIEJEWSKI R.: Explaining vulnerabilities to adversarial machine learning through visual analytics. *IEEE transactions on visualization and computer graphics 26*, 1 (2019), 1075–1085. 3

[NWD*20] NIE Y., WILLIAMS A., DINAN E., BANSAL M., WESTON J., KIELA D.: Adversarial nli: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (2020), pp. 4885–4901. 3

[PMGB22] PARMAR M., MISHRA S., GEVA M., BARAL C.: Don't blame the annotator: Bias already starts in the annotation instructions. *arXiv preprint arXiv:2205.00415* (2022). 2, 3

[PMPB22] PURI R. S., MISHRA S., PARMAR M., BARAL C.: How many data samples is an additional instruction worth? *arXiv preprint arXiv:2203.09161* (2022). 10

[PNH*18] POLIAK A., NARADOWSKY J., HALDAR A., RUDINGER R., VAN DURME B.: Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics* (2018), pp. 180–191. 3, 4

[PNI*18] PETERS M. E., NEUMANN M., IYYER M., GARDNER M., CLARK C., LEE K., ZETTLEMOYER L.: Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* (New Orleans, Louisiana, June 2018), Association for Computational Linguistics, pp. 2227–2237. URL: https://aclanthology.org/N18-1202, doi:10.18653/v1/N18-1202. 1

[RGA23] ROGERS A., GARDNER M., AUGENSTEIN I.: Qa dataset explosion: A taxonomy of nlp resources for question answering and reading comprehension. *ACM Computing Surveys 55*, 10 (2023), 1–45. 1

[RM21] REYNOLDS L., MCDONELL K.: Prompt programming for large language models: Beyond the few-s hot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems* (2021), pp. 1–7. 2

[ROU04] ROUGE L. C.: A package for automatic evaluation of summaries. In *Proceedings of Workshop on Text Summarization of ACL, Spain* (2004). 5

[SMM12] SEDLMAIR M., MEYER M., MUNZNER T.: Design study methodology: Reflections from the trenches and the stacks. *IEEE transactions on visualization and computer graphics 18*, 12 (2012), 2431–2440. 4

[SS21] SCHICK T., SCHÜTZE H.: Few-shot text generation with natural language instructions. Association for Computational Linguistics. 2

[SSK*17] SCHWARTZ R., SAP M., KONSTAS I., ZILLES L., CHOI Y., SMITH N. A.: The effect of different writing tasks on linguistic style: A case study of the roc story cloze task. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)* (2017), pp. 15–25. 3

[STG*20] SHRIDHAR M., THOMASON J., GORDON D., BISK Y., HAN W., MOTTAGHI R., ZETTLEMOYER L., FOX D.: Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2020), pp. 10740–10749. 2

[SVN*21] SUHR A., VANIA C., NANGIA N., SAP M., YATSKAR M., BOWMAN S., ARTZI Y.: Crowdsourcing beyond annotation: Case studies in benchmark data collection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts* (2021), pp. 1–6. 2

[SYH20] STASASKI K., YANG G. H., HEARST M. A.: More diverse dialogue datasets via diversity-informed data collection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (2020), pp. 4958–4968. 3, 4

[TYLB*] TALMOR A., YORAN O., LE BRAS R., BHAGAVATULA C., GOLDBERG Y., CHOI Y., BERANT J.: Commonsenseqa 2.0: Exposing the limits of ai through gamification. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*. 3

[TZD*] TRIANTAFILLOU E., ZHU T., DUMOULIN V., LAMBLIN P., EVCI U., XU K., GOROSHIN R., GELADA C., SWERSKY K., MANZAGOL P.-A., ET AL.: Meta-dataset: A dataset of datasets for learning to learn from few examples. In *International Conference on Learning Representations*. 2

[Van19] VANDENHOF C.: A hybrid approach to identifying unknown unknowns of predictive models. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing* (2019), vol. 7, pp. 180–187. 3

[WBZ*] WEI J., BOSMA M., ZHAO V., GUU K., YU A. W., LESTER B., DU N., DAI A. M., LE Q. V.: Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*. 2, 3

[WLGP20] WELLER O., LOURIE N., GARDNER M., PETERS M. E.: Learning from task descriptions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2020), pp. 1361–1375. 2

[WM06] WALKER D., MYRICK F.: Grounded theory: An exploration of process and procedure. *Qualitative health research 16*, 4 (2006), 547–559. 8

[WMA*22] WANG Y., MISHRA S., ALIPOORMOLABASHI P., KORDI Y., MIRZAEI A., ARUNKUMAR A., ASHOK A., DHANASEKARAN A. S., NAIK A., STAP D., ET AL.: Benchmarking generalization via in-context instructions on 1,600+ language tasks. *arXiv preprint arXiv:2204.07705* (2022). 1, 3, 4

[WRF*19] WALLACE E., RODRIGUEZ P., FENG S., YAMADA I., BOYD-GRABER J.: Trick me if you can: Human-in-the-loop generation of adversarial examples for question answering. *Transactions of the Association for Computational Linguistics 7* (2019), 387–401. 3, 4

[WXC*20] WANG Q., XU Z., CHEN Z., WANG Y., LIU S., QU H.: Visual analysis of discrimination in machine learning. *IEEE Transactions on Visualization and Computer Graphics 27*, 2 (2020), 1470–1480. 3

[YZFBC*20] YAGHOUB-ZADEH-FARD M.-A., BENATALLAH B., CASATI F., BARUKH M. C., ZAMANIRAD S.: Dynamic word recommendation to obtain diverse crowdsourced paraphrases of user utterances. In *Proceedings of the 25th International Conference on Intelligent User Interfaces* (2020), pp. 55–66. 3, 4