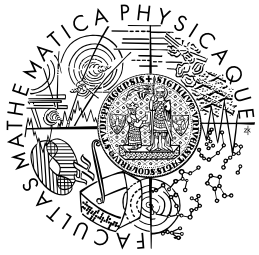


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Petr Laitoch

Text Classification with Limited Training Data

Institute of Formal and Applied Linguistics

Supervisor of the master thesis: RNDr. Jiří Hana, Ph.D.
Study programme: Computer Science
Study branch: Computational Linguistics

Prague 2021

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ....

Author's signature

RNDr. Jiří Hana, Ph.D., my supervisor, deserves acknowledgement and heartfelt gratitude for constant support, feedback and guidance on this thesis.

Title: Text Classification with Limited Training Data

Author: Petr Laitoch

Institute: Institute of Formal and Applied Linguistics

Supervisor: RNDr. Jiří Hana, Ph.D., Institute of Formal and Applied Linguistics

Abstract: The aim of this thesis is to minimize manual work needed to create training data for text classification tasks. Various research areas including weak supervision, interactive learning and transfer learning explore how to minimize training data creation effort. We combine ideas from available literature in order to design a comprehensive text classification framework that employs keyword-based labeling instead of traditional text annotation. Keyword-based labeling aims to label texts based on keywords contained in the texts that are highly correlated with individual classification labels. As noted repeatedly in previous work, coming up with many new keywords is challenging for humans. To accommodate for this issue, we propose an interactive keyword labeler featuring the use of word similarity for guiding a user in keyword labeling. To verify the effectiveness of our novel approach, we implement a minimum viable prototype of the designed framework and use it to perform a user study on a restaurant review multi-label classification problem.

Keywords: NLP text classification weakly supervised learning

Contents

Introduction

Text classification is a task in natural language processing consisting of assigning categories (sometimes called labels) to text based on its content. Stand-alone text classification is becoming increasingly important across industry by allowing automatic organization and understanding of texts such as emails, social media posts, support tickets, news articles and customer reviews. Classification of restaurant reviews is illustrated in Figure 1. Many tasks in natural language processing, including chatbot intent detection, sentiment analysis and spam detection are instances of text classification.

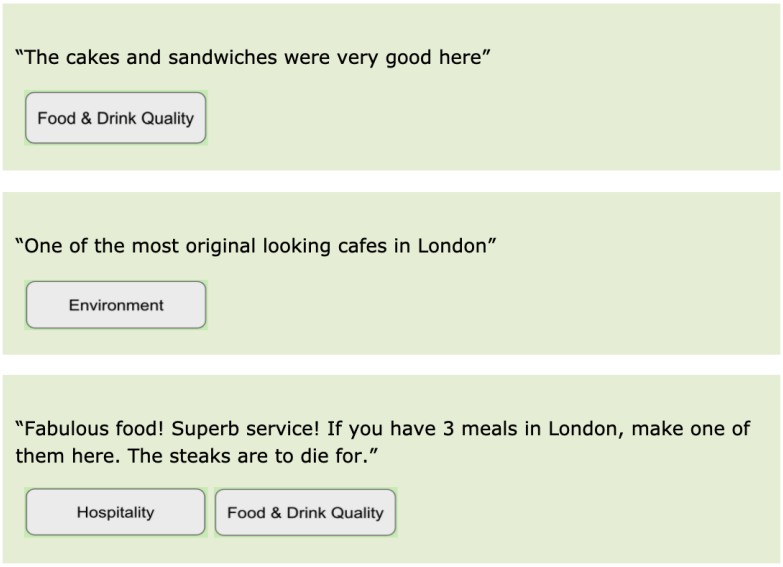


Figure 1: Examples of Text Classification

Supervised learning is the typical method used for classifying texts. A sufficiently large in-domain annotated dataset is needed to perform supervised learning. Often, an annotated dataset is not readily available. To acquire an annotated dataset, one usually hires contractors as annotators and has them annotate data points from an unannotated dataset one text at a time. This process of human labeling is costly and time-consuming. For some tasks, domain experts, whose time is limited, are required to label the data. Furthermore, if requirements for the text classification task change in time, it is often necessary to repeat the whole annotation process.

Multiple research efforts are underway to reduce annotation effort in machine learning. Transfer learning (subsection 1.4.2) transfers knowledge gained when solving one machine learning problem onto a different but related problem. Weakly supervised learning (section 2.1) collects noisy labels from various sources and ensembles them into a single classifier. Active learning (subsection 2.4.2) iteratively determines which data instances to present to an annotator for labeling to best improve the classifier being trained. Interactive learning (section 2.2) provides human annotators with tooling to annotate in a more efficient manner. Transfer learning is the only one of the methods currently in the mainstream.

In this thesis, we concentrate on minimizing the annotation effort required to

classify texts. To achieve this goal, we design an interactive keyword-based text classification framework. No annotated data is required to classify texts. Instead, interactive learning is used to gather noisy signals which are then turned into a text classifier by weakly supervised learning. A high-level diagram of the framework is presented in Figure 2.

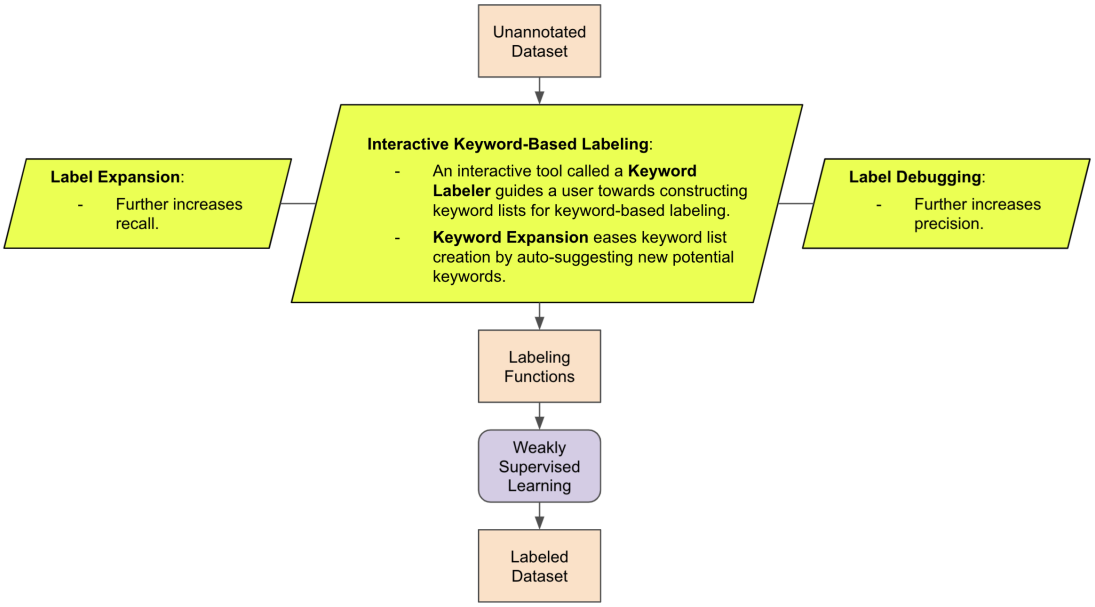


Figure 2: High-level Diagram of our Interactive Keyword-Based Text Classification Framework

The framework is based upon *Keyword-Based Labeling*, which is used to noisily label the unannotated dataset, many texts at a time. As demonstrated in Figure 3, keyword-based labeling aims to gather keywords whose presence in a text from a particular domain indicates that the text belongs to a particular label with a high level of accuracy. Labels are then automatically assigned when keywords from the label's keyword list are present in the text.

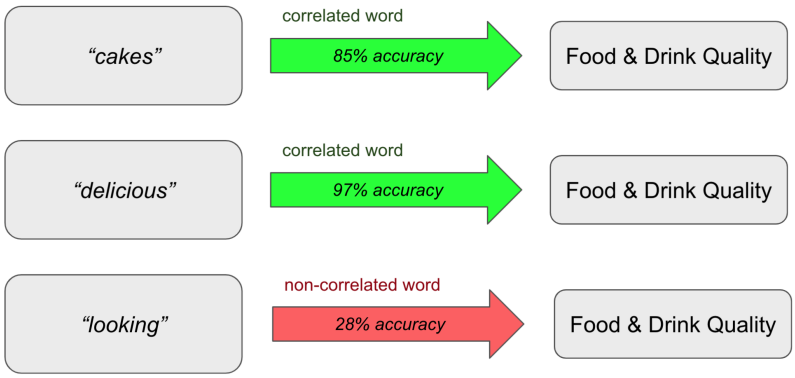


Figure 3: Keyword-Based Labeling

Constructing keyword lists for keyword-based labeling is a non-trivial task. We propose the use of an interactive tool, called a *Keyword Labeler*, that would guide human annotators towards constructing the accurate keyword lists as quickly

and easily as possible. A description of the keyword labeler is present in section 3.1. Besides keyword list management and unannotated dataset integration, an important component of the keyword labeler is *Keyword Expansion*. Keyword expansion auto-suggests new potential keywords to add to the keyword list based on previously identified keywords. As a result, the annotator can add a large group of related keywords to the list based on only a few seed keywords.

Keyword lists constructed using the keyword labeler can be used directly as a text classifier. However, we suggest the use of methods from areas such as weak supervision, transfer learning and active learning in our framework:

- *Weakly Supervised Learning* is used to construct text classifiers from keyword lists. We use a weak supervision concept called a labeling function. A labeling function programmatically labels an input by a classification label or abstains. We create a labeling function for each keyword. Using weakly supervised learning, we then de-noise and ensemble the labeling functions and train a single classifier.

- *Label Expansion* utilizes texts that have been labeled so far to infer sets of texts that should receive that same label. Each such set should be explainable. Small subsets of the inferred sets of texts are presented interactively to the user for manual verification. Verified sets are either used as additional labeling functions or can be used for keyword expansion, to further increase explainability. Label expansion expands the amount of labeled texts in the dataset and thus increases recall.

- *Label Debugging* is a feature of the interactive framework that helps to identify mislabeled texts and provides methods for repressing mislabelings. As a result, label debugging increases precision while attempting to maintain recall.

To validate that our interactive keyword-based text classification framework is a viable alternative to traditional text classification via annotation and supervised learning, we implement a minimum viable keyword labeler prototype and use it to conduct a user-study, where users construct keyword lists for classifying restaurant reviews. Aided by findings from the user study, we build specifications for an improved non-prototype version of the keyword labeler. Implementing the improved keyword labeler and building the full framework featuring weakly supervised learning, label expansion and label debugging is left for future work.

To summarize, our main contributions are as follows:

1. We survey existing methods for reducing annotation effort applicable to text classification. (chapter 2)

2. We design an interactive framework for text classification with no annotated data. The framework design combines ideas gathered from the survey, prototype and user study. (chapter 3)

3. We implement a prototype keyword labeler, featuring keyword expansion via word similarity. (chapter 4)

4. We conduct a user study where users classify restaurant reviews into predefined topics using the keyword labeler prototype. (chapter 5)

1. Background Concepts

In this chapter, we explain background concepts needed to understand the rest of the thesis. In section 1.1, we introduce basic machine learning concepts. In section 1.2, we discuss the manual process of obtaining annotated data. In section 1.3, we provide formulas for evaluating both binary and multi-label classification and for computing inter-annotator agreement. In section 1.4, we talk about today's natural language processing trends.

Newly defined terminology is written in **bold-face**. Readers familiar with machine learning and natural language processing may skip this chapter.

1.1 Machine Learning

Definition

Machine learning was coined in 1959 by Arthur Samuel, who stated that "it gives computers the ability to learn without being explicitly programmed". A more recent 1997 definition by Tom Mitchell states that: "A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E."

Machine learning algorithms require **training data** as input in order to generate a **machine learning model** during a process called **training**. Training is done once and can be computationally expensive. The trained model acts as a function $f : \mathbf{X} \rightarrow \mathbf{Y}$, where $\mathbf{X}$ is the input space and $\mathbf{Y}$ is the set of possible predictions. The trained model can be used to generate **predictions** for previously unseen data (i.e. data not part of the training data). This is called **inference**. In addition to each prediction, many models also output a **confidence score** which estimates the probability of the prediction being correct.

Typically, machine learning algorithms require that data points be transformed into **feature vectors**. Feature vectors are vectors of a fixed size that ideally encode all the information from a data point that is required to make the correct prediction. Feature vectors are generated during a phase called **feature extraction**.

Model Training

In machine learning tasks, we aim to **train a model** that will act as a predictor function $f : \mathbf{X} \rightarrow \mathbf{Y}$, where the input space $\mathbf{X}$ is a set of all feature vectors and the output space $\mathbf{Y}$ is the set of all possible predictions/annotations.

When training a machine learning model, the general expectation is that the model will learn to **generalize**. Model generalization means that the model will learn the underlying statistical patterns in the data to make predictions instead of simply remembering the training data. The opposite of generalization is **overfitting**. When a model over-fits on the training data, it makes close-to-perfect predictions on the training data but it is unable to make good predictions on any unseen data.

For a model to have a chance at generalization, the input space must contain data from the same **domain**, i.e. the same statistical distribution. For example, if a model is trained to distinguish between cat and dog pictures and we suddenly show a picture of a spaceship, the model cannot generate a reasonable prediction.

Training can be viewed as automatic tuning of model parameters, for example of weights of a neural network. The amount of such parameters is called **model size**. A **complicated model** has many parameters while a **simple model** has few parameters. As a general rule, overly complicated models are prone to overfitting, while overly simple models are not capable of capturing complexities of the task at hand.

Datasets

A **dataset** is a collection of **data points**. All data points in the dataset should contain the same type of data. A dataset may be annotated or unannotated. An **annotation** of a data point is a prediction for the data point that we already know. Typically, annotation is performed either by human annotators or by previously-trained high-quality ML models. Annotations are used as part of the training data.

- An annotated data set $\mathfrak{D}^{\mathbf{A}}$ with n data points is defined as

  $$\mathfrak{D}^{\mathbf{A}} := \{(x_i, y_i) : i \in \{1, ..., n\}\},$$

  where x_i are data points and y_i are annotations. All annotations in the dataset should be of the same type. We say an annotated dataset is **balanced** (or symmetric) when each label / class has approximately the same amount of data points.

- An unannotated data set $\mathfrak{D}$ with n data points is defined as

  $$\mathfrak{D} := \{x_i : i \in \{1, ..., n\},$$

  where x_i are data points.

Evaluation

Evaluation of **model performance** is the computation of **evaluation metrics** (see more information in section 1.3) by comparing model predictions to gold-standard data. High-quality annotation can be used as **gold-standard data** during evaluation.

Gold-standard data are the best available predictions we can gather under reasonable conditions. Typically, gold-standard data are manually created by humans, or are results of an empirical measurement.

In addition to training data, machine learning algorithms typically also require a set of **hyper-parameters** to train the model. Hyper-parameter choice can significantly affect model performance. Many factors including the type and amount of training data available affect optimal hyper-parameter choice. The process of finding the hyper-parameters needed to train the best-performing model is called **hyper-parameter tuning**.

Dataset Splitting

When performing a single machine learning task, it is a common practice to use random sampling to split a dataset into three subsets. A **training set**, a **development set** and a **test set**.

The training set is always used for model training, it is used as training data. To prevent over-fitting, we should always evaluate the trained model on the development set. Since none of the data in the development set was seen during training, we can assume that good model performance on the development set corresponds to a model that generalizes well.

To find the best performing machine learning model for the machine learning task, we need to select the best machine learning algorithm and tune its hyper-parameters. To this end, multiple models are trained on the training set and are then evaluated on the development set. The best-performing model on the development set is chosen as the best performing machine learning model for the task.

Since multiple models were evaluated on the development set and we chose the best-performing one, it is possible that our model was over-fit on the development set. To verify actual final modal performance, we need to evaluate on the test set. The test set must be used only for final model evaluation and should be kept secret until then.

Machine Learning Tasks

A **machine learning task** defines the type of predictions that are required to solve the problem at hand. For example, in the "spam or ham" classification task, we must predict for each data point (= email text) if the email should be assigned the class "spam" if the email is spam. Otherwise, we assign the class "ham". Many different machine learning algorithms exist to solve various types of machine learning tasks.

Types of machine learning tasks differ by requiring various types of data as input and generating various types of predictions as output. Some of the most common machine learning tasks are classification, regression and clustering.

- **Supervised learning** is one of the most common types of machine learning tasks. It trains a model directly from an annotated dataset's input-output pairs (x_i, y_i). It includes the **regression** and **classification** tasks.
  - In **classification**, the output space $\mathbf{Y}$ is a set of n labels l_i: $\mathbf{Y} = \{l_i : i \in \{1, ..., n\}\}$. A trained classification model is called a **classifier**. An element of the output space is called either a **class**, a **label**, a **tag**, or a **classification category**.
  - In **regression**, the output space $\mathbf{Y}$ is an interval on the set of real numbers defined by $min_y <= y <= max_y$.
- In **semi-supervised learning**, there is both an annotated and an unannotated training set. Semi-supervised learning algorithms can distinguish two steps. First, a model is trained on unannotated data in the **pre-training**

step. Then, the model training continues on the annotated data in the **fine-tuning** step.

- In **unsupervised learning**, only an unannotated training set is used. A typical example of unsupervised learning is **clustering**, where the goal is to group data points into a number of similar groups called **clusters**.

Classification

As defined in the previous section, classification can formally be viewed as a function $f : \mathbf{X} \to \mathbf{Y}$, where $\mathbf{X}$ is the input space and $\mathbf{Y}$ is the output space. As shown in Figure 1.1, we distinguish multiple types of classification depending on the output space $\mathbf{Y}$. More specifically, on the amount of labels L and on label cardinality K:

	K=2	$K \geq 2$
L=1	binary	multi-class
$L \geq 1$	multi-label	multi-output†

†also known as multi-target, multi-dimensional

Figure 1.1: Classification Nomenclature

- **Binary classification** assigns a single label to each data point from only two possible classes. Formally, $|\mathbf{Y}| = 2$, for example $\mathbf{Y} = \{\text{“}true\text{“}, \text{“}false\text{“}\}$ or $\mathbf{Y} = \{\text{“}spam\text{“}, \text{“}ham\text{“}\}$.

- **Multi-class classification** assigns a single label to each data point from multiple (more than two) possible classes . Formally, $|\mathbf{Y}| > 2$, for example $\mathbf{Y} = \{\text{“}cat\text{“}, \text{“}dog\text{“}, \text{“}mouse\text{“}, \text{“}elephant\text{“}\}$.

- **Multi-label classification** assigns multiple (more than one) label to each data point, making a binary classification decision for each label. For example, $\{\text{“}sports\text{“}, \text{“}politics\text{“}, \text{“}not\_finance\text{“}, \text{“}not\_crime\text{“}\} \in \mathbf{Y}$. Multi-label classification with L labels can be cast into L binary classification problems – one for each label.

  - **Tagging** is essentially equivalent to multi-label classification. In tagging, we assign a subset of labels to each data point. Formally, $\mathbf{Y} \subseteq \mathbf{T}$, where $\mathbf{T}$ is the set of all tags. For example, let $\mathbf{T} = \{\text{“}sports\text{“}, \text{“}politics\text{“}, \text{“}finance\text{“}, \text{“}crime\text{“}\}$. A data point x may be assigned the tags $\mathbf{T_x} = \{\text{“}sports\text{“}, \text{“}politics\text{“}\}$. This is equivalent to multi-label classification assigning the labels $\{\text{“}sports\text{“}, \text{“}politics\text{“}, \text{“}not\_finance\text{“}, \text{“}not\_crime\text{“}\}$, as above.

- **Multi-output classification** is multi-label and multi-class classification put together. It assigns multiple labels, each from many possible classes. An example is classifying an image by both foreground animal and background scenery, e.g. $\{\text{“}cat\text{“}, \text{“}forest\text{“}\}$, or $\{\text{“}dog\text{“}, \text{“}apartment\text{“}\}$.

1.2 Annotation

As indicated in the previous section, annotation plays a significant role in machine learning. Annotation is a manual process, where people provide supervision to machine learning models by manually assigning labels to individual data points in a dataset. These people are called annotators.

Some annotation tasks have simple specifications, such as classifying images as either cat pictures or dog pictures. However, many tasks have more complicated and less understandable specifications. These tasks require the expertise of **domain experts** to perform the annotation correctly. For example, classifying medical documents may need the expertise of a doctor. Furthermore, many tasks contain ambiguities concerning label boundaries. Communication with the client or manager in charge of the task will be necessary to resolve the ambiguities. In that case, we consider the client or manager to be the domain expert. Time of domain experts is usually too limited to be able to annotate a dataset sufficiently large to allow for sufficient supervised model performance.

Due to the limited time of domain experts, annotation is performed by annotators hired part-time. Often, multiple annotators are hired for the same task at once. It is necessary that all annotators understand the dataset and labels equally correctly and that they can ask questions about ambiguities in the instructions, which frequently arise. Continuous communication among the multiple annotators and between annotators and domain experts is necessary to achieve good quality annotation. The annotators are usually tasked to annotate a small shared subset of the data to compare their performance. If annotation is done incorrectly, or if the problem specifications change, often the only reasonable option is to repeat the whole annotation process.

Annotation guidelines

Annotation guidelines are a form of task specification written for annotators. For annotation guidelines to be easily readable and understandable, they are often written as a collection of keywords and example data point excerpts for each classification label. Annotators annotate data points based on their human interpratation of these guidelines.

Throughout the process of specifying the task, it is useful to do **dataset exploration**. It is useful to both randomly sample and search the dataset. Searching is useful for examining rare categories. However, searching provides a biased view of the dataset. For unbiased exploration of rare categories, we must randomly sample in bulk, quickly identify rare categories and only then can we examine them more closely.

Annotation guidelines may also be used to reduce all possible ambiguities that can occur during annotation. Annotators gain insight into the dataset structure while annotating and can formulate precise questions concerning the ambiguities. They can then cooperate with domain experts to resolve the ambiguities and accordingly adapt the guidelines. This should be done in the first iteration of annotation, since changing annotation guidelines may result in the need to re-annotate a large portion of the data.

Inter-annotator agreement

Two annotators should ideally be able to agree on the correct annotation of any text when precisely following annotator guidelines and ask for rulings when not sure. However, various reasons including annotation difficulty, human error and incorrect interpretation of guidelines may lead to differing annotations. Agreement among annotatators is measured via inter-annotator agreement. Cohen's kappa, a popular metric for measuring inter-annotator agreement, is presented in subsection 1.3.1.

With proper training and management of human annotators, inter-annotator agreement can become a proxy measurement of task difficulty – we call this performance benchmark **human-level performance**. For some tasks, achieving close to 100% accuracy is easy for humans and machine learning models should aim to achieve that level as well. For other tasks, often when the task isn't well defined, it is only possible to achieve accuracy as high as inter-annotator agreement allows.

Crowdsourcing

Crowdsourcing [Yuen et al., 2011, Quinn and Bederson, 2011] can be defined as distributed problem-solving over the Internet. A task is outsourced to a crowd of unskilled workers. Crowdsourcing is best applied to tasks that have been previously accomplished by hiring a large temporary workforce. Instead, the tasks are split into smaller, more manageable sub-tasks to be completed by distributed workers over the Internet. Each worker must only need a short introduction to the task. An example of a popular crowdsourcing platform is Amazon Mechanical Turk[1].

To successfully crowdsource a task, one must provide a clear, short and unambiguous task definition. It is necessary to limit the needed attention span of the crowdworker. Crowdsourcing platforms usually provide templates to design simple annotation GUIs. Crowdworkers come and go in a matter of just a few data points and concentrate on quick and easy tasks due to the monetary model. It is not feasible for crowdworkers to perform complicated tasks requiring long training time and reading long guidelines. Only tasks achievable by the general public are possible to crowdsource, domain experts are not available on crowdsourcing platforms.

In the context of machine learning, annotation is a task that fits together with crowdsourcing extremely well. Many annotation tasks can be performed with little training of each individual crowdworker. Crowdworkers can provide bad-quality annotations. Due to the large number and relative cheapness of crowdworkers, a single annotation may be performed by multiple people if needed. With crowdsoursing, it is not possible to cooperate with annotators to iteratively improve annotation guidelines, resolve ambiguities, or present them with instructions and onboarding tasks longer than a few minutes.

[1] https://www.mturk.com/

1.3 Evaluation

We first describe the computation of inter-annotator agreement in subsection 1.3.1. Inter-annotator agreement gives a dataset noise estimate for the annotation, which is equal to the upper bound on any possible machine learning model's accuracy. In subsection 1.3.2, we describe how to compute evaluation metrics for a binary classifier. We further extend the binary classifier metrics to multi-label in subsection 1.3.3.

1.3.1 Inter-Annotator Agreement

In machine learning tasks, it often isn't possible to empirically measure gold-standard classification task labels. In such situations, we must employ human annotators as the only viable method of obtaining annotations to be used as gold-standard data.

Inter-annotator agreement measures how well two or more annotators can make the same annotation decision for a given classification label. One of the most popular metrics for inter-annotator agreement is Cohen's kappa:

- **Cohen's kappa** is defined as

$$\kappa := \frac{p_o - p_e}{1 - p_e},$$

  where p_o is the observed probability of agreement among raters (equivalent to accuracy) and p_e is the probability of agreement among raters expected by random chance.

If $\kappa = 0$, there is no other agreement among raters other than that expected by random chance. $\kappa < 0$ indicates disagreement, $\kappa > 0$ indicates some level of agreement, $\kappa = 1$ indicates complete agreement.

1.3.2 Binary Classification Evaluation

When evaluating binary classification problems, it is common practice to re-state the problem into one with a positive case *CLASS* and a negative case *NOT_CLASS*. Then, the following values can be computed:

- Positive predicted labels are considered **true positives (TP)** when their respective gold-standard labels are positive as well.
- Negative predicted labels are considered **true negatives (TN)** when their respective gold-standard labels are negative as well.
- Positive predicted labels are considered **false positives (FP)** when their respective gold-standard labels are actually negative.
- Negative predicted labels are considered **false negatives (FN)** when their respective gold-standard labels are actually positive.

		Actual class	
		Cat	**Non-cat**
Predicted class	**Cat**	**5 true positives**	2 false positives
	Non-cat	3 false negatives	**3 true negatives**

Figure 1.2: Confusion Matrix Example

Binary classification can be easily evaluated using just TP, TN, FP and FN counts. An easily understandable visualization of these four values is a table layout called a **confusion matrix**. See an example of a confusion matrix in Figure 1.2.

Let the **total data point count** be denoted $total$. It is clear that:

$$total := true\_positives + false\_negatives + false\_positives + true\_negatives$$

The following evaluation metrics can be computed using the values of a confusion matrix and provide more insight:

- **Accuracy** is the ratio of correct predictions to total predictions. It is a very intuitive evaluation metric that works well on balanced and symmetric datasets.

$$accuracy := \frac{true\_positives + true\_negatives}{total}$$

- **Precision** is the ratio of correctly predicted positive data points to total data points predicted positive, i.e. a count of correct positive predictions. High precision relates to a low false positive rate.

$$precision := \frac{true\_positives}{true\_positives + false\_positives}$$

- **Recall** is the ratio of correctly predicted positive data points to total data points actually positive. (i.e. how many positive data points did we predict correctly?)

$$recall := \frac{true\_positives}{true\_positives + false\_negatives}$$

- **F-measure / F-1** is the harmonic mean of precision and recall. For datasets with an uneven class distribution, it is much more useful than accuracy.

$$F - 1 := 2 * \frac{precision * recall}{precision + recall}$$

1.3.3 Multi-Label Classification Evaluation

Multi-label classification evaluation can be performed as a simple extension of binary classification evaluation by using the **per-label evaluation** method, where we evaluate each label separately. Per-label evaluation provides the most insight. However, it is often useful to have aggregate metrics for all labels, e.g. when tuning classifier hyper-parameters across all labels. Some popular aggregate multi-label classification evaluation metrics are as follows:

- A data point is considered an **exact match** when all predicted labels are the same as gold-standard labels.
- A data point is considered a **Top-N match** when N predicted labels with the most confidence match are also gold-standard. Top-1 match and Top-2 match are quite popular metrics.
- **Jaccard index**, also known as Jaccard similarity coefficient, measures the similarity between two sets. The Jaccard index of sets A and B is defined as:

  $$J(A, B) := \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

  In tagging, we compute the Jaccard index of the predicted tag set and gold-standard tag set for each data point. We then average out the Jaccard indices of all data points across the annotated dataset $\mathfrak{D}^{\mathbf{A}}$:

  $$Jaccard\_index(\mathfrak{D}^{\mathbf{A}}) = \frac{\sum_{(x_i, y_i) \in \mathfrak{D}^{\mathbf{A}}} J(x_i, y_i)}{|\mathfrak{D}^{\mathbf{A}}|}$$

- With **macro-averaging**, we can compute aggregated accuracy, precision, recall and F-1 measures across all labels. We do it by computing per-label accuracy, precision, recall and F-1 measures and averaging them across all labels. Thus, all labels are treated equally. This can be unfair with an uneven label distribution.
- With **micro-averaging**, we can compute aggregated accuracy, precision, recall and F-1 measures across all labels as well. However, instead of averaging the computed metrics, we instead sum up confusion matrices for all labels. We then compute single-label metrics for the aggregate confusion matrix. With micro-averaging, rare classes are barely taken into account in the final score.

1.4 Natural Language Processing

Natural language processing, often shortened to NLP, is a subfield of artificial intelligence that grants computers the ability to work with human language. Most NLP problems are solved by using machine learning. Examples of NLP tasks follow:

- **Text Classification**
- **Topic Detection**
- **Dialog Agent Creation**

NLP has drastically evolved in the past couple of years with the adoption of numerous deep learning methods [Young et al., 2017]. Traditional NLP systems relied heavily on hand-crafted features and shallow models such as support vector machines or logistic regression. This evolution was sparked by the success of word embeddings (subsection 1.4.1) and the move from hand-crafting to deep learning models (subsection 1.4.2).

1.4.1 Word Embeddings

Traditional statistical NLP often suffered from the **curse of dimensionality** when representing a language vocabulary of size $|V|$ as a one-hot-encoded $|V|$-dimensional vector. This led to learning **distributed representations** of words in low-dimensional space [Bengio et al., 2003]. These distributed representations are now called **word vectors** or **word embeddings**. "Distributed" comes from **distributional semantics**, the study of statistical distributions in which words occur in a text corpus. Distributional semantics captures a word's meaning purely from its context throughout a text corpus, hypothesising that similar words occur in similar contexts. Words are embedded into a low-dimensional similarity space so that similar words (= words that occur in similar contexts) are close together in the similarity space by the **cosine similarity measure** (= cosine of the angle between two vectors of a vector space), as seen in Figure 1.3.

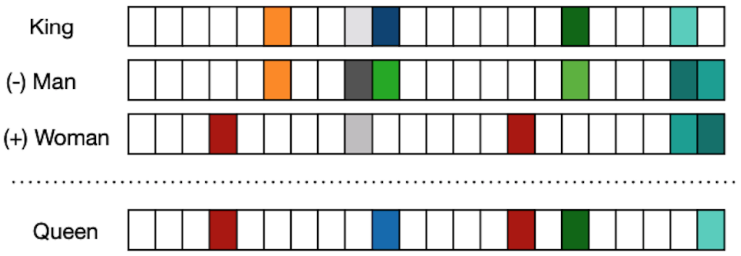


Figure 1.3: Distributional vectors, source: http://veredshwartz.blogspot.sg

Word embeddings are typically used as the first layer of a deep learning model. They are pre-trained by optimizing an auxiliary objective in a large unlabeled corpus.

Pre-computed word embeddings are readily available. They are generated on a large dataset spanning multiple domains. Word embedding **fine-tuning** uses pre-computed word embeddings as a starting point and does more training on a smaller in-domain dataset. Word embeddings can also be trained from scratch on a medium-sized in-domain dataset.

Word Similarity

Word vectors can also be used to compute **word similarity**. Usually, we measure the **cosine similarity** or other similarity measure between word vectors to get a word similarity score.

Word2Vec

Word2vec [Mikolov et al., 2013] is a word embedding system that truly popularized word embeddings. It is a particularly computationally-efficient predictive model for learning word embeddings from raw text. Word2vec uses two learning models, as shown in Figure 1.4. Both are built using a 3-layer neural network consisting of an input layer, a hidden layer and an output layer. After the models train, the computed embeddings are in the hidden layer.

1. **Continuous bag-of-words**: The input is the context of a word: Two preceding and two following words, the output is the word itself.
2. **Skip-gram**: The input is a one-hot-encoded word, the output is the word's context: Two preceding and two following words.

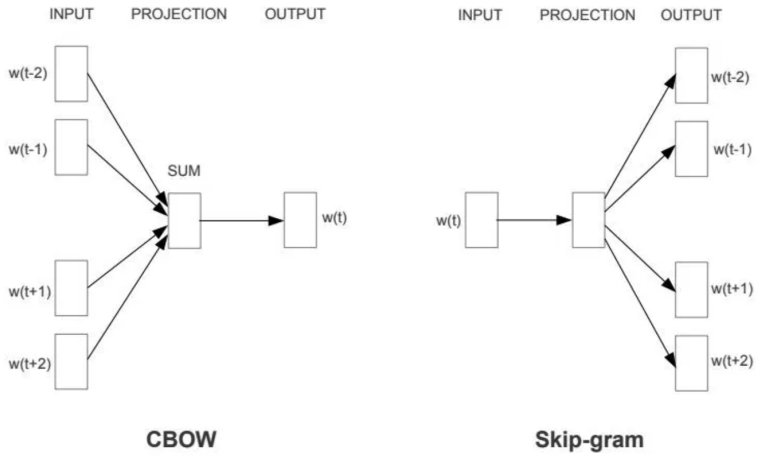


Figure 1.4: Continuous Bag-of-Words and Skip-Gram Diagram

Text Classification

FastText [Joulin et al., 2016] is an library for text representation and classification. It is designed to be fast and lightweight. It is able to both train word embeddings and to classify text.

1.4.2 Transfer Learning

Transfer learning enhances supervised learning by transferring knowledge gained when solving one problem onto a different but related problem. Currently, a popular form of transfer learning is the use of large-scale language models. These models learn from extensive amounts of data and are specific to a given human

language. Since training large-scale models is costly, models for many natural languages are readily available to be fine-tuned to a particular task in their language. Transfer learning offers increased performance compared to supervised learning with equal amounts of training data. However, in-domain labeled data is still required. Currently popular model architectures are:

- ULMFiT [Howard and Ruder, 2018],
- GPT-2 [Radford et al., 2019],
- BERT [Devlin et al., 2018],
- XLNet [Yang et al., 2019].

We choose to describe at least one of the models in more detail: BERT – as it's the most popular to date.

BERT

BERT [Devlin et al., 2018] is an extremely popular language model. It uses both left and right context of a sentence. The transformer architecture [Vaswani et al., 2017] is used to learn language representations using two objectives: BERT has multiple stages of training:

1. Vocabulary selection
2. Pre-training; It is a lengthy processes. BERT needs to be trained for days on a TPU (longer on other hardware). The output of pre-training is a model checkpoint file trained on a given language. There are multiple pre-trained BERT models available for multiple languages. Pre-training code is available.[2] In case a language doesn't have a model available, or if the vocabulary of the available model is too different from the data domain.

   Both of the following objectives are trained at once, as is shown in Figure 1.5.

   (a) **Masked language model**: Tokens from the input sequence are randomly masked, the model learns to predict them. 15% of the tokens in each input sequence are randomly changed as follows:
      i. with 80% probability, the token is replaced by [MASK].
      ii. with 10% probability, the token is replaced by another random token.
      iii. with 10% the token remains unchanged.

   (b) **Next sentence prediction**: Given two sentences, the model must predict if the first sentence A follows the other sentence B or not. The sequences are chosen such that 50% of the time, B actually is the next sentence and 50% of the time, B is a different random sentence. Note that although named "next sentence prediction", two continuous spans of text are fed to the model, not actual sentences.

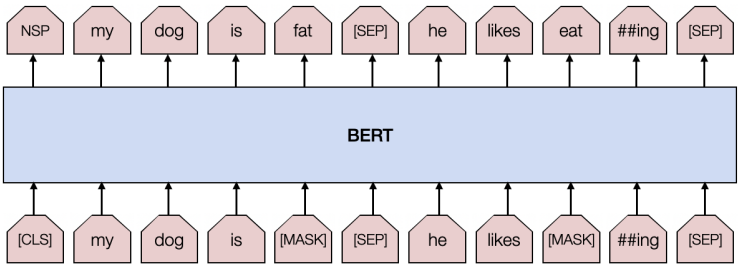


Figure 1.5: Bert Training Input-Output Example

3. Fine-tuning; It is the process of using the pre-trained model to train a model on a different machine learning task. Transfer learning should occur from the pre-trained base model to the fine-tuned model. It is a less resource-consuming process compared to pre-training (only 1 hour on a TPU). A popular BERT fine-tuning implementation is available in the HuggingFace Transformers library.[3] Machine learning tasks fine-tunable for use with the Transformers library include: Sequence classification, token classification, multiple choice and question answering.

1.4.3 Other NLP Concepts

We briefly mention some other natural language processing concepts important for this thesis:

- **Model explainability** – the ease with which it is possibly to explain model decisions to humans.
- **Word Sense Induction** – classification of words into word senses.
- **Search engines** – tools used to index collections of text documents. They allow to look up specific documents using queries. Examples of popular search engine implementations include *Elasticsearch* and *Lucene*.
- **Sentence similarity** can be measured via:
  - Statistical methods researched in search engine literature: TF-IDF, BM25, etc.
  - Pre-trained distributed sentence representations such as: GenSen [Subramanian et al., 2018], InferSent [Conneau et al., 2018], or Universal Sentence Encoder [Cer et al., 2018].
- **Few-Shot Learning** – trains machine learning models with a very small amount of training data. This is contrary to the supervised learning standard of training on large amounts of data.

[2]https://github.com/google-research/bert
[3]https://github.com/huggingface/transformers

2. Literature Survey: Reducing Annotation Effort in Text Classification

This chapter summarizes work done in machine learning that aims to minimize the annotation effort required to develop machine learning models. Our rapid text annotation framework from chapter 3 is highly inspired by works covered in this chapter.

Weakly supervised learning (section 2.1) is an active area of research which minimizes annotation effort using multiple noisy sources. Interactive learning (section 2.2) creates tools for domain experts to allow for fast annotation. In particular, we cover the SLP framework which combines interactive learning with weakly supervised learning for the purposes of text classification. Work extending weakly supervised learning is surveyed in two sections: In section 2.3, there are methods for increasing recall of weak models; In section 2.4, there are methods for increasing precision of weak models.

2.1 Weakly Supervised Learning

Hand-labeling datasets is a significant bottleneck when developing machine learning applications using supervised learning (see section 1.2). Weakly supervised learning aims to minimize annotation effort by moving towards a programmatic or otherwise semi-automatic generation of labels in machine learning.

Weak supervision refers to generating labels that are noisy or otherwise sub-par, hence "weak". There are multiple methods for obtaining noisy labels in a fast and cheap manner. Weak labels from multiple sources are combined together via an ensembling method. The claim of weakly supervised learning is that obtaining these weak labels is considerably cheaper and combining them provides performance on par with traditional dataset hand-labeling.

Multiple types of weak supervision may be considered [Zhou, 2018]:

- **Inaccurate supervision** denotes a scenario where labels are not always **ground-truth** (the ideal expected result in machine learning).
- **Incomplete supervision** deals with training data having ground-truth label only for a subset of the data.
- **Inexact supervision** works with coarse-grained labels that are not fine enough for the task at hand.

In this thesis, we concentrate on inaccurate weak supervision.

2.1.1 Data Programming

The **data programming** paradigm [Ratner et al., 2017b], further developed within the Snorkel framework,[1] [Ratner et al., 2017a] introduces integral techniques of weak supervision.

Labeling Functions

Data programming hopes to facilitate the creation of training data sets quickly and easily by creating **labeling functions**. Instead of annotating texts one-by-one, a programmer can write labeling functions. A labeling function (see examples in Figure 2.1) is a function written in a programming language that takes a data point as input and either assigns a label or **abstains**. Abstaining means that a label on that data point is not assigned. The difference between a labeling function and a classifier is that a labeling function is intended to be short and easy. Furthermore, labeling functions are expected to only label a part of the dataset, oftentimes noisily.

```
def lambda_1(x):
    return 1 if (x.gene,x.pheno) in KNOWN_RELATIONS_1 else 0

def lambda_2(x):
    return -1 if re.match(r'.*not␣cause.*', x.text_between) else 0

def lambda_3(x):
    return 1 if re.match(r'.*associated.*', x.text_between)
            and (x.gene,x.pheno) in KNOWN_RELATIONS_2 else 0
```

Figure 2.1: Example labeling functions written when extracting gene-disease relations from scientific literature, from Ratner et al. [2017b]

Labeling Function Evaluation

Without **gold labels** (labels which are hand-labeled with a reliable ground-truth value), we can measure the following characteristics of labeling functions:

- *Polarity*: The set of unique labels this labeling function outputs.
- *Coverage*: The fraction of the dataset this labeling function labels.
- *Overlaps*: The fraction of the dataset where this labeling function and at least one other labeling function label.
- *Conflicts*: The fraction of the dataset where this labeling function and at least one other labeling function label but disagree.

If gold labels are provided, we can further measure the following characteristics of labeling functions:

- *Correct*: The number of data points this labeling function labels correctly.

[1] http://snorkel.com

- *Incorrect*: The number of data points this labeling function labels incorrectly.
- *Empirical Accuracy*: The ratio of correct to all labels this labeling function labels.

Ensembling Methods

Data programming provides techniques for **denoising** labeling functions by **ensembling** them. Typically, one creates a considerable amount of labeling functions that label the training data inaccurately and/or inexactly. It is encouraged to use many different sources of knowledge to create the labeling functions. Ideally, there should be overlaps among various labeling functions so that denoising can happen successfully.

The ensembling method first estimates the accuracy of the individual labeling functions. Using the estimated accuracies, labeling function outputs are combined together for each data point, so that they output a coherent labeled training set.

We offer a selection of ensembling methods in increasing complexity:

- *Majority vote* – selects the label which was chosen by the majority of labeling functions. When labeling functions label independently with respect to the true label class and have roughly the same labeling accuracies on the dataset, majority voting provides optimal results. Due to the simplicity of majority voting, it is often used as a baseline ensembling method.
- *Weighed voting* – provides optimal ensembling for independent labeling functions with known differing accuracies. Accuracies are used as voting weights. The label whose sum of labeling function weights is the highest is chosen.
- *Data programming* [Ratner et al., 2017b] – provides an ensembling method to estimate accuracies of independent labeling functions. Furthermore, dependent labeling functions may be modeled as well if a user specifies dependency types among the labeling functions.
- *Snorkel*[2] [Ratner et al., 2017a] – uses a generative model to estimate labeling function accuracies even for dependent labeling functions without needing to specify dependencies.
- *FlyingSquid*[3] [Fu et al., 2020] – can find a closed-form solution for estimating labeling function accuracies with dependent learning functions. Compared to Snorkel, this closed-form solution computes systems of simple equations instead of needing to compute stochastic gradient descent. FlyingSquid is two orders of magnitude faster than Snorkel with comparable performance.

Ensembling methods proposed above provide theoretical analysis of their modeling advantage compared to majority voting. The number of labeling functions,

[2]https://www.snorkel.org/
[3]https://hazyresearch.stanford.edu/flyingsquid

their accuracy, abstain rate, ratio of overlaps, number of conflicting labels and more affect modeling advantage when ensembled. In general, labeling functions with at least 0.5 to 0.8 accuracy with overlaps and conflicts are needed for effective denoising.

PU-learning

PU-learning, short for positive unlabeled learning, attempts to learn classifiers given only positive examples and unlabeled data. A recent survey on PU-learning methods is available: Bekker and Davis [2020]. To the best of our knowledge, research on PU learning in the context of labeling function ensembling is yet to be explored.

Snorkel Framework

The **Snorkel** framework [Ratner et al., 2017a] provides a workflow for training machine learning models using data programming and weak supervision. Multiple user studies in cooperation with the industry were performed to verify Snorkel's effectiveness [Ratner et al., 2017a, Bach et al., 2019, Dunnmon et al., 2019].

As seen in Figure 2.2, we expect an unlabeled dataset to be available. Unlabeled data points are preprocessed into a context hierarchy. Snorkel calls user-generated labeling functions on the context hierarchy instead of on the raw data so that pre-processing need not be executed multiple times. Snorkel provides a labeling function interface in the Python programming language. A domain expert should use various weak supervision strategies to write labeling functions. Outputs of labeling functions are stored in a label matrix. Snorkel trains a **generative model** that learns to compute probabilistic labels for each data point. This Snorkel-generated probabilistic training data is used to train a **discriminative model** – an arbitrary machine learning model chosen to solve the underlying machine learning task.

In addition to the mentioned core functionality, the Snorkel labeling function interface provides support for transformation and slicing functions. Transformation functions do data augmentation. Slicing functions identify important dataset subsets.

Snorkel Flow

A new platform developed by the Snorkel team, Snorkel Flow,[4] allows the user to build and deploy AI applications via weakly supervised learning end-to-end. The platform consists of four steps:

1. GUI to Label, augment and programmatically build training data
2. Automatic ensembling
3. Training and deployment of machine learning models
4. Evaluation, analysis and monitoring

[4] https://snorkel.ai/platform/

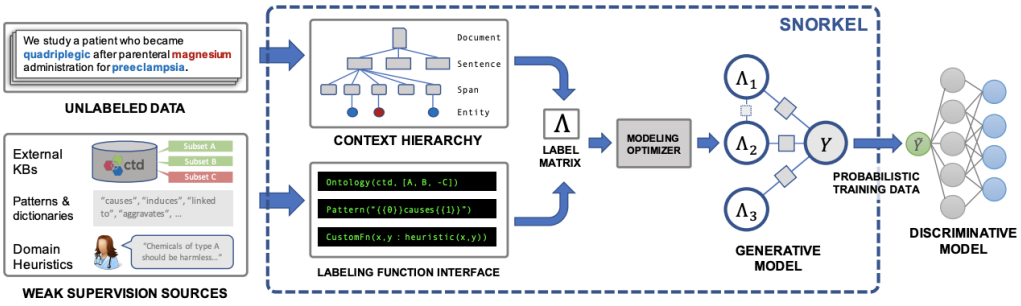


Figure 2: An overview of the Snorkel system. (1) SME users write *labeling functions (LFs)* that express weak supervision sources like distant supervision, patterns, and heuristics. (2) Snorkel applies the LFs over unlabeled data and learns a generative model to combine the LFs' outputs into probabilistic labels. (3) Snorkel uses these labels to train a discriminative classification model, such as a deep neural network.

Figure 2.2: Snorkel Framework Overview

Snorkel Data Labeling Tutorial

The Snorkel Data Labeling Tutorial[5] is the initial resource for new Snorkel users. The tutorial encourages users to employ an iterative process when writing labeling functions. Each iteration consists of: Ideation, refining, evaluation (spot checking performance on training set data points) and debugging. The user should proceed as follows:

1. Explore the training dataset by random sampling. Gather ideas for writing new labeling functions.
2. Write an initial version of a labeling function.
3. Evaluate performance of the labeling function on the training set: Polarity, coverage, overlaps, conflicts, correct, incorrect and Snorkel accuracy estimates. Spot check performance on training set data points.
4. Refine the labeling function. Balance accuracy and coverage in the process.
5. Repeat.

The domain expert is advised to employ keyword search, pattern matching, third-party models, distant supervision and crowd-worker labels as weak supervision strategies.

2.1.2 Weak Supervision Strategies

Labeling functions may be created using various **weak supervision strategies**. These include:

- **Heuristics** – short and simple manually written snippets of source code
- **Distant Supervision** – a supervision strategy involving knowledge graphs

[5]Snorkel Data Labeling Tutorial: https://www.snorkel.org/use-cases/01-spam-tutorial (extracted September 2020)

- **Crowdsourcing** – annotation by many non-experts through an online crowdsourcing platform
- **Existing ML models** – existing models designed for similar tasks and data

In the following sub-sections, we describe individual weak supervision strategies in more detail.

Heuristics

Usually, domain experts manually write labeling functions for text using the following methods:

- Keyword/keyphrase matching
- Regular expressions
- Matching of symbols occurring in text: URLs, emojis, Twitter hashtags, etc.
- Matching of NLP-generated labels: Syntactic tagging, named entity recognition, topic models, etc.
- Knowledge graphs and ontologies

For data modalities other than text, it is frequent to do traditional feature engineering and threshold feature values.

There are multiple recent attempts to ease heuristic labeling function creation for non-programmer experts. Several GUIs have been proposed for this purpose:

- *BabbleLabble* [Hancock et al., 2018] – instructs annotators to provide natural language explanations for each labeling decision. A semantic parser converts the explanations into labeling functions.
- *Reef* [Varma and Ré, 2018] – a system for automatic generation of heuristics for labeling functions from a small annotated and large unannotated dataset.
- *Ruler* [Evensen et al., 2020] – offers a graphical user interface to interactively create labeling functions non-programmatically by span-level text annotation.

The general methods outlined above for writing a text labeling function still leave most of the work up to the domain expert. These methods also rely on expertise in natural language processing which limits possible adoption. The GUIs proposed have not gained any traction after publication, as of yet.

It is a general consensus, that little guidance exists when it comes to labeling function creation. Multiple works, including some by Snorkel authors acknowledge this issue:

- "There is usually little formal structure or guidance for how these labeling functions are created by users." [Cohen-Wang et al., 2019]
- "Little is known about user experience in writing labeling functions and how to improve it." [Evensen et al., 2020]
- "Through our engagements with users at large companies, we find that experts spend a significant amount of time designing these weak supervision sources." [Varma and Ré, 2018]

Distant Supervision

Relation extraction is a task in information retrieval which predicts relations for entities in a sentence. For example, we could extract the *person_born_in_country* relation for the person "George W. Bush" and country "USA" from the sentence "George W. Bush was born in the United States".

Distant supervision [Smirnova and Cudré-Mauroux, 2018, Riedel et al., 2010] is a frequently mentioned weak supervision strategy in the Snorkel Framework. Introduced by Mintz et al. [2009], distant supervision generates training data for a relation extraction task from a large semantic database, usually Freebase. Annotated data is generated by considering all pairs of entities that appear in some Freebase relation. Sentences in a large unlabeled corpus containing both entities are used as distantly-annotated training data.

Distant supervision was also adopted by several other NLP tasks. Husby and Barbosa [2012] use distant supervision for topic classification. Again, a semantic database is used as a source of knowledge. It is possible to aid classifiers with topics present in the semantic database. Chenthamarakshan et al. [2011] employ knowledge bases for text classification. However, to the best of current knowledge, it is not clear how to easily extend distant supervision to an arbitrary task beyond relation extraction. For this reason, although distant supervision is a valid weak supervision strategy, perhaps it can only be used to a limited extend in general.

Crowdsourcing

It is possible to use data programming for crowdworker annotation de-noising. Each crowdworker is considered to be a noisy labeling function. Overlaps needed for labeling function de-noising are obtained by annotating a single data point by multiple crowdworkers. Ensembling of labeling functions gives de-noised labels.

Third party models

We exemplify the use of third party models as a weak supervision strategy on classification of Instagram posts in the fashion domain [Hammar et al., 2018].

As seen in Figure 2.3, multiple commercial APIs are used. Each third party model is considered a labeling function. Additional labeling functions are created by keyword matching to a fashion ontology based on either word embeddings or the Levenstein distance.

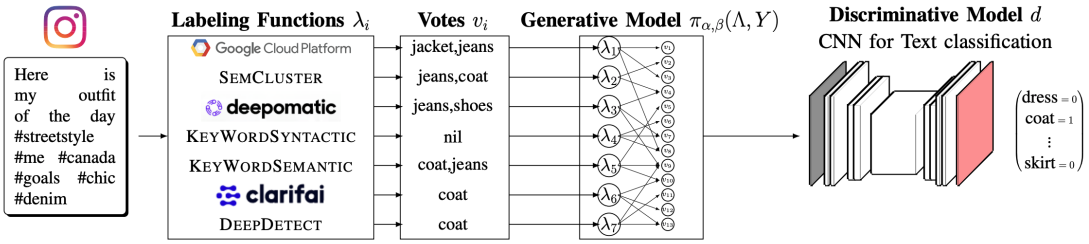


Figure 2.3: Weak supervision pipeline for Instagram fashion post classification, from Hammar et al. [2018]

Weak Supervision in Large Corporations

Large organizations typically have a large range of resources such as trained models, knowledges bases and heuristics available, see Figure 2.4.

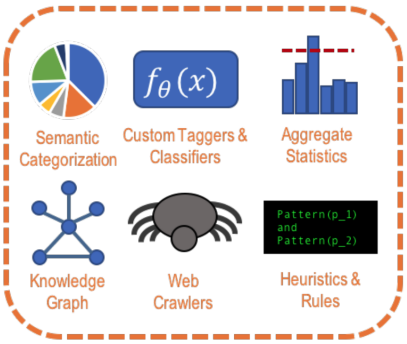


Figure 2.4: Examples of weak supervision resources available on Google's platform, from Bach et al. [2019]

A study of weak supervision at industrial scale by Bach et al. [2019] studies the use of weakly supervised learning in large organizations. Benefits include in particular:

- *Flexible ingestion of organizational knowledge*: Large organizations that deal with machine learning typically have many models that are quite similar. For example, hundreds of similar classifiers may exist for only slightly differing domains or languages. By using these similar classifiers as sources of weak supervision, the study shows how a new classifier may be quickly developed by training on less annotated data with improved results over previous models.

- *Cross-feature production serving*: We call machine learning features **non-servable** if they are unavailable for a production model but available at training time. Non-servable features are often too slow to compute during inference or are otherwise expensive to obtain. **Servable** features are usually inexpensive real-time signals readily available during inference. Weak supervision can allow a transfer of knowledge from non-servable features to servable feature, since the generative model can create a probabilistic

training set of servable features using non-servable features as weak supervision signals. Learning from non-servable features when training a machine learning model is also known as coaching.

2.2 Interactive Learning

Interactive learning [Simard et al., 2014] is a machine learning method where a human teacher guides an interactive learning system while continuously getting feedback from the same system. Interactive learning is a generalization of active learning. An active learning system chooses which data points the user should label and prompts the user to annotate data points – the user acts solely as an oracle. In interactive learning, actions the human teacher can perform are unconstrained. An interactive learning system should be quick and responsive, since a human user is involved.

Guided Learning

Guided learning is a sub-category of interactive learning. Attenberg and Provost [2010] designed a text classification guided learning system where domain experts use a search engine interface to search through unannotated training data to find examples relevant to a specific target class. The domain experts then label search results.

Dialog Intent Detection

A crucial step when creating dialog agents (a.k.a. chatbots) is defining intents and building intent classifiers. An intent represents a high-level purpose – e.g. a set of semantically similar sentences – for which the chatbot can provide the same response. For example, "good morning" and "hi" should both be labeled as "greetings". Characteristics of the intent classification task include:

- Short unstructured texts – people often only write several words or sentences in chat dialogs.
- Lots of unannotated data – historical chat logs are a rich source of unannotated data.
- Lack of annotated data – intents are designed by hand, they are nowhere to be found except by human annotation. In-domain training examples are needed, this is a bottleneck for chatbot development.
- Highly imbalanced classes – very few positive examples present in chat logs, labeling data in sequence is thus expensive.
- High number of classes – tens to hundreds of intents.

Guided Learning for Dialog Intent Detection

Williams et al. [2015] apply interactive learning to intent detection for dialog systems. An example dialog of their system is in Figure 2.5. The user can interactively search through unlabeled data, label training instances, train and evaluate classifiers.

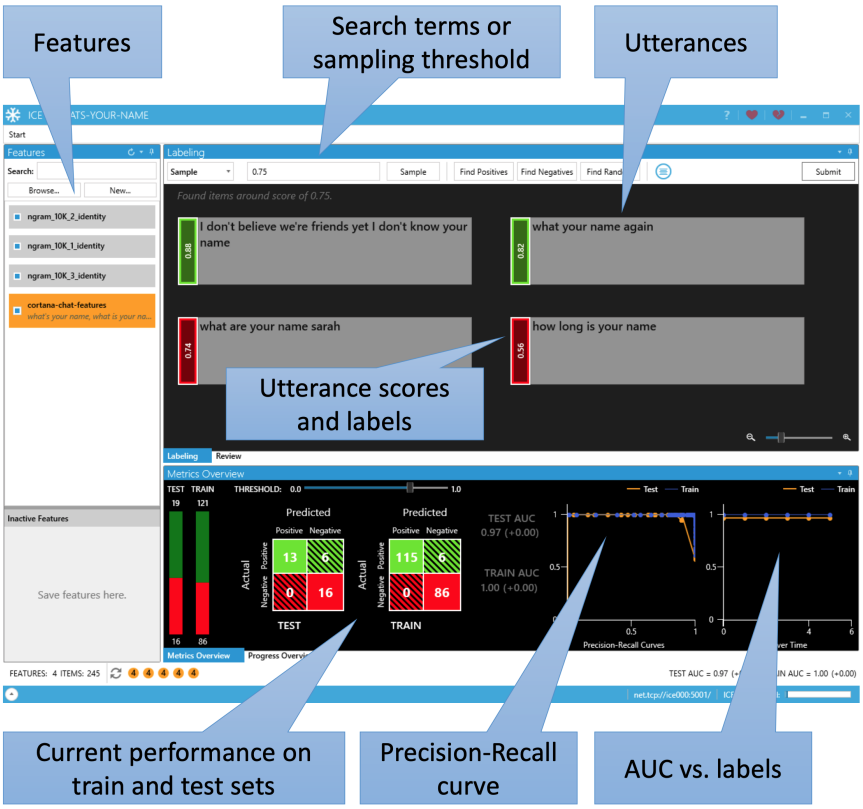


Figure 2.5: ICE [Simard et al., 2014]: An interactive learning tool adapted for intent detection, from Williams et al. [2015]

2.2.1 Search, Label, Propagate (SLP)

Mallinar et al. [2019] combine interactive learning with data programming. They propose the Search, Label, Propagate (SLP) framework to reduce manual labor required for intent detection. Large chat logs must be available. An ElasticSearch search box user interface (Figure 2.6) guides the user's creation of heuristic labeling functions. A human expert actively searches for queries (keywords and phrases) that should correspond to a particular class of the classification problem. The user is asked to label a subset of the results found by the search engine after each search. The framework then automatically creates labeling functions based on the search queries and labels.

As shown in Figure 2.7, chat logs are first indexed and stored in a data store. This is done using ElasticSearch. In the search step, the user uses a search engine to search for chat messages that belong to a given intent. In the label step, a subset of the search results are given to the user for labeling. Not all search results are shown to the user. In the propagate step, the rest of the search results

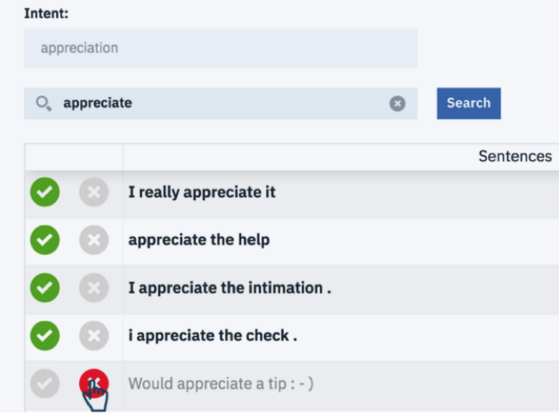


Figure 2.6: GUI of the SLP framework prototype, from Mallinar et al. [2019]

that were not shown to the user get labeled automatically. The user may continue again from the search step so that more data gets labeled.

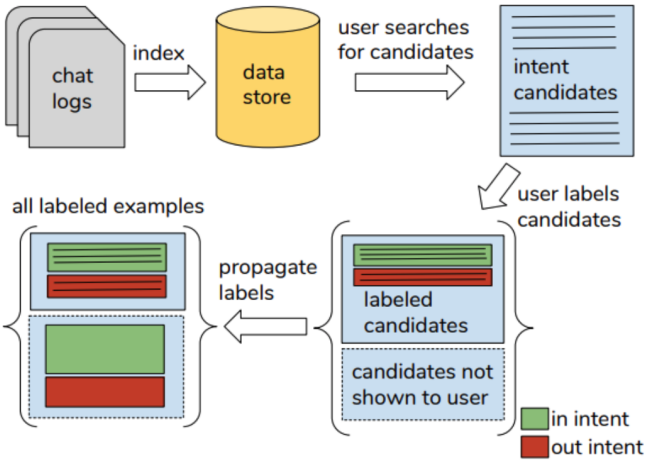


Figure 2.7: Search, Label, Propagate Overview, from Mallinar et al. [2019]

Search, Label and Propagate steps in detail:

1. **Search**: A domain expert is provided with a search box user interface and is asked to write queries that would find statements of a particular intent from the chat logs. ElasticSearch is used as the search engine. Users in the user study mostly used short phrases and keywords to search. Occasionally, boolean operators and/or quotations were used as well. Search results are returned using exact match. However, multiple search algorithms such as Okapi-BM25, lexical similarity or semantic similarity were proposed to experiment with in future research.

2. **Label**: Top-N ($N = 100$) search results are chosen by the search engine. A candidate subset of size k ($k = 10$) is sampled from the search results by randomly sampling $1/3 * k$ candidates from the bottom, middle and top of the retrieved top-N list. $k = 10$ is a good setting due to user attention span. The candidate labels are given to the user for labeling and are later

used as strong labels. The user doesn't get to see search results outside the candidate subsets.

3. **Propagate**: Elements of the labeled candidate subset are extended to the whole result neighborhood using a thresholding approach. It is a presupposition of the propagate step that highly precise neighborhoods are pulled from the chat logs.

A prototype system was built and a user study was performed on proprietary real-word data. The user study compared the full Search, Label, Propagate pipeline vs Search & Label steps only vs manual labeling assisted with search. Each user in the study labeled 3 intents at 8 minutes per intent. On average, they made 9.09 queries per intent. The study verified that the Search & Label performed significantly better than Label only and the full SLP pipeline performed significantly better than Search & Label only.

The following issues were presented as user feedback to the performed study:

- "Users were confused about handling precision / recall and positive / negative examples."
- "Users had difficulty in coming up with queries due to corpus unfamiliarity."
- "Users are not sure when to stop labeling an intent and move to the next one."
- "Users desire immediate feedback for how each query impacts the results."
- "Some users were unnecessarily rephrasing queries by adding more specific words, e.g. "meeting", "schedule meeting", "schedule meeting time"."

2.3 Label Expansion

In this section, we survey a selection of methods that aim to increase recall of weak models. These are the existing methods we found are most similar to the needs of the Label Expansion step of the framework presented in chapter 3.

2.3.1 Pairwise Feedback

Constrained clustering techniques introduce pairwise feedback to improve clustering performance in a semi-supervised setting. Pairwise feedback, in the form of pairwise linkage constrains, is a set of *must-link* pairs and a set of *cannot-link* pairs that indicate belonging of data point pairs to the same class. Inspired by this concept, Boecking and Dubrawski [2019] incorporate pairwise feedback into the data programming framework to increase labeling accuracy. Pairwise feedback is used to tie labeling functions together across different samples in order to improve latent class variable modeling of the data programming generative model.

Pairwise linkage constrains are encoded as a symmetric matrix $\mathbf{A} \in \{-1, 0, 1\}^n$ where n is the dataset size. Often, only *must-link* pairs are available and pairwise linkage is thus encoded as $\mathbf{A} \in \{0, 1\}^n$. The values of $\mathbf{A}$ are encoded as follows:

- *Must-link* – data points i and j that should be in the same class are encoded as $\mathbf{A}_{i,j} = 1$
- *Cannot-link* – data points i and j that should **not** be in the same class are encoded $\mathbf{A}_{i,j} = -1$
- *Abstain* – no feedback for class membership of data points i and j is encoded as $\mathbf{A}_{i,j} = 0$

Domain-specific heuristics created by a domain expert may be used for constructing pairwise linking constraints for the problem at hand. Furthermore, Mutual k-Nearest Neighbor (MKNN) graphs of a similarity function that holds across the dataset may be used to construct the constraints. The choice of similarity function is left up to the domain expert.

Boecking and Dubrawski [2019] test data programming with pairwise feedback on the 20 Newsgroups text classification dataset.[6] Sixteen keyword-based labeling functions gathered from domain experts gained 46.94% accuracy using a majority vote and 56.45% accuracy using Snorkel. MKNN pairs constructed from TF-IDF similarity (see chapter 1) by computing 10 nearest neighbors increased the accuracy all the way to 75.98%.

2.3.2 Epoxy

Large language models such as BERT can be used as the discriminative model in weak supervision. After fine-tuning for the classification problem on annotated data, the language model is equipped to find synonyms, relax word order, or otherwise provide predictions beyond the scope of heuristics-based labeling functions. Although favorable to training from scratch, fine-tuning still requires non-negligible training time. This hinders the possibility for the data programmer to interactively develop labeling functions with feedback from the discriminative model.

Label propagation [Zhu and Ghahramani, 2002, Iscen et al., 2019] is a technique in semi-supervised learning, where labels from annotated data are propagated through unannotated data using an embedding space. Epoxy[7] [Chen et al., 2020] takes an inspiration from label propagation. As seen in Figure 2.8, labels generated by weak supervision are propagated across unlabeled data using pre-trained language model embeddings to guide the propagation. Embeddings need not be tuned. High-accuracy low-coverage labeling function predictions are extended to nearby unannotated training data points, increasing coverage.

The advantage of this approach is its interactivity. No model fine-tuning is necessary and the whole procedure takes less than half a second. An evaluation was made proving that the accuracy of a generative model enhanced by Epoxy label propagation is comparable to having a fine-tuned BERT as a discriminative model.

[6] https://scikit-learn.org/stable/datasets/index.html#newsgroups-dataset
[7] https://github.com/HazyResearch/epoxy

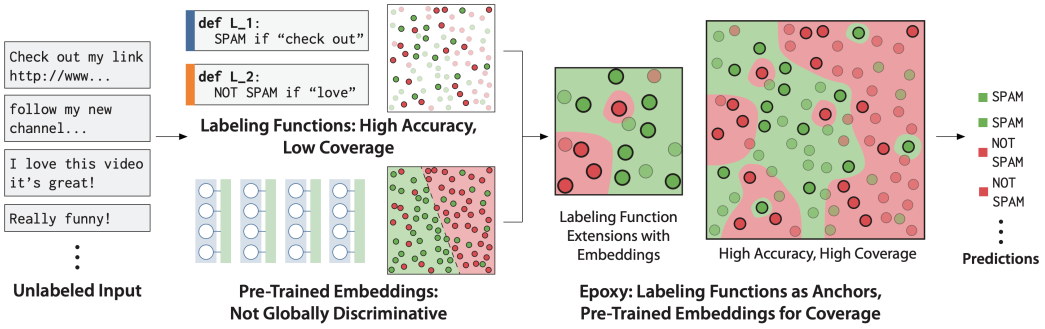


Figure 2.8: Epoxy Overview, from Chen et al. [2020]

2.3.3 Passage Ranking

Passage ranking is a task in information retrieval where a query with associated candidate passages are given and a model must learn to rank the passages by relevance. Most common passage ranking baselines are unsupervised methods TF-IDF and BM-25. State-of-the-art methods make use of supervised learning. However, hand-labeling is difficult. Xu et al. [2019] propose to use weak supervision methods on the passage ranking task and propose successful weak supervision strategies.

To define labeling functions, the ranking problem is transformed into a classification problem that decides if a passage is strongly related to a query. Using a similarity function between the query and candidate passages, a labeling function is automatically created using positive and negative sampling. For the top-1 passage, the labeling functions outputs a positive label. For the bottom half of passage, a negative label is returned. For other passages, the labeling function abstains.

The following similarity functions are used as sources of weak supervision:

1. BM25 score
2. TF-IDF score
3. cosine similarity of Universal Sentence Encoder representations
4. cosine similarity of the last hidden layer of pre-trained BERT

2.3.4 Iterative Expansion for Text

The authors of SLP (subsection 2.2.1) follow with an iterative data programming method [Mallinar et al., 2020] which introduces an iterative batch label-expansion procedure for Snorkel by combining traditional Data Programming combined with Elasticsearch's "More Like This" feature to expand labels in batches and thus augment the created training data sets. It is essentially an iterative procedure that uses search to rank and select unlabeled examples by leveraging labeled examples. The newly selected examples form a weak supervision strategy and thus form a labeling function.

2.4 Label Debugging

In this section, we survey a selection of methods that aim to increase precision of weak models. These are the existing methods we found to be most similar to the needs of the Label Debugging step of the framework presented in chapter 3.

2.4.1 Socratic Learning

Weak supervision uses a generative model to estimate an accuracy parameter for each weak supervision source, even without any ground truth data. The generative model is then able to assign probabilistic training labels to unlabeled data based on the associated weak supervision sources and their estimated accuracy parameters. Socratic learning [Varma et al., 2017a] realizes that the assumption that weak supervision sources have uniform accuracy over the whole dataset rarely holds. In particular, if weak supervision sources are user-developed heuristics, users can only cover subsets of the data for which they have an immediate understanding.

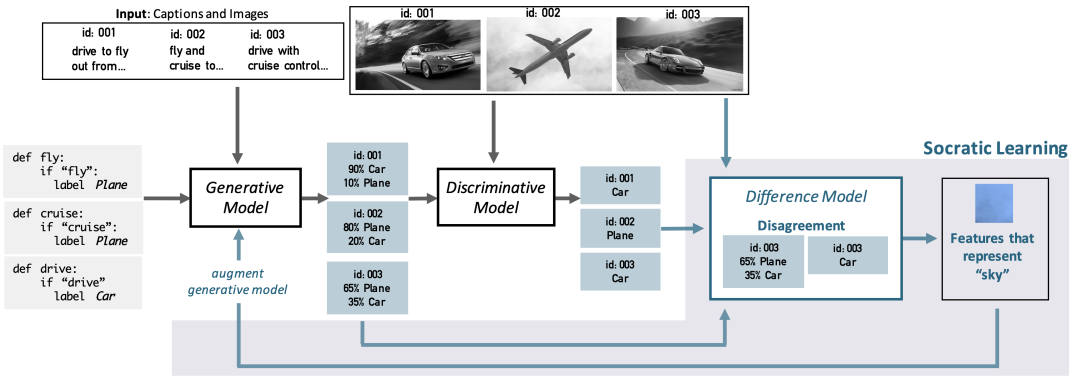


Figure 2.9: Socratic Learning Toy Example, from Varma et al. [2017a]

Latent subsets in the training data, where weak supervision sources perform differently than on average, are identified using a difference model. The difference model computes differences of Snorkel's generative and discriminative models. The latent subsets found in the training data can help improve the generative model. Socratic learning (see Figure 2.9) tries to improve the model automatically. However, by examining features selected by the difference model, experts can improve performance via user-defined heuristics even more than the automatic model.

Flipper

An extension of Socratic learning called Flipper [Varma et al., 2017b] uses latent semantic analysis (LSA) to provide keywords/phrases/topics to help users with improving heuristics based on the features identified by Socratic learning.

2.4.2 Active Learning

Active learning [Settles, 2009, 2012, Olsson, 2009, Escudeiro, 2012] is a machine learning paradigm where the learning algorithm chooses the data from which it learns. A human annotator is used as an oracle. The active learning system queries the user for annotation of data points it chooses.

Interactive Programmatic Labeling for Weak Supervision

Cohen-Wang et al. [2019] combines data programming with active learning by proposing two basic strategies for the human expert: 1) focus on data points where current labeling functions abstain the most; 2) focus on data points where current labeling functions disagree the most.

Hybridization of Active Learning and Data Programming

Nashaat et al. [2018] combine active learning and data programming into a single hybrid framework. First, data programming is performed in a standard way. Then, active learning is used to improve conflicts among labeling functions.

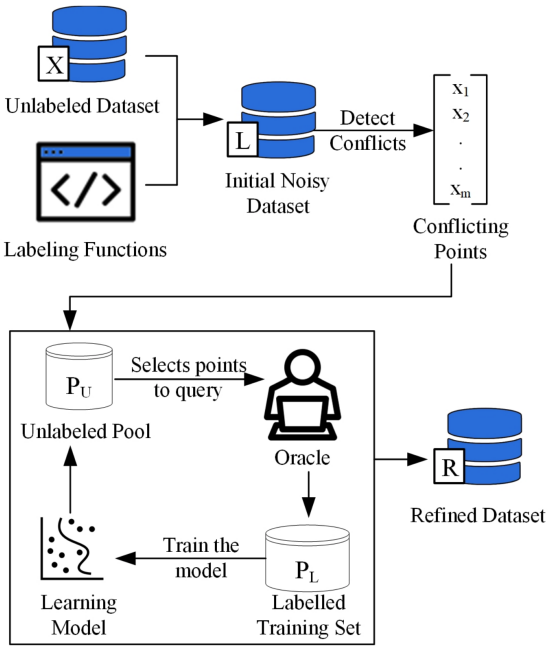


Figure 2.10: Hybridization of Active Learning and Data Programming Overview, from Nashaat et al. [2018]

3. Interactive Keyword-Based Text Classification Framework

In this chapter, we present an interactive framework for text classification via keyword-based labeling. A diagram of the framework can be seen in Figure 3.1.

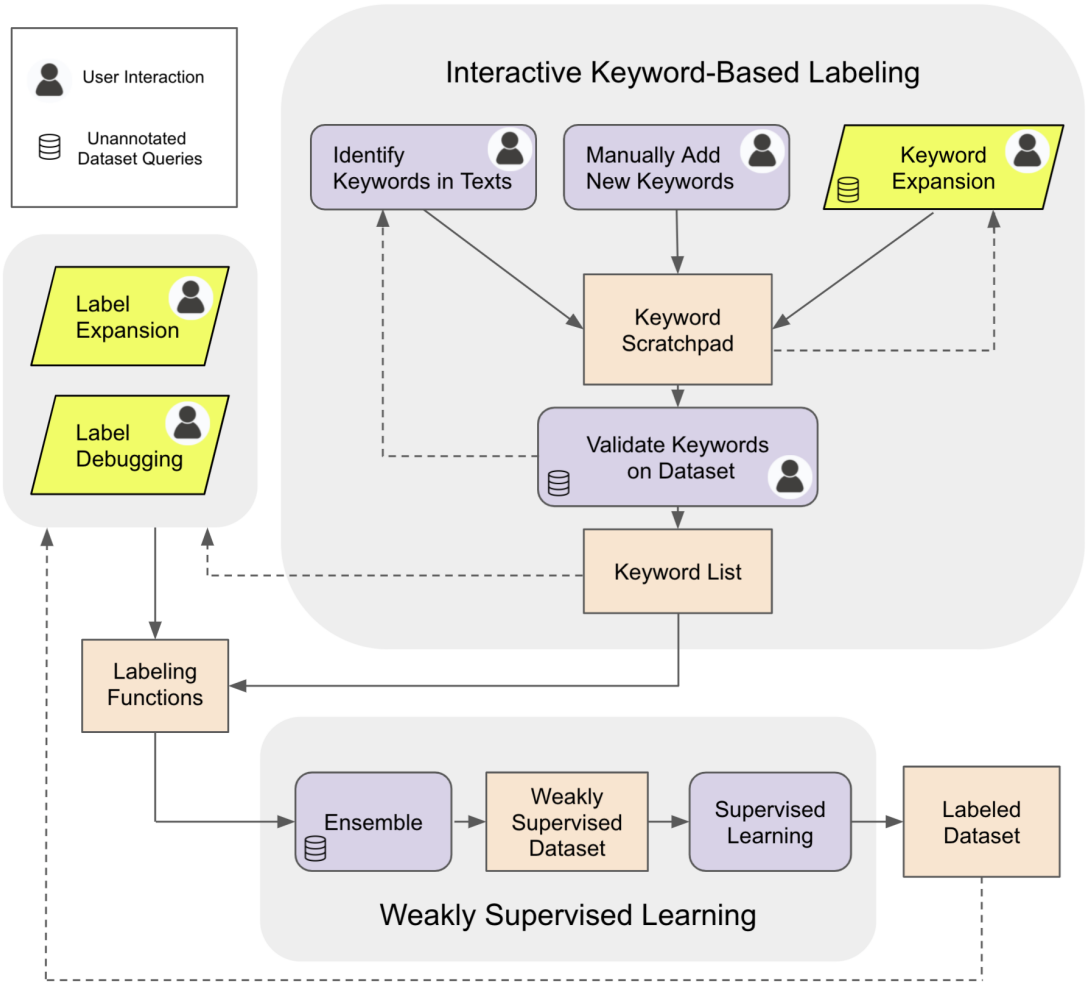


Figure 3.1: Detailed Diagram of the Interactive Keyword-Based Text Classification Framework

The aim of this framework is to remove the need for annotation when training text classifiers. Interactive keyword-based labeling (section 3.1), which takes a database of unannotated texts as input, is used as a centerpiece of the framework to interactively guide a user towards generating keyword lists quickly and easily. The keyword lists are then used to automatically assign weak labels to a dataset of texts via weakly supervised learning (section 3.2). The weakly labeled dataset is used to train a final text classifier (section 3.3). The user can interactively tune the classifier and increase its performance via label expansion (section 3.4) and label debugging (section 3.5).

No programming or machine learning background is needed to use the framework. As opposed to traditional annotation where annotating wrongly requires re-annotation, keyword labeling enables iterative development. By continually adding keywords, expanding on them and debugging the model, results improve iteratively. As an addition, keyword-based labeling results in highly explainable models.

Note: The design of this framework is influenced by experience gained while conducting the user study (chapter 5).

3.1 Interactive Keyword-Based Labeling

Text classification traditionally requires texts to be manually labeled and used as training data for supervised learning algorithms. As illustrated in Figure 3.2, in keyword-based labeling, people find keywords whose presence in the text highly correlates with the assignment of a particular label. Following terminology from weakly supervised learning, each keyword found is used as a heuristic labeling function which noisily labels a subset of the training data. The resulting keyword lists function as simple text classifiers. If a given text contains any keyword from a keyword list for a particular label, then it is assigned that label.

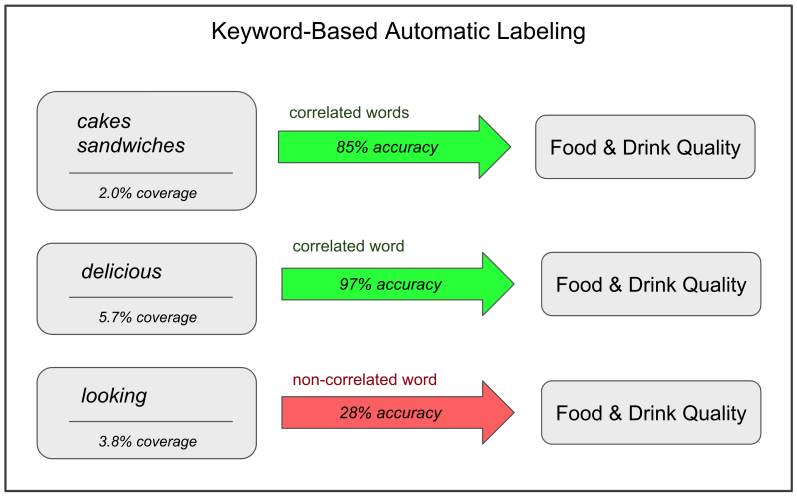


Figure 3.2: Keyword Labeling

As far as we know, attempting to perform keyword labeling without proper tooling is problematic. One must be familiarized with the dataset to start providing good keywords. Also, keyword importance and quality is difficult to assess without a validation procedure. And most importantly, people have a hard time coming up with keywords and very quickly run out of keyword ideas on their own. We offer the following quotes gathered during the literature survey performed in chapter 2 as proof of the described shortcomings of keyword labeling without proper tooling:

- ”Users had difficulty in coming up with queries due to corpus unfamiliarity.” Mallinar et al. [2019]

- "Users desire immediate feedback for how each query impacts the results." Mallinar et al. [2019]
- "There is usually little formal structure or guidance for how these labeling functions are created by users" Cohen-Wang et al. [2019]
- "Little is known about user experience in writing labeling functions and how to improve it" Evensen et al. [2020]
- "Through our engagements with users at large companies, we find that experts spend a significant amount of time designing these weak supervision sources" Varma and Ré [2018]

To overcome the mentioned shortcomings of keyword labeling, we propose the use of an interactive tool – a keyword labeler. The keyword labeler guides the user to interactively create and validate a keyword list for each desired classification label. It enhances the labeling experience by providing a workflow consisting of multiple keyword ideation options, as well as keyword validation. The interactive tool is described in more detailed in the following subsections.

What is a keyword?

There is a certain freedom of choice of what we can consider a keyword. Some possible options follow:

- *Word tokens* – The input text is tokenized into tokens that correspond to individual words. Punctuation is omitted. Keywords can be made case-insensitive. This option was implemented in our keyword labeler prototype.
- *Lemmas* – Usually, one doesn't want to distinguish between singular and plural noun forms when adding keywords. Lemmas might be a better solution than word tokens, at least when it comes to nouns.
- *Multi-word expressions* – Most participants of the user study suggested that they would highly welcome the possibility of using multi-word expressions as keywords, possibly bi-grams being enough.
- *Word Senses* – Homonymy can disqualify potential keywords. Some keywords are only correlated when used in one sense of the word and not the other. It could be a good idea to include word sense disambiguation into label debugging.

Multi-Label Classification

The keyword labeler allows for multi-label classification. One keyword list is generated per label. The user always works on one label at a time. The user has the possibility to switch between labels they are currently working on. A separate keyword scratchpad and keyword list is provided to the user for each label. A concise summary and evaluation of classification performance of all labels is available to the user.

3.1.1 Keyword Scratchpad

When people are asked to come up with keywords for keyword labeling without any tools or instructions available, they quickly run out of ideas. To boost user ideation, we propose that a keyword scratchpad be the dominant component of the keyword labeler. This scratchpad can be simply implemented as a text box. It is a place that encourages users to write down all their keyword ideas, brainstorm, and more. The scratchpad serves as a buffer of keywords that will later be validated in the *Keyword Validation* step. As only a subset of keyword ideas may be validated at once, and more potential keywords are encountered during validation, a keyword scratchpad is needed to keep track of them before they can start being validated.

Adding Keywords to the Scratchpad

Multiple avenues exist for adding keywords to the scratchpad:

1. Manually type in a keyword.
2. Add a keyword identified when reading through sample data points in the *Keyword Validation* step.
3. Add keyword suggestions generated by *Keyword Expansion*.

The user interface should allow seamless adding of keywords in all cases. A single keyword is only allowed to be present in the scratchpad once.

Keyword List

The keyword list is the output format of the keyword labeler. There is one keyword list for each classification label. It contains keyword groups with keywords that correlate with the label. A keyword group is moved from the keyword scratchpad to the keyword list upon successful validation.

The keyword list also supports keyword group enabling and disabling. By disabling keyword groups with low accuracies, the user may increase precision while decreasing recall depending on the task specification.

Keyword Coverage

An important feature of the keyword buffer is to display keyword coverage for all words added to the buffer. By keyword coverage, we mean the percentage of texts from the dataset that the keyword occurs in. Keyword coverage serves as a strong indicator of what keywords to focus on. Keywords with a high coverage deserve to be validated first and on a bigger dataset sample. They are also prime candidates for keyword expansion, label expansion and label debugging. Keyword coverage should be visibly displayed both for individual keywords and for whole keyword groups, both in the scratchpad and in the keyword list.

Keyword Grouping

The keyword scratchpad has the option to group keywords together. Even though keyword grouping does not impact classification done by the set of keywords, it has multiple benefits. The most impactful benefit may be the possibility to validate whole keyword groups at a time. For instance, suppose that we want to identify text pertaining to "cuisine". The user had the idea to add ingredient keywords – think "salt", "oregano", "parsley", etc. Using keyword expansion on the dataset, over 50 similar keywords were found. It is convenient to have all ingredient keywords grouped together, both in the scratchpad and in the resulting keyword list, for multiple reasons:

1. *Reduce cognitive load*: Reading through long lists of keywords is difficult. If keywords are reasonably grouped, one only needs to scan through keyword groups without the need to look at every word.

2. *Keyword validation*: One of the most important reasons to group keywords is to simplify keyword validation of long lists of similar keywords, such as those generated through keyword expansion. If the user uses good judgement, they should be able to identify groups of keywords that act in the same manner in keyword labeling. These groups can be safely validated as a whole. One needn't sample the dataset for multiple texts to validate one word at a time. Instead, random sampling is performed both to choose a keyword from the group and a text containing that keyword from the dataset. Sampling distributions of keywords should correspond to their coverages.

3. *Keyword expansion via similarity search*: By the nature of similarity search, its results likely belong to the same keyword group. Similarity search works best when a small number of similar words are provided together. As a result, keyword expansion and keyword grouping form a symbiotic relationship: The user should manually construct small keyword groups and expand them via similarity search.

4. *Model explainability*: Keyword groups give extra reasoning when looking back at why a text was labeled the way it was.

5. *Change in task specifications*: If a task's specification changes, it is simpler to change a few keyword groups from one label to another than to have to go through all keywords and decide if its label must be changed individually.

6. *Label expansion via sentence similarity*: The idea behind label expansion via sentence similarity is to take a set of seed sentences we (weakly) know the label for and extend the set by N-nearest neighbor similar sentences. The user chooses N. Texts labeled by keywords from a keyword group are a good candidate for seed sentences.

7. *Label debugging*: Automatically identify outliers within keyword groups and remove them.

3.1.2 Keyword Validation

Moving a keyword group from the keyword scratchpad to the keyword list is done via keyword validation. The user employs keyword validation to validate for themselves the approximate accuracy of that keyword group. There are two ways of defining this keyword accuracy:

1. The probability that a text should be classified as the selected label, given that the text is in-domain (i.e. it is present in the unannotated dataset) and given that the text contains at least one keyword from the keyword group.

2. Same as above with the additional constraint that all the keywords from the keyword group that are present in the given text directly influence that text being classified as that label. In other words, we mark keyword instances that are unrelated to the classification label as invalidating the keyword group.

We suggest that the user adopts the second definition of keyword accuracy, even though that definition is very subjective. Determining if a single word influences the classification label of a whole text may be controversial.

We approximate keyword group accuracy by random sampling the unannotated dataset. Keywords groups with higher coverages deserve larger sample sizes. Novice users should be encouraged to validate on slightly larger samples, so that they learn what can go wrong in keyword labeling. Higher sample sizes are also beneficial for keyword identification in text samples. As users become more familiar with keyword labeling, they should be able to adjust sample sizes on the go. Possibly, they should be able to easily sample keywords of their own choosing in the keyword group.

Keyword validation can have three outcomes:

1. The whole keyword group is added to the keyword list.

2. The whole keyword group is deleted and will not be present in the keyword scratchpad, nor the keyword list any longer.

3. The user interactively removes keywords from the keyword group they found uncorrelated with the classification label, or correlated differently compared to the rest of the group. The user should make sure that the chosen subset is properly validated and that the computed approximate accuracy is not skewed by the keyword group modifications made.

Sampled texts marked as validating or contradicting keyword group correctness should be kept for bookkeeping purposes.

Keyword Highlighting

All texts sampled when validating should have the following types of keywords highlighted, in different colors:

- keywords being validated;

- keywords present in the keyword list;
- keywords present in the keyword scratchpad.

The highlights of keywords being validated serve the user to quickly gage how many and which keywords being validated are present in each text. The other highlighted keywords serve as an indicator of how likely it is, that the text belongs to the classification label. Seeing the highlights points the users to start reading only the relevant parts of longer texts and lets them easily skip the rest. Keyword identification in the text samples is eased, since the user only needs to consider non-highlighted words in the proximity of highlighted words.

Keyword Identification in Text Samples

A very important byproduct of the user reading texts sampled from the dataset when doing keyword validation is the possibility of running into words in the text that may be used as keywords. The keyword labeler should have a simple one-click feature that will add a selected word from the text to the keyword scratchpad. Since sampled texts will mostly belong to the classification label, identifying keywords when reading validation text samples should be more beneficial than random sampling the dataset, where only a subset of texts belong to the label. It is advised to identify keyword ideas across all keywords being validated. Texts sampled for a single keyword group may only contain a small part of keywords that correlate with the classification label.

Rare Classes

In traditional annotation where data samples are annotated one at time, rare classes get much less attention than common classes. The keyword labeler does not allow the user to sequentially look through data points. Instead, data points are random sampled by using keywords correlated with the target class. As a result, the user can concentrate on the problem one class at a time and decide how much time to spend on each class regardless of how rare it is.

3.1.3 Keyword Expansion

Keyword expansion automatically suggests new keywords to add to the keyword scratchpad based on keywords present in the scratchpad or in the keyword list.

Keyword Expansion via Word Similarity

We suggest the use of word similarity for keyword expansion. Word similarity scores are used to suggest new keywords. A small amount of seed keywords that are semantically similar, must be provided by the user. Alternatively, every keyword group can serve as the set of seed keywords. Nearest neighbors by a similarity metric are used as keyword suggestions and the user can simply select the keywords they find relevant. Each choice made by the user further enhances the keyword suggestion list.

Many choices exist for the word similarity model. It is for example possible to use a pre-computed word vector, such as *word2vec*. The word vectors can also

be fine-tuned on the unannotated dataset. In our keyword labeler prototype, we decided to use word vectors computed using solely the unannotated dataset. We believe this is a good choice, because the keyword suggestions are very specific to the dataset. In addition, words that are not semantically similar in general, but have a correlation in the target domain are also suggested by the word similarity model. This yields very good keyword suggestions – even ones it would be close to impossible to think of for the user alone.

Word similarity suggestions based on a single word (or a very small group of words) often yields bad suggestions. However, we found that the list of many bad suggestions can often provide one or two good ones. When these are added to the word similarity search, results improve. It is thus possible to iteratively select keywords from the keywords suggestion list and end up with a good large keyword group at the end.

3.2 Labeling Function Ensembling

Keyword-based labeling functions are defined by a keyword and a label. If the keyword is present in a text-to-be-classified, the keyword-based labeling function returns the appropriate label. Otherwise, the function abstains. The labeling functions are ensembled using methods discussed in subsection 2.1.1

3.3 Supervised Classification

In weakly supervised learning, labeling function ensembling is used to weakly annotate the dataset. This weakly supervised dataset is then used to train an end supervised model, which is the main output of our framework.

We encourage the use of multiple supervised classifiers. There should be at least one simple classifier that would allow for near immediate training and inference. This fast classifier should be run in parallel to interactive labeling after each keyword list change. The framework enables the user to continuously monitor classifier performance. We suggest that when sampling texts during keyword validation, the model predictions should be shown to the user for each sample. If a dev set is provided, evaluation metrics should be available to the framework user on-the-fly.

State-of-the art models that employ transfer learning should also be run in parallel as often as possible, with their results visible. Such models should be able to generalize and provide much better results than simple keyword labeling. With many sophisticated text classification algorithms to choose from, we should explore if these more complicated algorithms, demanding more computational resources, achieve better results than the simpler model. If they do, they provide a higher realistic overall performance estimate. The algorithm that will be used for inference should be chosen based on evaluation results, model simplicity, ease of deployment, inference speed and extent of training resources needed for retraining.

Complicated models providing good results are great candidates for model generalization exploitation. We use the model to predict data points from the

training set. The difference between the predicted labels and labels generated by keyword labeling may be used for label expansion and label debugging.

3.4 Label Expansion

In the literature survey in section 2.3, we identified a selection of methods that may be used to add labels to unannotated data points on the basis of existing annotation. A confidence threshold is selected by the user to enrich the annotated training data. Here, we propose two methods for label expansion inspired by those in the survey:

- *Sentence Similarity*: Use a sentence similarity model to search for sentences on which labeling functions abstain that are similar to those already classified. The hope is that these similar sentences should belong to that same class.
- *Model Generalization*: A supervised classifier is trained on all labeled data. Then, we let the classifier label data on which the labeling functions are abstaining so far. The training dataset can then be expanded by predictions the classifier is confident in.

Two large benefits of creating text classification models via keyword labeling are explainability and ease of modification. We suggest two strategies not to lose these benefits:

1. Use texts that newly received a label by label expansion to auto-suggest keywords for keyword expansion.
2. Link label expansion to keyword groups. Label expansion generates labeling functions whose parameters are a set of keywords in a keyword group and a threshold.

3.5 Label Debugging

In the literature survey in section 2.4, we identified a selection of methods that may be used to correct misclassifications. We propose to use such methods in our framework for label debugging. Label debugging improves model precision and bug-fixes individual misclassifications. When the recall for a particular class gets high enough, the pipeline user is encouraged to debug the annotated training set they created. Debugging increases precision while maintaining recall.

Besides methods covered in the survey, the following methods come to mind for label debugging:

1. *Word sense disambiguation*: Otherwise good keywords missclassify in certain cases due to homonymy. Word sense disambiguation can be used to classify only via certain word senses of keywords.

2. *Linguistic logical operator*: Allow the use of logical operators, such as AND, for combining keyword groups to model more complicated relationships in the text. Other operators could include using wild-cards and relations for various linguistic phenomena, such as part-of-speech tags, dependency tags, dependency relations, lemmas, etc.

3. *Keyword group outliers*: Find outlier keywords within keyword groups.

4. *Hard annotation*: Annotate a subset of texts labeled by a given labeling function and use it as hard labels for the classifier.

4. Keyword Labeler Prototype

Previously, in section 3.1, we presented a design for a keyword labeler. In this chapter, we present our prototype keyword labeler implementation. This prototype is implemented for the purpose of conducting the user study described in chapter 5.

4.1 Interactive Keyword Labeler

People find keyword labeling from the top of their head difficult and very quickly run out of keyword ideas, miss important keywords, or are generally slow at it. The purpose of this keyword labeler prototype is to provide users with an interactive tool and a set of strategies to allow for creating large and accurate keyword lists quickly and easily.

Implemented keyword labeler features include:

1. A single interactive tool that keeps track of keyword lists, accuracies, and more.
2. Manage keyword groups.
3. Provide coverage information.
4. Validate ideas by adding one keyword group at a time, annotating it and adjusting it in the meantime.
5. Auto-suggest keywords based on a few seed keywords using keyword-based similarity search.
6. Discourage browsing the dataset. Focus on showing dataset samples when validating adding keyword groups instead.
7. Encourage the user to focus on one label at a time.

4.2 Keyword Expansion via Similarity Search

Similar interactive tools were created in the past, see more information in section 2.2. One of the novel additions in our framework is keyword expansion via similarity search, which we hope will further ease keyword labeling and prolong generated keyword lists.

Word Similarity: *svd2vec*

We utilize the word similarity algorithm *svd2vec*,[1] available as a Python library that computes word vectors solely by training on unannotated data from a target domain. No pre-training or transfer learning is involved. The choice of this

[1] https://github.com/valentinp72/svd2vec

specific library is not of paramount importance to the prototype. The library was chosen due to availability and ease of use. At first glance, it provides sufficient results.

The computed vectors are used to compute word similarity via cosine distance. The svd2vec algorithm internally computes pointwise mutual information (PMI) for all words in the input dataset and then factorizes them by using Singular Value Decomposition (SVD) to get word vectors. The library is based on findings from Levy et al. [2015] who suggest that traditional methods like PMI and SVD can be as good as word embedding methods such as word2vec, when hyperparameters are chosen correctly.

Details on training *svd2vec* along with similarity search examples are available in the appendix section A.3.

4.3 User Instructions

An instruction manual for the keyword labeler prototype is included in the User Study Instructions in the appendix section A.2. We must note that the text itself was too dense to be easily readable by the user study participants. Instead, we instructed them by verbally reexplaining the instructions.

Key concepts the keyword labeler user should understand after reading the instructions are:

- correlation,
- accuracy and coverage,
- the need to validate keyword ideas; and
- keyword grouping.

Keyword labeling strategies discussed in the instructions include:

- Brainstorm potential keywords, look for keywords from "various points of view".
- Think of synonyms.
- Take note of new potential keywords while going through data samples.

4.4 Prototype Implementation

The prototype implementation can be found on GitHub.[2]

The prototype is written in Python. It has a sole dependency: the *svd2vec* library. It features a command line interface (CLI). Installation instructions are included in the source code README file.

[2]https://github.com/laitoch/master-thesis-prototype

User documentation including example commands are to be found in the user instructions in the appendix of this thesis (section A.2).

The prototype implementation was created with the sole purpose of conducting the user study. It is not meant as a full-featured keyword labeler implementation.

A dataset sample is available in the data folder:

- Unannotated dataset of restaurant reviews from Yelp.
- A small (100 data point) annotated test set including Environment and Hospitality labels.

5. User Study: Keyword Labeling of Restaurant Reviews

This chapter showcases a user study designed to evaluate the effectiveness of the keyword labeler prototype presented in chapter 4.

The keyword labeler prototype represents the cornerstone of the interactive keyword-based text classification framework proposed in chapter 3. By performing the user study, we wish to illustrate the potential benefits offered by the proposed framework and to validate the usefulness of its full implementation as future work.

User study findings are located as follows:

1. Evaluation of keyword-based labeling results in comparison with a supervised classification baseline (section 5.2).
2. Analysis of keyword lists created in the user study (section 5.4).
3. User feedback gathered through post-task interviews (section 5.5).
4. Specifications for an improved keyboard labeler (section 5.6).
5. Table of all keyword lists created in the user study (section A.4).

5.1 Performing the User Study

Each of the four participants sat down individually in a one-on-one session with the author of this thesis for a net duration of 3 hours. Short breaks were taken in between scheduled time segments upon request, which made the actual duration of the study surpass 3 hours. The time-frame was set up on-the-fly with the first participant. Time was spent as follows:

- *On-boarding Instructions* – 35 minutes – We explained keyword-based labeling principles along with technical details on using the keyword labeler prototype in the form of a one-on-one lesson. Participants were encouraged to ask questions along the way. When there were no questions, instructions were presented as a monologue.
- *On-boarding Tutorial* – 50 minutes – The participants experimented with the prototype, working on a tutorial task (Food&Drink Quality label). We made sure they understood all underlying principles, and became familiar with using the prototype. We explained additional tips and tricks and had them try them out. We guided participants to try out different workflow aspects to equitably fill the allocated 50 minutes.
- *Task 1 (Environment label)* – 50 minutes – Participants were asked to perform the first task (Environment label). Participants were notified when their progress slowed, for instance when they were doing something wrong,

or when adopting a sub-par workflow, or when incorrectly interpreting annotation guidelines. We did not otherwise influence their work. Keyword-based similarity search was banned from use for the first 30 minutes. In the last 20 minutes, they tried keyword-based similarity search.

- *Task 2 (Hospitality label)* – 30 minutes – Participants were asked to perform the second task (Hospitality label). We only communicated with them to clarify annotation guidelines.
- *Post-task interview* – 15 minutes – We interviewed the participants by asking open-ended questions about the keyword labeler, the workflows and strategies adopted, the tasks performed and their achieved results.

5.1.1 On-boarding

A non-trivial user on-boarding procedure is required to teach users how to properly perform keyword-based labeling. The on-boarding was performed verbally. A write-up of user study instructions that were re-formulated in the on-boarding is available in section A.2.

The following concepts must be explained to the users to allow for successful keyword labeling:

- *Keyword-Based Labeling* – classification, keywords, correlation.
- *Coverage, Accuracy, Keyword Validation* – find frequently occurring keywords that predict the desired label with a high accuracy. Validate that accuracy by random sampling the unannotated dataset.
- *Annotation Guidelines* – classification labels are explained intuitively by keywords and example sentences in the annotation guidelines. Annotators should consult with somebody if not sure what should be in the label and what shouldn't.
- *UI usage* – how to use the keyword labeler user interface.

Strategies that users were instructed to follow:

- *Ideation:*
  - Think of synonyms.
  - Browse text for ideas.
  - Group similar keywords.
  - Use keyword similarity search.
- *Validation:*
  - Manually annotate data samples containing the keyword being added.
  - Have a good mental model of what a keyword is.

- Decide if the keyword indicates the label, or if the label is just a coincidence.
- Validate groups of similar keywords together – if you are sure they really do behave in the same way.

To simplify prototype implementation, we decided to implement it as a command-line application. We believe that an intuitive GUI with a short well-written interactive tutorial would cut down on-boarding time significantly.

5.1.2 Tasks

For tasks in the user study, we decided to use a proprietary restaurant review dataset provided by the IT company *Geneea Analytics*.[1] The dataset contains a collection of restaurant reviews, some of them annotated. The annotation is done into 21 different classes and is a multi-class classification problem. The classes represent topics the text is talking about. The text can talk about multiple topics at the same time.

A subset of the unannotated dataset containing Yelp[2] restaurant reviews is released as sample data along with the implemented prototype. A dataset description is available in subsection A.1.1.

The tasks performed in the user study were presented through the following annotator guidelines:

Tutorial Task: Food & drink – quality

Keywords:

- taste, freshness, healthfulness, cold food, under/overcooked pasta/chicken

Examples:

- "Food was excellent."
- "All dishes were delicious"
- "To put it bluntly, it is hard to fail scrambled eggs, and for that price you would hope that at least they can make decent ones. But the eggs were a thick spongy white block, in fact, I don't even know how to do that myself, and the bacon was oily and overcooked..."
- "I'm not really a fan of cheese but this was really good!!"
- "You can tell the ingredients are fresh and the pizzas are all made with lots of love"

Task 1: Environment

Keywords:

- cleanliness, atmosphere, decor, noise, seating/parking availability, ...

Examples:

- "the environment is luxurious yet relaxed"
- "the music not that cheerful"
- "Too loud to enjoy"
- "prob our favourite london tea venue in terms of decor/ambience"

[1] https://geneea.com/
[2] https://www.yelp.com/dataset/

Task 2: Hospitality

Keywords:

- nice/rude staff; great staff; staff attitude

Examples:

- "The hospitality is simply superb"
- "really friendly and efficient staff who have to manage working (with a smile!) in very tight space"
- "The welcome was friendly but a little cold initially from our waitress but as the evening went on she was v friendly and very informed regarding the menu."

5.2 User Study Results

In this section, we evaluate results of the two restaurant review classification tasks performed in the user study via keyword labeling. The participants of the user study and their associated keyword-based classifiers are denoted A, B, C and D.

We start with an evaluation on the original test set (subsection 5.2.1) provided with the restaurant review dataset. Due to some missing labels in the annotated data (subsection 5.2.2), we created a new test set and used it for a second evaluation (subsection 5.2.3). We finish by a conclusion (subsection 5.2.4).

5.2.1 Original Test Set

We first evaluate classification via keyword labeling on the test set provided along with the restaurant review dataset (see subsection A.1.2).

Binary classification via the popular fasttext classifier is performed as a supervised learning baseline. We split data from Annotator 1 into a training set of size 2540 and validation set of size 501. The data has 1003 texts labeled Hospitality and 866 texts labeled Environment (out of 3041 texts). We use the validation set to automatically tune the classifier using fasttext's built-in autotune feature. Autotune is arbitrarily set to a duration of 600 seconds.

The test set consists of 607 texts that are separate from the training and validation sets.

The classifier with highest F-1 score is consider the best. We mark highest F-1 score using **bold-face**.

	Original Test Set Environment Label				
Participant	fasttext	A	B	C	D
Precision	81.7%	66.0%	79.1%	64.9%	72.7%
Recall	81.7%	58.4%	50.3%	72.8%	69.4%
F-1	**81.7%**	62.0%	61.5%	68.6%	71.0%

Fasttext provides best F-1 scores for the Environment label. However, an examination of conflicting data samples between keyword-based classifiers and the

original test set uncovered a problem with the annotation. A more fitting test set is needed to evaluate keyword-based labeling of the Environment label.

	Original Test Set Hospitality Label				
Participant	fasttext	A	B	C	D
Precision	77.4%	76.6%	77.1%	74.2%	69.7%
Recall	77.3%	92.1%	89.7%	94.5%	92.1%
F-1	77.3%	**83.6%**	82.9%	83.1%	79.3%

Keyword labeling provides best F-1 scores for the Hospitality label. All participants of the study were able to surpass the supervised fattext classifier.

5.2.2 Original Test Set Issues

The original data is annotated in a multi-label setting, with 21 possible labels. Many texts in the dataset mention multiple topics and should thus receive multiple labels. When annotating the original dataset, the annotators were instructed that it is sufficient to label three most prominent topics. Annotators sometimes selected more than three labels. However, if a text covered many topics, annotators would sometimes not label brief non-prevalent topics. As is seen in the inter-annotator agreement, choosing which label to skip was often made as a judgement call. The following text illustrates the problem:

> "This place is always great! A lovely atmosphere where you can relax for dinner or do homework – just a block from ASU's Downtown campus. The music is great too and they have great happy hour specials all for $5 too! Brian was a great server and took care of us. The rotisserie chicken, the downtown devil fries, and the corn bisque soup were are all incredible! "

Most participants of the user study identified *"music"* as an accurate keyword for the Environment label. Keyword labeling thus assigns the Environment label to the above text. This is correct, since the text informs us that "the place has great music". However, the annotated dataset only lists the labels Food Drink Quality, Value Discounts Promotions and Hospitality. The Environment label is missing in the annotation.

5.2.3 New Test Set

In order to fairly evaluate keyword-based labeling, we decide to manually annotate a new test set containing only the Environment and Hospitality labels. The new test set contains 100 texts.

We cannot retrain the fasttext classification model to match the new test set since that would require us to annotate 3000 more data points. Instead, we use the fasttext models trained on the original annotation. This gives the fasttext baseline a disadvantage on the new test set.

	New Test Set Environment Label				
Participant	fasttext	A	B	C	D
Precision	76.0%	72.7%	90.9%	80.6%	83.3%
Recall	76.0%	70.6%	58.8%	85.3%	73.5%
F-1	76.0%	71.6%	71.4%	**82.9%**	78.1%

The F-1 scores of user study participants are significantly higher on the new test set. The fasttext classifier's F-1 score drops in comparison with the original test set. Two participants out of four surpass the F-1 score of the fasttext baseline. The other two participants perform worse. To be fair, this is the first keyword labeling task the participants performed and they still weren't completely clear on all aspects of keyword labeling at the time.

Participant C even surpasses the F-1 score the baseline classifier received on the original dataset. Overall, supervised classification and classification via keyword-based labeling perform similarly on the Environment label.

	New Test Set Hospitality Label				
Participant	fasttext	A	B	C	D
Precision	78.0%	84.9%	84.4%	83.3%	73.0%
Recall	78.0%	91.8%	77.6%	91.8%	93.9%
F-1	78.0%	**88.2%**	80.9%	87.3%	82.1%

Participant F-1 scores are higher on the new test set. Fasttext scores similarly to the original test set. Just as on the original test set, all participants achieve better F-1 scores than the supervised fasttext classifier on the Hospitality label.

5.2.4 Conclusion

At first glance, keyword-based classification performs similarly to fasttext on the Environment label. However, multiple issues remain to make it a safe comparison:

1. missing labels in the training set;
2. low keyword-labeling skill of some participants (this was their first task);
3. ban of keyword expansion during the first 30 minutes of keyword labeling and lack of keyword expansion experience in the other 20 minutes.

Text classification via keyword-based labeling performs significantly better than supervised classification when assigning the Hospitality label to restaurant reviews. All participants of the user study were able to create keyword-based text classifiers in 30 minutes that surpassed the F-1 score of the popular fasttext text classifier trained on over 3000 texts. Participant C even achieved a 10 point F-1 score increase over the supervised approach.

5.3 Disabling Bad Keyword Groups

To simplify the user study, participants were instructed to add the keyword groups one at a time. They were to validate the group's accuracy before adding the group. However, they were not required to analyze the results after the fact. We promised to do this subtask for them. To fulfil this promise, we disabled such groups before studying the results in section 5.2.

First, we describe example bad keywords that participants erroneously added to their keyword lists (subsection 5.3.1). Then, we show labeling results prior to keyword disabling to illustrate how much a bad keyword can spoil the results (subsection 5.3.2).

5.3.1 Bad Keywords Found

Examples of bad keywords:

- *table* – A table is part of the environment. However, people often use it in the context: "They brought us food to the table". Tables are present in almost all restaurants. People mention them without describing the tables themselves, or any other part of the interior.
- *smell* – general purpose word – it doesn't have to be a smell of the restaurant, but can be smell of the food, of the street, of the staff, etc. However, the smell of the restaurant would qualify for environment. The participant just didn't generalize when they thought of the word / saw it in a sentence.
- *restaurant, place, great* – all have something to do with all texts, not just with the Environment label. The participant didn't fully understand what to do when adding these keywords, it was the beginning of the first task. They were instructed on the error during the task, but after the keyword was added.
- *located* – down to interpretation – should location description of a restaurant be part of the Environment label? This is a good question, the answer should be added to the annotator guidelines. We've ruled that it shouldn't, but we believe it to be an arbitrary decision.

An error of Participant D on the Environment label shows how important keyword validation and keyword disabling can be. While having great results otherwise, the participant made a single error that significantly reduced the end classifier performance. More than halfway through keyword-labeling, the participant added one keyword group of two keywords: *great (cov. 19.1%), place (cov. 20.2%).* This instantly gave him the worst performance. After disabling the group, D was second best.

5.3.2 Results Prior to Disabling

We compare classification results prior to keyword disabling vs after keyword disabling:

	Original Test Set Environment Label							
	Prior to Keyword Disabling				After Keyword Disabling			
Participant	A	B	C	D	A	B	C	D
Precision	58.3%	**43.4%**	62.6%	**42.2%**	66.0%	**79.1%**	64.9%	**72.7%**
Recall	68.8%	87.9%	74.6%	83.8%	58.4%	50.3%	72.8%	69.4%
F-1	63.1%	58.1%	68.1%	56.1%	62.0%	61.5%	68.6%	71.0%

	New Test Set Environment Label							
	Prior to Keyword Disabling				After Keyword Disabling			
Participant	A	B	C	D	A	B	C	D
Precision	58.3%	**49.3%**	77.5%	**46.5%**	72.7%	**90.9%**	80.6%	**83.3%**
Recall	82.4%	100.0%	91.2%	97.1%	70.6%	58.8%	85.3%	73.5%
F-1	68.3%	66.0%	83.8%	62.9%	71.6%	71.4%	82.9%	78.1%

All participants added some bad keywords when labeling the Environment label. This is understandable, as the participants were still learning to do keyword-based labeling properly. Note the extremely high precision differences in the Environment label results with and without keyword group disabling for Participants B and D (marked in **bold-face**).

	Original Test Set Hospitality Label				
Participant	A	B	C	D	D after Keyword Disabling
Precision	76.6%	77.1%	69.4%	65.8%	69.7%
Recall	92.1%	89.7%	94.9%	93.5%	92.1%
F-1	83.6%	82.9%	80.2%	77.2%	79.3%

	New Test Set Hospitality Label				
Participant	A	B	C	D	D after Keyword Disabling
Precision	84.9%	84.4%	83.3%	70.8%	73.0%
Recall	91.8%	77.6%	91.8%	93.9%	93.9%
F-1	88.2%	80.9%	87.3%	80.7%	82.1%

Almost no keyword group disabling was needed for the Hospitality label. We only needed to do a small keyword disabling fix for Participant D.

5.4 Keyword List Analysis

In this section, we discuss various aspects of the keyword lists created in the user study.

5.4.1 Important Keywords

Both tasks contained keywords that could alone correctly label a large portion of the data. We collected most occurring keywords from the keyword lists, not taking into account keywords from disabled keyword groups. We list top *important keywords* for both tasks:

Important Keywords					
Environment Label			***Hospitality Label***		
keyword	***coverage***	***participants***	***keyword***	***coverage***	***participants***
atmosphere	5.3%	A,B,C,D	*service*	20.1%	A,B,C,D
outside	2.5%	D	*staff*	13.3%	A,B,C,D
clean	2.2%	A,D	*friedly*	11.4%	A,B,C,D
decor	1.9%	B,C,D	*waiter*	3.1%	A,B,C,D
music	1.6%	A,C,D	*waitress*	2.7%	A,B,C,D
seating	1.3%	B,C,D	*attentive*	2.7%	A,C,D
view	1.1%	C	*customer*	2.6%	B,D
space	1.1%	B,C	*manager*	2.1%	C,D
patio	1.0%	C	*server*	2.1%	A,C
ambiance	0.9%	C	*helpful*	2.0%	A,C,D
quiet	0.9%	A,B,C,D	*pleasant*	1.8%	D
loud	0.8%	C,D	*employees*	1.4%	B
interior	0.8%	C	*greeted*	1.4%	C
comfortable	0.8%	A,B,C	*owner*	1.2%	D
parking	0.7%	A,B,C,D	*team*	1.3%	C,D
outdoor	0.5%	A,B,C,D	*polite*	0.9%	C,D
environment	0.5%	B,C	*professional*	0.7%	C,D

Counts of important keywords found per participant:

	Important Keywords Found	
Participant	Environment	Hospitality
A	7	8
B	9	7
C	15	13
D	10	14
Total	17	17

We made the following observations based on important keyword analysis:

- The Hospitality task contains much more frequent important keywords compared to the Environment task. It is thus easier to get good results when doing keyword-based labeling.
- Keyword coverages for a given label seem to follow Zipf's law.
- Multiple participants missed keywords that could've been easy to add:

- *environment* – this was the name of the label
- *ambiance* – this keyword was present in the annotator guidelines

5.4.2 Keyword Expansion

Keyword expansion enabled the participants of the study to assemble large keyword lists of similar words. The keywords in such lists were accurate at keyword-based labeling and while their individual coverages were low, total coverages of the groups were higher than most individual important keywords found.

Examples are provided below. Please refer to the appendix section A.4 for the full keyword lists:

- Environment, Participant B: *cleanliness (0.2%), decor (1.9%), noise (0.3%)* yielded keywords including *quiet (0.9%), noisy (0.4%), lively (0.4%), intimate (0.3%), quieter (0.1%), ...*
- Environment, Participant D: *dancing (0.1%)* yielded a group of 29 keywords totaling *2.2% cov.* related to music and dancing *trendy, tunes, DJ, song, band, audience, stage, karaoke, oldies, ...*
- Environment, Participant D: *outdoor, chairs* yielded *stools, couch, decorated, outside, couches, comfy, benches, heaters, cushions, indoor, sofa, lamps, ...*
- Hospitality, Participant A: *7.8% cov.*, almost 40 keywords: *helpful, rude, attentive, attitude, ...*
- Hospitality:
  - List of restaurant professions: *chefs, bartender, hostess, ...*
  - Lists of adjectives characterising human attitudes and interactions: *helpful, professional, knowledgable, arrogant, insulting, defensive, ignored, responded, irritating, cheerful, attentive, helpful, ...*

5.5 Participant Feedback

In this section, we summarize what we learned while observing participants during the user study, communicating with them during it and by conducting post-study interviews.

Pros

The participants liked the following aspects of interactive keyword-based labeling:

- *Coverage* is important.
- *Reading texts* in keyword validation is important – it is a way for the user to both browse input data and validate keyword ideas.

- *Similarity search* is not essential but it is extremely helpful. Participants stated, that it sometimes guided them in a new direction of thought when similarity search failed to present a nice list of synonyms and instead presented mostly random and distantly related word – i.e. when we would've deemed similarity search unsuccessful.
- *Highlights in texts* – users use them as an indicator (feedback) when annotating:
  - *Keywords being validated highlights*: Show what part of the texts to focus on.
  - *Keywords in the keyword list highlights*: Show how likely the text really belongs to the label. Also, they show which words from the text were and weren't added to the scratchpad yet.
- *Keyword-based labeling is fun* – After grasping all the concepts, the users found finding keywords more amusing, than laborious. More a crossword-puzzle than boring labor or frustrating form-filling.

Cons

The participants disliked the following:

- *On-boarding* – was a difficult process, there were many things to wrap their heads around.
- *A nice GUI* – would be a big improvement over the prototype CLI.
- *Quote search missing* – having even just bi-gram search would be beneficial. Possibly AND and OR search would help as well.
- *Keyword Groups* are confusing – especially at first, the participants were confused about why they should group keywords.
- *Low control of keyword validation* – after participants really started to understand what keyword validation is good for, they would've liked to have more control over what keywords to sample in the keyword group. In the prototype, only random sampling with keyword coverages as weights was available. This issue was connected to confusion about what keywords to group.
- *Keyword suggestions* only give good results in some cases. It takes several different searches to get used to the algorithm.
- *Keyword suggestions UI* – checkboxes / clickable words to add remove keyword suggestions needed.
- *Understanding annotation guidelines* is non-trivial. Participants were confused about what should and shouldn't be in a given label. They had to ask for clarifications.

- *Metrics* besides keyword coverage and keyword group coverage were deemed unimportant and were ignored. This includes: Estimated accuracy, total coverage, total accuracy, overlaps, etc. This could be due to the fact that many metrics were presented at once at the beginning and the participants weren't ask to disable keywords to improve precision at the end.

Workflows Adopted by Users

A checklist of most noticeable participant behaviors and strategies:

- View Annotation Guidelines and write down all potential keywords.
- View coverages of potential keywords.
- Think and write synonyms from the back of the head.
- Think about various synonym sets / views of the subject.
- *add_keywords* on a single keyword at a time to validate/annotate one keyword at a time.
- Take note of potential keywords while reading.
- Being aware that it's a competition, that it's on time.
- Stopping annotation/validation immediately when realizing that the idea does not work.
- Some users trusted *bulk_add_keywords* blindly, others tried hard to validate all suggested keywords.

5.6 Specifications for an Improved Keyword Labeler

A list of specifications a keyword labeler should have in addition to those present in the keyword labeler prototype:

- *Graphical User Interface* – Have a powerful, easy to use Graphical User Interface (GUI).
- *Integrated scratchpad* – A major component of the GUI should be a keyword scratchpad. The scratchpad automatically computes and shows coverages for every keyword and keyword group.
- *Keyword Expansion* – On-the-fly adding of multiple suggestions to keyword groups in one click. Similarity search input is easily controlled by selecting individual keywords from the scratchpad group. Suggestions interactively change without a lag. Automatic ordering of keywords by occurrences. Allows to increase keyword suggestion size. Keyword expansion is an essential step happening before and/or after keyword group validation.

- *Keyword Validation* – One-click add word from text to scratchpad. More control over with which keywords to sample from when adding a keyword group.

Conclusion

In this thesis, we designed a framework aiming to minimize manual work needed to annotate training data in text classification. The framework requires no annotated data. Instead, the user of the framework uses the central component of the framework called a keyword labeler to create keyword lists that are then used to classify texts. Other components of the framework improve performance of text classifiers built using the keyword lists. Ideas for methods to be used in these other compoments come from a literature survey pertaining to weakly supervised learning, interactive learning, and more.

If a keyword from a keyword list is present in a text, that text is classified using that keyword list's label. This is called keyword-based labeling. The keyword labeler is designed as an interactive tool featuring methods for building good keyword lists quickly and easily for an in-domain unannotated dataset. Notably, we propose the use of word similarity to automatically suggest new keywords to the user based on existing keywords.

To verify that people can easily create large keyword lists in a short amount of time using our designed keyword labeler, we implemented a keyword labeler prototype and used it to conduct a user study. Participants of the user study first gained experience using the keyword labeler prototype. Then, they created keyword lists for classifying texts from a restaurant review dataset, where the classification labels indicate if certain topics were discussed in the review.

In 30 minutes of interactive keyword-based labeling, participants of the user study created keyword-based text classifiers surpassing the F-1 score of a standard supervised classifier trained on over 3000 data points. Keyword suggestions considerably aided users when creating these keyword lists. As a result, the user study validated that:

1. Interactive keyword-based labeling tools allow users to quickly and easily create large and accurate keyword lists, which provide very competitive baselines in text classification tasks.

2. Word similarity-based keyword suggestions significantly improve the keyword labeler.

Based on experience gained from conducting the user study, we assembled a list of specifications for a better, more user-friendly keyword labeler.

As future work, we suggest to:

1. Implement the suggested improved keyword labeler.

2. Implement the complete keyword-based labeling framework.

Keyword-based labeling is a competitive alternative to text classification via annotating texts and training supervised models. It is more explainable and allows for much simpler retraining when task specifications change. It can be operated by anybody, no programming nor machine learning background is needed. We believe it has potential to become a full-featured do-it-yourself text classification solution for anybody with an available text dataset.

Bibliography

Josh Attenberg and Foster Provost. Why label when you can search? alternatives to active learning for applying human resources to build classification models under extreme class imbalance. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, page 423–432, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450300551. doi: 10.1145/1835804.1835859. URL https://doi.org/10.1145/1835804.1835859.

Stephen H. Bach, Daniel Rodriguez, Yintao Liu, Chong Luo, Haidong Shao, Cassandra Xia, Souvik Sen, Alex Ratner, Braden Hancock, Houman Alborzi, Rahul Kuchhal, Chris Ré, and Rob Malkin. Snorkel drybell: A case study in deploying weak supervision at industrial scale. In *Proceedings of the 2019 International Conference on Management of Data*, SIGMOD '19, page 362–375, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450356435. doi: 10.1145/3299869.3314036. URL https://doi.org/10.1145/3299869.3314036.

Jessa Bekker and Jesse Davis. Learning from positive and unlabeled data: a survey. *Machine Learning*, 109(4):719–760, Apr 2020. ISSN 1573-0565. doi: 10.1007/s10994-020-05877-5. URL http://dx.doi.org/10.1007/s10994-020-05877-5.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3(null):1137–1155, March 2003. ISSN 1532-4435.

Benedikt Boecking and Artur Dubrawski. Pairwise feedback for data programming, 2019. URL http://arxiv.org/abs/1912.07685.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. Universal sentence encoder, 2018. URL https://arxiv.org/pdf/1803.11175.

Mayee F. Chen, Daniel Y. Fu, Frederic Sala, Sen Wu, Ravi Teja Mullapudi, Fait Poms, Kayvon Fatahalian, and Christopher Ré. Train and you'll miss it: Interactive model iteration with weak supervision and pre-trained embeddings, 2020.

Vijil Chenthamarakshan, Prem Melville, Vikas Sindhwani, and Richard D. Lawrence. Concept labeling: Building text classifiers with minimal supervision. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, page 1225–1230. AAAI Press, 2011. ISBN 9781577355144.

Benjamin Cohen-Wang, Stephen Mussmann, Alex Ratner, and Chris Ré. Interactive programmatic labeling for weak supervision, August 5 2019. URL https://bencw99.github.io/files/kdd2019_dcclworkshop.pdf.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data, 2018. URL https://arxiv.org/pdf/1705.02364.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Jared Dunnmon, Alexander Ratner, Nishith Khandwala, Khaled Saab, Matthew Markert, Hersh Sagreiya, Roger E. Goldman, Christopher Lee-Messer, Matthew P. Lungren, Daniel L. Rubin, and Christopher Ré. Cross-modal data programming enables rapid medical machine learning. *CoRR*, abs/1903.11101, 2019. URL http://arxiv.org/abs/1903.11101.

Nuno Filipe Fonseca Vasconcelos Escudeiro. Semi-automatic classification: using active learning for efficient class coverage, 2012. URL https://repositorio-aberto.up.pt/bitstream/10216/73245/2/25912.pdf.

Sara Evensen, Chang Ge, and Çagatay Demiralp. Ruler: Data programming by demonstration for document labeling. In Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020*, pages 1996–2005. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.findings-emnlp.181. URL https://www.aclweb.org/anthology/2020.findings-emnlp.181/.

Daniel Y. Fu, Mayee F. Chen, Frederic Sala, Sarah M. Hooper, Kayvon Fatahalian, and Christopher Ré. Fast and three-rious: Speeding up weak supervision with triplet methods, 2020.

Kim Hammar, Shatha Jaradat, Nima Dokoohaki, and Mihhail Matskin. Deep text mining of instagram data without strong supervision. *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, Dec 2018. doi: 10.1109/wi.2018.00-94. URL http://dx.doi.org/10.1109/WI.2018.00-94.

Braden Hancock, Paroma Varma, Stephanie Wang, Martin Bringmann, Percy Liang, and Christopher Ré. Training classifiers with natural language explanations. *CoRR*, abs/1805.03818, 2018. URL http://arxiv.org/abs/1805.03818v4.

Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *CoRR*, abs/1801.06146, 2018. URL http://arxiv.org/abs/1801.06146v5.

Stephanie D. Husby and Denilson Barbosa. Topic classification of blog posts using distant supervision. In *Proceedings of the Workshop on Semantic Analysis in Social Media*, EACL '12, page 28–36, USA, apr 2012. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W12-0604.

Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. *CoRR*, abs/1904.04717, 2019. URL http://arxiv.org/abs/1904.04717.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *CoRR*, abs/1607.01759, 2016. URL http://arxiv.org/abs/1607.01759.

Omer Levy, Yoav Goldberg, and Ido Dagan. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225, 2015. doi: 10.1162/tacl_a_00134. URL https://www.aclweb.org/anthology/Q15-1016.

Neil Mallinar, Abhishek Shah, Rajendra Ugrani, Ayush Gupta, Manikandan Gurusankar, Tin Kam Ho, Q. Vera Liao, Yunfeng Zhang, Rachel K.E. Bellamy, Robert Yates, Chris Desmarais, and Blake McGregor. Bootstrapping conversational agents with weak supervision. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:9528–9533, Jul 2019. ISSN 2159-5399. doi: 10.1609/aaai.v33i01.33019528. URL http://dx.doi.org/10.1609/aaai.v33i01.33019528.

Neil Mallinar, Abhishek Shah, Tin Kam Ho, Rajendra Ugrani, and Ayush Gupta. Iterative data programming for expanding text classification corpora, Apr. 2020. URL https://ojs.aaai.org/index.php/AAAI/article/view/7045.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. URL http://arxiv.org/abs/1301.3781.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, page 1003–1011, USA, 2009. Association for Computational Linguistics. ISBN 9781932432466.

Mona Nashaat, Aindrila Ghosh, James Miller, Shaikh Quader, Chad Marston, and Jean-François Puget. Hybridization of active learning and data programming for labeling large industrial datasets. *2018 IEEE International Conference on Big Data (Big Data)*, pages 46–55, Dec 2018. doi: 10.1109/BigData.2018.8622459. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8622459&isnumber=8621858.

Fredrik Olsson. A literature survey of active machine learning in the context of natural language processing, 2009. URL https://www.ccs.neu.edu/home/vip/teach/MLcourse/4_boosting/materials/SICS-T--2009-06--SE.pdf.

Alexander J. Quinn and Benjamin B. Bederson. Human computation: A survey and taxonomy of a growing field. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, page 1403–1412, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450302289. doi: 10.1145/1978942.1979148. URL https://doi.org/10.1145/1978942.1979148.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Alexander Ratner, Stephen H. Bach, Henry R. Ehrenberg, Jason Alan Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. *CoRR*, abs/1711.10160(3):269–282, November 2017a. ISSN 2150-8097. doi: 10.14778/3157794.3157797. URL http://arxiv.org/abs/1711.10160.

Alexander Ratner, Christopher De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. Data programming: Creating large training sets, quickly. In *Advances in neural information processing systems*, 2017b. URL https://arxiv.org/abs/1605.07723v3.

Sebastian Riedel, Limin Yao, and Andrew McCallum. Modeling relations and their mentions without labeled text. In *Proceedings of the 2010 European Conference on Machine Learning and Knowledge Discovery in Databases: Part III*, ECML PKDD'10, page 148–163, Berlin, Heidelberg, 2010. Springer-Verlag. ISBN 3642159389.

Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009. URL http://digital.library.wisc.edu/1793/60660.

Burr Settles. *Active Learning*. Morgan & Claypool Publishers, june 2012. ISBN 1608457257.

Patrice Y. Simard, David Maxwell Chickering, Aparna Lakshmiratan, Denis Xavier Charles, Léon Bottou, Carlos Garcia Jurado Suarez, David Grangier, Saleema Amershi, Johan Verwey, and Jina Suh. ICE: enabling non-experts to build models interactively for large-scale lopsided problems. *CoRR*, abs/1409.4814, 2014. URL http://arxiv.org/abs/1409.4814.

Alisa Smirnova and Philippe Cudré-Mauroux. Relation extraction using distant supervision: A survey. *ACM Computing Surveys (CSUR)*, 51(5):1–35, November 2018. ISSN 0360-0300. doi: 10.1145/3241741. URL https://doi.org/10.1145/3241741.

Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J. Pal. Learning general purpose distributed sentence representations via large scale multi-task learning. *CoRR*, abs/1804.00079, 2018. URL http://arxiv.org/abs/1804.00079v1.

Paroma Varma and Christopher Ré. Snuba: Automating weak supervision to label training data. *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, 12(3):223–236, nov 2018. ISSN 2150-8097. doi: 10.14778/3291264.3291268. URL https://doi.org/10.14778/3291264.3291268.

Paroma Varma, Bryan He, Dan Iter, Peng Xu, Rose Yu, Christopher De Sa, and Christopher Ré. Socratic learning: Augmenting generative models to incorporate latent subsets in training data, 2017a. URL https://arxiv.org/pdf/1610.08123v4.pdf.

Paroma Varma, Dan Iter, Christopher De Sa, and Christopher Ré. Flipper: A systematic approach to debugging training sets. In *Proceedings of the 2nd Workshop on Human-In-the-Loop Data Analytics*, HILDA'17, New York, NY, USA, 2017b. Association for Computing Machinery. ISBN 9781450350297. doi: 10.1145/3077257.3077263. URL https://doi.org/10.1145/3077257.3077263.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Jason Williams, Nobal B. Niraula, Pradeep Dasigi, Aparna Lakshmiratan, Carlos Garcia Jurado Suarez, Mouni Reddy, and Geoffrey Zweig. Rapidly scaling dialog systems with interactive learning. In *Natural language dialog systems and intelligent assistants*, pages 1–13. Springer, January 2015. URL https://www.microsoft.com/en-us/research/publication/rapidly-scaling-dialog-systems-with-interactive-learning/.

Peng Xu, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. Passage ranking with weak supervision, 2019. URL http://arxiv.org/abs/1905.05910.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237, 2019. URL http://arxiv.org/abs/1906.08237.

Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *CoRR*, abs/1708.02709, 2017. URL http://arxiv.org/abs/1708.02709v8.

Man-Ching Yuen, Irwin King, and Kwong-Sak Leung. A survey of crowdsourcing systems. In *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*, pages 766–773. IEEE, 2011. doi: 10.1109/PASSAT/SocialCom.2011.203.

Zhi-Hua Zhou. A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53, 2018. doi: https://dx.doi.org/10.1093/nsr/nwx106.

Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation, 2002. URL http://www.scholar.google.com/url?sa=U&q=http://www.cs.cmu.edu/~zhuxj/pub/propagate.ps.gz.

List of Figures

A. Appendix

A.1 Restaurant Review Dataset

A.1.1 Dataset Description

The dataset contains a collection of restaurant reviews assembled from multiple sources. A subset of the dataset containing Yelp reviews is distributed with this thesis along with the keyword labeler prototype source code (see chapter 4). The dataset was assembled by the IT company *Geneea Analytics*[1] to work on multi-label classification of restaurant reviews into 21 classes. The classes represent topics the text is talking about. The texts frequently discuss multiple topics at the same time.

Example of a review:

> "Great place with fresh food on my commute to King's Buildings. I generally pop in quickly for a takeaway sandwich, and have not been disappointed with the quality or the quantity of food up to now. The staff are friendly and efficient, and I can get whatever items I want in my sandwich. I just have to be careful not to let it fall apart in my hands as I eat! The sitdown café looks really nice too- a nice selection of hot drinks, cakes, and soups. I will pop back to give it a try some day soon."

List of labels:

- Experience-general
- Experience-environment
- Experience-hospitality
- Experience-order accuracy
- Experience-payment
- Experience-experience / pick–up & online & delivery
- Experience-speed
- Experience-other
- Food & drink-general
- Food & drink-availability
- Food & drink-health & safety
- Food & drink-ingredients
- Food & drink-menu
- Food & drink-quality
- Food & drink-size
- Food & drink-other
- Value-general
- Value-discounts & promotions

[1] https://geneea.com/

- Value-rewards
- Value-other
- Other-other

Browsing through the dataset, we observed that there likely exist unique keywords whose presence in the text indicates a class: e.g. "hospitality" indicates the class "Experience-hospitality". This inspired us to attempt to use keyword-based labeling for this classification problem.

A.1.2 Annotation

Geneea Analytics also provided an annotated dataset along with the problem. Three annotators were employed to label the data. The annotation is not distributed along with this thesis. However, we provide an analysis of the annotation, including inter-annotator agreement. This allows us to understand the problem in more detail.

Observations:

- Food&Drink-Quality is the most common category with 38-46% of reviews.
- The Experience-Environment label commonly appears in the data, in approximately 14% of the reviews. It has the highest agreement among annotators among common labels.
- The Value-General label has the most agreement among annotators among rare categories. Only approximately 3% of reviews receive this label.
- It seems that Annotator 3 mixed up categories Food&Drink – Quality with Food&Drink-General. Overall, this annotator performed the worst annotation.
- Due to sample size for measuring inter-annotator agreement (IAA) being only 150 samples, it is impossible to estimate the IAA for very rare categories, such as Food&Drink-Availability. However, it is a good sign that both Annotator 2 and Annotator 1 have identified similar data point amounts for the individual categories.

The data follows:

Category	Subcategory	Total count	Total percentage
food&drink	quality	4274	41.41%
experience	hospitality	2056	19.92%
experience	environment	1502	14.55%
experience	speed	455	4.41%
value	general	356	3.45%
food&drink	menu	297	2.88%
experience	general	225	2.18%
other	-	205	1.99%
food&drink	general	177	1.71%
experience	order accuracy	174	1.69%
food&drink	size	148	1.43%
food&drink	ingredients	125	1.21%
experience	other	105	1.02%
value	discounts & promotions	63	0.61%
food&drink	availability	45	0.44%
experience	pick-up & online & delivery	41	0.4%
food&drink	health & safety	26	0.25%
experience	payment	18	0.17%
food&drink	other	13	0.13%
value	rewards	13	0.13%
value	other	4	0.04%

Inter-Annotator Agreement					
Category	Subcategory	Annotator 2 Annotator 1	Annotator 2 Annotator 3	Annotator 3 Annotator 1	Annotator 2 Annotator 3 Annotator 1
food&drink	quality	0.482	0.003	0.005	0.057
experience	hospitality	0.603	0.73	0.608	0.646
experience	environment	0.689	0.653	0.737	0.693
experience	speed	0.54	0.457	0.233	0.412
value	general	0.571	0.784	0.651	0.671
food&drink	menu	0.371	0.286	0.169	0.283
experience	general	0.183	-0.019	-0.043	0.038
other	-	0.241	0.326	0.133	0.212
food&drink	general	0.0	0.0	0.002	0.001
experience	order accuracy	1.0	1.0	1.0	1.0
food&drink	size	0.716	0.72	0.641	0.691
food&drink	ingredients	-0.01	0.32	-0.01	0.13
experience	other	-0.018	-0.018	0.229	0.08
value	discounts & promotions	0.0	0.0	0.664	0.33
food&drink	availability	N/A	0.0	0.0	0.0
experience	pick-up & online & delivery	N/A	N/A	N/A	N/A
food&drink	health & safety	N/A	N/A	N/A	N/A
experience	payment	1.0	-0.007	-0.007	0.329
food&drink	other	N/A	0.0	0.0	0.0
value	rewards	N/A	N/A	N/A	N/A
value	other	0.0	N/A	0.0	0.0

Annotated Dataset Class Counts (Training Set)							
Category	Subcategory	Annotator 1		Annotator 2		Annotator 3	
		Count	Percentage	Count	Percentage	Count	Percentage
food&drink	quality	1664	38.71%	2607	45.95%	3	0.86%
experience	hospitality	806	18.75%	1156	20.37%	94	26.93%
experience	environment	614	14.28%	819	14.43%	69	19.77%
experience	speed	244	5.68%	203	3.58%	8	2.29%
value	general	160	3.72%	187	3.30%	9	2.58%
food&drink	menu	147	3.42%	148	2.61%	2	0.57%
experience	general	96	2.23%	126	2.22%	3	0.86%
other	-	149	3.47%	54	0.95%	2	0.57%
food&drink	general	23	0.54%	2	0.04%	152	43.55%
experience	order accuracy	85	1.98%	89	1.57%		
food&drink	size	95	2.21%	51	0.90%	2	0.57%
food&drink	ingredients	27	0.63%	96	1.69%	2	0.57%
experience	other	79	1.84%	24	0.42%	2	0.57%
value	discounts & promotions	25	0.58%	37	0.65%	1	0.29%
food&drink	availability	24	0.56%	21	0.37%		
experience	pick-up & online & delivery	19	0.44%	22	0.39%		
food&drink	health & safety	15	0.35%	11	0.19%		
experience	payment	7	0.16%	11	0.19%		
food&drink	other	7	0.16%	6	0.11%		
value	rewards	12	0.28%	1	0.02%		
value	other	1	0.02%	3	0.05%		

Annotated Dataset Class Counts (IAA Set)							
Category	Subcategory	Annotator 1		Annotator 2		Annotator 3	
		Count	Percentage	Count	Percentage	Count	Percentage
food&drink	quality	66	44.0%	80	53.3%	1	0.7%
experience	hospitality	26	17.3%	31	20.7%	40	26.7%
experience	environment	33	22.0%	24	16.0%	32	21.3%
experience	speed	1	0.7%	2	1.3%	1	0.7%
value	general	3	2.0%	4	2.7%	4	2.7%
food&drink	menu	5	3.3%	3	2.0%	1	0.7%
experience	general	4	2.7%	2	1.3%	3	2.0%
other	-	4	2.7%	1	0.7%	1	0.7%
food&drink	general	2	1.3%			64	42.7%
experience	order accuracy						
food&drink	size	5	3.3%	2	1.3%	1	0.7%
food&drink	ingredients	1	0.7%	1	0.7%	1	0.7%
experience	other					1	0.7%
value	discounts & promotions						
food&drink	availability						
experience	pick-up & online & delivery						
food&drink	health & safety						
experience	payment						
food&drink	other						
value	rewards						
value	other						

A.2 User Study Instructions

In the following pages, we include instructions used to perform the user study. They contain an explanation of keyword labeling, instructions to the keyword labeler protoytpe, strategies for suggested workflows and specifications of tasks. The text itself was not presented to the users, as we found that the cognitive load of studying the text was too difficult. Instead, these instructions were verbaly explained to the participants of the user study.

User study: Text Classification using Keyword-Based Labeling

Text classification traditionally requires texts to be manually labeled to be used as training data for machine learning algorithms. In keyword-based labeling, people find keywords whose presence in the text highly correlates with the assignment of a particular label. We provide an interactive tool that leverages an unannotated dataset of texts in the restaurant review domain to guide the user in keyword-based labeling. The tool also offers a similarity search function to the user. In this user study, we would like to determine the effectiveness of this alternative approach when compared to traditional manual annotation.

Contents:

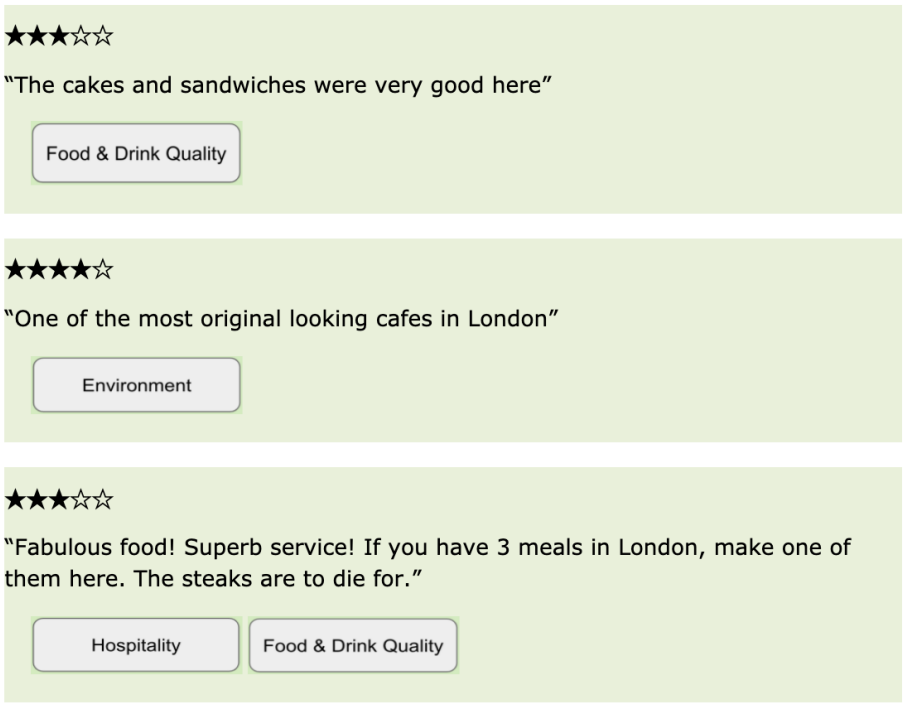
1. Keyword-Based Labeling

Restaurant Review Labeling

We are concerned with assigning labels to restaurant reviews. The labels represent selected topics reviewers often write about:

- *"Food & Drink Quality"* - Texts describing the quality of foods or drinks.
- *"Environment"* - Texts describing a restaurant's interior or atmosphere.
- *"Hospitality"* – Texts describing the service received are labeled.

A review can have multiple labels when it addresses multiple topics.



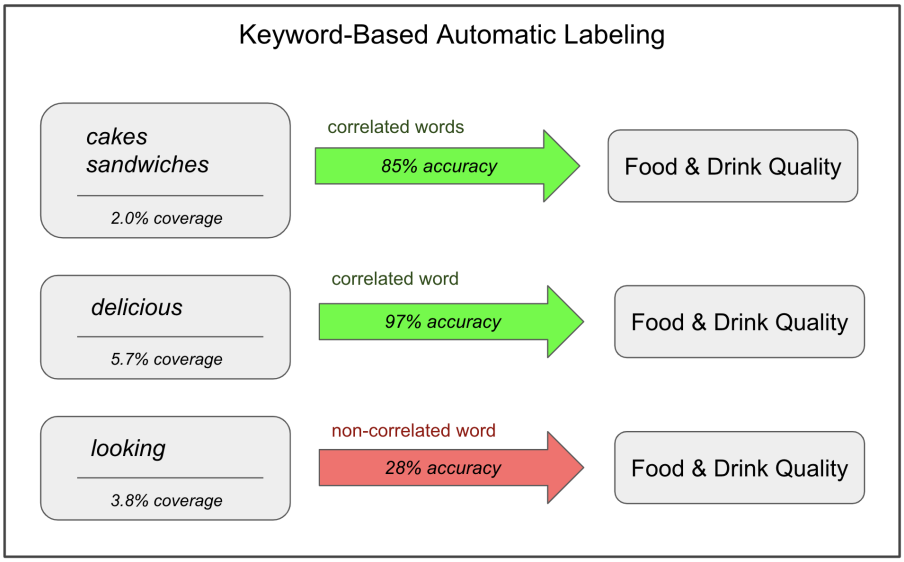
Examples of restaurant reviews with assigned labels

Keywords for Labeling

In restaurant reviews, certain words correlate with specific labels. For example, the word *"delicious"* highly correlates with the label *"Food & Drink Quality"*.

In keyword-based labeling, we guide users in discovering keywords that are highly indicative of a particular label when present in the text of a review. Using the keywords, we can automatically predict if the label should be assigned to a new, previously unseen text.

A keyword's chance of predicting the label correctly is called accuracy. We are only concerned with keywords with noticeably higher accuracy than words non-correlated with the label. (Words that appear in texts independently of the label (=non-correlated words) have accuracy equal to the label's frequency in the dataset.)



Accuracy & Coverage

Keyword coverage is the percentage of texts in the dataset containing the keyword. We compute keyword coverage using our dataset of restaurant reviews.

Finding keywords with both high coverage and accuracy is ideal. A keyword with low accuracy will only introduce incorrect labels when predicting an unseen text. A keyword with low coverage will rarely be seen in an unseen text and will thus provide very little improvement.

We would like total keyword coverage to approach label coverage.

Annotation

To estimate the accuracy of a keyword, we ask the user to annotate (=manually label) some texts containing the keyword randomly sampled from the restaurant review dataset. Estimated accuracy is equal to the percentage of texts that really belonged to the category.

Rarely occurring keywords deserve less annotation effort

Don’t spend too much time on annotating a single keyword. However, if only a small number of texts is annotated per keyword, the estimated accuracy is highly dependent on the few texts we ran into by chance. As a compromise, we let the user annotate very few texts for keywords occurring very rarely in the data but require more annotation for more frequent texts.

Keyword Groups

To simplify adding multiple related keywords, we allow adding of keyword groups. The number of samples to annotate is determined by the total occurrence of the keyword group. Texts containing words from the keyword group are sampled together, which allows the user not to sample a majority of keywords in the group. Only related words should be added; otherwise the “bad” words will spoil the estimated accuracy of the whole group and the “good” words won’t be used. Keyword group annotation works well if keywords have similar coverages and you expect them to have similar accuracies.

Look for new keyword ideas in sample texts

Without guidance, we suspect that coming up with keywords is hard for any user. We believe that annotation is a crucial step in coming up with new keywords. The user of this study should always take note (with pen-and-paper or a text editor) all potential keywords they come across when reading texts during annotation. Otherwise, they may run out of inspiration quickly. Annotation guidelines explaining the category should give enough inspiration for the first few keywords.

2. CLI Tool for Keyword-Based Labeling

We've implemented a tool for Keyword-Based Automatic Restaurant Review Labeling as a command-line application (CLI). A restaurant review dataset is included.

Run

Open the tutorial project *"user_study_example"*:

```
$ ./run_pipeline.py user_study_example
(food_drink_quality)
```

As shown in the prompt, the **"Food & Drink Quality"** label is pre-set in this project. This project exists just to try out the software. Later, other projects will be used for the actual user study.

Like any command-line application, the tool offers a prompt for inputting commands, most notably:

- `add_keywords`,
- `print_keywords`,
- `bulk_add_keywords`.

After each command finishes execution, the prompt will appear again.

Tab-Completion

This CLI supports tab-completion. Pressing Tab will autocomplete the command:

```
(food_drink_quality) p[Tab]
→
(food_drink_quality) print_keywords
```

Interrupt

Commands requiring additional user input may be interrupted by pressing CTRL-C:

```
(food_drink_quality) bulk_add_keywords
Enter keyword(s) for similarity search: [CTRL-C]
(food_drink_quality)
```

Exit

To exit, use either the exit command or press CTRL-D:

```
$ ./run_pipeline.py user_study_example
(food_drink_quality) exit
$
```

```
$ ./run_pipeline.py user_study_example
```

```
(food_drink_quality) [CTRL-D]
$
```

Autosave

After each action, the CLI automatically saves its state. The autosave is loaded automatically after CLI restart.

Command: `add_keywords`

To add, for example, the keywords *“cakes”* and *“sandwiches”* as a single keyword group, run:

```
(food_drink_quality) add_keywords
Enter keyword(s) to add: cakes sandwiches

    -keyword-                    -coverage-
-------------------------------------------------------------
     cakes                      0.6% (828/139875)
     sandwiches                 1.5% (2151/139875)

-------------------------------------------------------------
   Keyword group coverage:      2.0% (2829/139875)
-------------------------------------------------------------

Are these texts categorized as 'food_drink_quality' correctly? (y=yes / n=no / s=skip)

(1/10) three different sandwiches were ordered and enjoyed very much. the bread (white and wheat) was
very fresh tasting. (y=yes / n=no / s=skip)

... 9 more texts for annotation follow ...
```

Tips:

- Words already present in a previously added keyword group cannot be added again.
- If the text is long, just read sentences close to the occurrence of the keyword (highlighted red) in the text to save time.
- Write down all potential keywords you run across while annotating. For convenience, we always yellow-highlight existing keywords since they don’t need to be considered anymore.

Command: `print_keywords`

Shows all keyword groups, accuracies and coverages:

```
(food_drink_quality) print_keywords

-------------------------------------------------------------------------------------------------------------------
--                                          food_drink_quality                                                   --
-------------------------------------------------------------------------------------------------------------------
keyword_group_id: 1  enabled: True   coverage: 2.0% (2829/139875)  overlaps:  0.0%   est. accuracy: 70%  annotated: 10/10
-------------------------------------------------------------------------------------------------------------------
       -keyword-              -coverage-         -overlaps-
       cakes             0.6% (828/139875)         18.1%
       sandwiches        1.5% (2151/139875)        7.0%
```

```
-------------------------------------------------------------------------------------------------------------------------
-- Total Estimated Accuracy:  70.0% (= average of accuracies weighted by coverages)
-- Total Keyword Coverage:     2.0% (2829/139875)
-- Estimated Label Coverage:   1.4% (Estimated Accuracy * Total Keyword Coverage)
-- Actual Label Coverage:     75.0% (constant value; measured on a labeled dataset)
--
-- The goal is to achieve as high Estimated Label Coverage as possible (best is 75.0% = Actual Label Coverage),
-- while maintaining high Estimated Accuracy.
-------------------------------------------------------------------------------------------------------------------------

(food_drink_quality)
```

Tips:

- Overlap measures how many of the texts covered by the keyword or keyword groups would also be added by the other keyword groups nonetheless.

Command: `bulk_add_keywords`

This command helps the user by automatically suggesting new potential keywords. The command generates a list of 40 additional keywords for each set of keywords you try to add with. The command works by looking up similar words to the keywords provided.

Bulk adding *“sandwiches”* example:

```
(food_drink_quality) bulk_add_keywords
Enter keyword(s) for similarity search:
sandwiches

-index-  -similar_word-            -coverage-
  0      sandwiches              1.5% (2151/139875)
-------------------------------------------------------
  1      scones                  0.2% (326/139875)
  2      clotted                 0.0% (68/139875)
  3      Scones                  0.0% (19/139875)
  4      Afternoon               0.1% (202/139875)
  5      tier                    0.0% (30/139875)
  6      china                   0.0% (62/139875)
  7      deli                    0.2% (248/139875)
  8      turkey                  0.4% (558/139875)
  9      Subway                  0.2% (313/139875)
  10     Sandwiches              0.1% (123/139875)
  11     cakes                   0.6% (828/139875)
  12     sandwich                3.6% (5081/139875)
  13     pickle                  0.3% (422/139875)
  14     Sub                     0.3% (379/139875)
  15     Sandwich                0.2% (319/139875)
  16     Deli                    0.0% (53/139875)
  17     provolone               0.0% (62/139875)
  18     Wilde                   0.0% (14/139875)
  19     Hook                    0.2% (220/139875)
  20     jams                    0.0% (26/139875)
  21     sub                     2.7% (3712/139875)
  22     tarts                   0.0% (54/139875)
  23     pianist                 0.1% (89/139875)
  24     savouries               0.0% (15/139875)
  25     Tea                     0.3% (372/139875)
  26     delis                   0.0% (16/139875)
  27     pastrami                0.1% (118/139875)
  28     bagel                   0.1% (145/139875)
  29     swiss                   0.0% (44/139875)
  30     Ladder                  0.1% (164/139875)
```

```
  31     baguettes              0.0% (33/139875)
  32     teas                   0.2% (239/139875)
  33     mustard                0.3% (431/139875)
  34     steamer                0.1% (79/139875)
  35     latkes                 0.0% (12/139875)
  36     replenished            0.0% (42/139875)
  37     Georgian               0.0% (31/139875)
  38     Claridges              0.0% (29/139875)
  39     salon                  0.0% (13/139875)
  40     corned                 0.1% (77/139875)

Enter keyword index. Keywords from 0 to index (including) will be added:
3
Do you wish to confirm keywords one-by-one before adding them? If not, all will be added
(Y/n)
y

Will the following keywords categorize texts as 'food_drink_quality' correctly?

1. / 3: enjoyed a treat here recently with a delicious afternoon tea. they offered us a
choice of tables in the lounge part where we felt the chairs were quite low so we ate in
the conservatory part. we were served an amuse bouche which was unusual with the texture of
panacotta but made with cauliflower. there was a good selection of sandwiches and delicious
scones with jam and cream (the best bit!). as it was the queen's birthday the cakes all had
a royal theme including 'the queen's favourite fruit cake'. they brought out more
sandwiches and cakes when we asked and a very posh 'doggy bag' in the form of a smart
goring box to take the leftovers home! all accompanied by a variety of teas and a walk in
the pretty garden afterwards. very good. (y=yes=keep/n=no=discard/m=more)

... Annotation as in `add_keywords` follows ...
```

If you are not satisfied with the suggestion list above, try entering multiple related keywords instead of a single keyword. This usually gives better suggestions, as seen below:

```
(food_drink_quality) bulk_add_keywords
Enter keyword(s) for similarity search:
sandwiches burgers fries mustard

-index-  -similar_word-          -coverage-
  0      sandwiches             1.5% (2151/139875)
  0      burgers                1.0% (1453/139875)
  0      fries                  1.9% (2682/139875)
  0      mustard                0.3% (431/139875)
-------------------------------------------------------
  1      bun                    0.5% (660/139875)
  2      Burger                 0.4% (624/139875)
  3      patty                  0.2% (243/139875)
  4      cheeseburger           0.1% (146/139875)
  5      rings                  0.2% (326/139875)
  6      Burgers                0.1% (166/139875)
  7      patties                0.1% (120/139875)
  8      buns                   0.2% (262/139875)
  9      tots                   0.0% (53/139875)
  10     curds                  0.0% (56/139875)
  11     pickle                 0.3% (422/139875)
  12     hamburger              0.1% (153/139875)
  13     mayo                   0.3% (478/139875)
  14     poutine                0.2% (264/139875)
  15     Fries                  0.2% (274/139875)
```

```
  16     ketchup                0.2% (233/139875)
  17     juicy                  0.4% (615/139875)
  18     mayonnaise             0.1% (105/139875)
  19     sandwich               3.6% (5081/139875)
  20     Sandwich               0.2% (319/139875)
  21     burger                 2.3% (3168/139875)
  22     pastrami               0.1% (118/139875)
  23     Dijon                  0.0% (27/139875)
  24     relish                 0.0% (58/139875)
  25     Juicy                  0.0% (37/139875)
  26     chuck                  0.0% (20/139875)
  27     Reuben                 0.0% (30/139875)
  28     Priest                 0.0% (31/139875)
  29     Cheeseburger           0.0% (21/139875)
  30     In-N-Out               0.0% (15/139875)
  31     tater                  0.0% (38/139875)
  32     Philly                 0.1% (91/139875)
  33     gravy                  0.3% (455/139875)
  34     nuggets                0.0% (42/139875)
  35     Hero                   0.1% (72/139875)
  36     ranch                  0.1% (183/139875)
  37     spears                 0.0% (16/139875)
  38     Mustard                0.0% (14/139875)
  39     Sandwiches             0.1% (123/139875)
  40     gyros                  0.0% (65/139875)

... Continues as before ...
```

Tips:

- Words already present in a keyword group are highlighted yellow.
- A word can only exist in a single keyword group. Yellow-highlighted suggestions are automatically excluded when creating new keyword groups.
- Interrupt the command with CTRL-C to add/change keywords used for similarity search.
- To create a keyword group with just a few similar words, consider interrupting the command (CTRL-C) and adding the desired words using the `add_keywords` command.

Command: `further_annotate [keyword_group_id]`

Continue annotation of a keyword group.

Command: `remove_keyword_group`

Remove keyword groups created by mistake using the command *`remove_keyword_group`*:

```
(food_drink_quality) remove_keyword_group
Enter keyword group id to remove: 4
Removed keyword group id 4
(food_drink_quality)
```

Tips:

- Find keyword group ids using `print_keywords`.

Commands: `enable_keyword_group`, `disable_keyword_group`

You can use the commands `*enable_keyword_group*`, `*disable_keyword_group*` to enable/disable keyword groups by id. Keywords from the disabled groups are still highlighted yellow and cannot be added into new groups. Disabled keyword groups don't count toward the total accuracy and coverage as computed by `print_keywords`. Disabling and enabling keyword groups is useful for tuning total accuracy and coverage.

3. Tips & Tricks

1. Always browse restaurant reviews for new keywords when annotating them. Use a text editor or pen-and-paper to write down all potential keywords.

2. Sometimes, so many potential keywords are found that annotating them becomes the bottleneck. Group similar words and add one keyword group at a time, skipping annotation. Only after all are added, start annotating. The keywords you added will appear in yellow highlight in subsequent annotation, simplifying identification of new keywords when annotating the keyword groups with skipped annotation.

3. Double-check each word you are about to add to a new keyword group. Keyword groups don't allow adding and removing keywords. Removing a keyword group removes the group's annotation as well.

4. Similarity search works much better with multiple words in the search query. Start either with a single keyword or a few related keywords. Search for similar keywords. Often, only a few useful similar words are found. Add them to the search query and repeat the search. The new search hopefully finds much more relevant keywords. These can all be added together, making bulk-adding keywords more effective.

5. Try not to mix keywords with varying accuracies and coverages together in a single group.

6. Depending on desired final accuracy and coverage, disable keyword groups that do not provide enough precision for the added coverage. Delete keywords with precision so low, that they are of no use. Seeing bad keywords highlighted in yellow when annotating is distracting.

7. After a while of diving into searching for many similar keywords, take a step back and attack the problem from a different direction. It's easy to start adding heaps of low-coverage high-accuracy keywords that are all similar. It is more important to cover all different areas of a label. Look back at what you added, re-read annotation guidelines and think: "Is there a high-coverage high-accuracy?"

8. Don't waste too much time fiddling with disabling/enabling keywords. This can be done after the study. It is more important to find as many high-quality keywords as possible and prove it by annotation.

4. User Study Task Specification

Tutorial

50 minutes: Get yourself familiar with the Keyword-Based Labeling Tool on the “**Food & Drink Quality”** label. Re-read the *“Tips & Tricks”* section of these instructions:

```
$ ./run_pipeline.py user_study_example
(food_drink_quality)
```

First Task

30 + 20 minutes: Achieve as high as possible accuracy & coverage on the **“Experience Environment”** label. Don’t use `*bulk_add_keywords*` in the first 30 minutes.

```
$ ./run_pipeline.py user_study_environment
(experience_environment)
```

Second Task

30 minutes: Achieve as high as possible accuracy & coverage on the **“Experience Hospitality”** label.

```
$ ./run_pipeline.py user_study_hospitality
(experience_hospitality)
```

5. Annotation Guidelines

Food & Drink – Quality

taste, freshness, healthfulness, cold food, under/overcooked pasta/chicken, ...

Examples:

- “Food was excellent.”
- “All dishes were delicious”
- “To put it bluntly, it is hard to fail scrambled eggs, and for that price you would hope that at least they can make decent ones. But the eggs were a thick spongy white block, in fact, I don’t even know how to do that myself, and the bacon was oily and overcooked...”
- “I’m not really a fan of cheese but this was really good!!”
- “You can tell the ingredients are fresh and the pizzas are all made with lots of love”

Experience – Environment

cleanliness, atmosphere, decor, noise, seating/parking availability, ...

Examples:

- “the environment is luxurious yet relaxed”
- “the music not that cheerful”
- “Too loud to enjoy”
- “prob our favourite london tea venue in terms of decor/ambience”

Experience – Hospitality

nice/rude staff; great staff; staff attitude

Examples:

- “The hospitality is simply superb”
- “really friendly and efficient staff who have to manage working (with a smile!) in very tight space”
- “The welcome was friendly but a little cold initially from our waitress but as the evening went on she was v friendly and very informed regarding the menu.”

A.3 Example of Similar Words: *svd2vec*

This section contains a list of similar words to a seed keyword *toilet*. An svd2vec model trained on the provided restaurant review dataset is used to generate the similarity list. The list contains 200 words to showcase the similarity model. Our keyword labeler prototype shows by default only 40 most similar words.

#	Similar Word	Score
0	**toilet**	1.00
1	toilets	0.58
2	bathroom	0.54
3	unclean	0.54
4	laying	0.53
5	wipe	0.52
6	dirty	0.52
7	towels	0.51
8	soap	0.51
9	stains	0.48
10	gents	0.48
11	filthy	0.47
12	washroom	0.47
13	restroom	0.47
14	broken	0.47
15	napkins	0.46
16	cleaner	0.46
17	bathrooms	0.45
18	Bathroom	0.45
19	flush	0.45
20	wiped	0.45
21	sink	0.44
22	restrooms	0.44
23	wash	0.44
24	disgusting	0.44
25	smelly	0.43
26	bleach	0.43
27	urinal	0.43
28	napkin	0.43
29	washing	0.43
30	Toilets	0.43
31	dispenser	0.42
32	Bathrooms	0.42
33	floor	0.42
34	rag	0.42
35	Ladies	0.41
36	stall	0.41
37	wear	0.41
38	lids	0.40
39	tissue	0.40
40	wiping	0.40
41	unsanitary	0.40
42	cleaning	0.40
43	trash	0.40
44	towel	0.40
45	sanitary	0.40
46	cloth	0.40
47	dust	0.40
48	graffiti	0.39
49	coat	0.39
50	stalls	0.39
51	cleaned	0.39
52	stairs	0.39
53	swept	0.39
54	paper	0.38
55	littered	0.38
56	lock	0.38
57	hygiene	0.38
58	surfaces	0.38
59	dispensers	0.38
60	facilities	0.37
61	loo	0.37
62	ladies	0.37
63	men	0.37
64	gloves	0.37
65	forks	0.36
⋮	⋮	⋮
66	apron	0.36
67	disgusted	0.36
68	holders	0.36
69	entrance	0.36
70	debris	0.35
71	smelling	0.35
72	benches	0.35
73	steel	0.35
74	loos	0.35
75	staircase	0.35
76	signs	0.35
77	holder	0.34
78	mop	0.34
79	incense	0.34
80	puddle	0.34
81	floors	0.34
82	Dirty	0.34
83	supplies	0.34
84	condition	0.34
85	hallway	0.33
86	visible	0.33
87	disabled	0.33
88	garbage	0.33
89	handling	0.33
90	grossed	0.33
91	bin	0.33
92	inspection	0.33
93	everywhere	0.33
94	smelled	0.33
95	safety	0.33
96	flies	0.33
97	climb	0.32
98	dress	0.32
99	smelt	0.32
100	mirror	0.32
101	doors	0.32
102	straws	0.32
103	utter	0.32
104	rotten	0.32
105	paint	0.32
106	lap	0.32
107	ground	0.32
108	trays	0.31
109	push	0.31
110	users	0.31
111	wearing	0.31
112	silverware	0.31
113	Men	0.31
114	boots	0.31
115	crumbs	0.30
116	touching	0.30
117	drain	0.30
118	bucket	0.30
119	shower	0.30
120	bare	0.30
121	washed	0.30
122	wallpaper	0.30
123	needing	0.30
124	spoiling	0.30
125	trail	0.29
126	spill	0.29
127	grubby	0.29
128	dump	0.29
129	containers	0.29
130	Looked	0.29
131	slipping	0.29
132	halfway	0.29
⋮	⋮	⋮
133	cockroach	0.29
134	clue	0.29
135	contamination	0.29
136	sweeping	0.29
137	jeans	0.29
138	rid	0.29
139	odor	0.29
140	sweep	0.29
141	spilled	0.28
142	leads	0.28
143	dirt	0.28
144	worn	0.28
145	foul	0.28
146	sprayed	0.28
147	ruins	0.28
148	overlook	0.28
149	buckets	0.28
150	robbed	0.28
151	appearance	0.28
152	plastic	0.28
153	consuming	0.28
154	scraped	0.28
155	unfriendly	0.27
156	purse	0.27
157	descend	0.27
158	slippery	0.27
159	Tables	0.27
160	wobbly	0.27
161	mopping	0.27
162	sliding	0.27
163	ashamed	0.27
164	tub	0.27
165	utensils	0.27
166	wandering	0.27
167	bins	0.27
168	sticky	0.26
169	freeze	0.26
170	shoes	0.26
171	washrooms	0.26
172	pretend	0.26
173	cutlery	0.26
174	spilt	0.26
175	tear	0.26
176	lol	0.26
177	sundae	0.26
178	vent	0.26
179	areas	0.26
180	smell	0.26
181	lid	0.26
182	severe	0.26
183	women	0.26
184	IHOP	0.26
185	Pret	0.26
186	jacket	0.26
187	carpet	0.26
188	pants	0.26
189	castle	0.26
190	disgrace	0.26
191	lake	0.26
192	Avoid	0.26
193	low	0.26
194	saki	0.26
195	Ps	0.26
196	canteen	0.26
197	code	0.25
198	sweaty	0.25
199	hair	0.25
200	Floor	0.25

Table A.1: Long Similarity List of Words Similar to "**toilet**"

A.4 Keyword Lists Created in the User Study

Environment Label – User A

	Test Set 1	Test Set 2
True Pos.	19.6% (119/607)	28.0% (28/100)
False Neg.	8.9% (54/607)	6.0% (6/100)
False Pos.	14.0% (85/607)	20.0% (20/100)
True Neg.	57.5% (349/607)	46.0% (46/100)
Accuracy	77.1%	74.0%
Precision	58.3%	58.3%
Recall	68.8%	82.4%
F1	63.1%	68.3%
Gold Cov.	28.5% (173/607)	34.0% (34/100)
Pred. Cov.	33.6% (204/607)	48.0% (48/100)
Training Set Cov.	23.8% (33335/139875)	

DISABLED	Test Set 1	Test Set 2
True Pos.	16.6% (101/607)	24.0% (24/100)
False Neg.	11.9% (72/607)	10.0% (10/100)
False Pos.	8.6% (52/607)	9.0% (9/100)
True Neg.	62.9% (382/607)	57.0% (57/100)
Accuracy	79.6%	81.0%
Precision	66.0%	72.7%
Recall	58.4%	70.6%
F1	62.0%	71.6%
Gold Cov.	28.5% (173/607)	34.0% (34/100)
Pred. Cov.	33.6% (204/607)	33.0% (33/100)
Training Set Cov.	16.3% (22848/139875)	

Keyword Group 3 - Disable
Coverage: 7.2% (10034/139875)
Accuracy: 10% Annotated: 10/10
Test Set 1 Prec.: 44.9% Recall:12.7%
Test Set 2 Prec.: 27.3% Recall: 8.8%

-keyword-	-coverage-
table	6.8% (9506/139875)
dirty	0.5% (680/139875)

Keyword Group 4
Coverage: 2.3% (3165/139875)
Accuracy: 90% Annotated: 10/10
Test Set 1 Prec.: 86.7% Recall:7.5%
Test Set 2 Prec.: . % Recall: . %

-keyword-	-coverage-
clean	2.2% (3130/139875)
tidy	0.0% (68/139875)

Keyword Group 5
Coverage: 1.7% (2392/139875)
Accuracy: 86% Annotated: 7/7
Test Set 1 Prec.: 66.7% Recall:3.5%
Test Set 2 Prec.:100.0% Recall: 2.9%

-keyword-	-coverage-
comfortable	0.8% (1138/139875)
quiet	0.9% (1290/139875)

Keyword Group 6 - Disable
Coverage: 1.8% (2552/139875)
Accuracy: 14% Annotated: 7/7
Test Set 1 Prec.: 41.7% Recall:2.9%
Test Set 2 Prec.:100.0% Recall: 8.8%

-keyword-	-coverage-
glass	1.7% (2441/139875)
fridge	0.1% (121/139875)

Keyword Group 7
Coverage: 5.5% (7701/139875)
Accuracy: 80% Annotated: 10/10
Test Set 1 Prec.: 86.2% Recall:14.5%
Test Set 2 Prec.:100.0% Recall:20.6%

-keyword-	-coverage-
spacious	0.2% (299/139875)
atmosphere	5.3% (7433/139875)

Keyword Group 8
Coverage: 0.7% (1026/139875)
Accuracy: 60% Annotated: 5/5
Test Set 1 Prec.: 42.9% Recall:1.7%
Test Set 2 Prec.: . % Recall: . %

-keyword-	-coverage-
noise	0.3% (374/139875)
buzzy	0.1% (204/139875)
bright	0.3% (456/139875)

Keyword Group 9 - Disable
Coverage: 2.1% (2943/139875)
Accuracy: 80% Annotated: 10/10
Test Set 1 Prec.: 31.2% Recall:2.9%
Test Set 2 Prec.: 50.0% Recall: 5.9%

-keyword-	-coverage-
car	0.3% (357/139875)
parking	0.7% (959/139875)
street	1.3% (1814/139875)

Keyword Group 10
Coverage: 1.7% (2432/139875)
Accuracy: 86% Annotated: 7/7
Test Set 1 Prec.: 64.3% Recall:5.2%
Test Set 2 Prec.:100.0% Recall:11.8%

-keyword-	-coverage-
music	1.6% (2178/139875)
TV	0.2% (280/139875)

Keyword Group 11 - Disable
Coverage: 0.7% (945/139875)
Accuracy: 80% Annotated: 5/5
Test Set 1 Prec.: 40.0% Recall:1.2%
Test Set 2 Prec.: . % Recall: . %

-keyword-	-coverage-
air	0.3% (375/139875)
smell	0.4% (540/139875)
scent	0.0% (28/139875)
ventilation	0.0% (39/139875)

Keyword Group 12
Coverage: 0.4% (568/139875)
Accuracy: 100% Annotated: 3/3
Test Set 1 Prec.: 0.0% Recall:0.0%
Test Set 2 Prec.: . % Recall: . %

-keyword-	-coverage-
ground	0.4% (502/139875)
dustbin	0.0% (1/139875)
garage	0.0% (65/139875)

Keyword Group 13
Coverage: 0.1% (204/139875)
Accuracy: 0% Annotated: 1/1
Test Set 1 Prec.: NaN Recall:0.0%
Test Set 2 Prec.: . % Recall: . %

-keyword-	-coverage-
elevator	0.0% (56/139875)
lift	0.1% (141/139875)
escalator	0.0% (11/139875)

Keyword Group 14
Coverage: 1.2% (1744/139875)
Accuracy: 86% Annotated: 7/7
Test Set 1 Prec.: 20.0% Recall:1.2%
Test Set 2 Prec.: . % Recall: . %

-keyword-	-coverage-
dust	0.0% (55/139875)
wiped	0.1% (134/139875)
stains	0.0% (18/139875)
wipe	0.1% (81/139875)
grossed	0.0% (33/139875)
dispenser	0.1% (83/139875)
trash	0.1% (157/139875)
glove	0.0% (15/139875)
filthy	0.1% (85/139875)
silverware	0.0% (55/139875)
unclean	0.0% (15/139875)
unsanitary	0.0% (20/139875)
towels	0.0% (53/139875)
rag	0.0% (15/139875)
sweeping	0.0% (58/139875)
soap	0.0% (69/139875)
wiping	0.0% (53/139875)
cleaned	0.2% (266/139875)
flies	0.1% (87/139875)
mopping	0.0% (29/139875)
lids	0.0% (43/139875)
cleaning	0.2% (255/139875)
garbage	0.1% (101/139875)
stained	0.0% (38/139875)
urinal	0.0% (12/139875)
Behind	0.0% (20/139875)
washing	0.1% (81/139875)
laying	0.0% (36/139875)
Gross	0.0% (6/139875)
straws	0.0% (38/139875)
holders	0.0% (15/139875)
dusty	0.0% (22/139875)
Yuck	0.0% (6/139875)

Keyword Group 15
Coverage: 2.4% (3371/139875)
Accuracy: 40% Annotated: 10/10
Test Set 1 Prec.: 72.7% Recall:9.2%
Test Set 2 Prec.: 66.7% Recall:11.8%

-keyword-	-coverage-
carpet	0.0% (31/139875)
ceiling	0.1% (172/139875)
lighting	0.2% (347/139875)
walls	0.3% (378/139875)
painted	0.1% (90/139875)
tiles	0.0% (28/139875)
exposed	0.1% (95/139875)
neon	0.0% (42/139875)
panelling	0.0% (15/139875)
framed	0.0% (20/139875)
furniture	0.1% (136/139875)
paintings	0.0% (67/139875)
brick	0.1% (164/139875)
wall	0.5% (650/139875)
mirrors	0.0% (35/139875)
fabric	0.0% (27/139875)
chandeliers	0.0% (32/139875)
booths	0.2% (216/139875)
adorning	0.0% (20/139875)
lights	0.2% (295/139875)
curtains	0.0% (21/139875)
Lighting	0.0% (21/139875)
paint	0.0% (53/139875)
cloths	0.0% (36/139875)
concrete	0.0% (37/139875)
flowers	0.1% (109/139875)
ceilings	0.1% (78/139875)
décor	0.2% (249/139875)
tones	0.0% (16/139875)
wallpaper	0.0% (11/139875)
makeover	0.0% (22/139875)
cushions	0.0% (30/139875)
scattered	0.0% (49/139875)
cavernous	0.0% (24/139875)
brass	0.0% (23/139875)
plants	0.0% (41/139875)
vintage	0.1% (85/139875)
fixtures	0.0% (29/139875)
mirror	0.0% (22/139875)
sofas	0.0% (45/139875)
artwork	0.0% (59/139875)
brighter	0.0% (20/139875)
tacky	0.0% (54/139875)

Keyword Group 16
Coverage: 1.3% (1829/139875)
Accuracy: 57% Annotated: 7/7
Test Set 1 Prec.: 71.4% Recall:2.9%
Test Set 2 Prec.:100.0% Recall:11.8%

-keyword-	-coverage-
staircase	0.0% (53/139875)
homey	0.1% (80/139875)
cool	1.2% (1702/139875)

Keyword Group 17
Coverage: 0.7% (918/139875)
Accuracy: 100% Annotated: 5/5
Test Set 1 Prec.: 75.0% Recall:1.7%
Test Set 2 Prec.:100.0% Recall: 5.9%

-keyword-	-coverage-
indoors	0.0% (58/139875)
outdoors	0.0% (48/139875)
indoor	0.1% (148/139875)
outdoor	0.5% (632/139875)
balcony	0.1% (74/139875)
furnished	0.0% (47/139875)

Keyword Group 18
Coverage: 1.0% (1409/139875)
Accuracy: 86% Annotated: 7/7
Test Set 1 Prec.: 62.5% Recall:2.9%
Test Set 2 Prec.: . % Recall: . %

-keyword-	-coverage-
chairs	0.4% (498/139875)

```
bench              0.1% (110/139875)
park               0.3% (355/139875)
garden             0.3% (474/139875)

Keyword Group 19
Coverage: 0.6% (838/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 100.0% Recall:0.6%
Test Set 2 Prec.:100.0%  Recall: 8.8%
-------------------------------------
-keyword-              -coverage-
cozy               0.6% (824/139875)
swamped            0.0% (15/139875)
```

Environment Label – User B

```
|-----------------------------------------|
|          |  Test Set 1   |  Test Set 2  |
|----------+---------------+--------------|
| True Pos.|25.0% (152/607)|34.0% (34/100)|
|False Neg.| 3.5% ( 21/607)| 0.0% ( 0/100)|
|False Pos.|32.6% (198/607)|35.0% (35/100)|
| True Neg.|38.9% (236/607)|31.0% (31/100)|
|----------+---------------+--------------|
|  Accuracy|     63.9%     |     65.0%    |
| Precision|     43.4%     |     49.3%    |
|    Recall|     87.9%     |    100.0%    |
|        F1|     58.1%     |     66.0%    |
|-----------------------------------------|
| Gold Cov.|28.5% (173/607)|34.0% (34/100)|
|Pred. Cov.|57.7% (350/607)|69.0% (69/100)|
|-----------------------------------------|
| Training Set Cov.| 47.7% (66704/139875) |
|-----------------------------------------|


|-----------------------------------------|
| Disabled |  Test Set 1   |  Test Set 2  |
|----------+---------------+--------------|
| True Pos.|14.3% ( 87/607)|20.0% (20/100)|
|False Neg.|14.2% ( 86/607)|15.0% (14/100)|
|False Pos.| 3.8% ( 23/607)| 2.0% ( 2/100)|
| True Neg.|67.7% (411/607)|64.0% (64/100)|
|----------+---------------+--------------|
|  Accuracy|     82.0%     |     84.0%    |
| Precision|     79.1%     |     90.9%    |
|    Recall|     50.3%     |     58.8%    |
|        F1|     61.5%     |     71.4%    |
|-----------------------------------------|
| Gold Cov.|28.5% (173/607)|34.0% (34/100)|
|Pred. Cov.|57.7% (350/607)|22.0% (22/100)|
|-----------------------------------------|
| Training Set Cov.| 12.2% (17099/139875) |
|-----------------------------------------|

Keyword Group 1
Coverage: 3.1% (4380/139875)
Accuracy:  90%  Annotated: 10/10
Test Set 1 Prec.: 73.7%  Recall:8.1%
Test Set 2 Prec.: 83.3%  Recall:14.7%
-------------------------------------
-keyword-              -coverage-
seating            1.3% (1874/139875)
parking            0.7% (959/139875)
availability       0.0% (57/139875)
environment        0.5% (649/139875)
venue              0.7% (976/139875)

Keyword Group 2 - Disabled
Coverage:20.2% (28216/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.: 39.8%  Recall:28.3%
Test Set 2 Prec.: 43.2%  Recall:47.1%
-------------------------------------
-keyword-              -coverage-
place             20.2% (28216/139875)

Keyword Group 4 - Disabled
Coverage:14.3% (20072/139875)
Accuracy:  70%  Annotated: 10/10
Test Set 1 Prec.: 51.5%  Recall:30.1%
Test Set 2 Prec.: 28.6%  Recall: 5.9%
-------------------------------------
-keyword-              -coverage-
restaurant        14.3% (20072/139875)

Keyword Group 5 - Disabled
Coverage:14.0% (19584/139875)
Accuracy:  30%  Annotated: 10/10
Test Set 1 Prec.: 32.2%  Recall:16.2%
Test Set 2 Prec.: 60.0%  Recall:35.3%
-------------------------------------
-keyword-              -coverage-
located            0.9% (1269/139875)
local              1.9% (2644/139875)
look               3.6% (5091/139875)
looking            3.8% (5318/139875)
smell              0.4% (540/139875)
smelling           0.1% (76/139875)
feel               4.2% (5938/139875)
feeling            1.0% (1457/139875)

Keyword Group 6
Coverage: 5.3% (7433/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.: 84.6%  Recall:12.7%
Test Set 2 Prec.:100.0%  Recall:20.6%
-------------------------------------
-keyword-              -coverage-
atmosphere         5.3% (7433/139875)

Keyword Group 7
Coverage: 4.3% (6041/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.: 88.9%  Recall:13.9%
Test Set 2 Prec.:100.0%  Recall: 8.8%
-------------------------------------
-keyword-              -coverage-
cleanliness        0.2% (219/139875)
decor              1.9% (2672/139875)
noise              0.3% (374/139875)
quiet              0.9% (1290/139875)
wallpaper          0.0% (11/139875)
noisy              0.4% (590/139875)
lively             0.4% (494/139875)
echo               0.0% (30/139875)
intimate           0.3% (488/139875)
quieter            0.1% (146/139875)
dingy              0.0% (43/139875)
television         0.0% (42/139875)

Keyword Group 8 - Disabled
Coverage: 7.8% (10923/139875)
Accuracy: NAN   annotated:  0/10
Test Set 1 Prec.: 44.4%  Recall:11.6%
Test Set 2 Prec.: 83.3%  Recall:14.7%
-------------------------------------
-keyword-              -coverage-
pub                1.4% (1916/139875)
building           0.5% (759/139875)
college            0.1% (126/139875)
park               0.3% (355/139875)
corner             0.7% (1027/139875)
road               0.3% (484/139875)
hotel              1.1% (1482/139875)
store              1.2% (1609/139875)
street             1.3% (1814/139875)
town               0.9% (1277/139875)
city               0.7% (1033/139875)
village            0.0% (57/139875)

Keyword Group 9 - Disabled
Coverage: 5.1% (7149/139875)
Accuracy: NAN   annotated:  0/10
Test Set 1 Prec.: 43.3%  Recall:7.5%
Test Set 2 Prec.: 75.0%  Recall: 8.8%
-------------------------------------
-keyword-              -coverage-
space              1.1% (1473/139875)
land               0.1% (73/139875)
outdoor            0.5% (632/139875)
location           3.7% (5145/139875)

Keyword Group 10
Coverage: 0.8% (1138/139875)
Accuracy: NAN   annotated:  0/5
Test Set 1 Prec.: 66.7%  Recall:2.3%
Test Set 2 Prec.:100.0%  Recall: 2.9%
-------------------------------------
-keyword-              -coverage-
comfortable        0.8% (1138/139875)
```

Environment Label – User C

```
|-----------------------------------------|
|          |  Test Set 1   |  Test Set 2  |
|----------+---------------+--------------|
| True Pos.|21.3% (129/607)|31.0% (31/100)|
|False Neg.| 7.2% ( 44/607)| 3.0% ( 3/100)|
|False Pos.|12.7% ( 77/607)| 9.0% ( 9/100)|
| True Neg.|58.8% (357/607)|57.0% (57/100)|
|----------+---------------+--------------|
|  Accuracy|     80.1%     |     88.0%    |
| Precision|     62.6%     |     77.5%    |
|    Recall|     74.6%     |     91.2%    |
|        F1|     68.1%     |     83.8%    |
|-----------------------------------------|
| Gold Cov.|28.5% (173/607)|34.0% (34/100)|
|Pred. Cov.|33.9% (206/607)|40.0% (40/100)|
|-----------------------------------------|
| Training Set Cov.| 23.8% (33238/139875) |
|-----------------------------------------|


|-----------------------------------------|
| Disabled |  Test Set 1   |  Test Set 2  |
|----------+---------------+--------------|
| True Pos.|20.8% (126/607)|29.0% (29/100)|
|False Neg.| 7.7% ( 47/607)| 5.0% ( 5/100)|
|False Pos.|11.2% ( 68/607)| 7.0% ( 7/100)|
| True Neg.|60.3% (366/607)|59.0% (59/100)|
|----------+---------------+--------------|
|  Accuracy|     81.1%     |     88.0%    |
| Precision|     64.9%     |     80.6%    |
|    Recall|     72.8%     |     85.3%    |
|        F1|     68.7%     |     82.9%    |
|-----------------------------------------|
| Gold Cov.|28.5% (173/607)|34.0% (34/100)|
|Pred. Cov.|32.0% (194/607)|36.0% (36/100)|
|-----------------------------------------|
| Training Set Cov.| 22.9% (31967/139875) |
|-----------------------------------------|

Keyword Group 1
Coverage: 5.3% (7433/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.: 84.6%  Recall:12.7%
Test Set 2 Prec.:100.0%  Recall:20.6%
-------------------------------------
-keyword-              -coverage-
atmosphere         5.3% (7433/139875)

Keyword Group 2
Coverage: 2.0% (2846/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.: 100.0% Recall:9.8%
Test Set 2 Prec.:100.0%  Recall: 8.8%
-------------------------------------
-keyword-              -coverage-
decor              1.9% (2672/139875)
decoration         0.1% (180/139875)

Keyword Group 3
Coverage: 1.6% (2178/139875)
Accuracy:  86%  Annotated:  7/7
Test Set 1 Prec.: 61.5%  Recall:4.6%
Test Set 2 Prec.:100.0%  Recall:11.8%
-------------------------------------
-keyword-              -coverage-
music              1.6% (2178/139875)

Keyword Group 4
Coverage: 1.3% (1874/139875)
Accuracy: NAN   annotated:  5/7
Test Set 1 Prec.: 57.1%  Recall:2.3%
Test Set 2 Prec.:100.0%  Recall: 8.8%
-------------------------------------
-keyword-              -coverage-
seating            1.3% (1874/139875)

Keyword Group 5
```

```
Coverage: 0.9% (1212/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 100.0% Recall:1.2%
Test Set 2 Prec.:100.0%  Recall: 8.8%
-------------------------------------
-keyword-              -coverage-
ambiance           0.9% (1212/139875)

Keyword Group 6
Coverage: 0.7% (959/139875)
Accuracy:  80%  Annotated:  5/5
Test Set 1 Prec.: 75.0%  Recall:1.7%
Test Set 2 Prec.: 66.7%  Recall: 5.9%
-------------------------------------
-keyword-              -coverage-
parking            0.7% (959/139875)

Keyword Group 7
Coverage: 1.7% (2398/139875)
Accuracy: 100%  Annotated:  7/7
Test Set 1 Prec.: 50.0%  Recall:2.3%
Test Set 2 Prec.:100.0%  Recall: 5.9%
-------------------------------------
-keyword-              -coverage-
loud               0.8% (1161/139875)
quiet              0.9% (1290/139875)

Keyword Group 8
Coverage: 3.7% (5145/139875)
Accuracy:  70%  Annotated: 10/10
Test Set 1 Prec.: 23.8%  Recall:2.9%
Test Set 2 Prec.: 50.0%  Recall: 2.9%
-------------------------------------
-keyword-              -coverage-
location           3.7% (5145/139875)

Keyword Group 9
Coverage: 0.7% (916/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 100.0% Recall:3.5%
Test Set 2 Prec.: 50.0%  Recall: 2.9%
-------------------------------------
-keyword-              -coverage-
Interior           0.0% (60/139875)
window             0.6% (856/139875)

Keyword Group 10
Coverage: 1.6% (2291/139875)
Accuracy:  71%  Annotated:  7/7
Test Set 1 Prec.: 60.0%  Recall:3.5%
Test Set 2 Prec.:100.0%  Recall: 8.8%
-------------------------------------
-keyword-              -coverage-
charming           0.4% (523/139875)
comfortable        0.8% (1138/139875)
romantic           0.5% (664/139875)

Keyword Group 11
Coverage: 0.8% (1119/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 50.0%  Recall:1.2%
Test Set 2 Prec.:   . %  Recall:  . %
-------------------------------------
-keyword-              -coverage-
floor              0.8% (1119/139875)

Keyword Group 12
Coverage: 0.7% (1022/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 80.0%  Recall:2.3%
Test Set 2 Prec.:   . %  Recall:  . %
-------------------------------------
-keyword-              -coverage-
noise              0.3% (374/139875)
environment        0.5% (649/139875)

Keyword Group 13
Coverage: 0.8% (1098/139875)
Accuracy:  80%  Annotated:  5/5
Test Set 1 Prec.: 83.3%  Recall:2.9%
Test Set 2 Prec.:   . %  Recall:  . %
-------------------------------------
-keyword-              -coverage-
relaxed            0.8% (1098/139875)

Keyword Group 14
Coverage: 0.6% (871/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 100.0% Recall:1.7%
Test Set 2 Prec.:100.0%  Recall: 2.9%
-------------------------------------
-keyword-              -coverage-
vibe               0.6% (808/139875)
vibes              0.0% (65/139875)

Keyword Group 15
Coverage: 1.0% (1370/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 66.7%  Recall:2.3%
Test Set 2 Prec.:100.0%  Recall: 8.8%
-------------------------------------
-keyword-              -coverage-
patio              1.0% (1370/139875)

Keyword Group 16
Coverage: 1.1% (1493/139875)
Accuracy: 100%  Annotated:  7/7
Test Set 1 Prec.: 66.7%  Recall:4.6%
Test Set 2 Prec.:   . %  Recall:  . %
-------------------------------------
-keyword-              -coverage-
upstairs           0.6% (804/139875)
downstairs         0.5% (740/139875)
cellar             0.1% (137/139875)

Keyword Group 17
Coverage: 0.8% (1055/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 100.0% Recall:2.9%
Test Set 2 Prec.:100.0%  Recall: 2.9%
-------------------------------------
-keyword-              -coverage-
interior           0.8% (1055/139875)

Keyword Group 18
Coverage: 0.3% (373/139875)
Accuracy: 100%  Annotated:  3/3
Test Set 1 Prec.: 100.0% Recall:1.2%
Test Set 2 Prec.:   . %  Recall:  . %
-------------------------------------
-keyword-              -coverage-
design             0.3% (373/139875)

Keyword Group 19
Coverage: 0.6% (843/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 75.0%  Recall:1.7%
Test Set 2 Prec.:   . %  Recall:  . %
-------------------------------------
-keyword-              -coverage-
spacious           0.2% (299/139875)
wooden             0.2% (345/139875)
stools             0.1% (162/139875)
stool              0.0% (55/139875)

Keyword Group 20 - Disable
Coverage: 0.5% (711/139875)
Accuracy:  80%  Annotated:  5/5
Test Set 1 Prec.: 100.0% Recall:0.6%
Test Set 2 Prec.:   . %  Recall:  . %
-------------------------------------
-keyword-              -coverage-
central            0.5% (711/139875)

Keyword Group 21
Coverage: 1.6% (2190/139875)
Accuracy: 100%  Annotated:  7/7
Test Set 1 Prec.: 87.5%  Recall:4.0%
Test Set 2 Prec.:   . %  Recall:  . %
-------------------------------------
-keyword-              -coverage-
view               1.1% (1534/139875)
views              0.5% (754/139875)

Keyword Group 22
Coverage: 1.1% (1473/139875)
Accuracy: 100%  Annotated:  7/7
Test Set 1 Prec.: 100.0% Recall:4.0%
Test Set 2 Prec.:   . %  Recall:  . %
-------------------------------------
-keyword-              -coverage-
space              1.1% (1473/139875)

Keyword Group 23
Coverage: 0.9% (1246/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 80.0%  Recall:4.6%
Test Set 2 Prec.:   . %  Recall:  . %
-------------------------------------
-keyword-              -coverage-
noisy              0.4% (590/139875)
crowded            0.5% (692/139875)

Keyword Group 24
Coverage: 0.6% (900/139875)
Accuracy:  80%  Annotated:  5/5
Test Set 1 Prec.: 100.0% Recall:0.6%
Test Set 2 Prec.:100.0%  Recall: 5.9%
-------------------------------------
-keyword-              -coverage-
hip                0.2% (265/139875)
tv                 0.1% (116/139875)
buzz               0.2% (282/139875)
rustic             0.2% (240/139875)

Keyword Group 25 - Disable
Coverage: 1.4% (2018/139875)
Accuracy: 100%  Annotated:  7/7
Test Set 1 Prec.: 60.0%  Recall:3.5%
Test Set 2 Prec.: 50.0%  Recall: 2.9%
-------------------------------------
-keyword-              -coverage-
glow               0.0% (20/139875)
cavernous          0.0% (24/139875)
panelling          0.0% (15/139875)
industrial         0.1% (137/139875)
Victorian          0.0% (52/139875)
basement           0.2% (273/139875)
Light              0.1% (83/139875)
sharp              0.1% (91/139875)
chandelier         0.0% (18/139875)
John               0.1% (186/139875)
German             0.1% (111/139875)
crystal            0.0% (38/139875)
Downstairs         0.0% (46/139875)
ceilings           0.1% (78/139875)
building           0.5% (759/139875)
Dark               0.0% (69/139875)
flowers            0.1% (109/139875)

Keyword Group 26
Coverage: 1.1% (1524/139875)
Accuracy: NAN   annotated:  0/7
Test Set 1 Prec.: 87.5%  Recall:4.0%
Test Set 2 Prec.:100.0%  Recall: 5.9%
-------------------------------------
-keyword-              -coverage-
intimate           0.3% (488/139875)
outdoor            0.5% (632/139875)
hanging            0.2% (321/139875)
lamps              0.0% (45/139875)
shade              0.0% (64/139875)
```

Environment Label – User D

```
|-----------------------------------------|
|          |  Test Set 1   |  Test Set 2  |
|----------+---------------+--------------|
| True Pos.|23.9% (145/607)|33.0% (33/100)|
|False Neg.| 4.6% ( 28/607)| 1.0% ( 1/100)|
|False Pos.|32.8% (199/607)|38.0% (38/100)|
| True Neg.|38.7% (235/607)|28.0% (28/100)|
|----------+---------------+--------------|
|  Accuracy|     62.6%     |     61.0%    |
| Precision|     42.2%     |     46.5%    |
|    Recall|     83.8%     |     97.1%    |
|        F1|     56.1%     |     62.9%    |
|-----------------------------------------|
| Gold Cov.|28.5% (173/607)|34.0% (34/100)|
|Pred. Cov.|56.7% (344/607)|71.0% (71/100)|
|-----------------------------------------|
| Training Set Cov.| 43.9% (61424/139875) |
|-----------------------------------------|

|-----------------------------------------|
| Disabled |  Test Set 1   |  Test Set 2  |
|----------+---------------+--------------|
| True Pos.|19.8% (145/607)|25.0% (25/100)|
|False Neg.| 8.7% ( 28/607)| 9.0% ( 9/100)|
|False Pos.| 7.4% (199/607)| 5.0% ( 5/100)|
| True Neg.|64.1% (235/607)|61.0% (61/100)|
|----------+---------------+--------------|
|  Accuracy|     83.9%     |     86.0%    |
| Precision|     72.7%     |     83.3%    |
|    Recall|     69.4%     |     73.5%    |
|        F1|     71.0%     |     78.1%    |
|-----------------------------------------|
| Gold Cov.|28.5% (173/607)|34.0% (34/100)|
|Pred. Cov.|27.2% (165/607)|30.0% (30/100)|
|-----------------------------------------|
| Training Set Cov.| 19.9% (27784/139875) |
|-----------------------------------------|

Keyword Group 2
Coverage: 5.3% (7433/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.: 84.6%  Recall:12.7%
Test Set 2 Prec.:100.0%  Recall:20.6%
-----------------------------------
-keyword-              -coverage-
atmosphere         5.3% (7433/139875)

Keyword Group 3
Coverage: 1.9% (2672/139875)
Accuracy: 100%  Annotated:  7/7
Test Set 1 Prec.: 100.0% Recall:8.7%
Test Set 2 Prec.:100.0%  Recall: 8.8%
-----------------------------------
-keyword-              -coverage-
decor              1.9% (2672/139875)

Keyword Group 4
Coverage: 1.7% (2336/139875)
Accuracy:  86%  Annotated:  7/7
Test Set 1 Prec.: 61.5%  Recall:4.6%
Test Set 2 Prec.:100.0%  Recall:11.8%
-----------------------------------
-keyword-              -coverage-
music              1.6% (2178/139875)
entertainment      0.1% (178/139875)

Keyword Group 5
Coverage: 0.2% (224/139875)
Accuracy: 100%  Annotated:  1/1
Test Set 1 Prec.:   NaN  Recall:0.0%
Test Set 2 Prec.:   . %  Recall:  . %
-----------------------------------
-keyword-              -coverage-
games              0.2% (224/139875)

Keyword Group 6
Coverage: 0.0% (7/139875)
Accuracy: 100%  Annotated:  1/1
Test Set 1 Prec.:   NaN  Recall:0.0%
Test Set 2 Prec.:   . %  Recall:  . %
-----------------------------------
-keyword-              -coverage-
darts              0.0% (7/139875)

Keyword Group 7
Coverage: 2.1% (2996/139875)
Accuracy:  90%  Annotated: 10/10
Test Set 1 Prec.: 60.0%  Recall:3.5%
Test Set 2 Prec.:100.0%  Recall: 5.9%
-----------------------------------
-keyword-              -coverage-
singer             0.0% (57/139875)
loud               0.8% (1161/139875)
quiet              0.9% (1290/139875)
noisy              0.4% (590/139875)

Keyword Group 8
Coverage: 0.5% (698/139875)
Accuracy:  67%  Annotated:  3/3
Test Set 1 Prec.: 0.0%   Recall:0.0%
Test Set 2 Prec.:100.0%  Recall: 2.9%
-----------------------------------
-keyword-              -coverage-
guitar             0.0% (24/139875)
accordion          0.0% (8/139875)
singing            0.1% (132/139875)
crowd              0.4% (539/139875)

Keyword Group 9
Coverage: 2.2% (3130/139875)
Accuracy:  90%  Annotated: 10/10
Test Set 1 Prec.: 86.7%  Recall:7.5%
Test Set 2 Prec.:   . %  Recall:  . %
-----------------------------------
-keyword-              -coverage-
clean              2.2% (3130/139875)

Keyword Group 10
Coverage: 0.8% (1055/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 81.8%  Recall:5.2%
Test Set 2 Prec.:   . %  Recall:  . %
-----------------------------------
-keyword-              -coverage-
tidy               0.0% (68/139875)
spacious           0.2% (299/139875)
crowded            0.5% (692/139875)

Keyword Group 11
Coverage: 0.2% (248/139875)
Accuracy: 100%  Annotated:  1/1
Test Set 1 Prec.: 100.0% Recall:0.6%
Test Set 2 Prec.:100.0%  Recall: 5.9%
-----------------------------------
-keyword-              -coverage-
quaint             0.2% (248/139875)

Keyword Group 12
Coverage: 2.0% (2748/139875)
Accuracy:  86%  Annotated:  7/7
Test Set 1 Prec.: 73.3%  Recall:6.4%
Test Set 2 Prec.: 66.7%  Recall: 5.9%
-----------------------------------
-keyword-              -coverage-
fun                1.4% (2000/139875)
vibe               0.6% (808/139875)

Keyword Group 13
Coverage: 0.3% (374/139875)
Accuracy:  67%  Annotated:  3/3
Test Set 1 Prec.: 50.0%  Recall:0.6%
Test Set 2 Prec.:   . %  Recall:  . %
-----------------------------------
-keyword-              -coverage-
noise              0.3% (374/139875)

Keyword Group 14
Coverage: 0.7% (959/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 75.0%  Recall:1.7%
Test Set 2 Prec.: 66.7%  Recall: 5.9%
-----------------------------------
-keyword-              -coverage-
parking            0.7% (959/139875)

Keyword Group 15 - Disabled
Coverage:33.9% (47403/139875)
Accuracy:  70%  Annotated: 10/10
Test Set 1 Prec.: 36.3%  Recall:44.5%
Test Set 2 Prec.: 42.0%  Recall:61.8%
-----------------------------------
-keyword-              -coverage-
great             19.1% (26663/139875)
place             20.2% (28216/139875)

Keyword Group 17
Coverage: 1.5% (2146/139875)
Accuracy:  71%  Annotated:  7/7
Test Set 1 Prec.: 55.6%  Recall:2.9%
Test Set 2 Prec.: 75.0%  Recall: 8.8%
-----------------------------------
-keyword-              -coverage-
seating            1.3% (1874/139875)
booth              0.2% (306/139875)

Keyword Group 18
Coverage: 2.2% (3066/139875)
Accuracy:  70%  Annotated: 10/10
Test Set 1 Prec.: 63.2%  Recall:6.9%
Test Set 2 Prec.:100.0%  Recall:14.7%
-----------------------------------
-keyword-              -coverage-
dancing            0.1% (135/139875)
trendy             0.3% (425/139875)
watching           0.5% (765/139875)
dancers            0.0% (37/139875)
pong               0.0% (32/139875)
tunes              0.0% (48/139875)
dj                 0.0% (33/139875)
DJ                 0.1% (144/139875)
songs              0.0% (59/139875)
bands              0.0% (49/139875)
ping               0.0% (42/139875)
dance              0.1% (174/139875)
band               0.2% (263/139875)
performing         0.0% (25/139875)
plays              0.0% (64/139875)
audience           0.0% (28/139875)
videos             0.0% (34/139875)
song               0.0% (68/139875)
danced             0.0% (26/139875)
played             0.2% (266/139875)
acts               0.1% (75/139875)
stage              0.1% (118/139875)
karaoke            0.0% (51/139875)
speakers           0.0% (31/139875)
blasting           0.0% (23/139875)
oldies             0.0% (18/139875)
playlist           0.0% (20/139875)
stag               0.0% (29/139875)
play               0.3% (470/139875)

Keyword Group 20
Coverage: 4.4% (6220/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.: 60.0%  Recall:8.7%
Test Set 2 Prec.:100.0%  Recall:20.6%
-----------------------------------
-keyword-              -coverage-
outdoor            0.5% (632/139875)
chairs             0.4% (498/139875)
stools             0.1% (162/139875)
couch              0.0% (40/139875)
decorated          0.3% (489/139875)
lively             0.4% (494/139875)
outside            2.5% (3436/139875)
couches            0.0% (49/139875)
comfy              0.1% (194/139875)
benches            0.1% (92/139875)
heaters            0.1% (96/139875)
windows            0.2% (306/139875)
cushions           0.0% (30/139875)
indoor             0.1% (148/139875)
sofa               0.0% (47/139875)
lamps              0.0% (45/139875)
```

Hospitality Label – User A

	Test Set 1	Test Set 2
True Pos.	44.3% (269/607)	45.0% (45/100)
False Neg.	3.8% (23/607)	4.0% (4/100)
False Pos.	13.5% (82/607)	8.0% (8/100)
True Neg.	38.4% (233/607)	43.0% (43/100)
Accuracy	82.7%	88.0%
Precision	76.6%	84.9%
Recall	92.1%	91.8%
F1	83.7%	88.2%
Gold Cov.	48.3% (293/607)	49.0% (49/100)
Pred. Cov.	57.8% (351/607)	53.0% (53/100)
Training Set Cov.	45.3% (63302/139875)	

No groups to disable

Keyword Group 1
Coverage:30.6% (42826/139875)
Accuracy: 100% Annotated: 10/10
Test Set 1 Prec.: 78.8% Recall:49.7%
Test Set 2 Prec.: 86.4% Recall:38.8%

-keyword-	-coverage-
service	20.1% (28106/139875)
staff	13.3% (18637/139875)

Keyword Group 2
Coverage: 5.7% (7908/139875)
Accuracy: 80% Annotated: 10/10
Test Set 1 Prec.: 54.8% Recall:7.9%
Test Set 2 Prec.: 83.3% Recall:10.2%

-keyword-	-coverage-
waiter	3.1% (4382/139875)
waitress	2.7% (3740/139875)

Keyword Group 3
Coverage:12.5% (17451/139875)
Accuracy: 100% Annotated: 10/10
Test Set 1 Prec.: 83.6% Recall:20.9%
Test Set 2 Prec.: 87.5% Recall:14.3%

-keyword-	-coverage-
friendly	11.4% (15911/139875)
Friendly	1.1% (1569/139875)

Keyword Group 4
Coverage: 7.8% (10932/139875)
Accuracy: 100% Annotated: 10/10
Test Set 1 Prec.: 75.0% Recall:11.3%
Test Set 2 Prec.:100.0% Recall:10.2%

-keyword-	-coverage-
helpful	2.0% (2859/139875)
unfriendly	0.1% (175/139875)
annoying	0.2% (315/139875)
aggressive	0.1% (82/139875)
rude	0.9% (1289/139875)
irritating	0.0% (52/139875)
assisted	0.0% (49/139875)
cheerful	0.2% (223/139875)
attentive	2.7% (3760/139875)
bothering	0.1% (81/139875)
rudest	0.0% (35/139875)
condescending	0.0% (47/139875)
loudly	0.1% (116/139875)
stressed	0.1% (81/139875)
dismissive	0.0% (39/139875)
openly	0.0% (20/139875)
shouted	0.0% (60/139875)
unhelpful	0.0% (50/139875)
disrespect	0.0% (19/139875)
rudeness	0.0% (54/139875)
rudely	0.1% (128/139875)
insisting	0.0% (24/139875)
speech	0.0% (18/139875)
woman	0.3% (448/139875)
ignorant	0.0% (35/139875)
jokes	0.0% (55/139875)
respectful	0.0% (57/139875)
haired	0.0% (28/139875)
tone	0.1% (116/139875)
arrogant	0.1% (98/139875)
quit	0.0% (52/139875)
blond	0.0% (33/139875)
pushy	0.1% (94/139875)
behaviour	0.0% (63/139875)
acted	0.1% (179/139875)
attitudes	0.0% (49/139875)
interrupting	0.0% (33/139875)
attitude	0.5% (717/139875)
insulting	0.0% (32/139875)
unprofessional	0.1% (101/139875)
goodbye	0.0% (61/139875)
answers	0.0% (51/139875)
humor	0.0% (35/139875)
curt	0.0% (22/139875)
task	0.0% (49/139875)
blonde	0.1% (108/139875)
inconveniencing	0.0% (27/139875)
obnoxious	0.0% (51/139875)
tipping	0.0% (60/139875)

Keyword Group 5
Coverage: 1.8% (2464/139875)
Accuracy: 86% Annotated: 7/7
Test Set 1 Prec.: 63.6% Recall:2.4%
Test Set 2 Prec.:100.0% Recall: 6.1%

-keyword-	-coverage-
chefs	0.4% (561/139875)
bartenders	0.2% (230/139875)
bartender	0.4% (564/139875)
hostess	0.4% (519/139875)
employee	0.5% (668/139875)

Keyword Group 6
Coverage: 0.6% (846/139875)
Accuracy: 80% Annotated: 5/5
Test Set 1 Prec.: 80.0% Recall:1.4%
Test Set 2 Prec.: . % Recall: . %

-keyword-	-coverage-
impeccable	0.2% (277/139875)
unobtrusive	0.0% (62/139875)
talkative	0.0% (24/139875)
buzzy	0.1% (204/139875)
chatty	0.1% (110/139875)
formal	0.1% (176/139875)

Keyword Group 7
Coverage: 1.0% (1450/139875)
Accuracy: 100% Annotated: 7/7
Test Set 1 Prec.: 80.0% Recall:1.4%
Test Set 2 Prec.:100.0% Recall: 2.0%

-keyword-	-coverage-
smile	0.6% (836/139875)
smiling	0.2% (332/139875)
hospitality	0.2% (216/139875)
hospitable	0.1% (85/139875)

Keyword Group 8
Coverage: 3.7% (5169/139875)
Accuracy: 100% Annotated: 10/10
Test Set 1 Prec.: 68.0% Recall:5.8%
Test Set 2 Prec.: 90.9% Recall:20.4%

-keyword-	-coverage-
cashier	0.8% (1155/139875)
server	2.1% (2983/139875)
servers	0.9% (1296/139875)

Hospitality Label – User B

	Test Set 1	Test Set 2
True Pos.	43.2% (262/607)	38.0% (38/100)
False Neg.	4.9% (30/607)	11.0% (11/100)
False Pos.	12.9% (78/607)	7.0% (7/100)
True Neg.	39.0% (237/607)	44.0% (44/100)
Accuracy	82.2%	82.0%
Precision	77.1%	84.4%
Recall	89.7%	77.6%
F1	82.9%	80.9%
Gold Cov.	48.3% (293/607)	49.0% (49/100)
Pred. Cov.	56.0% (340/607)	45.0% (45/100)
Training Set Cov.	42.1% (58837/139875)	

Keyword Group 1
Coverage:21.9% (30662/139875)
Accuracy: 100% Annotated: 10/10
Test Set 1 Prec.: 78.1% Recall:30.5%
Test Set 2 Prec.: 72.7% Recall:16.3%

-keyword-	-coverage-
friendly	11.4% (15911/139875)
staff	13.3% (18637/139875)
unfriendly	0.1% (175/139875)
rude	0.9% (1289/139875)

Keyword Group 2
Coverage: 4.4% (6166/139875)
Accuracy: 90% Annotated: 10/10
Test Set 1 Prec.: 75.0% Recall:7.2%
Test Set 2 Prec.:100.0% Recall: 2.0%

-keyword-	-coverage-
smile	0.6% (836/139875)
customer	2.6% (3602/139875)
personnel	0.1% (101/139875)
racist	0.0% (15/139875)
employees	1.4% (1904/139875)

Keyword Group 3
Coverage:20.1% (28106/139875)
Accuracy: 100% Annotated: 10/10
Test Set 1 Prec.: 78.2% Recall:35.6%
Test Set 2 Prec.: 94.1% Recall:32.7%

-keyword-	-coverage-
service	20.1% (28106/139875)

Keyword Group 4
Coverage: 5.7% (7914/139875)
Accuracy: 100% Annotated: 10/10
Test Set 1 Prec.: 54.8% Recall:7.9%
Test Set 2 Prec.: 83.3% Recall:10.2%

-keyword-	-coverage-
waitress	2.7% (3740/139875)
waiter	3.1% (4382/139875)
ethic	0.0% (7/139875)

Keyword Group 5
Coverage: 0.2% (250/139875)
Accuracy: 100% Annotated: 1/1
Test Set 1 Prec.: 100.0% Recall:0.3%
Test Set 2 Prec.: . % Recall: . %

-keyword-	-coverage-
friendliness	0.2% (250/139875)

Keyword Group 6
Coverage: 0.4% (561/139875)
Accuracy: 100% Annotated: 3/3
Test Set 1 Prec.: 50.0% Recall:0.3%
Test Set 2 Prec.:100.0% Recall: 2.0%

-keyword-	-coverage-
hospitable	0.1% (85/139875)
bartenders	0.2% (230/139875)
ignorant	0.0% (35/139875)
hospitality	0.2% (216/139875)

Keyword Group 7
Coverage: 1.4% (1933/139875)
Accuracy: 100% Annotated: 7/7
Test Set 1 Prec.: 60.0% Recall:2.1%
Test Set 2 Prec.: . % Recall: . %

-keyword-	-coverage-
rudest	0.0% (35/139875)
humor	0.0% (35/139875)
costumer	0.0% (52/139875)
talkative	0.0% (24/139875)
attitudes	0.0% (49/139875)
guidance	0.0% (25/139875)
interaction	0.1% (89/139875)
rudeness	0.0% (54/139875)
industry	0.1% (150/139875)
manners	0.0% (41/139875)
Kayla	0.0% (12/139875)
encountered	0.1% (148/139875)
speech	0.0% (18/139875)
costumers	0.0% (19/139875)
treating	0.0% (59/139875)
RUDE	0.0% (17/139875)
behavior	0.0% (46/139875)
hi	0.0% (55/139875)
approachable	0.0% (36/139875)
language	0.1% (143/139875)

```
Michelle            0.0% (35/139875)
dealing             0.1% (113/139875)
hearted             0.0% (22/139875)
disrespect          0.0% (19/139875)
owed                0.0% (17/139875)
appreciative        0.0% (41/139875)
tending             0.0% (25/139875)
aggressive          0.1% (82/139875)
bye                 0.0% (39/139875)
teenagers           0.0% (35/139875)
quit                0.0% (52/139875)
yelled              0.1% (71/139875)
openly              0.0% (20/139875)
leader              0.0% (30/139875)
kindness            0.0% (38/139875)
Ms.                 0.0% (23/139875)
enthusiasm          0.1% (74/139875)
jokes               0.0% (55/139875)
angry               0.1% (148/139875)
```

Hospitality Label – User C

```
|-----------------------------------------|
|          |  Test Set 1   |  Test Set 2  |
|----------+---------------+--------------|
| True Pos.|45.6% (277/607)|45.0% (45/100)|
|False Neg.| 2.5% ( 15/607)| 4.0% ( 4/100)|
|False Pos.|20.1% (122/607)| 9.0% ( 9/100)|
| True Neg.|31.8% (193/607)|42.0% (42/100)|
|----------+---------------+--------------|
|  Accuracy|     77.4%     |     87.0%    |
| Precision|     69.4%     |     83.3%    |
|    Recall|     94.9%     |     91.8%    |
|        F1|     80.2%     |     87.4%    |
|-----------------------------------------|
| Gold Cov.|48.3% (293/607)|49.0% (49/100)|
|Pred. Cov.|65.7% (399/607)|54.0% (54/100)|
|-----------------------------------------|
| Training Set Cov.| 51.5% (71971/139875) |
|-----------------------------------------|

|-----------------------------------------|
| Disabled |  Test Set 1   |  Test Set 2  |
|----------+---------------+--------------|
| True Pos.|45.6% (277/607)|45.0% (45/100)|
|False Neg.| 2.5% ( 15/607)| 4.0% ( 4/100)|
|False Pos.|20.1% (122/607)| 9.0% ( 9/100)|
| True Neg.|31.8% (193/607)|42.0% (42/100)|
|----------+---------------+--------------|
|  Accuracy|     77.4%     |     87.0%    |
| Precision|     69.4%     |     83.3%    |
|    Recall|     94.9%     |     91.8%    |
|        F1|     80.2%     |     87.4%    |
|-----------------------------------------|
| Gold Cov.|48.3% (293/607)|49.0% (49/100)|
|Pred. Cov.|65.7% (399/607)|54.0% (54/100)|
|-----------------------------------------|
| Training Set Cov.| 46.9% (65630/139875) |
|-----------------------------------------|

Keyword Group 1
Coverage: 13.3% (18637/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.: 75.0% Recall: 17.5%
Test Set 2 Prec.: 66.7% Recall:  8.2%
----------------------------------------
-keyword-           -coverage-
staff              13.3% (18637/139875)
----------------------------------------

Keyword Group 2
Coverage: 6.8% (9485/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.: 55.1% Recall:  9.2%
Test Set 2 Prec.: 87.5% Recall:  8.2%
----------------------------------------
-keyword-           -coverage-
waiter              3.1% (4382/139875)
waitress            2.7% (3740/139875)
waiters             1.1% (1548/139875)
waitresses          0.4% (541/139875)
----------------------------------------

Keyword Group 3
Coverage: 1.8% (2518/139875)
Accuracy: NaN   Annotated:  0/7
Test Set 1 Prec.:  57.1% Recall:  1.4%
Test Set 2 Prec.: 100.0% Recall:  2.0%
----------------------------------------
-keyword-           -coverage-
confronted          0.0% (32/139875)
aggressive          0.1% (82/139875)
rude                0.9% (1289/139875)
rudeness            0.0% (54/139875)
behaviour           0.0% (63/139875)
aggressively        0.0% (24/139875)
rudely              0.1% (128/139875)
rudest              0.0% (35/139875)
insulted            0.0% (35/139875)
disrespectful       0.0% (30/139875)
shouted             0.0% (60/139875)
behavior            0.0% (46/139875)
unhelpful           0.0% (50/139875)
unprofessional      0.1% (101/139875)
demanded            0.0% (60/139875)
dismissive          0.0% (39/139875)
arguing             0.0% (63/139875)
demanding           0.0% (43/139875)
shouting            0.1% (107/139875)
yelled              0.1% (71/139875)
stormed             0.0% (23/139875)
argument            0.0% (52/139875)
arrogant            0.1% (98/139875)
waving              0.0% (29/139875)
bothering           0.1% (81/139875)
insulting           0.0% (32/139875)
loudly              0.1% (116/139875)
racist              0.0% (15/139875)
threatened          0.0% (22/139875)
Rude                0.0% (59/139875)
disrespect          0.0% (19/139875)
----------------------------------------

Keyword Group 4
Coverage: 20.1% (28106/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.:  78.2% Recall: 35.6%
Test Set 2 Prec.:  94.1% Recall: 32.7%
----------------------------------------
-keyword-           -coverage-
service            20.1% (28106/139875)
----------------------------------------

Keyword Group 5
Coverage: 11.4% (15911/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.:  82.8% Recall: 18.2%
Test Set 2 Prec.:  83.3% Recall: 10.2%
----------------------------------------
-keyword-           -coverage-
friendly           11.4% (15911/139875)
----------------------------------------

Keyword Group 6
Coverage: 4.0% (5615/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.:  59.1% Recall: 4.5%
Test Set 2 Prec.:  90.0% Recall: 18.4%
----------------------------------------
-keyword-           -coverage-
manager             2.0% (2740/139875)
barista             0.0% (63/139875)
server              2.1% (2983/139875)
assistant           0.0% (58/139875)
----------------------------------------

Keyword Group 7
Coverage: 1.9% (2717/139875)
Accuracy:  71%  Annotated:  7/7
Test Set 1 Prec.:  71.4% Recall: 3.4%
Test Set 2 Prec.:   Nan  Recall: 0.0%
----------------------------------------
-keyword-           -coverage-
apologized          0.2% (245/139875)
annoyed             0.2% (261/139875)
inform              0.1% (132/139875)
remade              0.0% (56/139875)
thanked             0.1% (116/139875)
shrug               0.0% (16/139875)
apologize           0.1% (187/139875)
frustration         0.0% (35/139875)
apology             0.2% (252/139875)
insisted            0.1% (207/139875)
remake              0.1% (72/139875)
defensive           0.0% (19/139875)
inquired            0.1% (76/139875)
acknowledged        0.1% (134/139875)
apologizing         0.0% (36/139875)
flagged             0.0% (42/139875)
refunded            0.0% (49/139875)
fixing              0.0% (42/139875)
declined            0.1% (100/139875)
apologetic          0.1% (162/139875)
angry               0.1% (148/139875)
repeatedly          0.1% (87/139875)
ignored             0.2% (317/139875)
responded           0.1% (121/139875)
repeated            0.1% (117/139875)
----------------------------------------

Keyword Group 8
Coverage: 1.3% (1797/139875)
Accuracy: 100%  Annotated:  7/7
Test Set 1 Prec.:  77.8% Recall: 2.4%
Test Set 2 Prec.: 100.0% Recall: 2.0%
----------------------------------------
-keyword-           -coverage-
team                1.3% (1797/139875)
----------------------------------------

Keyword Group 9
Coverage: 7.8% (10906/139875)
Accuracy: NaN   Annotated:  0/10
Test Set 1 Prec.:  76.6% Recall: 12.3%
Test Set 2 Prec.: 100.0% Recall: 12.2%
----------------------------------------
-keyword-           -coverage-
polite              0.9% (1231/139875)
helpful             2.0% (2859/139875)
knowledgeable       0.6% (871/139875)
nervous             0.1% (78/139875)
attentive           2.7% (3760/139875)
knowledgable        0.2% (240/139875)
informative         0.1% (149/139875)
courteous           0.4% (539/139875)
sommelier           0.2% (308/139875)
guided              0.1% (85/139875)
recommendations     0.3% (435/139875)
overbearing         0.0% (60/139875)
greetings           0.0% (55/139875)
explaining          0.2% (237/139875)
outgoing            0.0% (26/139875)
Cashier             0.1% (164/139875)
chatty              0.1% (110/139875)
chat                0.3% (415/139875)
hospitable          0.1% (85/139875)
smiling             0.2% (332/139875)
----------------------------------------

Keyword Group 10
Coverage: 3.8% (5324/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.:  80.0% Recall: 6.8%
Test Set 2 Prec.: 100.0% Recall: 6.1%
----------------------------------------
-keyword-           -coverage-
prompt              0.4% (583/139875)
greeted             1.4% (1912/139875)
professional        0.7% (996/139875)
warmly              0.2% (268/139875)
manner              0.2% (243/139875)
greeting            0.4% (504/139875)
greets              0.1% (106/139875)
promptly            0.4% (568/139875)
Greeted             0.1% (100/139875)
timely              0.3% (369/139875)
enthusiastic        0.1% (207/139875)
----------------------------------------

Keyword Group 11
Coverage: 1.3% (1757/139875)
Accuracy:  86%  Annotated:  7/7
Test Set 1 Prec.:  40.0% Recall: 0.7%
Test Set 2 Prec.:   0.0% Recall: 0.0%
----------------------------------------
-keyword-           -coverage-
thank               1.3% (1757/139875)
----------------------------------------

Keyword Group 12
Coverage: 0.4% (564/139875)
Accuracy: NaN   Annotated:  0/3
Test Set 1 Prec.:  0.0% Recall: 0.0%
Test Set 2 Prec.:100.0% Recall: 2.0%
----------------------------------------
-keyword-           -coverage-
bartender           0.4% (564/139875)
----------------------------------------

Keyword Group 13
Coverage: 0.7% (993/139875)
Accuracy: 100%  Annotated:  5/5
```

```
Test Set 1 Prec.:  33.3% Recall: 0.3%
Test Set 2 Prec.: 100.0% Recall: 2.0%
---------------------------------------
```

```
-keyword-          -coverage-
tip                0.7% (951/139875)
tipping            0.0% (60/139875)
```

```
---------------------------------------
```

Hospitality Label – User D

```
|-----------------------------------------|
|          |  Test Set 1   |  Test Set 2  |
|----------+---------------+--------------|
| True Pos.|45.0% (273/607)|46.0% (46/100)|
|False Neg.| 3.1% ( 19/607)| 3.0% ( 3/100)|
|False Pos.|23.4% (142/607)|19.0% (19/100)|
| True Neg.|28.5% (173/607)|32.0% (32/100)|
|----------+---------------+--------------|
|  Accuracy|     73.5%     |     78.0%    |
| Precision|     65.8%     |     70.8%    |
|    Recall|     93.5%     |     93.9%    |
|        F1|     77.2%     |     80.7%    |
|-----------------------------------------|
| Gold Cov.|48.3% (293/607)|49.0% (49/100)|
|Pred. Cov.|68.4% (415/607)|65.0% (65/100)|
|-----------------------------------------|
| Training Set Cov.| 53.4% (74719/139875) |
|-----------------------------------------|

|-----------------------------------------|
| Disabled |  Test Set 1   |  Test Set 2  |
|----------+---------------+--------------|
| True Pos.|32.8% (199/607)|37.0% (37/100)|
|False Neg.|15.3% ( 93/607)|12.0% (12/100)|
|False Pos.|16.5% (100/607)|16.0% (16/100)|
| True Neg.|35.4% (215/607)|35.0% (35/100)|
|----------+---------------+--------------|
|  Accuracy|     68.2%     |     72.0%    |
| Precision|     66.6%     |     69.8%    |
|    Recall|     68.2%     |     75.5%    |
|        F1|     67.3%     |     72.5%    |
|-----------------------------------------|
| Gold Cov.|48.3% (293/607)|49.0% (49/100)|
|Pred. Cov.|49.3% (299/607)|53.0% (53/100)|
|-----------------------------------------|
| Training Set Cov.| 39.4% (55135/139875) |
|-----------------------------------------|

|----DISABLED + SERVICE ADDED-------------|
| Disabled |  Test Set 1   |  Test Set 2  |
|----------+---------------+--------------|
| True Pos.|44.3% (199/607)|46.0% (46/100)|
|False Neg.| 3.8% ( 93/607)| 3.0% ( 3/100)|
|False Pos.|19.3% (100/607)|19.0% (19/100)|
| True Neg.|32.6% (215/607)|32.0% (32/100)|
|----------+---------------+--------------|
|  Accuracy|     76.9%     |     80.0%    |
| Precision|     69.7%     |     73.0%    |
|    Recall|     92.1%     |     93.9%    |
|        F1|     79.4%     |     82.1%    |
|-----------------------------------------|
| Gold Cov.|48.3% (293/607)|49.0% (49/100)|
|Pred. Cov.|63.6% (386/607)|63.0% (65/100)|
|-----------------------------------------|
| Training Set Cov.| 49.4% (74719/139875) |
|-----------------------------------------|

Keyword Group 1
Coverage:13.3% (18637/139875)
Accuracy: 100%  Annotated: 10/10
Test Set 1 Prec.: 75.0%  Recall:17.5%
Test Set 2 Prec.: 66.7%  Recall: 8.2%
---------------------------------------
-keyword-              -coverage-
staff             13.3% (18637/139875)

Keyword Group 2
Coverage: 2.6% (3602/139875)
Accuracy:  90%  Annotated: 10/10
Test Set 1 Prec.: 68.4%  Recall:4.5%
Test Set 2 Prec.:100.0%  Recall: 2.0%
---------------------------------------
-keyword-              -coverage-
customer          2.6% (3602/139875)

Keyword Group 3
Coverage: 0.9% (1289/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 100.0% Recall:0.7%
Test Set 2 Prec.:100.0%  Recall: 2.0%
---------------------------------------
-keyword-              -coverage-
rude              0.9% (1289/139875)

Keyword Group 4 - Disable
Coverage: 1.8% (2466/139875)
Accuracy: 100%  Annotated:  7/7
Test Set 1 Prec.: 46.2%  Recall:2.1%
Test Set 2 Prec.: 50.0%  Recall: 2.0%
---------------------------------------
-keyword-              -coverage-
pleasant          1.8% (2466/139875)

Keyword Group 5
Coverage: 0.8% (1108/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.: 66.7%  Recall:0.7%
Test Set 2 Prec.:   . %  Recall:  . %
---------------------------------------
-keyword-              -coverage-
horrendous        0.0% (36/139875)
ass               0.1% (132/139875)
atrocious         0.0% (19/139875)
disinterested     0.1% (87/139875)
vile              0.0% (14/139875)
arrogant          0.1% (98/139875)
incompetent       0.0% (39/139875)
dismissive        0.0% (39/139875)
unhelpful         0.0% (50/139875)
clueless          0.0% (42/139875)
Terrible          0.1% (150/139875)
rudest            0.0% (35/139875)
appalling         0.1% (76/139875)
Avoid             0.1% (188/139875)
insulting         0.0% (32/139875)
insult            0.1% (85/139875)
AVOID             0.0% (30/139875)

Keyword Group 6
Coverage: 0.3% (458/139875)
Accuracy: 100%  Annotated:  3/3
Test Set 1 Prec.: 100.0% Recall:0.3%
Test Set 2 Prec.:   . %  Recall:  . %
---------------------------------------
-keyword-              -coverage-
inept             0.0% (21/139875)
behavior          0.0% (46/139875)
idiot             0.0% (23/139875)
disrespectful     0.0% (30/139875)
rudeness          0.0% (54/139875)
unprofessional    0.1% (101/139875)
rudely            0.1% (128/139875)
speech            0.0% (18/139875)
behaviour         0.0% (63/139875)

Keyword Group 7
Coverage:18.4% (25774/139875)
Accuracy: NAN   annotated:  0/10
Test Set 1 Prec.: 56.0%  Recall:20.9%
Test Set 2 Prec.: 77.8%  Recall:28.6%
---------------------------------------
-keyword-              -coverage-
helpful           2.0% (2859/139875)
knowledge         0.2% (342/139875)
manager           2.0% (2740/139875)
waiter            3.1% (4382/139875)
waitress          2.7% (3740/139875)
asked             4.6% (6498/139875)
him               1.3% (1766/139875)
job               0.7% (1040/139875)
She               1.8% (2555/139875)
he                4.6% (6374/139875)
company           0.3% (478/139875)
owner             1.2% (1692/139875)
store             1.2% (1609/139875)
professional      0.7% (996/139875)
knowledgable      0.2% (240/139875)

Keyword Group 8
Coverage:17.5% (24456/139875)
Accuracy: NAN   annotated:  0/10
Test Set 1 Prec.: 82.1%  Recall:29.8%
Test Set 2 Prec.: 90.0%  Recall:18.4%
---------------------------------------
-keyword-              -coverage-
friendly         11.4% (15911/139875)
attentive         2.7% (3760/139875)
welcoming         1.7% (2348/139875)
attitude          0.5% (717/139875)
relaxed           0.8% (1098/139875)
presented         1.0% (1463/139875)
patronizing       0.0% (19/139875)
demanding         0.0% (43/139875)
demeanor          0.0% (22/139875)
polite            0.9% (1231/139875)
cared             0.1% (128/139875)
smiling           0.2% (332/139875)
engaging          0.1% (105/139875)

Keyword Group 9
Coverage: 0.5% (708/139875)
Accuracy: 100%  Annotated:  5/5
Test Set 1 Prec.:   NaN  Recall:0.0%
Test Set 2 Prec.:100.0%  Recall: 2.0%
---------------------------------------
-keyword-              -coverage-
Horrible          0.1% (126/139875)
unfriendly        0.1% (175/139875)
Bad               0.1% (159/139875)
Worst             0.1% (196/139875)
disgusted         0.0% (67/139875)

Keyword Group 10
Coverage: 1.3% (1825/139875)
Accuracy: 100%  Annotated:  7/7
Test Set 1 Prec.: 77.8%  Recall:2.4%
Test Set 2 Prec.:100.0%  Recall: 2.0%
---------------------------------------
-keyword-              -coverage-
bouncers          0.0% (28/139875)
team              1.3% (1797/139875)

Keyword Group 11 - Disable
Coverage:28.1% (39290/139875)
Accuracy: NAN   annotated:  0/10
Test Set 1 Prec.: 66.3%  Recall:43.8%
Test Set 2 Prec.: 69.0%  Recall:40.8%
---------------------------------------
-keyword-              -coverage-
happy             3.7% (5178/139875)
awesome           1.3% (1879/139875)
accommodate       0.3% (369/139875)
piss              0.0% (25/139875)
service          20.1% (28106/139875)
ignore            0.1% (125/139875)
experience        6.4% (8887/139875)
crappy            0.1% (146/139875)
horrible          0.4% (495/139875)
behavior          0.0% (0/139875)
employ            0.0% (20/139875)
jobs              0.1% (71/139875)


```