LabelVizier: Interactive Validation and Relabeling for Technical Text Annotations

Xiaoyu Zhang*† University of California, Davis
Xiwei Xuan*† University of California, Davis
Alden Dima‡ National Institute of Standards and Technology
Thurston Sexton‡ National Institute of Standards and Technology
Kwan-Liu Ma* University of California, Davis

2023 IEEE 16th Pacific Visualization Symposium (PacificVis) | 979-8-3503-2124-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/PacificVis56936.2023.00026

ABSTRACT

With the rapid accumulation of text data produced by data-driven techniques, the task of extracting "data annotations"—concise, high-quality data summaries from unstructured raw text—has become increasingly important. The recent advances in weak supervision and crowd-sourcing techniques provide promising solutions to efficiently create annotations (labels) for large-scale technical text data. However, such annotations may fail in practice because of the change in annotation requirements, application scenarios, and modeling goals, where label validation and relabeling by domain experts are required. To approach this issue, we present *LabelVizier*, a human-in-the-loop workflow that incorporates domain knowledge and user-specific requirements to reveal actionable insights into annotation flaws, then produce better-quality labels for large-scale multi-label datasets. We implement our workflow as an interactive notebook to facilitate flexible error profiling, in-depth annotation validation for three error types, and efficient annotation relabeling on different data scales. We evaluated our workflow in assisting the validation and relabelling of technical text annotation with two use cases and four expert reviews. The results show that *LabelVizier* is applicable in various application scenarios, and users with different knowledge backgrounds have diverse preferences for the tool usage.

Keywords: Workflow Design, Technical Language Processing, Data Annotation, Model Interpretation

1 INTRODUCTION

Data-driven approaches have pervaded manufacturing in the age of Industry 4.0, producing a large amount of digitized data in the form of unstructured technical text [44]. For example in machine maintenance, machine operators and repairing technicians often create maintenance work orders (MWOs) to record their maintenance activities. However, the rich text of asset management history in MWOs usually sits untouched because of the potential inconsistency, incompleteness, or incorrectness [4] in the descriptive text. Compared to raw unstructured text, a set of high-quality annotations summarizing the content is preferred for "robust and reproducible" [4] analysis of large-scale technical text. In particular, these annotations can be utilized for the systematic problem identification and classification, root cause analysis, and product life cycle prediction [18], which provides precious insights and facilitate the key performance index (KPI) assessment and budget planning process. For instance, the statistics of the label "*too_hot*" in a heating, ventilation, and air conditioning (HVAC) system maintenance log dataset (see Sec. 6.1) could indicate how well the air conditioning system has been maintained and thus inform maintenance budget planning. This is also a critical research topic in technical language processing (TLP) [14].

However, it is not easy to create quality annotations and many important annotated datasets are riddled with labelling errors [38]. Given the exponentially increasing volume of unstructured text, researchers have gradually discarded conventional manual annotation approaches and turned to more efficient state-of-the-art machine learning (ML) techniques or commercial crowd-sourcing [26] platforms. Particularly, recent advances in weak supervision [41, 50] promise efficient large-scale text annotation. However, it is necessary to sufficiently validate and improve the annotations generated by such methods before delivering them to down-streaming tasks. Limited research efforts have been devoted to validation and relabeling of such large-scale technical text annotation. To facilitate this process, we developed *LabelVizier*, a human-in-the-loop workflow encapsulated as a visual analytic solution that supports reliable and efficient annotation validation and relabelling for domain practitioners to meet their specific application requirements.

LabelVizier helps identify and correct three types of annotations errors: (1) duplicate, (2) wrong, and (3) missing labels. Inspired by the practice of debugging in software engineering, we profile the potential errors in the existing labels and devise visual analytics procedures to facilitate an efficient skimming of the labels and their context based on the domain expert's annotation preferences. We supplement this validation process by training a surrogate model to approximate the agnostic annotation process, visualizing the prediction metadata to expose potential errors, and providing LIME explanations [42] for root cause analysis. For the user-identified annotation errors, we support flexible relabelling of the dataset on the corpus, sub-group, and record levels. We implement this workflow as a web-based interactive notebook containing editable function blocks and an interactive visual analytic interface designed in close collaboration with the two domain experts on our team. We demonstrate how *LabelVizier* can benefit different application scenarios in two use cases and evaluate them with expert reviews from four domain practitioners. The results show that the domain experts appreciated the efficiency and accessibility of *LabelVizier* and are interested in using *LabelVizier* for their text-based analysis tasks. This work has the potential to impact a number of data-driven fields that emphasize annotation quality and, in particular, benefit multidisciplinary areas that deal with critical problems such as maintaining the vital infrastructure and ensuring community resilience.

Our main contributions can be summarized as follows:

- We propose a human-in-the-loop workflow that supports domain practitioners to efficiently conduct validation and relabeling tasks for large-scale technical text annotations from weak supervision.
- We encapsulate this workflow as a web-based interactive notebook with a visual analytic interface that facilitates the identification of annotation errors and relabeling for different scales of data.
- We collect insights from domain experts in different application domains and observe various preferences corresponding to their backgrounds, which shed light on directions for improving *LabelVizier* and cater to the needs of diverse domain practitioners.

*Equally contributed.
†e-mail: {xybzhang, xwxuan, klma}@ucdavis.edu
‡e-mail: {alden.dima, thurston.sexton}@nist.gov

2165-8773/23/$31.00 ©2023 IEEE
DOI 10.1109/PacificVis56936.2023.00026

2 Related Work

2.1 Technical Language Processing

Process monitoring, diagnostics, and prognostics have gained prevalence with the increased emphasis on smart manufacturing, and reduced machine downtime. This trend—coupled with lower cost, more accessible sensors and data storage solutions—has increased the volume of maintenance data [5]. Despite the potential benefits, companies frequently struggle to adopt advanced manufacturing technologies due to cost of and lack of technical expertise in data analysis [27]. Simple yet powerful solutions for data analysis are necessary to aid manufacturers improve their practices. There has been an increasing focus on sensor data and predictive maintenance using AI techniques [6, 47]. However, these works often neglect a large part of maintenance data: natural language contained short-text maintenance logs, which leads organizations to turn to NLP.

Technical text, however, poses challenges to commonly used NLP methods. Technical fields are often low-resource settings from an NLP perspective; they lack available resources such as annotated data and algorithms appropriate for specific analyses [15]. Transfer learning is the traditional strategy for addressing low-resource domains in machine learning [14]. Models that were generated from annotated data from resource-rich domains are adapted for the low-resource domain. Transfer learning approaches often assume limited differences between two different domains. But the technical text that appears in industrial information systems deviates considerably from "standard" English [14], full of expressions like "1 W Mech Insp Ball Mill BM001" and "DSHT Cons Thkner rplace bed press".

The lexical, grammatical, and terminological differences between "standard" English and industrial technical text have spawned bespoke domain-specific NLP adaptations that are largely outside of mainstream NLP [14]. TLP is a human-in-the-loop, iterative approach that addresses perceived shortcomings of applying standard NLP (natural language processing) to technical text data [14]. Originating with manufacturing maintenance, it is an adaptation of NLP that focuses on the technical text communicated within specialized domains. TLP emphasizes the practical importance of semantic information and extends its system boundaries beyond algorithms and pipelines to include human input and community resources [4]. The short-text from maintenance work orders (MWOs) are important analysis corpora for TLP [34, 45]. They record in detail the maintenance history of equipment and collectively capture vital information about inspections, diagnoses, and corrective actions [4]. Annotation methods for MWOs have been the subject of recent research in TLP. Tools, such as Nestor[1] have been developed to support the manual injection of critical real-world knowledge by allowing for the annotation of the MWO text descriptions via tagging to facilitate automated categorization and analyses. Machine learning systems can then use these tags as a signal to help ensure correct outcomes [14].

2.2 Large-Scale Text Annotation

The exponential growth of text data has made the current manual text annotation approaches, e.g., crowd-sourcing [26], deficient in meeting the pressing demands for high-quality large-scale annotations [32, 52]. As an alternative, researchers have developed the weak supervision techniques [19,32,41] that leverage human-defined labeling functions (LFs) [41], small labeled datasets [50], or existing text paradigms with multi-type metadata [36] for more efficient text annotating. However, most of these approaches trade off labeling speed or cost with annotation quality [32, 36, 41, 50], and the generated labels are mainly evaluated by numerical performance matrices, such as accuracies [41], F1 scores [50]. Without human review, it is uncertain whether such annotations are of sufficient quality for real-world applications. In light of the deficiencies of manual and automatic text annotation approaches, a series of semi-automatic text annotation frameworks have been proposed, allowing humans to annotate large-scale text data with the help of automatic modules, which can be coordinated labeling modules [54] or deep learning techniques such as attention model [10], human-validated labeling functions [17, 43], and transductive semi-supervised learning [12]. However, the annotation quality of such frameworks still lacks human validation—they either only verify quantitative performance matrices [54], or sample a small subset for humans to inspect results [12, 17]. Although there are a few works for improving the annotation quality [2, 33, 48], they are mainly designed for image or video data and hence not directly applicable to technical language datasets. Given the importance of high-quality annotations [16, 35], a human-centered tool is needed to support the validation of large-scale text annotations.

[1] https://nist.gov/services-resources/software/nestor

2.3 Technical Text Visualization

In the past decade, the idea of applying visualization and visual analytics to technical text analysis has been broadly embraced. Manufacturing enterprises are becoming aware of the value of maintenance records they collect and are supporting visualization research [1, 8, 9, 22]. Academic researchers have developed visual analytical strategies for maintenance records [53] and error logs [30, 31, 37, 46]. In particular, La VALSE [21] and MELA [46] are scalable visualization tools with multiple visualization interfaces incorporating different logs for interactive event analysis. ViBR [7] provides a visual summary of large bipartite relationships by via minimum description lengths and is used for vehicle fault diagnostics. However, existing solutions have prerequisites on either the text format or the quality of the labels. Some assume that there exists a well-defined set of labels [24] to train a classification modelfor the annotation task or assume a trivial effort to define these labels in the pre-processing stage [13, 55] when they are not available as input. Others expect that the text can be generated from grammar or rules so that the labels can be derived from clustering [21].

In this work, we address inconsistent technical text created by human maintainers that contains domain jargon and labels of unknown reliability. Unlike other approaches, we do not have prerequisite text formats nor do we make assumptions about the labels or their quality. We also do not rely on the text's grammatical structure

3 Validation and Relabeling for Text Annotations

In this section, we define the problem we wish to solve and clarify our assumptions. We also derive the annotation error types and design requirements based on the industrial experience of two of the authors and an exploration of a machine maintenance log dataset.

3.1 Problem Definition

Like software development, TLP often requires machine-assistance in the validation of *large*, *complex*, and *context-specific* sets of text. To fulfill this need, we aim to facilitate the validation of annotations and the correction of labelling errors by designing visual analytic techniques for domain practitioners.

Our target users are domain practitioners and data analysts in need of assessing and improving the annotations for large-scale technical text data. We expect they have the analytical skills to interpret the model performance metrics and interact with *LabelVizier*. We also make two assumptions about the technical log text and their labels:

- There exists a finite set of labels $L = \{l_1, l_2, ..., l_n\}$. And the mapping from each record s_i to the labels is defined by $l_i = f(s_i)$, where $l_i \subseteq L$. In the context of this paper, $f(s_i)$ is agnostic and the quality of L requires expert verification.
- There exists a finite set of label categories $C = \{c_1, c_2, ..., c_n\}$. Each label l in L belongs to one label category in C. Note that the "label category" in our context is a higher-level taxonomy of labels. For instance, label "*air-conditioner*" belongs to "Item" and "*too_hot*" belongs to "Problem" in Sec. 6.1.

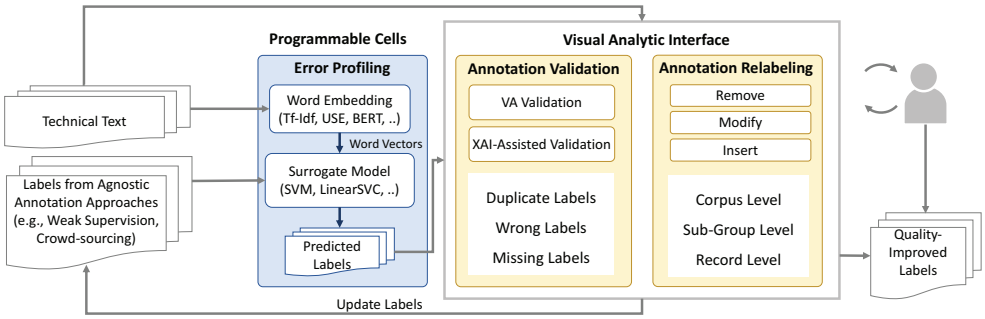

Figure 1: The *LabelVizier* workflow consists of three phases: Error Profiling, Annotation Validation, and Annotation Relabeling. It is implemented as a web-based interactive notebook which takes technical text and the corresponding labels as input. With the actionable insights provided by the surrogate model, XAI method, and visualization, users can identify three error types and improve annotation quality at three different data scales.

Given there is no formalized taxonomy of labeling errors in the TLP domain, we target three dominant types of annotation errors distilled from two coauthors' long-term industrial experience:

- **Duplicate Labels** share duplicated words (e.g., "*temperature*" and "*room_temperature*") and/or express semantic meanings (e.g., "*too_cold_building*" and "*temperature_too_cold*").
- **Wrong Labels** involve labels with conflicting meanings (e.g., one record is labeled with "*too_cold*" and "*too_hot*" simultaneously). It can also refer to an unreasonable label name (e.g., "*building_building*") where the label refers to a non-existent class.
- **Missing Labels** refer to labels that should be assigned to technical records but are absent (see examples in Sec. 6.1). Missing labels are relatively hard to detect if there are already other labels assigned to the record.

The output of our workflow is a set of labels with improved quality.

3.2 Design requirements

After much discussion on a weekly base, our team, which included two TLP domain experts, agreed to four design requirements for *LabelVizier* to address the problem defined in Sec. 3.1:

R1 **Label Overview**: As the first step of label debugging, *LabelVizier* needs to provide users with a summary overview of all technical text and labels. The visual interface needs to present label distribution in the finite label set L and illustrate their categories c (if available) intuitively. The visual interface also needs present the currently assigned labels of one or multiple record(s) s_i in different levels of detail and from different perspectives per user demand to support intensive context comprehension.

R2 **Label Quality Screening**: *LabelVizier* need to support an efficient evaluation of the quality of existing annotations. In particular, the visual interface needs to allow users to quickly locate labels that potentially fall into the three types of errors (see Sec. 3.1). After that, it should help users confirm the error by providing sufficient context information about the related labels and explaining how they were assigned to specific records.

R3 **Interactive Relabeling Support**: Once the errors are identified and confirmed, *LabelVizier* needs to interactively collect the user's relabeling suggestions and apply them to specific scales of the dataset per user request. In particular, users should be able to make suggestions to remove or modify an existing label or insert new labels according to their best judgment. After that, such modifications should be applied to entire corpus, a sub-group, or an individual record per user demand.

R4 **Accessibility and Flexibility**: *LabelVizier* should be accessible to domain practitioners of varying backgrounds. On the one hand, the basic functionalities of the visual interface should be intuitive enough for users without a computing background during the validation and relabeling tasks. On the other hand, *LabelVizier* should provide users with in-depth information on demand and the flexibility to adjust the data processing or model training settings so that the analysis process also satisfies experts with more computing experience and special analysis needs.

3.3 Datasets

We involve two TLP datasets, **HVAC** and **NLU**, in this paper:

HVAC is an internal dataset from our industrial collaborators with over 21,000 pieces of maintenance records from an HVAC system. Each record contains two text fields: "LONG_DESCRIPTION" and "DESCRIPTION". "LONG_DESCRIPTION" describes the detailed maintenance information, including the problem, the solution, the maintainer, the corresponding machine, etc., while "DESCRIPTION" is a concise version, which is often a sentence or a set of keywords. There are also eight categories of labels available for each record, including "P" (Problem), "S" (Solution), "I" (Item), "PI" (Problem Item), "SI" (Solution Item), "X" (Irrelevant), "U" (Unknown), and "NA". For example, the category "P" includes labels such as "*too_hot*", "*leak*", and the category "SI" includes labels such as "*adjust thermostat*", "*replace valve*", etc. These labels were produced by a weak supervision method, and their quality remains agnostic.

NLU [3] contains over 25,000 human-robot interaction records and the corresponding labels, collected from a voice AI agent serving in an intelligent home system. Each record includes three text fields: "question" is a pre-designed human-robot interaction question; "answer" and "answer_normalized" contain the original and normalized user answers, respectively. There are three categories of labels, including "scenario", "intent", and "suggested_entities". For example, the category "scenario" includes labels such as "*weather*", "*music*", and the category "intent" includes labels such as "*request*", "*send email*", etc. These labels were generated from a crowd-sourcing platform, and their quality requires validation as well.

4 Methodology

4.1 Workflow

We designed the *LabelVizier* workflow as an iterative framework with three major phases: (1) Error Profiling, (2) Annotation Validation, and (3) Annotation Relabeling. A regular analysis process starts from the **Error Profiling** phase, in which we train a surrogate model with the technical text and their existing labels to approximate the prior annotation process. Then, users can conduct the first round of **Annotation Validation** through the integrated visual analytic interface, where multiple coordinated views are provided to assist an efficient investigation of labels (**R1**) and detection of three types of errors. After that, users can move on to the **Annotation Relabeling** phase and relabel the identified results at three different levels: corpus level, sub-group level, and record level (**R3**). A more detailed description of our visual and interactive support on these three levels is provided in Sec. 5. After the first pass of the three phases, users can iterate between **Annotation Validation** and **Annotation**

Relabeling for multiple rounds till the annotation quality converges with their standard of satisfaction. It is also worth mentioning that *LabelVizier* simplifies the input and output of phase(1) so that users only need to make minor hyperparameter adjustments to execute different use cases with various analysis purposes (**R4**).

4.2 Surrogate Model for Error Profiling

In the first phase of the *LabelVizier* workflow, we train a surrogate ML model [11] to approximate the generation process of existing labels in the dataset. To ensure that the surrogate model can achieve satisfactory performance and reflect potential annotation issues, we tuned the model architecture with the interactive notebook to fit the specific dataset. Then, by visualizing the model's intermediate results (e.g., prediction probability [29]) in the second and third workflow phases, we help users uncover potential annotation flaws. In this way, users start their label validation from those suspicious labels related to unusual model behaviors (**R2**) and locate a group of potential labeling errors for inspection. After addressing these labels, users can retrain the surrogate model with the better-quality dataset to obtain a reusable model incorporating domain knowledge from human experts and save it for future annotation tasks.

The error profiling phase of *LabelVizier* require the surrogate model to be: (1) lightweight, so that the model tuning is time-effective; (2) accurate in producing similar results to existing annotations. For (1), we utilize lightweight and time-effictive word-embedding and ML methods to process text data and train the annotation classifier. For instance, to process the input technical text data, we adopt computationally efficient and widely-used word embedding techniques, including TF-IDF (term frequency-inverse document frequency) [51] and truncatedSVD (Singular value decomposition) [20], to encode the original text into real-valued vectors. For (2), we iterate multiple processes with different model training settings, and audit quantitative performance matrices in the validation split until reaching the best result. Thus, we ensure the surrogate model achieves satisfactory performance, i.e., the alignment between the predicted labels and the existing annotations is reasonable for the surrogate model to simulate the annotating process and provide hints for users. Specifically, the average hamming loss is 0.02, the micro f1 score is 0.8044, and the average macro f1 is 0.6703, where smaller hamming loss, and larger micro & macro f1 scores indicate better performance. Besides, the predicted probabilities *predProba* of the fitted LinearSVC $fittedModel$ is obtained to act as the clue of finding suspicious labels.

4.3 Model Behavior Explanation

LabelVizier utilizes one of the state-of-the-art eXplainable Artificial Intelligence (XAI) techniques–LIME (Local Interpretable Model-Agnostic Explanations) [42]–to support the annotation validation. For each of the constructed surrogate models, LIME performs perturbation-based analysis over a given text record and presents the explanation by highlighting the rationale behind the model's prediction. It exposes the weakness of the model and the pitfalls of the input technical text and thus could help users more accurately inspect a potential annotation error and make a reasonable relabeling decision. The LIME explanation is integrated into the "Explain" tab of Record Projection View *LabelVizier* and is triggered when users select a record from the "Categorize" tab for further inspection (more details in Sec. 5 and examples Sec. 6).

4.4 Implementation

To maximize its accessibility for sharing and flexibility for customization (**R4**), we implement the *LabelVizier* workflow as a computational notebook. We use multiple Python data analysis libraries including Pandas [39], Numpy [25], and Joblib [28] for data processing and intermediate metrics analysis. In addition, the word embedding techniques (TF-IDF and truncated SVD) and ML methods (LinearSVC) discussed in Sec. 4.2 are implemented with scikit-learn [40], and the LIME technique is with LimeTextExplainer [42]. To ensure smooth integration and faster rendering speed, we embed the visual interface (Fig. 2) in the computational notebook with Plotly's JavaScript Graphing Library and Plotly Dash. And we use the t-distributed stochastic neighbor embedding (t-SNE) algorithm for dimensionality reduction when visualizing the high-dimensional word vectors and confidence vectors for the Record Projection View. We also deliberately separate the functions in the notebook so that users can easily plug in any word embedding and dimensionality reduction algorithms for their specific analysis needs with minor programming.

5 Visual Analytic Interface

To fulfill the design requirements in Sec. 3.2 and concretize the *LabelVizier* workflow in Sec. 4.1, we design a visual analytic interface (see Fig. 2) that contains three major components: (A) Label Investigation View, (B) Record Projection View, and (C) Inspection & Operation View. In this section, we demonstrate how we can coordinate these views to locate the three types of errors and perform multi-level validation and relabeling on annotations.

5.1 Annotation Validation

With the coordination among different components of *LabelVizier*, users can efficiently validate the annotation quality and identify the three major types of error introduced in Sec. 3.1 (**R2**).

5.1.1 Duplicate Label Detection

To support duplicate label detection, we designed the Label Investigation View (Fig. 2 (A)) and the "Inspect" tab of Inspection & Operation View. We choose the sunburst diagram for Label Investigation View to provide an overview of the hierarchical relationship between labels and their category (**R1**), as well as the distribution of the labels across categories – the size of the label sectors at the outermost layer represents the number of records in the dataset assigned with the corresponding label. We also encode the possibility of duplicate labels into the sectors colors to provide a priority recommendation for the user inspection. This duplication possibility is the average of a ratio of co-occurrence number $Co(l_i, l_j)$ to the total appearance $Num(l_i)$ of each label l_i in the category:

$$P_{duplication} = \frac{1}{n_{categ}} \sum_{i=1}^{n_{categ}} \left(\frac{1}{n_{occur}} \sum_{j=1}^{n_{occur}} \frac{Co(l_i, l_j)}{Num(l_i)} \right) \quad (1)$$

The sunburst diagram is expandable per user request. And in the zoom-in view, we embed a chord diagram to illustrate the label co-occurrence in the same record, which is a strong indicator of duplicate labels (e.g., Fig. 3 (A)). In this chord diagram, the co-occurring labels are connected with white chords, with their thickness representing the co-occurrence frequency. For example, the chord between the label "*room too_hot*" and "*too_hot room*" is thicker than that between "*room too_hot*" and "*water leak*", indicating a heavier co-occurrence pattern and potential duplication of the former pair. A larger number of thicker chords also indicates a higher possibility of existing duplicated labels in this category, which corresponds to the brighter sector color described above.

Users can further inspect the context of any suspect labels by clicking on them and checking the updated data table under the "Inspect" tab in the Inspection & Operation View (Fig. 2 (C)). This data table presents all the records across the dataset assigned with the selected label. In this way, users can efficiently locate and evaluate the correctness of potential problematic labels.

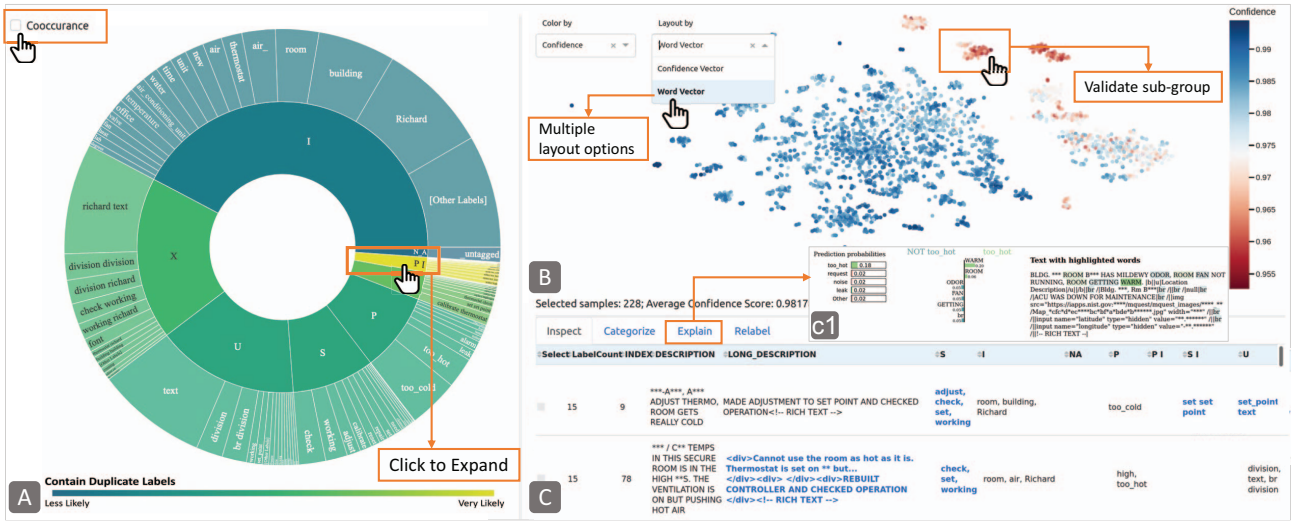

Figure 2: The *LabelVizier* interface with the HVAC dataset. (A) **Label Investigation View** visualizes the label and category hierarchy relationships; each category can be expanded to present label co-occurrences (see Fig. 3 (A)). (B) **Record Projection View** presents the record distribution oto support sub-group validation, layout by model confidence vectors or input word vectors. The color represents "model prediction confidence" or "record info density". (C) **Inspection & Operation View** includes multiple tabs, "Inspect" for record and label inspection, "Categorize" for category-based investigation, "Explain" for model behavior interpretation (c1), and "Relabel" for relabeling operations.

5.1.2 Wrong Label Detection

Wrong labels can be detected with the Record Projection View (Fig. 2 (B)) in coordination with the "Categorize" and "Explain" tabs of Inspection & Operation View. To provide a two-dimensional (2D) overview for all records (**R1**), we apply the t-SNE [49] algorithm to project the customized record vectors onto the 2D space and visualize each of them as a dot. The customized record vector can be a "word vector" or a "confidence vector". When the "word vector" is used for layout, the distance among the dots indicates the semantic closeness of descriptions in their corresponding records. When the "confidence vector" is used for layout, the distance among dots indicates the model behavior towards similarity when predicting labels for the corresponding records. We provide two options to color the record projections — "information density" and "confidence score". The "information density" is more useful in locating missing labels, so we will discuss it in Sec. 5.1.3. The "confidence score" is the mean value of all dimensions of the aforementioned "confidence vector", which could expose the records containing more labels predicted with low confidence, and thus provide hints to locate sub-groups that potentially contain labeling mistakes (Fig. 2 (B)).

Once a cluster with low confidence is identified and selected, a heatmap under the "Categorize" tab in Inspection & Operation View (Fig. 4 (c1)) will be triggered, where each row refers to a single record; each column represents one label category, and the color indicates the model's average confidence score.

To support deeper understanding of the model reasoning process, we provide LIME explanations under the "Explain" Tab (Fig. 2 (c1)). The explanation includes three parts: the left bar chart visualizes the top five predicted labels and their prediction probabilities; the middle bar chart visualizes the "score of contribution" of the input words to the top label; the right side shows more context information and the original text record, where the positive and negative contributors are highlighted with different colors. Combining these three kinds of information, we aim to help users verify whether the rationale behind the model's decision aligns with their knowledge (**R2**).

5.1.3 Missing Label Detection

The detection of missing labels also involves the Record Projection View (Fig. 2 (B)) and "Inspect" tabs of Inspection & Operation View. We designed the "information density" metric to highlight records more likely to have missing labels. This metric is determined by the ratio of label count and the input text length:

$$D_{Info} = log(\frac{Count(labels)}{WordCount(text)}) \quad (2)$$

Once the users locate and select a cluster of records with low "information density", the "Inspect" tab of Inspection & Operation View will be updated for verification of the label missing issue. It is also worth mentioning that higher "information density" can insufficiently indicate the existence of duplicate labels, but further verification is required with the process in Sec. 5.1.1.

5.2 Annotation Relabelling

After confirming a labeling error, users can improve the annotation quality of the dataset (**R3**) in three different data scales: corpus level, sub-group level, and record level:

- **Corpus level relabeling** updates the label across the entire technical text dataset. It is achieved by clicking on a label from Label Investigation View and use the "Relabel" tab of Inspection & Operation View to "remove", "modify", or "insert" it. For example, the label "*building_building*" is considered to be a "wrong label" error at the corpus level. Users can select it from Label Investigation View and remove it from all affected records.
- **Sub-group Level relabeling** involves a sub-group of records within the dataset. It is achieved by selecting a sub-group of records with the lasso tool in Record Projection View and relabeling them with the "Relabel" tab. The number of records in one sub-group can vary from a dozen to hundreds.
- **Record level relabeling** updates individual record(s) associated with a specific label error. For example, a user looks over the records through the "Inspect" tab and notice two records missing the label "alarm". They can select these two record(s) with the checkbox and relabel them under the updated "Relabel" tab. These relabeling operations are only applied to the selected records.

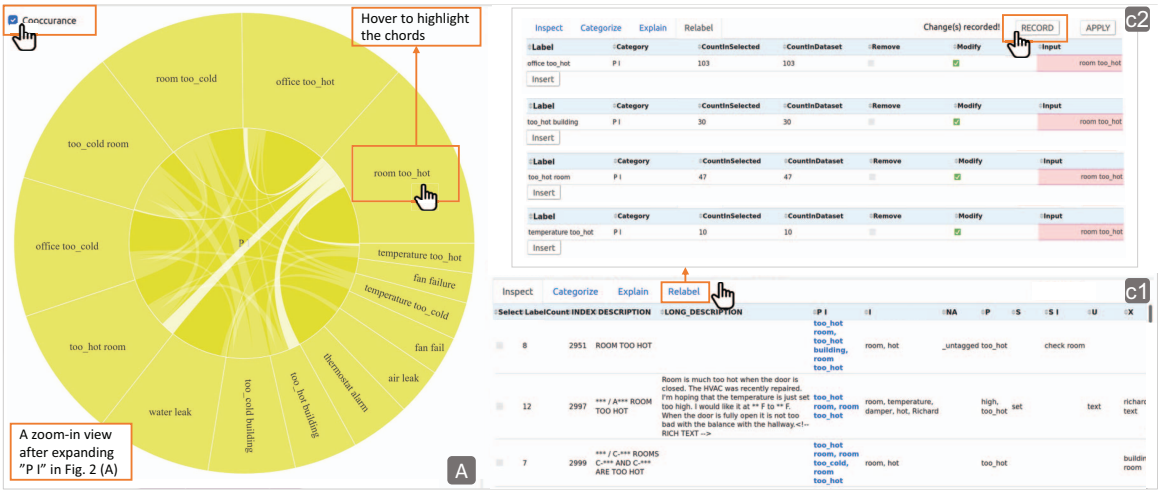

Figure 3: Duplicate label validation with the HVAC dataset (Sec. 6.1). The chord diagram (A) shows that labels "*office too_hot*", "*room too_hot*", "*too_hot building*", and "*too_hot room*" co-occur frequently. Duplication is confirmed via record details in (c1) and fixed in "Relabel" tab (c2).

To eliminate waiting for dataset updates and projection re-rendering, the visual interface is only re-rendered when the user request to apply the changes. To achieve this, we record the relabeling operations as a history list and sequentially apply them to the database per request.

6 Use Case Scenarios

This section describes two use case scenarios where *LabelVizier* assists domain experts in validating the quality of technical text annotations and conducting efficient relabeling.

6.1 Case 1: Maintenance Management for HVAC System

This use case involves Amy, a maintenance manager who monitors machine maintenance records to track maintenance-related issues and to plan for future maintenance resources (e.g. budgets, maintainers, etc.). With the HVAC dataset, we demonstrate how she uses *LabelVizier* to validate the label quality of MWOs and make maintenance management based on the more accurate labels.

After finishing data processing and surrogate model training in the programmable cells of *LabelVizier*, Amy starts validating labels through the interface. Because the frequency of similar labels reflects the prevalence of a maintenance issue and influences decision priorities and budgets, Amy chooses to screen duplicate labels at first. She notices from the Label Investigation View (Fig. 2 (A)) that the category "PI" is the most likely to contain duplicate labels, so she expands it to check for label co-occurrance (Fig. 3 (A)). As indicated by the chord thickness, "*office too_hot*", "*room too_hot*", "*temperature too_hot*", "*too_hot building*", and "*too_hot room*" co-occur very frequently. Because these labels have similar semantic meanings, Amy further inspects their context in the "Inspect" table (Fig. 3 (c1)) and confirms that they are duplicates. Such duplication will overemphasize air-conditioner-related "*too hot*" issues and may cause excessive allocation of maintenance resources. Amy removes the redundant labels and unifies the rest with "*room too_hot*" with the "Relabel" (Fig. 3 (c2)). Now that Amy has created more accurate and consistent labels for temperature-related problems, she counts their frequency, evaluates the problem severity, and decides to arrange for regular examination of all air-conditioners across the company.

Amy moves on to the Record Projection View (Fig. 2 (B)) to screen for wrong labels which are another type of error that can mislead maintenance planning. Amy notices that there are several clusters with lower confidence scores (Fig. 4 (B)), indicating that the surrogate model performed worse and might predict the wrong labels for the corresponding records. After selecting one, she learns from the updated "Inspect" tab that the labels "*Richard*" in category "I" and "*br richard*" in category "X" appear in many records. Amy browses the context to check if this is related to a maintainer's name or is an improper description but finds the phrase "br richard" doesn't appear to be semantically relevant to either "DESCRIPTION" or "LONG_DESCRIPTION". Seeing this issue also appears in many other clusters, Amy decides to further investigate its cause with the "Categorize" and "Explain" functions. Under the "Categorize' tab, she clicks on the cell indicating lowest prediction confidence in category "I" and "X" (Fig. 4 (c1)) to trigger a LIME explanation. As shown in the left bar chart in Fig. 4 (c2), the model's top prediction for the selected record is the label "*br richard*". However, the middle and right part shows that most positive contributors to this prediction are HTML metadata such as "TEXT" and "RICH". Amy realizes that the wrongly-predicted label "*br richard*" might originate from the presensce of HTML tags. After seeing similar LIME explanations for more records in this cluster, her hypothesis is confirmed – the model referred to HTML tags to predict the wrong labels. Amy removes these wrong labels (Fig. 4 (c3)) and decides to conduct a thorough cleaning of the dataset later to remove HTML tags.

Amy also checks the info density and does not find any severe label missing issues. Then she reloads the updated labels into *LabelVizier* and confirms that the new annotations are satisfactory. Finally, Amy executes the code cells of the computational notebook (Sec. 4.4) to re-train the surrogate model with the relabeled dataset. In this way, she preserves her domain knowledge in the model that can be used to annotate any future maintenance records.

6.2 Case 2: Data Cleansing for NLU Model Training

In this section, we demonstrate how *LabelVizier* can help modify the annotations of the training data for a specific application scenario. This use case involves Steven, a data engineer working in a company that provides conversational agent (CA) services. Steven coordinates the large-scale crowd-sourcing process to provide high-quality training data for the natural language understanding (NLU) model [3] embedded in the CAs, which summarizes the semantic content of user utterances by mapping it to structured, abstract representations (labels) that support the decision making process.

Steven uses *LabelVizier* to validate and debug the crowd-sourced annotation results before delivering the dataset for the downstream machine learning tasks. The NLU model requires all labels to be independent and accurate so that the voice AI agent can exclu-

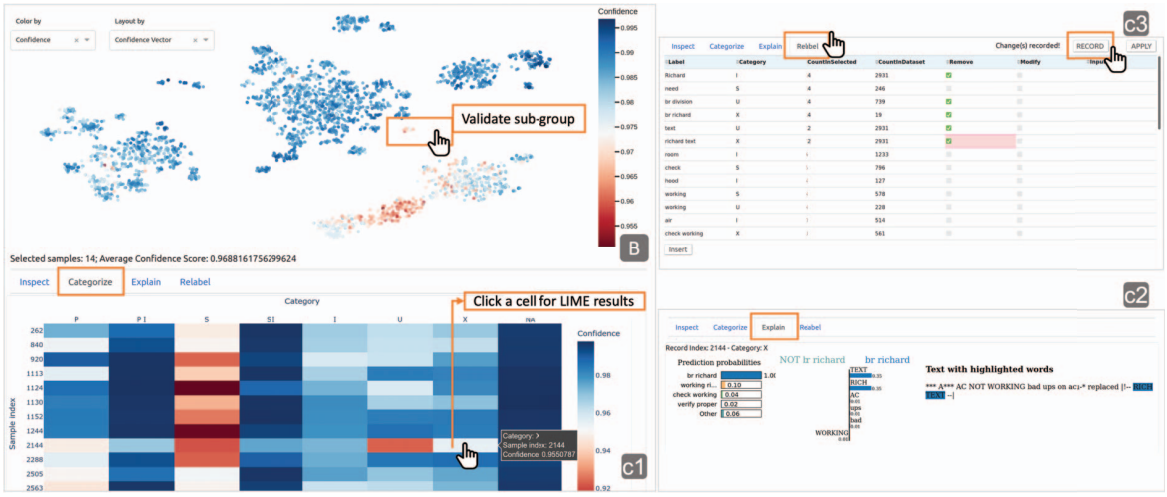

Figure 4: Wrong label detection in HVAC dataset using *LabelVizier* (Sec. 6.1). Users can select a sub-group with lower confidence in (B) and inspect model confidence of each category in (c1). Then they can click on cells in (c1) to activate LIME explanation in (c2) for model behavior interpretation. After confirming the error, users can remove the wrong labels with (c3).

sively query them in the search engine and provide correct answers to the users. To remove those dependent labels with semantic duplication, so Steven starts by looking for them via the Label Investigation View. According to the coloring of the categories, the category "suggested_entities" is most likely to include duplicate, so Steven expands this category to inspect the label co-occurrences. The thick chord indicates that the labels "*currency_source*" and "*currency_target*" heavily overlap (Fig. 5 (A)). Steven inspects the detailed context of the corresponding records with the Inspection & Operation View and finds that most of these records are related to currency exchange questions. Although the two labels "*currency_source*" and "*currency_target*" appear reasonable, Steven still decides to merge them into the single label "*currency_source_and_target*" to facilitate the down-streaming task.

When using the "Inspect" tab to investigate the duplicate label issues above, Steven notices that missing labels are a common with this dataset. He moves on to the Record Projection View to facilitate find more missing labels. He chooses the options of "Color by Info Density" and "Layout by Word Vector" to highlight clusters with similar semantic meanings and lower info density (Fig. 5 (b1)). Then he uses the lasso tool to select the most notable cluster and observes from the Inspection & Operation View (Fig. 5 (c1)) that all the records in this cluster were labeled as "*QA*" under the category "scenarios" and "*object_query*" under the category "intent", but not assigned with any labels under the category "suggested_entities". With a second look at the input "questions", "answers", along with the model's reasoning process from the "Explain" tab, Steven figures out the cause — the model only captured the information from the records' shared "question" but ignored the "answers". To fill the missing labels, Steven zooms into this cluster in the Record Projection View (Fig. 5 (b2)) and uses the lasso tool to select each sub-clusters with similar semantic meaning. After selecting one sub-cluster (Fig. 5 (b2)), Steven notices similar semantic meanings of the selected records – for "answer_normalized", those records have sentences such as "*how do you think the world ends*", "*tell me how the world begin*" and "*do you believe in god*", etc. Considering these questions were asked by their CA users, Steven believes "*apocalypticism*" or "*philosophy*" would be proper labels for the category "suggested_entities". He inserts them into the selected sub-group of records with the help of the "Relabel" tab. Steven conducts the same operation to the few other low-info-density clusters, and fix the label missing issues accros the entire dataset.

Finally, Steven applies his relabeling operations to the dataset and updates the interface. After confirming the quality of the annotations, he delivers the dataset to the machine learning engineers for the downstream training tasks. With *LabelVizier*, Steven optimizes the annotation quality from crowd-sourcing results and avoids the potential flaws that can bias the training of the voice AI agent.

7 Expert Reviews

LabelVizier was developed with the participation of TLP domain experts (Sec. 3.2) over the course of two years. To evaluate the generalisability of our workflow and reveal insights from or practical value to domain practitioners, we invited another two TLP domain experts (E1 and E2) and two experts from other domains (E3 and E4) into our expert review studies.

7.1 Expert Demographics

All experts were experienced in data analysis and performed data annotation tasks in their daily work. E1 and E2 was research engineers from the TLP community, who were familiar with and had worked on the analysis of the HVAC dataset (Sec. 6.1) before the study for several years. E3 was an economist and statistician who analyzed large-scale tabular datasets to gain insights into community resilience. E4 was a research social scientist who manually annotated large-scale datasets about risk perception and evacuation decision-making, and was in need of speeding up this process.

7.2 Tasks and Setup

We conducted two pilot studies to simulate the remote setup, test the *LabelVizier* execution environment (online Colab via a local Jupyter Notebook) and adjust the content of the tutorial sessions. Then we finalized a semi-structured, open-ended expert review in which each expert was asked to explore one of the two datasets described in Sec. 6. Based on their domain of expertise and familiarity with the dataset, E1 and E2 used the HVAC Dataset while E3 and E4 used the NLU Dataset. We shared the original dataset and its documentation[2] with E3 and E4 before the study so that they could get familiar with it in advance. Because the existing annotation in the NLU Dataset is relatively clean, we used an adapted version with two manually inserted errors for each error type in Sec. 3.1 in the study.

[2]https://github.com/xliuhw/NLU-Evaluation-Data

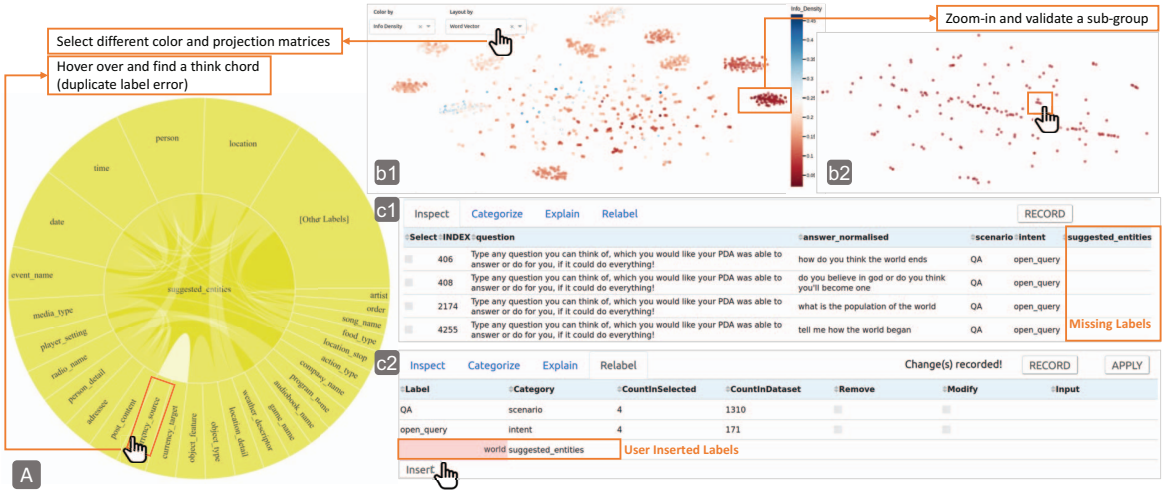

Figure 5: Finding duplicate and missing labels in NLU dataset using *LabelVizier* (Sec. 6.2). The chord diagram in the Label Investigation View (A) reveals the duplicate labels "*currency_source*" and "*currency_target*". The clusters with low information density in the Record Projection View (b1) highlight records with missing labels. The "Inspect" view (c1) shows that the "suggested_entities" for the selected records in (b2) are missing. User-suggested labels can be inserted through the "Relabel" tab (c2).

The study was conducted online via a video conferencing where the domain experts accessed *LabelVizier* from Google Colab Notebook[3] via their personal computers. We shared the tutorial document with the experts no less than two days before the study.The online study session started with a 25-min tutorial session that combined an introductory presentation, a live demonstration, and the mini-tasks. An example mini-task for the HVAC dataset was "Please use *LabelVizier* to find one pair of duplicated labels under the category 'PI' (Problem Item), and then suggest how to modify it with the 'Relabel' tab". After the tutorial session, the experts were asked to freely explore their assigned dataset to validate and relabel the annotations (20-25 minutes). During this process, the experts followed the think-aloud protocol to verbalize their thinking and suggestions. Finally, the experts responded to a questionnaire with their demographic information and general feedback of *LabelVizier*.

7.3 Observations

In our study, all experts appreciated the value of *LabelVizier* for facilitating annotation refinement and expressed willingness to use it in their daily work, describing it as *a very good tool* (E2) that was "*helpful at a high level of quickly and...pleasantly...identifying issues than just scrolling through a spreadsheet*" (E4). Meanwhile, we observed that domain experts with differing backgrounds interacted distinctly with *LabelVizier* and sometimes provided divergent comments towards the same features. We categorize the their behaviors and feedback during the exploratory and describe them below.

Learning Curve. Based on their familiarity with the dataset and the tool, domain experts required differing times to overcome the learning curve (**R4**). For instance, E1 and E2 had previously worked on the HVAC dataset with other annotation tools, so they spent less time grasping *LabelVizier* compared to the other two experts (E3, E4). E2 expressed great interest in the methodology "*under the hood*" and asked many technical questions to understand the underlying mechanism during the tutorial session. Although E3 and E4 needed more hands-on instructions about using our tool, they were capable of replicating the moderator's operations and accomplishing the exploration task after the tutorial sessions. They praised our tutorial session design, saying "*it helped very much...after (the moderator) demonstrated, it very easy to replicate*" (E2).

[3]https://colab.research.google.com/

Functionality. In all four studies, the experts were able to efficiently evaluate the quality of the annotations (**R2**) and successfully accomplish the relabelling (**R3**) by coordinating information from the three major views of *LabelVizier* (Fig.2). Moreover, E1, E2, and E4 successfully mastered the relatively complex "Categorize" and "Explain" functions and utilized them to understand the root cause of a potential wrong label. We also received requests for more delicate annotation manipulations and more complicated information support from experts with shorter learning curves, such as modifying the name of a label category (E2) or showing the percentage value of the duplicated labels (E1, E2). However, experts with longer learning curves requested simpler operations and more exploration guidance from the tool, such as simplified projection view (E3) or "*pop-up reminders...to remind people what these different tools are for in a really obvious way*" (E4). How to support more delicate label manipulation as well as ensure the accessibility of *LabelVizier* ((**R4**)) is an inspiring topic that we will discuss in Sec. 8.

Visualization. Interestingly, experts with different backgrounds and experience using (semi-) automatic annotation tools also showed different preferences towards our two major visualization components – the Label Investigation View and the Record Projection View. Though all experts expressed their favor of the Label Investigation View, saying they "particularly like the chord diagram" (E3) because it was "*very helpful*"(E1), "*intuitive enough*" (E2) and they "*haven't seen labels presented in this way*" (E4), E2 mentioned "*the co-occurrence is less useful because I don't have enough flexibility to dive down into why there's that co-occurrence*." For the Record Projection View, experts knowing more about machine learning (E1 and E2) picked it up faster and appreciated its value in finding wrong and missing labels better. "*I think the projection view is super useful*," said E2. They "*would like to see even more options of projection spaces and be able to play around with those*." In contrast, E3 felt the same function was too complex and required "*a lot of playing around*." E4 didn't even get a chance to try out the different projection options because of the time constraint.

Interaction. During the study, we received precious interaction improvement suggestions, including more cross-view coordination, operation history tracking, and typing suggestions. E1, E2, and E4 suggested more flexible interactions, such as cross-view Boolean operations and subset highlighting. For example, when E2 inspected the label "*time*," they mentioned that this label might involve differ-

ent types of redundancy according to their prior knowledge about the dataset. As a result, they requested Boolean operations between the Label Investigation View and Record Projection View to sift those records of their specific need. E3 and E4 suggested providing ways to keep track of the editing history, such as an undo function (E3), a history list (E4), and some hints of what the user has just clicked (E4). E2 suggested adding typing suggestions for relabel tab, such as auto-complete or alternative recommendation functions. These suggestions reflected the experts' tacit knowledge gained from their long-term annotation practice and will direct us to a more accountable annotation tool in the next development iteration.

8 Discussion

The observations and feedback from the expert reviews indicated that *LabelVizier* provides a means for domain practitioners to validate and relabel the technical text annotations "*quickly*" and "*pleasantly*". They also suggest potential future work for our workflow and tool. Below, we organize the lessons learned.

Accessibility v.s. Functionality. We observed in our expert review that users with less machine learning and (semi-) automatic annotation tool experience may go through longer learning curves with *LabelVizier*. They requested more exploration guidance or relabelling recommendations when using the tool, while the other group of users requested more complex functions, saying *LabelVizier* was "*good to find gross errors, but not for perfectionism*"(E2). This is understandable because we required domain experts with diverse backgrounds to learn a relatively complex system within a limited time. We plan to alleviate this problem by leveraging user modeling techniques [23] to analyze the user behavior and guide them to start from different levels of complexity. This way, it will also be safe to extend *LabelVizier* with more intricate functions, as recommended.

Automation v.s. Human Trust. As computer science researchers, we tended to incorporate more automation in *LabelVizier* during the development process, which was discouraged by our collaborators with technical text annotation backgrounds. We observed that most of the data analysts tended to be "*over conservative*" (E2) and had to closely check the raw text before "*starting to believe the systems is working*"(E1). They also said that many cases were ambiguous, so they tended to examine more context before making relabelling decisions. Because of this, *LabelVizier* currently still involves considerable manual work, as demonstrated in Sec. 6. *LabelVizier* also only provides recommendations and explanations instead of one-step relabelling suggestions to supply users with a comfortable amount of information. Indeed, there were no complaints about too much manual work during the expert review but we did receive praise that our tool helped "*focus their energy*"(E4).

Application Domains. *LabelVizier* was originally designed to serve as a component of technical language processing, but it is generalizable to other annotation verification tasks. The error profiling process can take any natural language descriptions and their labels as input and allow users to perform validation and relabelling via the interface. If label categories are available, as was for our use cases (Sec. 3.1), there will be two layers in the Sunburst diagram of Label Investigation View. Otherwise, the Sunburst diagram will devolve into a pie chart, with other *LabelVizier* functionality remaining unchanged.

9 Conclusion

We presented *LabelVizier*, a human-in-the-loop workflow that can help domain experts efficiently validate and improve the quality of multi-labeled technical text annotations. *LabelVizier* utilizes a web-based interactive notebook to enable flexible data processing and model training, and integrates a visual analytic system to leverage human knowledge in annotation relabeling. The interface coordinates different visual components for multi-type error detection (duplicate, missing, and wrong labels) in different dataset scopes (corpus level, sub-group level, and record level), and provides a human-centered solution targeting the quality enhancement for large-scale text annotations. We demonstrate the usability of *LabelVizier* via two use case cases, and four experts evaluated the effectiveness of our workflow through a study consisting of one-on-one qualitative evaluations. We believe our work will encourage the design of visual analyticsfor other domain-driven problems and inspire future research efforts in creating higher-quality annotations for larger-scale text datasets.

NIST Disclaimer

The use of any products described in this paper does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that products are necessarily the best available for the purpose.

Acknowledgement

We are grateful to the domain experts who volunteered for our expert reviews and the anonymous reviewers that helped improve this paper. This research is sponsored in part by the Grant No. 70NANB21H170 from the U.S. Department of Commerce, National Institute of Standards and Technology and a gift from Bosch Research.

References

[1] B. Alsallakh and L. Ren. Powerset: A comprehensive visualization of set intersections. *IEEE Transactions on visualization and computer graphics*, 23(1):361–370, 2016.

[2] A. Bäuerle, H. Neumann, and T. Ropinski. Classifier-guided visual correction of noisy labels for image classification tasks. In *Computer Graphics Forum*, vol. 39, pp. 195–205. Wiley Online Library, 2020.

[3] D. Braun, A. H. Mendez, F. Matthes, and M. Langen. Evaluating natural language understanding services for conversational question answering systems. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pp. 174–185, 2017.

[4] M. P. Brundage, T. Sexton, M. Hodkiewicz, A. Dima, and S. Lukens. Technical language processing: Unlocking maintenance knowledge. *Manufacturing Letters*, 27:42–46, 2021.

[5] M. P. Brundage, T. Sexton, M. Hodkiewicz, K. C. Morris, and J. Arinez. Where do we start? guidance for technology implementation in maintenance management for manufacturing. *J. of Manufacturing Science and Engineering*, 141(9):091005, 2019. doi: 10.1115/1.4044105

[6] T. P. Carvalho, F. A. Soares, R. Vita, R. d. P. Francisco, J. P. Basto, and S. G. Alcalá. A systematic literature review of machine learning methods applied to predictive maintenance. *Computers & Industrial Engineering*, 137:106024, 2019.

[7] G. Y.-Y. Chan, P. Xu, Z. Dai, and L. Ren. ViBR: Visualizing bipartite relations at scale with the minimum description length principle. *IEEE Trans. on Visualization and Computer Graphics*, 25(1):321–330, 2019. doi: 10.1109/tvcg.2018.2864826

[8] S. Chandrasegaran, X. Zhang, M. P. Brundage, and K.-L. Ma. Using text visualization to aid analysis of machine maintenance logs. *Model-Based Enterprise Summit (MBE 2020)*, p. 76, 2020.

[9] Y. Chen, P. Xu, and L. Ren. Sequence synopsis: Optimize visual summary of temporal event data. *IEEE transactions on visualization and computer graphics*, 24(1):45–55, 2017.

[10] M. Choi, C. Park, S. Yang, Y. Kim, J. Choo, and S. R. Hong. Aila: Attentive interactive labeling assistant for document classification through attention-based deep neural networks. In *Proceedings of the 2019 CHI conference on human factors in computing systems*, pp. 1–12, 2019.

[11] A. Cozad, N. V. Sahinidis, and D. C. Miller. Learning surrogate models for simulation-based optimization. *AIChE Journal*, 60(6):2211–2227, 2014.

[12] M. Desmond, M. Muller, Z. Ashktorab, C. Dugan, E. Duesterwald, K. Brimijoin, C. Finegan-Dollak, M. Brachman, A. Sharma, N. N. Joshi, et al. Increasing the speed and accuracy of data labeling through an ai assisted interface. In *26th International Conference on Intelligent User Interfaces*, pp. 392–401, 2021.

[13] S. Di, R. Gupta, M. Snir, E. Pershey, and F. Cappello. Logaider: A tool for mining potential correlations of hpc log events. In *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, pp. 442–451. IEEE, 2017.

[14] A. Dima, S. Lukens, M. Hodkiewicz, T. Sexton, and M. P. Brundage. Adapting natural language processing for technical text. *Applied AI Letters*, 2(2):e33, June 2021.

[15] A. Dima and A. Massey. Keyphrase extraction for technical language processing. *Journal of Research of National Institute of Standards and Technology*, 126(126053), Mar. 2022.

[16] A. Dumitrache, L. Aroyo, and C. Welty. Achieving expert-level annotation quality with crowdtruth. In *Proc. of BDM2I Workshop, ISWC*, 2015.

[17] S. Evensen, C. Ge, and C. Demiralp. Ruler: Data programming by demonstration for document labeling. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1996–2005, 2020.

[18] K. Fort. *Collaborative Annotation for Reliable Natural Language Processing: Technical and Sociological Aspects*. John Wiley & Sons, 2016.

[19] D. Fu, M. Chen, F. Sala, S. Hooper, K. Fatahalian, and C. Ré. Fast and three-rious: Speeding up weak supervision with triplet methods. In *International Conference on Machine Learning*, pp. 3280–3291. PMLR, 2020.

[20] G. H. Golub and C. Reinsch. Singular value decomposition and least squares solutions. In *Linear algebra*, pp. 134–151. Springer, 1971.

[21] H. Guo, S. Di, R. Gupta, T. Peterka, and F. Cappello. La VALSE: Scalable log visualization for fault characterization in supercomputers. In *Proc. EGPGV*, pp. 91–100. Eurographics, 2018. doi: 10.5555/3293524.3293533

[22] S. Guo, K. Xu, R. Zhao, D. Gotz, H. Zha, and N. Cao. Eventthread: Visual summarization and stage analysis of event sequence data. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):56–65, 2017.

[23] S. Ha, S. Monadjemi, R. Garnett, and A. Ottley. A unified comparison of user modeling techniques for predicting data interaction and detecting exploration bias. *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[24] H. Hamooni, B. Debnath, J. Xu, H. Zhang, G. Jiang, and A. Mueen. Logmine: Fast pattern recognition for log analytics. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pp. 1573–1582, 2016.

[25] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020. doi: 10.1038/s41586-020-2649-2

[26] Jeff Howe. Crowdsourcing: A definition, 2006. [Online; accessed 28-Oct-2022].

[27] X. Jin, B. A. Weiss, D. Siegel, and J. Lee. Present status and future growth of advanced maintenance technology and strategy in us manufacturing. *International J. of Prognostics and Health Management*, 7(Special Issue on Smart Manufacturing PHM), 2016.

[28] Joblib Development Team. Joblib: running python functions as pipeline jobs, 2020.

[29] D. Jordan, M. Steiner, E. F. Kochs, and G. Schneider. A program for computing the prediction probability and the related receiver operating characteristic graph. *Anesthesia & Analgesia*, 111(6):1416–1421, 2010.

[30] S. P. Kesavan, T. Fujiwara, J. K. Li, C. Ross, M. Mubarak, C. D. Carothers, R. B. Ross, and K.-L. Ma. A visual analytics framework for reviewing streaming performance data. In *IEEE Pacific Visualization Symposium (PacificVis)*, pp. 206–215, 2020.

[31] J. K. Li, T. Fujiwara, S. P. Kesavan, C. Ross, M. Mubarak, C. D. Carothers, R. B. Ross, and K.-L. Ma. A visual analytics framework for analyzing parallel and distributed computing applications. In *IEEE Visualization in Data Science*, pp. 1–9, 2019.

[32] P. Lison, J. Barnes, and A. Hubin. skweak: Weak supervision made easy for nlp, 2021.

[33] S. Liu, C. Chen, Y. Lu, F. Ouyang, and B. Wang. An interactive method to improve crowdsourced annotations. *IEEE transactions on visualization and computer graphics*, 25(1):235–245, 2018.

[34] S. Lukens, M. Naik, K. Saetia, and X. Hu. Best practices framework for improving maintenance data quality to enable asset performance analytics. *Annual Conference of the PHM Society*, 11(1), 2019.

[35] F. Matsuda, Y. Shinbo, A. Oikawa, M. Y. Hirai, O. Fiehn, S. Kanaya, and K. Saito. Assessment of metabolome annotation quality: a method for evaluating the false discovery rate of elemental composition searches. *PLoS One*, 4(10):e7490, 2009.

[36] D. Mekala, X. Zhang, and J. Shang. Meta: Metadata-empowered weak supervision for text classification. In *Proc. of Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

[37] H. T. P. Nguyen, A. Bhatele, N. Jain, S. Kesavan, H. Bhatia, T. Gamblin, K.-L. Ma, and P.-T. Bremer. Visualizing hierarchical performance profiles of parallel codes using callflow. *IEEE transactions on visualization and computer graphics*, 2019.

[38] C. G. Northcutt, A. Athalye, and J. Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks. Mar. 2021.

[39] T. pandas development team. pandas-dev/pandas: Pandas, Feb. 2020. doi: 10.5281/zenodo.3509134

[40] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[41] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, vol. 11, p. 269. NIH Public Access, 2017.

[42] 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, San Francisco, CA, USA, August 13-17, 2016*, pp. 1135–1144, 2016.

[43] T. Rietz and A. Maedche. Cody: An ai-based system to semi-automate coding for qualitative research. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pp. 1–14, 2021.

[44] J. Risch, A. Kao, S. R. Poteet, and J. Wu. Text visualization for visual text analytics. In *Visual data mining*, pp. 154–171. Springer, 2008.

[45] T. Sexton, M. P. Brundage, M. Hoffman, and K. C. Morris. Hybrid datafication of maintenance logs from ai-assisted human tags. In *Proc. IEEE Big Data*, pp. 1769–1777, 2017.

[46] Shilpika, B. Lusch, M. Emani, V. Vishwanath, M. E. Papka, and K.-L. Ma. MELA: A visual analytics tool for studying multifidelity hpc system logs. In *Proc. DAAC*, pp. 13–18. IEEE, 2019.

[47] G. A. Susto, A. Schirru, S. Pampuri, S. McLoone, and A. Beghi. Machine learning for predictive maintenance: A multi-classifier approach. *IEEE Transactions on Industrial Informatics*, 11(3):812–820, 2014.

[48] T. Tang, Y. Wu, Y. Wu, L. Yu, and Y. Li. Videomoderator: A risk-aware framework for multimodal video moderation in e-commerce. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):846–856, 2021.

[49] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[50] P. Varma and C. Ré. Snuba: Automating weak supervision to label training data. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, vol. 12, p. 223. NIH Public Access, 2018.

[51] Wikipedia contributors. Tf–idf — Wikipedia, the free encyclopedia, 2022. [Online; accessed 28-March-2022].

[52] T. Young, D. Hazarika, S. Poria, and E. Cambria. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, 13(3):55–75, 2018.

[53] X. Zhang, T. Fujiwara, S. Chandrasegaran, M. P. Brundage, T. Sexton, A. Dima, and K.-L. Ma. A visual analytics approach for the diagnosis of heterogeneous and multidimensional machine maintenance data. In *2021 IEEE 14th Pacific Visualization Symposium (PacificVis)*, pp. 196–205. IEEE, 2021.

[54] Y. Zhang, Y. Wang, H. Zhang, B. Zhu, S. Chen, and D. Zhang. Onelabeler: A flexible system for building data labeling tools. In *CHI Conference on Human Factors in Computing Systems*, pp. 1–22, 2022.

[55] Z. Zheng, Z. Lan, B. H. Park, and A. Geist. System log pre-processing to improve failure prediction. In *2009 IEEE/IFIP International Conference on Dependable Systems & Networks*, pp. 572–577. IEEE, 2009.