TEXTWISE: Text Exploration through Interactive Natural Language Processing for Wide-Ranging Insights and Semantic Exploration

Hema Pandey [1], Benjamin Kloepper [1], Ruben Huehnerbein [2], Muskaan Singh [3], Binh Vu [1], Sina Mehraeen [1], Mehrdad Jalali [1], and Swati Chandna [1,*]

[1] Applied Data Science and Analytics, SRH University Heidelberg, Heidelberg, Germany
[2] Industrial Data Analytics, ABB Corporate Research, Ladenburg, Germany
[3] School of Computing, Engineering and Intelligent Systems, Ulster University, Derry/Londonderry, UK
Email: hemapandey.srh@gmail.com (H.P.); kloepper@posteo.de (B.K.); ruben.huehnerbein@de.abb.com (R.H.); m.singh@ulster.ac.uk (M.S.); binh.vu@srh.de (B.V.); sina.mehraeen@srh.de (S.M); mehrdad.jalali@srh.de (M.J); swati.chandna@srh.de (S.C.)
*Corresponding author

***Abstract*—The task of generating accurately labeled datasets in Natural Language Processing (NLP) is notably challenging due to the high cost and extensive time requirements, compounded by the reliance on large volumes of unstructured data scraped from the web. Addressing this, our research introduces a novel framework utilizing Explanatory Interactive Machine Learning (XIL) and Explainable Artificial Intelligence (XAI). This framework enables the dynamic labeling of text data without predefined categories, significantly reducing the dependence on human annotators. Our methodology employs a topic modeling approach that allows a single annotator to label data efficiently with minimal oversight. In testing, this method trained a classifier on as few as 600 documents, achieving a precision of approximately 0.70. This precision is comparable to that of a classifier trained on a fully labeled dataset of 13,000 documents, demonstrating our system's effectiveness while using less than 5% of the labeled data typically required. These findings highlight how our approach not only enhances the transparency of the labeling process but also reduces its resource intensity, offering substantial improvements over traditional methods in both scalability and efficiency. This proof of concept paves the way for broader applications of explainable interactive NLP across various domains.**

***Keywords*—Explainable Artificial Intelligence (XAI), Explanatory Interactive Machine Learning (XIL), text mining, topic modelling, text labeling, unsupervised learning**

I. Introduction

In recent decades, Artificial Intelligence (AI) has made significant strides, with systems achieving human-like performance across various complex tasks. Much of this progress is due to advancements in deep learning, which has transformed fields such as image recognition, autonomous driving, and, notably, Natural Language Processing (NLP) [1, 2]. NLP, a critical area within AI, enables machines to understand, generate, and interact with human language, powering applications from sentiment analysis to real-time language translation. Since its early days in the 1960s, when basic rule-based approaches dominated [3], NLP has evolved dramatically with the development of sophisticated architectures like transformer-based models. These models now underpin state-of-the-art applications that handle nuanced language understanding with a level of fluency that was previously unimaginable.

However, despite these achievements, modern NLP models present significant challenges, particularly regarding transparency and data requirements. Many deep learning models function as "black boxes", making it difficult to interpret their decision-making processes. This opacity can lead to accountability, ethics, and compliance issues, particularly in sensitive applications. Furthermore, NLP models typically require vast amounts of labeled data to achieve high performance. Yet, obtaining large, high-quality annotated datasets is time-consuming and costly, especially as the volume of unstructured text data grows rapidly. This dependence on extensive labeled data has become a persistent obstacle for advancing NLP applications.

Traditional data annotation in NLP involves either manual labeling by experts or crowdsourcing. Manual labeling, though reliable, is highly resource-intensive and impractical at scale. Crowdsourcing, on the other hand, speeds up the process by distributing the workload across multiple annotators. However, crowdsourcing introduces its issues, including inconsistencies in quality, biases from annotator backgrounds, and additional costs associated with managing and verifying data accuracy. Moreover, sharing sensitive data for crowdsourced annotation can raise privacy concerns, limiting the feasibility of this approach for certain datasets. These limitations underscore

Manuscript received August 11, 2024; revised September 6, 2024; accepted February 21, 2025; published August 8, 2025.

the need for more efficient, scalable, and secure methods of data labeling.

One particular challenge with crowdsourced or multi-annotator labeling is variability in quality. The expertise and interpretation of annotators can vary, resulting in inconsistencies requiring further reconciliation processing. Additionally, managing a large team of annotators adds logistical complexities and costs, as quality control mechanisms—such as inter-annotator agreement checks—are needed to maintain labeling accuracy. In many cases, reliance on multiple annotators can introduce biases and errors that ultimately compromise the quality of the labeled dataset.

To address these challenges, this paper introduces a novel framework that enables high-quality data labeling with only a single expert annotator. By integrating Explanatory Interactive Machine Learning (XIL) [4] with Explainable Artificial Intelligence (XAI) [5], the framework reduces the need for multiple annotators and streamlines the labeling process. This approach leverages unsupervised NLP techniques to automatically generate initial labels for large volumes of unstructured text data, minimizing the need for extensive domain expertise upfront. These preliminary labels are then refined through iterative feedback from a single annotator who reviews model explanations, making adjustments to improve label accuracy.

The framework employs a structured workflow to optimize the annotator's input. Initially, a topic model automatically generates initial labels, which are then reviewed by the annotator. Using Local Interpretable Model-agnostic Explanations (LIME), the model provides transparent explanations for each assigned label, allowing the annotator to understand the model's reasoning and make informed corrections. These corrections are fed back into the system to refine the model iteratively. This interactive feedback loop enables the model to learn from each correction, gradually improving its accuracy and reducing the need for additional human input. By relying on a single expert instead of multiple annotators, this framework minimizes the issues of variability and bias while maintaining high labeling quality and consistency.

The proposed framework has broad applicability across various fields where large-scale text labeling is essential yet resource-intensive. For example, in materials science, it could streamline the classification of research articles, lab reports, and experimental data, helping researchers efficiently organize information based on material properties, applications, or composition types. This capability would enable scientists to focus more on discovery and analysis than data curation. Similarly, the framework could be used in healthcare to classify extensive datasets, such as medical literature or patient feedback, supporting healthcare providers in quickly identifying trends in patient outcomes, treatment efficacy, or common health concerns. These applications demonstrate the potential of the framework to enhance scalability and efficiency across fields reliant on large volumes of unstructured text data.

The primary contributions of this study are as follows: first, we present a scalable, explainable, and interactive data labeling framework that minimizes the need for multiple annotators, addressing the challenges of quality inconsistency, cost, and complexity associated with traditional annotation methods. Second, the framework demonstrates how integrating XIL and XAI can support efficient labeling with a single annotator by using automated label generation combined with transparent explanations, enabling scalability without sacrificing label accuracy.

The remainder of this paper is structured as follows: Section II reviews the foundational concepts and related literature in NLP, setting the context for the research and highlighting the existing challenges that this framework addresses. Section III details the XIL framework, describing how it incorporates XAI to improve traditional data-labeling methods. Section IV evaluates the framework's performance, demonstrating its efficiency and reliability compared to conventional multi-annotator approaches. Section V discusses broader implications, potential applications, limitations, and future research directions. Finally, Section VI summarizes the study's contributions and proposes future advancements in scalable, explainable NLP and AI.

II. Related Works

Knowledge Discovery from Text (KDT) has emerged as an essential field for extracting valuable insights from unstructured text data. The demand for efficient and accurate text mining techniques has intensified with the exponential growth of textual information on the internet and in various databases [6]. This section explores foundational concepts and methodologies in text mining, highlighting their importance in analyzing and deriving meaningful information from text data.

One approach that has gained attention in topic modeling is BERTopic, which combines BERT (Bidirectional Encoder Representations from Transformers), embeddings with class-based Term Frequency-Inverse Document Frequency (TF-IDF) weighting [7]. This method builds on traditional topic modeling by using pre-trained BERT embeddings to capture semantic relationships between words. BERTopic processes documents by first converting them into numerical representations, which are then reduced in dimensionality using the UMAP model [8], as clustering models often struggle with high-dimensional data. Afterward, the documents are clustered based on these embeddings through HDBSCAN [9], a density-based clustering technique. This clustering process results in a set of topics, each consisting of related documents. A class-based TF-IDF procedure is then applied to each topic, assigning weights to words to identify the most relevant terms within each topic. The class-based TF-IDF weight $W_{x,c}$ for a term x within a class c is calculated as:

$$W_{x,c} = \left|\left|tf_{x,c}\right|\right| \times log(1 + f_x) \quad (1)$$

where $tf_{x,c}$ is the term frequency of x within class c, f_x is the frequency of x, and $\|\cdot\|$ denotes the magnitude. As illustrated in Fig. 1, the BERTopic architecture involves several stages, allowing it to produce coherent and diverse topics across various datasets.

In parallel with advancements in modeling techniques, XAI methods, such as LIME, address the need for transparency in AI systems. LIME works by constructing simplified, interpretable models for complex model behaviors using perturbed instances of the data [11]. By employing visualization techniques, feature importance analysis, and counterfactual explanations, XAI aims to develop AI systems whose model decisions can be easily understood and trusted by end users [12]. XIL builds on this by enabling user interaction with the model, where users can provide corrections based on explanations to help the model iteratively improve [4]. CAIPI, shown in Fig. 2, is one implementation of XIL. It combines Active Learning (AL) [13] by presenting model predictions and explanations to users, allowing them to clarify ambiguous cases and thereby enhancing model reliability.

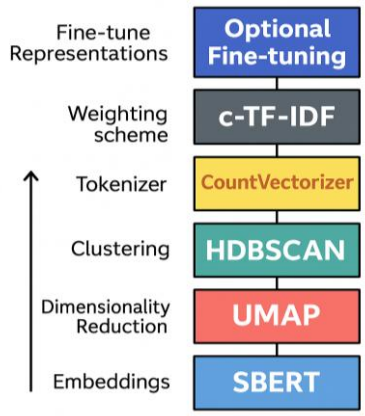

Fig. 1. BERTopic general architecture [10].

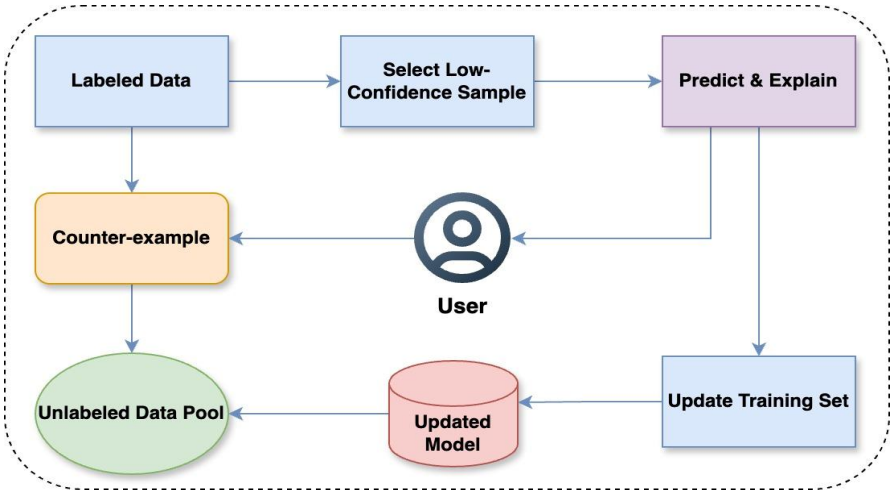

Fig. 2. CAIPI workflow [4].

Lapuschkin *et al.* [14] present a method to distinguish between genuine learning and reliance on spurious data patterns in machine learning models. The authors compare model performance on data from random processes and real data to identify reliance on spurious correlations. Transparent decision rationales in nonlinear models make it easier to assess validity and generalizability. They identify "Clever Hans" behavior in prevalent Computer Vision models, which may rely on non-linear confounding factors, emphasizing trust in the learner. Schramowski *et al.* [15] argue that classification accuracy is insufficient without understanding the reasoning behind predictions. They demonstrate through case studies using Convolutional Neural Networks (CNNs) that human interaction with the model can correct its predictions. Addressing ML trust, XIL was proposed [4], focusing on understanding the model's decision-making process to build trust. Experiments showed that XIL could enhance model prediction and explanatory power, particularly using SVMs. Lertvittayakumjorn and Toni [16] propose Explanation Based Human Debugging (EBHD) for cases where training data are suboptimal. EBHD involves explainability, interactive and human-in-the-loop learning, and knowledge integration. EBHD leverages NLP model explanations to detect bugs and utilizes user input to correct these issues, thus improving model performance. This approach highlights the significance of integrating human expertise to validate and enhance learned models, particularly when dealing with biases or anomalies in the training data.

Recent developments in NLP have popularized transformer-based Language Models (LMs). Devlin *et al.* [17] introduced BERT using a "masked language model" (MLM) pre-training goal to overcome the unidirectionality constraint in previous LMs. Ein-Dor *et al.* [18] analyzed BERT with AL techniques in challenging real-world situations like class imbalance and labeled data scarcity. BERT significantly improved recall when using the AL pipeline across various datasets. Friedrich *et al.* [19] focused on generating LM predictions with case-based reasoning explanations during inference. They introduced Proto-Trex Networks, which enhance model interpretability by adding a prototype layer to transformers, providing prototypical explanations by

comparing predictions to similar labeled training samples. Proto-Trex was also proposed for interactive learning settings. Maarten Grootendorst introduced BERTopic [8] in 2022, a dynamic topic model that creates topic representations in three stages. Unlike static topic models like LDA, it transforms documents into embedding representations, reduces dimensionality for better clustering, and extracts topic representations using a customized class-based TF-IDF variant. Despite model comprehension and evaluation advances, no "silver bullet" exists. Tenney *et al.* [20] provided a comprehensive open-source toolkit called "The Language Interpretability Tool (LIT)". LIT offers a unified user interface and components for visualizing and investigating NLP model behavior, supporting interactive study of entire datasets and individual data points.

Both Lapuschkin *et al.* [14] and Schramowski *et al.* [15] focused on image data using CNNs without addressing natural language data. While Teso and Kersting [4] explored text data (text classification) using SVMs, it did not discuss the effects of XIL on transformer-based language models. Although Lertvittayakumjorn and Toni [16] examined RoBERTa (a BERT variant) for explanation-based human debugging, it assumed model bugs like spurious correlations and labeling errors in text classification tasks without considering incorrect model explanations as a bug. Friedrich *et al.* [19] assessed the Proto-Trex approach for sentiment analysis; however, other NLP tasks remain unexplored. Ein-Dor *et al.* [18] analyzed BERT with AL techniques, yet lacked an explanatory interactive setting. Tenney *et al.* [20] provided the comprehensive open-source toolkit LIT, demonstrating BERT in a sentiment analysis task, although not fully exploring transformer models' potential. None of these works discuss the impact of the XIL approach on other NLP tasks such as text mining without predetermined labels or unsupervised learning tasks like topic modeling and clustering.

While existing methods, such as manual annotation and crowdsourcing, have traditionally been employed for text labeling, they often face scalability, cost, and labeling consistency limitations. Advanced models like BERTopic, which combines BERT embeddings with class-based TF-IDF weighting, improve topic coherence but still rely on significant amounts of labeled data and human intervention for quality control. In contrast, our proposed framework integrates XIL and XAI to streamline and enhance the labeling process. By employing a single annotator who iteratively refines automatically generated labels through model explanations, our approach significantly reduces labor costs, improves label consistency, and provides a transparent annotation process that adapts dynamically to new data. These advancements make our framework uniquely contribute to scalable and efficient NLP data labeling.

III. Methodology

This section proposes a pipeline, as illustrated in Fig. 3. The pipeline is divided into four steps. Step 1 uses the unlabeled pool of documents to train a topic model. Step 2 predicts the topics for the entire dev set documents and selects documents for the AL query. Step 3 explains the topic model's prediction for the queried documents and presents it to the user to obtain the correction. Step 4 generates augmented documents using the correction obtained in Step 3 before adding the augmented set to the initial training data. The topic model is then re-trained using this partially labeled dataset in semi-supervised mode.

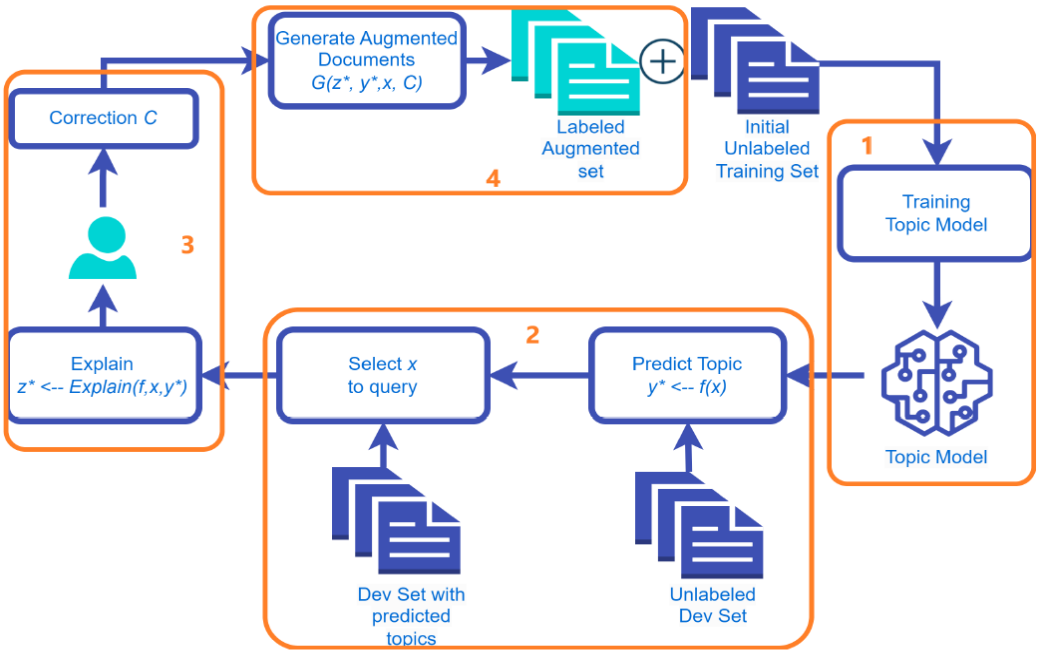

Fig. 3. Modeling pipeline for text mining using XIL.

A. Train Topic Model and Generate Initial Labels

The first step in the modeling pipeline is training a topic model with a fixed number of topics *N*, as identified in the data understanding phase. For datasets with unknown topics, such as the 20Newsgroups dataset, the initial training of the topic model helps determine the appropriate number of topics N for the subsequent pipeline steps. In this example, since the 20Newsgroups dataset contains 20 document categories, *N* is set to 20. After training the baseline topic model, topics are labeled based on the dominant document class within each topic. The distribution of documents for each topic across all document classes is analyzed to assign initial labels. For instance, most documents assigned to Topic 10 belong to the "talk.politics.guns" class. Therefore, Topic 10 is labeled as "politics guns". Similarly, other topics are labeled based on the dominant document class they represent, such as "space", "medicine", "crypto", and "politics mideast", among others. It's essential to note that the number of topics and document categories should be roughly equivalent. However, adjustments can be made during retraining if the number of labels exceeds the number of categories. Additionally, topics can represent more specific distributions within a class, such as "politics-europe" or "politics-usa". These labels are generated for user understanding, but the pipeline only utilizes numeric identifiers (1, 2, 3, ..., *n*) to identify topics, not the string labels ("crypto", "space", etc.).

B. Predict and Query from Dev Set

After labeling the generated topics, the next step involves predicting topics and their probabilities for all documents in the Dev set. Subsequently, documents are queried from the Dev set using the Uncertainty Sampling with Diversity Maximization (USDM) strategy [21] chosen for its effectiveness in selecting informative documents from the unlabeled Dev set in each iteration of AL. The topic probability serves as the measure of informativeness, with USDM aiming to select the least certain instances while maximizing diversity among them. For the 20Newsgroups dataset, each query should consist of the least certain documents from all 20 topics, ensuring adequate representation across all categories. Three variations of the USDM query strategy are tested to determine the most effective approach:

(1) **USDM**: USDM is applied without any modifications in this strategy. The two least certainly predicted documents are queried based on the predicted topic probabilities of each topic from the Dev set.

(2) **USDM with filtering**: This strategy involves filtering out documents with very low topic probabilities (<0.4) before querying. Subsequently, the two least certainly predicted documents are selected from the filtered Dev set, aiming to avoid using documents for which the model is most uncertain.

(3) **USDM with high probability documents**: In this variation, the most certainly predicted documents, along with the least uncertainly predicted ones, are queried from the Dev set based on the predicted topic probabilities of each topic. This approach leverages the behavior of the BERTopic model, which utilizes UMAP for dimensionality reduction. By forcing UMAP to place documents with high and low topic probabilities together, a significant impact on the embedding space is expected, potentially leading to the formation of clusters even among unlabeled documents.

Since 20 topics are generated for the 20Newsgroups dataset, each query comprises 40 documents for further processing. To prevent duplication, selected documents are marked as "picked" after each query iteration. The impact of these three strategies on the framework's performance is discussed in detail in the evaluation section.

C. Explain and Correct

The subsequent step after querying documents for processing involves explaining the topic predictions for the queried documents and presenting the input, predicted topic, and prediction explanation to the human annotator iteratively. Each iteration involves presenting information about 40 queried documents to the annotator, who then provides feedback in the form of correct labels and correction terms. LIME text explainer is employed for elucidating the topic model predictions. Being a model-agnostic explainer, LIME can generate explanations for any probabilistic model. To implement this, a wrapper is created around the BERTopic model to generate probabilities for each text instance. The details of this wrapper's implementation are elaborated in the implementation section. Fig. 4 showcases a sample explanation produced by the LIME text explainer for a specific instance from the 20Newsgroups dataset.

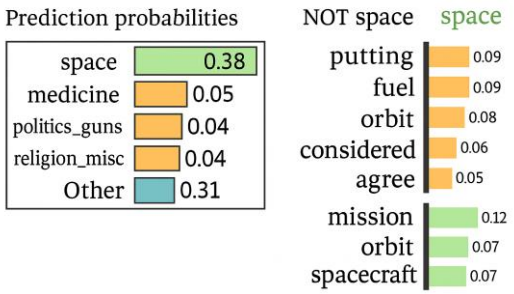

Fig. 4. LIME explanation for a text instance from 20Newsgroups Dev dataset.

D. Generated Augmented Examples

The subsequent step in the modeling pipeline involves accepting the annotator's corrections and generating augmented documents using both the original documents from the query and the corrections provided. Further details about the implementation of this step can be found in the implementation section. In each iteration, 40 documents are queried; for each queried document, M augmented documents are generated using pre-trained BERT-based embeddings. All the augmented documents inherit the same labels as the original document.

Subsequently, corrections provided by the annotator are applied to all augmented documents. Relevant terms are added to the documents, while irrelevant terms are removed. Thus, in a single iteration, a total of 40*M new documents was generated from the 40 originally queried documents. These 40*M newly generated documents and the 40 queried documents are incorporated into the initial training corpus for subsequent re-training. These explanation terms, along with their respective weights, are presented to the human annotator. Based on this information, the annotator provides corrections by specifying: 1) the true label; 2) Relevant terms to be added or retained from the explanation; and 3) Irrelevant terms to be removed. This feedback is crucial for generating augmented documents in the subsequent step of the modeling pipeline.

E. Model Re-Training

Once a small, labeled set has been generated from the correction and data augmentation process, the labeled corpus is integrated into the initial pool of unlabeled training data. This forms the training set for the subsequent iteration. The new training data now includes both labeled documents and the initial unlabeled documents. The topic model is re-trained in semi-supervised mode using this data. BERTopic offers several options to guide the creation of topics toward certain pre-specified topics. Fig. 5 illustrates the step sequence in semi-supervised topic modeling with BERTopic. Semi-supervised modeling enables steering the dimensionality reduction of the embeddings into a space that closely aligns with any provided labels.

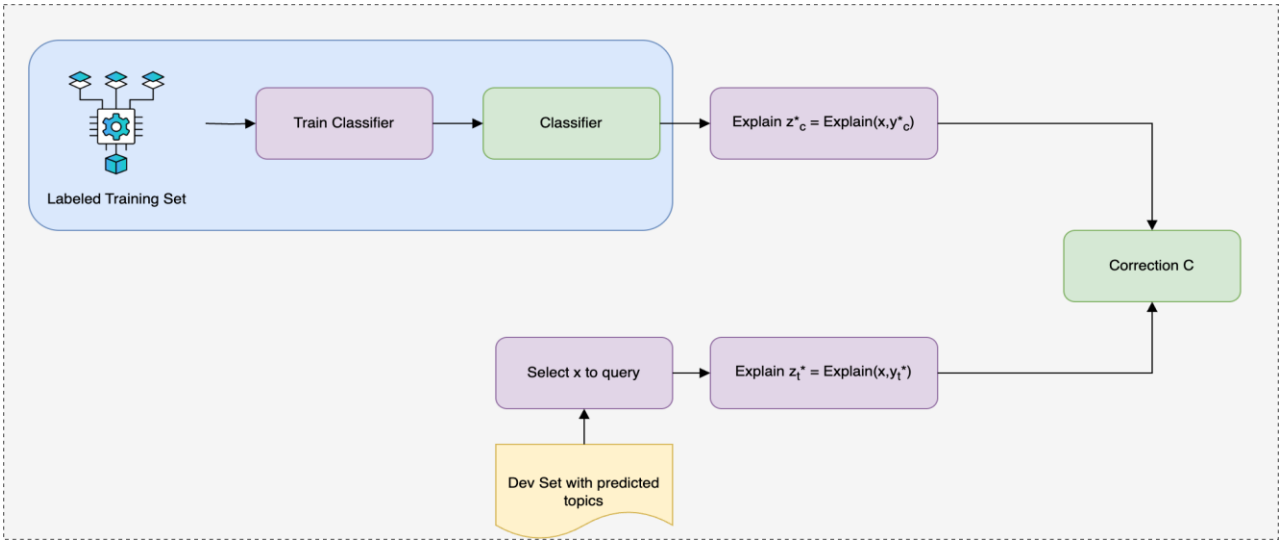

Fig. 5. Feedback simulator.

F. Classification Benchmark for 20Newsgroups Data

A classifier with reasonable performance can be utilized to establish a benchmark for the performance of the labeling task. Linear SVMs with TF-IDF vectorization have achieved benchmark performance for the 20Newsgroups dataset based on the latest data collected by *paperswithcode* [22]. Therefore, a linear SVM classifier is trained on the 20Newsgroups Dataset with TF-IDF using the 20Newsgroups training set. This classifier is then employed to predict and explain the predictions on the test dataset. The explanation terms with weights obtained from explaining the classifier's prediction are stored for later comparison with the Topic Model's explanations produced for test instances.

G. Agreement Index

Now that there are two sets of explanation terms and term weights (indicating the term importance in the explanation), one from the classifier and the other from the topic model, an Agreement Index can be calculated between these two sets (S_a and S_b) of explanation terms and their respective weights. The two sets can be represented as shown in the equations below:

$$[(a_1, w_{a1}), (a_2, w_{a2}), \dots, (a_n, w_{an})] \in S_a \quad (2)$$

$$[(b_1, w_{b1}), (b_2, w_{b2}), \dots, (b_n, w_{bn})] \in S_b \quad (3)$$

To calculate the agreement index, all terms existing in S_b but missing from S_a are added to S_a with weights of 0, and vice versa for S_b. After updating both sets to have the same terms, the agreement index is calculated as shown in the equation:

$$\text{Agreement Index}\,(S_a, S_b) = 1 - \frac{|\Sigma(w_a - w_b)|}{|\Sigma(w_a + w_b)|} \quad (4)$$

The agreement index rewards the occurrence of common terms (agreement) between the two sets and penalizes the non-occurrence (disagreement). The threshold agreement index value is decided by calculating the agreement index between the classifier's explanation and the topic model's explanation on a stratified sample of training data, followed by normalization for this sample. After plotting the distribution of the normalized agreement index, the threshold value was decided to be 0.5. A normalized agreement index value exceeding 0.5 indicates model agreement, while a value below 0.5 indicates disagreement.

H. Feedback Simulator

Providing corrections for every document manually is time-consuming and requires domain knowledge of each category. To mitigate these challenges and reduce reliance on human feedback, a simulation mechanism named Feedback Simulator is introduced. The Feedback Simulator simulates human feedback by training a classifier based on the true labels of the documents. This classifier, being a supervised model trained on the true labels, predicts labels and provides prediction explanations, which are then used to generate correction terms. Fig. 6 illustrates the conceptual diagram of the Feedback Simulator.

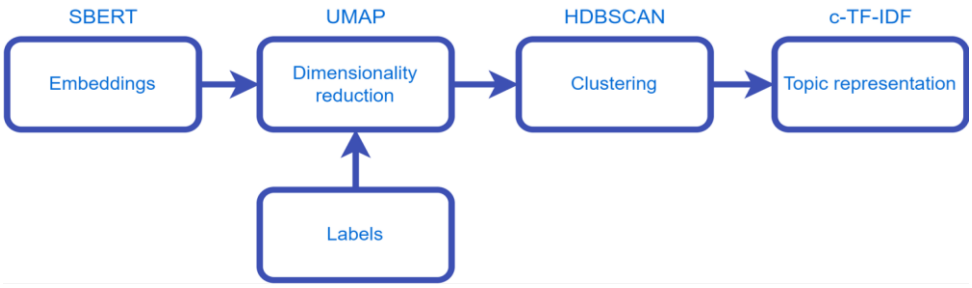

Fig. 6. BERTopic Topic Modeling steps sequence for semi-supervised mode.

A correction comprises three items: 1) True label; 2) Relevant terms to be added to the documents; 3) Irrelevant terms to be removed from the documents. The classifier's prediction y_c^* is considered as the true label of the document. Relevant terms to be added are the explanation terms provided by the classifier but not by the topic model, while irrelevant terms to be removed are the explanation terms given by the topic model but not by the classifier.

To demonstrate the practical application of our methodology, we applied the framework to the 20Newsgroups dataset, which consists of documents across topics like "politics", "technology", and "sports". Initial clusters were generated based on document embeddings in the topic modeling step, forming topic groups that loosely aligned with general categories. During the AL phase, documents with uncertain or ambiguous classifications were selected for review by the feedback simulator, which provided corrections to refine these classifications, allowing the model to learn iteratively. The framework continuously improved its classification accuracy through these iterative corrections, demonstrating its adaptability to large, unstructured datasets with minimal human intervention.

IV. Implementation

This section details the implementation of the methodology, starting with the general setup and then mapping the methodology steps to the Kedro pipelines. It also lists the essential Python packages utilized in the pipeline and provides source code listings for important methods.

A. General Implementation

The methodology pipeline is implemented in *Python 2*, leveraging its widespread usage, open-source nature, and flexibility. Python *Poetry* is employed as the dependency manager to ensure the correct and latest versions of external packages are used consistently throughout the project. *Kedro* serves as the pipeline manager, offering reproducibility, modularity, and portability. The source code is managed using *Azure Git*, accessible through the repository. The implementation relies on various Python packages, with *pandas*, *numpy*, *regex*, and *scikit-learn* being used extensively for data extraction, preparation, and manipulation. *Matplotlib* and *seaborn* are employed for data visualization. The pipeline is highly configurable, supporting different model hyperparameter variations via *Kedro* configuration *YAML* files.

B. Kedro Pipeline Mapping of Methodology Steps

The methodology steps are mapped to Kedro pipelines as follows:

- **Data extraction**: Implemented in the "data extraction" pipeline.
- **Data preparation**: Performed in the "data preparation" pipeline, utilizing pandas, numpy, scikit-learn, and regex packages for extraction and cleaning.
- **Model training**: Training of the benchmark classifier and topic model is conducted in the "model training" pipeline, utilizing scikit-learn's SGDClassifier for the classifier and the BERTopic package for the topic model.
- **Threshold value calculation**: The steps to calculate the threshold value of the agreement index on the training set are executed in the "explain evaluate train" pipeline.
- **Label and topic prediction on Dev set**: Implemented in the "explain evaluate dev" pipeline, which also prepares the Dev dataset for AL query.
- **AL query, feedback simulation, correction, and re-training**: Conducted in the "query compare correction retrain" pipeline.
- **Model evaluation after retraining**: Evaluation of the retrained model is performed in the "explain evaluate dev retrained" pipeline.

Fig. 7 highlights the one-time steps in blue and the iterative steps in green. Since Kedro does not support running pipelines iteratively, the iteration process is manually applied in "kedro batch.py" under the notebooks folder in the Kedro pipeline directory.

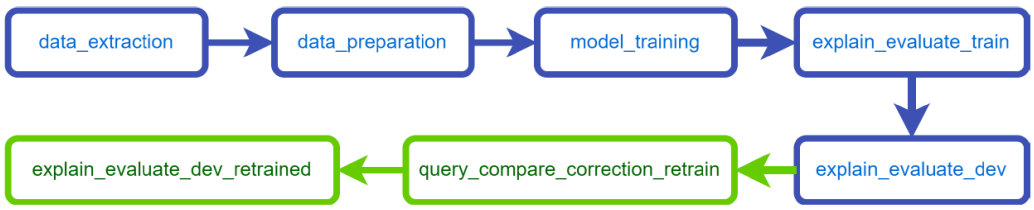

Fig. 7. Kedro pipelines.

C. Important Software Packages

Key Python packages used in the implementation include:

- **Word representations**: Created using *scikit-learn*'s *TfIdfVectorizer*, which transforms text raw content into features and generates a sparse matrix of tf-idf weights for each word in a document. Stop words are removed using the stop words parameter.
- **Benchmark classifier**: Implemented with *scikit-learn*'s *SGDClassifier*, tuned with a learning rate of 0.001 and l2 penalty. Integrated with the *TfIdfVectorizer* using *scikit-learn*'s Pipeline object.
- **Topic model**: Implemented using the *BERTopic* package, with modifications to fix random state issues in the default configuration. Consistency in word representations is maintained by reusing the *scikit-learn* TF-IDF vectorizer.
- **Explanations**: Generated using *LIME* text explainer, supporting *scikit-learn* classifiers. A wrapper around the topic model is created to produce input and output compatible with *LIME*.
- **Agreement index**: Implemented as per the algorithmic pseudo-code provided, facilitating the comparison of two explanation sets with weights and words.
- **Data augmentation**: Utilizes the *nlpaug* library, specifically the Contextual Word Embeddings Augmenter, which inserts/substitutes words using contextual word embeddings like BERT, DistilBERT, RoBERTA, or XLNet.

D. Model Evaluation

Following the re-training step, the task is to assess the framework's performance. As topic modeling is an unsupervised task, evaluating its performance after each iteration poses challenges due to the absence of a reference point or benchmark performance indicator. Conventional topic modeling or clustering evaluation metrics like coherence and perplexity are unsuitable since the objective is to label the dataset rather than cluster the documents. These metrics do not provide a reliable assessment of whether the labels accurately represent the document contents. To address this challenge, a classifier is employed as the reference point for the labeling task.

V. Data Preparation

In the CRISP-DM methodology, the data understanding phase follows the business understanding phase, ensuring alignment between goals and data insights. Initially, a general Exploratory Data Analysis (EDA) is conducted to comprehend various aspects of the input dataset, such as volume, distribution across categories (Fig. 8), and quality of text. Additionally, an initial understanding of underlying themes or topics in the unlabeled data is crucial for determining the number of topics for modeling.

The 20Newsgroups dataset used in this study is a widely recognized dataset in text classification and clustering research. It consists of approximately 18,828 documents across 20 distinct newsgroups, each representing a different topic, such as "politics," "technology," or "sports". The dataset is organized into files where each file corresponds to a specific newsgroup. Messages within the dataset include only the "From" and "Subject" headers, with duplicate messages removed to ensure data quality. Each document is assigned a newsgroup label, which serves as the classification target for text processing. This dataset is publicly available and can be accessed on Kaggle at https://www.kaggle.com/datasets/crawford/20-newsgroups.

Outliers, including empty documents and those exceeding 1000 words, were identified and considered for removal in the data preparation step. Observations revealed the presence of numbers, newline, and tab characters, which needed cleaning. Additionally, lengthy documents were trimmed to enhance processing efficiency, as excessively long documents can increase time and model complexity without significant gains in accuracy.

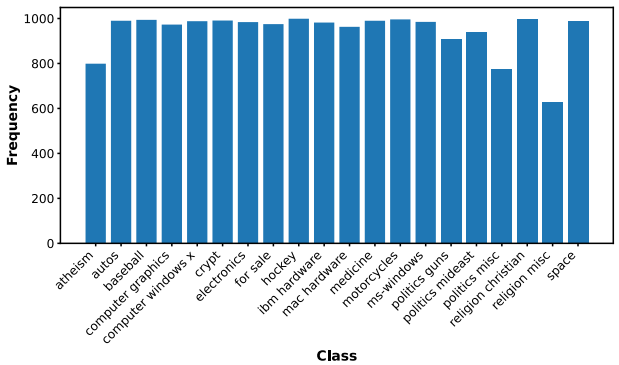

Fig. 8. Class distribution of 20Newsgroupss Data.

In the initial phase of the data preparation pipeline, data cleaning is conducted to ensure the data is ready for subsequent analysis. This process involves addressing issues identified during the data understanding phase. Initially, numeric characters, newline characters, tabs, and punctuation are removed from the dataset. Following this,

the number of words per document is calculated to identify outliers based on document size, which are filtered out along with any empty documents. Once the dataset is cleaned, it is partitioned into three subsets: train, dev, and test. The train set is utilized for the initial training of the topic model, while the dev set serves for user feedback and model refinement. Lastly, the test set is employed to evaluate the performance of the entire framework. The partitioning of the clean dataset into train, dev, and test sets follows a ratio of 70%, 15%, and 15%, respectively. Fig. 9 illustrates the sequential steps involved in the data cleaning process, ensuring clarity and coherence in the workflow. Additionally, Fig. 10 depicts a sample text document post-cleaning, demonstrating the effectiveness of the cleaning steps applied.

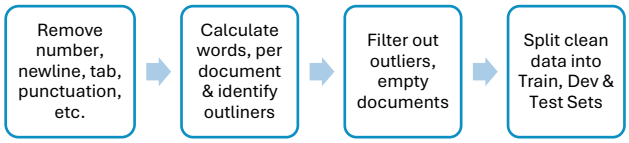

Fig. 9. Data cleaning steps.

i am sure some bashers of pens fans are pretty confused about the lack of any kind of posts about the recent pens massacre of the devils actually i am bit puzzled too and a bit relieved however i am going to put an end to non pitts-burghers relief with a bit of praise for the pens man they are killing those devils worse than i thought jagr just showed you why he is much better than his regular season stats he is also a lot fo fun to watch in the playoffs bowman should let jagr have a lot of fun in the next couple of games since the pens are going to beat the pulp out of jersey anyway i was very disappointed not to see the islanders lose the final regular season game pens rule

Fig. 10. Document text obtained for a sample document after data cleaning steps.

In the data preparation pipeline, text preprocessing plays a pivotal role in ensuring the quality of the input data. As part of this process, the document texts were initially converted to lowercase. Subsequently, tokenization was employed, breaking down the document text into smaller units known as tokens. Following tokenization, the removal of stop words became imperative to reduce the dimensionality of the corpus dictionary. Sklearn vectorizers offer predefined stop word lists for English, facilitating this elimination process based on the presence of words in the stop word list. Once preprocessing is complete, the next step involves feature extraction from the preprocessed text, where the document text is transformed into numerical representations understandable by machine learning models. This conversion is achieved using the Vector Space Model with TF-IDF. The Vector Space Model represents text documents or objects as vectors of identifiers, with document vectors constructed from the document text. Vectorization was carried out using scikit-learn's TfIdfVectorizer.

After completing the data cleaning step, 17,920 clean documents were obtained. To facilitate modeling and evaluation, distinct datasets are necessary. For the AL query process, 20 documents per iteration are needed, amounting to approximately 600 documents for 20–30 iterations. However, a specific AL query configuration mandates 40 documents per iteration, resulting in approximately 2,000 documents for the development set. To organize the data effectively, it is partitioned into three main sets: the train set comprising 70% of the data, utilized for training the initial topic model; the dev set, comprising 15% of the data, used for iterative retraining of the topic model after each AL query; and the test set, consisting of the remaining 15% of the data, employed for evaluating the entire framework. Instances within each set are chosen randomly. Further details regarding the metadata of these partitions can be found in Table I.

TABLE I. METADATA FOR TRAIN, DEV AND TEST SPLIT

Metadata	Train	Dev	Test
Percentage split	70	15	15
Number of documents	12544	2688	2688

VI. EVALUATION

A. Evaluation Using Agreement Index

In our evaluation strategy, after each iteration of the modeling pipeline, a topic model is generated using semi-supervised training with partial labels. This model assigns labels to the test set, and explanations for these labels are compared with those provided by the classifier using an agreement index. While BERTopic is utilized for topic modeling due to its effectiveness, explaining its predictions using LIME proves time-consuming, making this strategy impractical given hardware limitations. Hence, an alternative evaluation approach is sought to assess the achievement of business objectives.

To illustrate the labeling process, consider a document from the 20Newsgroups dataset initially categorized under the broad topic of "politics". In the first pass, the framework assigns this label based on prominent keywords within the text, such as "government" and "policy". However, during the AL phase, the framework identifies terms such as "rights" and "equality" within the document, which could imply a more specific focus within the political sphere, such as "civil rights". This ambiguity prompts the feedback mechanism to request human input, allowing the annotator to refine the label to "civil rights". As a result, the framework updates its understanding, enabling it to apply more precise labels to similar documents in subsequent iterations.

To evaluate how well our framework aligns with baseline models, we also performed comparisons across multiple documents where label assignments were initially unclear. For instance, a document in the "sports" category contains phrases like "game", "team", and "score", which the baseline classifier categorizes as "sports". However, the framework goes further, identifying additional terms like "season" and "playoffs", allowing it to assign a more specific subtopic of "sports playoffs". This refinement demonstrates the framework's ability to enhance general labels by recognizing terms that contribute to more nuanced classifications, providing a closer match with the document's actual content.

For instance, a document in the "sports" category contains phrases like "game", "team", and "score", which the baseline classifier categorizes as "sports". However, the framework goes further, identifying additional terms like "season" and "playoffs", allowing it to assign a more

specific subtopic of "sports playoffs". This refinement demonstrates the framework's ability to enhance general labels by recognizing terms that contribute to more nuanced classifications, providing a closer match with the document's actual content.

Another example involves a document initially labeled as "technology". Upon review, the framework identifies the terms "software", "update", and "version", and categorizes the document as relating to "software development". In the feedback phase, the annotator confirms this subtopic label, enabling the framework to recognize similar patterns in other documents. As this iterative process continues, the framework becomes increasingly accurate in distinguishing between general topics like "technology" and more specific classifications like "software development", improving the overall consistency and accuracy of the labeling.

B. Evaluation Using Classifier

To objectively evaluate the framework's performance in generating labeled data from unannotated text, we adopt a classification approach. We compare our framework's classifier with a benchmark classifier trained on true labels from the newsgroup dataset. Recent research suggests Linear SVMs with TF-IDF vectorization achieve benchmark performance for the 20Newsgroups dataset. This approach is feasible within our hardware limitations and offers straightforward evaluation through standard statistical measures.

C. Experimental Setup

In the implementation of our framework, we conducted experiments with three different USDM-based query strategies as shown in Table II. Additionally, we explored variations in the correction strategy, which comes into play only during simulated feedback scenarios, not with human annotators. *Unconditional Correction*: Correction terms and labels are provided for all documents regardless of agreement or disagreement between the feedback-simulating classifier and the topic model. *Correction at Model Disagreement:* Correction terms are provided only for documents where the models disagree. However, labels are provided for all documents. Agreement or disagreement is determined based on the normalized value of the agreement index between the explanation terms of the feedback-simulating classifier and the topic model. An agreement is indicated by a normalized agreement index value greater than or equal to 0.5, while disagreement is indicated by a value less than 0.5.

TABLE II. EXPERIMENTAL SETUP

Experiment ID	Query Strategy	Correction Strategy
1	USDM	Unconditional correction
2	USDM with filtering	Unconditional correction
3	USDM with high probability	Unconditional correction
4	USDM	Correction at model disagreement
5	USDM with filtering	Correction at model disagreement
6	USDM with high probability	Correction at model disagreement

D. Benchmarking

In the benchmarking phase, two benchmarks are established for evaluation:

- *Benchmark 1*: A Linear SVM classifier trained on 13,000 randomly selected documents (70% of the original labeled data) from the 20Newsgroups dataset with actual category labels.
- *Benchmark 2*: A Linear SVM classifier trained on a stratified sample of 600 documents (15 documents from each class) from the 20Newsgroups dataset, representing approximately 3.33% of the original labeled data.

Both benchmarks are tested on a test set consisting of 2,688 labeled instances. Table III presents the classification report for Benchmark 1 and Benchmark 2 with accuracy of 0.69 and 0.57, precision of 0.70 and 0.57, recall of 0.67 and 0.56, and F1-Score of 0.65 and 0.54, respectively.

TABLE III. CLASSIFICATION REPORT FOR BENCHMARK 1 AND BENCHMARK 2

Experiment ID	Accuracy	Precision	Recall	F1-Score
Benchmark 1	0.6927	0.7018	0.6725	0.6552
Benchmark 2	0.5721	0.5480	0.5575	0.5434

VII. RESULTS

This section presents the outcomes of experiments conducted in the previous section individually, followed by a summary table comparing all experiments. For each experiment, 40 documents per iteration were selected for correction, resulting in 1200 labeled documents by the end of 15 iterations. The performance changes of classification evaluation measures were recorded iteratively across all experiments. Fig. 11 graphically presents the evaluation, which involved plotting four classification metrics (accuracy, precision, recall, and F1-Score) against the iterations of model training. The x-axis represents iterations, and the y-axis denotes the metric values. The green line indicates precision, orange represents precision, blue signifies recall, and brown illustrates the F1-Score. Dotted lines of the same colors highlight the benchmark values discussed earlier. Each experiment is individually detailed, showing the metrics plotted for 15 iterations. A comparison with benchmark values is provided for each experiment, helping to evaluate performance improvements. The results demonstrate the performance trends and effectiveness of different combinations of query and correction strategies in the iterative labeling process.

Fig. 11(c) illustrates that Experiment 3, employing a combination of USDM with high probability documents and unconditional correction, achieved the highest accuracy (0.5066), recall (0.4899), and F1-Score (0.4707). This strategy selects uncertain documents alongside certain ones, influencing the topic model's training to cluster documents more effectively, resulting in improved evaluation metrics. Observing precision values across experiments, all surpassed the precision of Benchmark 2, which used a stratified sample of 600 documents from the training set. Experiment 5 achieved the highest precision

(0.6942) by employing USDM with filtering and correction only when models disagree. This strategy removes uncertain documents, leading to a more representative training set for the topic model and subsequent classifier. Notably, this precision was achieved using less than 5% of the documents compared to Benchmark 2, showcasing the framework's efficiency in achieving comparable precision with fewer labeled documents.

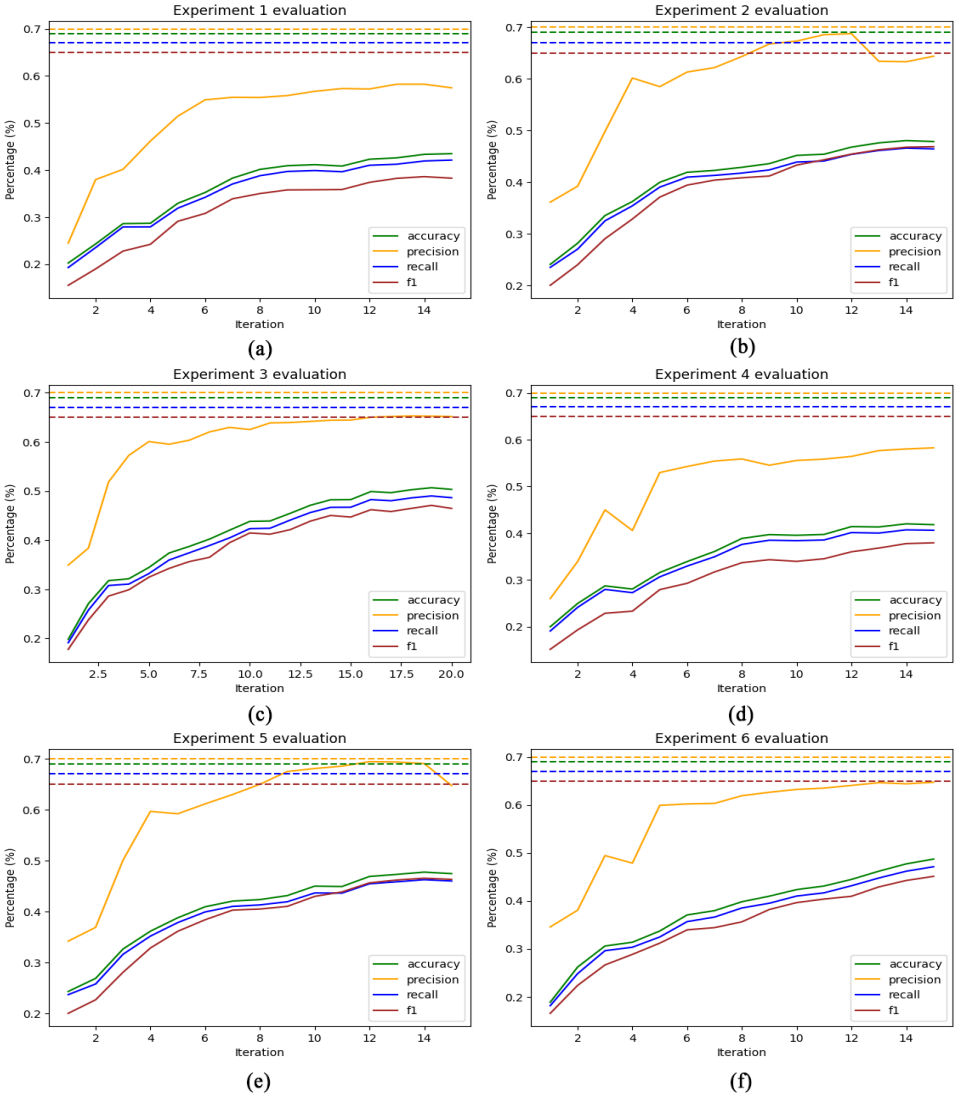

Fig. 11. Evaluation results.

VIII. Discussion

The framework successfully accomplishes the research goal of generating labeled data from an unlabeled dataset by implementing an XIL-based text mining approach. Experiment 3 yielded the highest accuracy (0.5066), recall (0.4899), and F1-Score (0.4707) by combining USDM with high probability documents and correcting all documents in a query, irrespective of model agreement. On the other hand, experiment 5 achieved the highest precision (0.6942) by using the query strategy of USDM with filtering and providing correction only when the models disagreed. This precision level matches the benchmark precision obtained with a training set of 13,000 labeled documents, despite training the evaluation classifier with only around 600 documents, which is less than 5% of the total. Comparing the number of labeled documents used for training with the precision achieved, it is evident that the framework requires fewer documents to achieve similar precision compared to supervised classification models. This section addresses the research question: "*How to create a prototype framework for text mining using an Explanatory Interactive Learning approach*?" Sub-questions RQ(a) through RQ(d) are examined below in the context of the experiment results.

RQ(a) assesses whether identifying labels using unsupervised NLP techniques can diminish the reliance on prior domain knowledge for text labeling tasks. By consistently assigning labels from a topic model to as few as 600 documents, the framework could train a classifier that matched the precision results of a classifier trained on a fully labeled dataset of 13,000 documents. This approach

eliminates the need to define categories upfront if a human were to perform the labeling task.

RQ(b) evaluates whether applying Explanatory Interactive NLP methods to text mining problems can achieve performance levels comparable to, if not better than, non-interpretable and non-explanatory learner methods. While the method falls short in terms of overall accuracy, recall, and F1-Score, it attains benchmark levels of precision (approximately 0.70) when compared to the benchmark classifier trained in a non-interpretable and non-explanatory setting.

RQ(c) investigates whether Explanatory Interactive NLP for text mining tasks can reduce the labeling data cost, considering the amount of labeled input data and user interaction required. The precision achieved by the evaluation classifier (trained on 600 labeled documents) matches that of the benchmark classifier (trained on approximately 13,000 labeled documents). This similar precision level was achieved using less than half the number of documents, indicating that the framework demands less data than supervised classification models to yield comparable results, thereby reducing labeling costs.

Compared to existing models, our framework presents several key advantages for efficient, high-quality labeling of unstructured text data, especially in scenarios with minimal labeled data and limited annotator involvement. Unlike traditional approaches, such as crowdsourcing or manual annotation, which are often costly and prone to inconsistencies, our framework leverages XIL and Explainable AI (XAI) principles to enhance labeling precision and scalability. For instance, tools like BERTopic perform topic modeling effectively but still require substantial human oversight to ensure quality, especially with large datasets. This requirement is met by iteratively refining labels through a dynamic feedback loop, which allows a single annotator to guide the model's learning process efficiently.

Compared with methods like Proto-Trex, which introduces prototype-based explanations for model predictions, or LIT, which provides interpretability tools primarily in sentiment analysis and text classification, our approach extends interactive learning to unsupervised tasks such as topic modeling and clustering.

This makes it highly versatile for scenarios where predefined labels are unavailable or limited. The integration of LIME further supports this process, making the framework accessible to non-expert annotators while ensuring reliable label refinement across iterations.

Moreover, the framework's adaptability opens possibilities for diverse applications beyond traditional NLP tasks. In healthcare, for example, it could streamline the classification of clinical documents and research articles, thereby aiding healthcare professionals in managing unstructured data more effectively. Similarly, the framework could help researchers categorize technical papers or patents in materials science by automatically identifying emerging research trends and themes, facilitating knowledge discovery, and accelerating innovation in the field.

Future research could explore integrating this framework with domain-specific ontologies or knowledge graphs, further enhancing labeling accuracy in specialized domains. Additionally, examining the application of this approach across other sectors, such as finance or legal, could demonstrate its broader applicability for organizing and extracting insights from large volumes of domain-specific textual data. By incorporating domain knowledge into the labeling process, the framework could be adapted to capture more nuanced themes, making it a powerful tool for interactive text mining across industries.

IX. Conclusion and Future Work

This study presents a novel framework using Explanatory Interactive Learning to improve the labeling of unstructured text data. By combining unsupervised topic modeling with limited human feedback, the framework effectively reduces the need for extensive labeled datasets. Experimental results demonstrate that our framework achieves a precision level comparable to traditional supervised classifiers trained on much larger labeled datasets. For instance, in Experiment 5, the framework achieved a precision of 0.6942 using only around 600 labeled documents—less than 5% of the data required by benchmark models to achieve similar performance. This efficiency demonstrates the potential of the framework to reduce labeling requirements by up to 95%, while maintaining comparable labeling quality to more data-intensive methods. Our approach, which utilizes models like BERTopic and explanations through LIME, enables a single annotator to guide the labeling process iteratively. The interactive learning mechanism, coupled with explainability, ensures the labels are accurate and trustworthy, enhancing model transparency. The framework's adaptability has promising applications across various fields. In materials science, it can assist in categorizing research papers, technical reports, and patents according to material properties or methods, facilitating knowledge discovery. It can support patient data categorization in healthcare, such as grouping clinical notes by symptoms or treatment outcomes, enabling more efficient data handling in clinical settings. Additionally, in domains like finance and legal research, this framework could aid in managing vast datasets by identifying and clustering relevant text topics with minimal annotation costs.

To expand this work, future research could explore integrating domain-specific knowledge resources, such as ontologies, to enhance the model's ability to recognize complex industry-specific terms. Moreover, incorporating advanced Large Language Models (LLMs) could further improve the framework's adaptability, allowing it to capture finer semantic nuances across specialized texts. Another promising avenue is to evaluate this framework with diverse datasets and real-world human annotations, enabling it to adapt dynamically and maintain high precision even in unique or evolving domains. By refining the interactive learning and topic modeling components, this framework holds significant potential to transform scalable text labeling across diverse fields.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

HP conducted the research; MS, BV, SC, SM, and MJ wrote the paper; RH, BK and MJ provided the expertise and topics for the project; all authors had approved the final version.

FUNDING

Work by Hema Pandey, Benjamin Klöpper and Ruben Hühnerbein was partially supported by the EU project EXPLAIN, funded by the Federal Ministry of Education and Research (grant 01—S22030A).

REFERENCES

[1] J. McCarthy. (2004). What is artificial intelligence. Stanford University. [Online]. Available: http://www.formal.stanford.edu/jmc/whatisai.html

[2] R. M. Cichy and D. Kaiser, "Deep neural networks as scientific models," *Trends in Cognitive Sciences*, vol. 23, no. 4, pp. 305–317, 2019.

[3] J. Hutchins. (December 2005). The history of machine translation in a nutshell. [Online]. 20(2009). p. 1. Available: https://mt-archive.net/10/Hutchins-2014.pdf

[4] S. Teso and K. Kersting, "Explanatory interactive machine learning," in *Proc. the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, 2019, pp. 239–245.

[5] D. Gunning, "Explainable Artificial Intelligence (XAI)," *Defense Advanced Research Projects Agency (DARPA)*, vol. 2, no. 2, 2017.

[6] A.-H. Tan *et al.*, "Text mining: The state of the art and the challenges," in *Proc. the Pakdd 1999 Workshop on Knowledge Discovery from Advanced Databases*, 1999, vol. 8, pp. 65–70.

[7] M. Grootendorst, "BERTopic: Neural topic modeling with a class-based TF-IDF procedure," arXiv preprint, arXiv:2203.05794, 2022.

[8] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform manifold approximation and projection for dimension reduction," arXiv preprint, arXiv:1802.03426, 2018.

[9] L. McInnes, J. Healy, and S. Astels, "HDBSCAN: Hierarchical density based clustering," *J. Open Source Softw*., vol. 2, no. 11, 205, 2017.

[10] M. Grootendorst. (Aug. 2024). The algorithm. *BERTopic*. [Online]. Available: https://maartengr.github.io/BERTopic/algorithm/algorithm.html

[11] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?' Explaining the predictions of any classifier," in *Proc. the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1135–1144.

[12] D. Gunning and D. Aha, "DARPA's Explainable Artificial Intelligence (XAI) program," *AI Magazine*, vol. 40, no. 2, pp. 44–58, 2019.

[13] B. Settles, "Active learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 6, no. 1, pp. 1–114, 2012.

[14] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R. Müller, "Unmasking clever Hans predictors and assessing what machines really learn," *Nature Communications*, vol. 10, no. 1, pp. 1–8, 2019.

[15] P. Schramowski, W. Stammer, S. Teso, A. Brugger, H.-G. Luigs, A.-K. Mahlein, and K. Kersting, "Right for the wrong scientific reasons: Revising deep networks by interacting with their explanations," arXiv preprint, arXiv:2001.05371, 2020.

[16] P. Lertvittayakumjorn and F. Toni, "Explanation-based human debugging of NLP models: A survey," *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 1508–1528, 2021.

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

[18] L. E. Dor, A. Halfon, A. Gera *et al.*, "Active learning for BERT: An empirical study," in *Proc. the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 7949–7962.

[19] F. Friedrich, P. Schramowski, C. Tauchmann, and K. Kersting, "Interactively providing explanations for transformer language models," arXiv preprint, arXiv:2110.02058, 2021.

[20] I. Tenney, J. Wexler, J. Bastings *et al.*, "The language interpretability tool: Extensible, interactive visualizations and analysis for NLP models," arXiv preprint, arXiv:2008.05122, 2020.

[21] Y. Yang, Z. Ma, F. Nie, X. Chang, and A. G. Hauptmann, "Multi-class active learning by uncertainty sampling with diversity maximization," *International Journal of Computer Vision*, vol. 113, pp. 113–127, 2015.

[22] Papers With Code. (Aug. 2024). Text classification on 20NEWS. [Online]. Available: https://paperswithcode.com/sota/text-classification-on-20news