Interactive Machine Teaching by Labeling Rules and Instances

Giannis Karamanolakis[*]
Amazon AGI
New York, NY 10001, USA
karamai@amazon.com

Daniel Hsu
Columbia University
New York, NY 10027, USA
djhsu@cs.columbia.edu

Luis Gravaano
Columbia University
New York, NY 10027, USA
gravano@cs.columbia.edu

Abstract

Weakly supervised learning aims to reduce the cost of labeling data by using expert-designed labeling rules. However, existing methods require experts to design effective rules in a single shot, which is difficult in the absence of proper guidance and tooling. Therefore, it is still an open question whether experts should spend their limited time writing rules or instead providing instance labels via active learning. In this paper, we investigate how to exploit an expert's limited time to create effective supervision. First, to develop practical guidelines for rule creation, we conduct an exploratory analysis of diverse collections of existing expert-designed rules and find that rule precision is more important than coverage across datasets. Second, we compare rule creation to individual instance labeling via active learning and demonstrate the importance of both across 6 datasets. Third, we propose an interactive learning framework, INTERVAL, that achieves efficiency by automatically extracting candidate rules based on rich patterns (e.g., by prompting a language model), and effectiveness by soliciting expert feedback on both candidate rules and individual instances. Across 6 datasets, INTERVAL outperforms state-of-the-art weakly supervised approaches by 7% in F1. Furthermore, it requires as few as 10 queries for expert feedback to reach F1 values that existing active learning methods cannot match even with 100 queries.

1 Introduction

Supervised machine learning models for text classification require large, hand-labeled training datasets, which are both expensive and time-consuming to obtain. Most efforts to reduce the reliance on large training datasets support just a single type of expert supervision, namely, to label individual instances one at a time (Seeger, 2006; Clark et al., 2018; Ruder and Plank, 2018; Berthelot et al., 2019; Peters et al., 2018; Devlin et al., 2019; Zhang and Yang, 2021; Zhang et al., 2022c).

To reduce the data labeling bottleneck, weakly supervised learning (WSL) (Zhang et al., 2022a) focuses on labeling rules that automatically generate weak labels for unlabeled instances. WSL works in two separate steps: (i) experts provide labeling rules; and (ii) labeling rules are used to train a machine learning model. Most work focuses on solving the second step and learn with noisy rules (Ratner et al., 2016, 2017; Karamanolakis et al., 2019; Bach et al., 2019; Awasthi et al., 2020). In practice, however, experts find it difficult to define sufficiently many rules in one shot (Varma and Ré, 2018). Considerable time and creativity are required for inspecting unlabeled instances and creating rules that add predictive value by effectively covering a substantial number of instances. Therefore, it is an open question whether experts should spend their limited time writing rules or instead providing instance labels, notably via active learning (Settles, 2009).

In this paper, we investigate how to efficiently exploit an expert's limited time for machine teaching. Our main idea is to *automatically* extract labeling rules with high coverage of unlabeled data, and then rely on domain expertise to validate the candidate rules. In contrast to active learning methods, where the machine queries the expert for labels of individual examples (Zhang et al., 2022c), providing feedback for each rule leads to multiple data labels, which we show here can boost classification performance faster.

Supporting rich forms of interaction is challenging, especially when the teaching budget is limited. First, given a restricted number of rules that can be created or validated by an expert, it is not clear what properties these rules should have to train an accurate model. For example, should one prioritize rules that cover many examples but with relatively low precision, or rules that have high

[*]Work done at Columbia prior to joining Amazon.

Transactions of the Association for Computational Linguistics, vol. 12, pp. 1441–1459, 2024. https://doi.org/10.1162/tacl_a_00707
Action Editor: Asli Celikyilmaz. Submission batch: 11/2023; Revision batch: 3/2024; Published 11/2024.

precision but lower coverage? Moreover, existing algorithms for rule extraction require substantial labeled data, and it is unclear how to extract and rank candidate rules when we are given just limited labeled data and perhaps a few expert-validated rules. In general, there are few guidelines in the literature for creating effective rules for efficient machine teaching. Additionally, the option to ask for feedback on both rules and instances requires balancing the costs and potential benefits of each type of feedback when there is a shared budget of expert interaction.

Our work addresses these open questions via the following contributions:

Characterization of Prevalent Patterns in Offline Machine Teaching. We analyze six datasets with expert-defined rules and evaluate multiple weak supervision methods under simulated low-resource settings. Specifically, we unify several weak supervision methods using a Teacher-Student abstraction, where a subset of the rules are considered in the teacher model for training a student model. By evaluating more than 1,000 Teacher-Student configurations per dataset, we associate Teacher properties with the Student's performance and, even though rules are dataset-specific, we find two prevalent patterns across datasets and methods that could inform guidelines for rule creation. First, we show that a higher-F1 Teacher does not necessarily lead to a higher-F1 Student. Second, we show the Teacher's precision is more important than coverage for training an accurate Student.

Automatic Rule Extraction via Prompting. We propose a method that extracts rules with rich predicates, expressed as conjunctions of n-grams, syntactic features, and prompt-based features. By prompting a pre-trained model (see Figure 1), our method extracts high-level features that might not explicitly appear in the text (e.g., ''terrible'' customer experience) and thus can discover common patterns across instances with no n-gram overlap. As we will show, by extracting both surface-level and higher-level features, our rule family achieves higher precision and coverage than n-gram rules. Our design focuses on rules that could be easily validated by a human and are highly effective.

Interactive Machine Teaching. We present a human-in-the-loop machine teaching framework

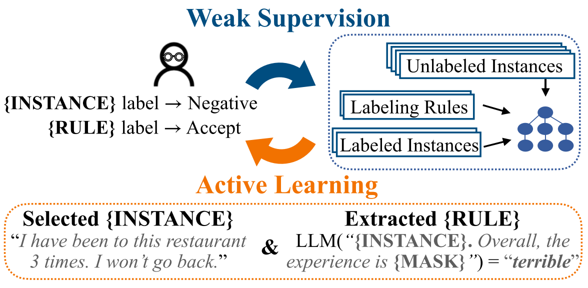

Weak Supervision
{INSTANCE} label → Negative
{RULE} label → Accept
Unlabeled Instances
Labeling Rules
Labeled Instances
Active Learning
Selected {INSTANCE}
"I have been to this restaurant 3 times. I won't go back."
&
Extracted {RULE}
LLM("{INSTANCE}. Overall, the experience is {MASK}") = "terrible"

Figure 1: Our INTERVAL framework supports interaction on both instances and automatically extracted rules (e.g., by prompting a large language model) for weakly supervised learning.

called INTERVAL,[1] which queries for expert feedback on both instances and rules, and uses all the available resources to train a classifier. We quantify the trade-off between labeling rules vs. instances and show that our framework is more efficient than existing WSL and active learning approaches even when starting with no expert-written rules. Our analysis demonstrates that feedback on both rules and instances is more effective than feedback on instances only (as in Active Learning) even when labeling rules are more expensive than labeling instances by up to 9 times.

The rest of this paper is organized as follows. Section 2 reviews related work on interactive machine teaching and defines our problem of focus. Section 3 presents our interactive machine teaching framework,[2] which queries for feedback on labeling rules and instance. Sections 4 and 5 evaluate our interactive method via experiments on six text classification datasets. Finally, Sections 6 and 7 conclude and suggest future work.

2 Problem Definition and Related Work

We now define our problem of focus (Section 2.1); we also discuss related work on non-interactive weak supervision and interactive learning with instance- and feature-level feedback (Section 2.2).

2.1 Problem Definition

Let $\mathscr{X}$ denote the feature space and $\mathscr{Y} = \{1, \ldots, K\}$ denote the label space for a K-class classification task. We consider a set of manually labeled examples $D_L = \{(s_l, y_l)\}$, where $s_l \in \mathscr{X}$

[1]INTERVAL: INTEractive Rule discoVery for weAkly supervised Learning.

[2]Our implementation is publicly available at `https://github.com/gkaramanolakis/interval`.

and $y_l \in \mathscr{Y}$, and a set of unlabeled examples $D_U = \{s_i\}$. We also consider a set of pre-defined expert-provided labeling rules $R = \{r^j\}$. A rule $r^j : \mathscr{X} \rightarrow \mathscr{Y} \cup \{\perp\}$ maps an example s_i into a label $z_i^j \in \mathscr{Y} \cup \{\perp\}$. Predicting $z_i^j = \perp$ indicates that r^j does not cover s_i. We are primarily interested in the scenario where the size of D_L is small in comparison to that of D_U, and where R contains just a few or no expert-provided rules, which is often the case for new tasks. Additionally, we assume that we have a budget of T ''cost'' units (e.g., time) for querying a subject matter expert for feedback on either an instance $s_i \in D_U$ (at a cost of T_I) or an automatically extracted rule r^j (at a cost of T_R), as we discuss in Section 2.2.

Our goal is to leverage D_L, D_U, and R, and interact with the expert within the specified budget T to train a classifier that, given an unseen test instance $s' \in \mathscr{X}$, predicts a label $y' \in \mathscr{Y}$.

2.2 Prior Work

Non-interactive Approaches. Non-interactive weak supervision approaches do not involve a human in the loop (i.e., $T = 0$ for our problem definition). Supervised learning methods consider just D_L, semi-supervised learning methods consider D_L and D_U (Nigam and Ghani, 2000; Lee, 2013; Gera et al., 2022), and WSL methods consider D_L, D_U, and R (Ratner et al., 2017; Bach et al., 2019; Badene et al., 2019; Fu et al., 2020; Awasthi et al., 2020; Karamanolakis et al., 2021). WSL uses rules in R (e.g., keyword-based patterns, regular expressions, heuristic labeling functions) to automatically generate weak training labels for unlabeled instances in D_U. As rules can be noisy, can have limited coverage, and different rules may generate conflicting labels for the same instance, WSL techniques estimate rule weights for noise-aware training (Zhang et al., 2022a). Our method also employs WSL, can work with any rule-weighting technique and further discovers new rules to expand the coverage of R.

Our method is also related to zero-shot and few-shot prompting methods, which use a template to modify the input s_i into a cloze-style or entailment question and leverage a pre-trained model to ''answer'' the question (Schick and Schütze, 2021; Yin et al., 2019; Liu et al., 2023). By directly using the outputs of the pre-trained model for classification, prompt-based techniques are sensitive to the selection of prompting templates (Gao et al., 2021; Ye et al., 2023), labeled examples (Zhao et al., 2021; Perez et al., 2021), and hyperparameters (Tam et al., 2021). Even prompting powerful models such as ChatGPT, the successor of InstructGPT (Ouyang et al., 2022), requires work to reach the performance of supervised (fine-tuned) models on text benchmarks (Bang et al., 2022). Our work explores prompting for rule creation during training instead of direct inference. Specifically, we use the pre-trained model's output to construct labeling rules, which we assume are only weakly indicative of the true labels. Through our approach prompting is required just for training and can work with any model for inference, thus enabling applications where deploying large language models might not be possible.

Our work is also related to rule extraction methods, which consider rules of various types such as keywords, named entities, and numeric expressions (Yangarber et al., 2000), synthetic relations (Snow et al., 2004), part-of-speech tags and hypernyms (Califf and Mooney, 2003), regular expression patterns (Augenstein et al., 2016), sequential patterns (Srikant and Agrawal, 1996; Jindal and Liu, 2008), and more recently, features extracted by prompting pre-trained models (Zhang et al., 2022b). Our method considers a rich family of rules based on n-grams, linguistic features (e.g., part of speech tags and named entities), and prompt-based features and focuses on efficient interaction by soliciting feedback on both candidate rules and instances.

Interactive Learning with Instance Feedback. One type of interaction that has been studied extensively in the literature is active learning, in which the machine queries the expert for just a small number of labels for examples that are chosen adaptively from abundant unlabeled data (Lewis and Gale, 1994; Cohn et al., 1996; Roy and McCallum, 2001; Dasgupta et al., 2007; Dasgupta and Hsu, 2008; Settles, 2009; Beygelzimer et al., 2010; Houlsby et al., 2011; Zhang and Chaudhuri, 2015; Shen et al., 2017; Kirsch et al., 2019; Ash et al., 2019; Brantley et al., 2020; Yuan et al., 2020; Dor et al., 2020; Margatina et al., 2021; Zhang et al., 2022c). Nearly all previous active learning methods solicit the expert's judgment to just label instances. In other words, they do not support feedback on labeling rules (i.e., $T_R = \infty$) and query for feedback on $\lfloor \frac{T}{T_I} \rfloor$ instance labels. Creating a sufficiently large training set would require separate feedback on many individual instances. On the other hand, validating a candidate *rule* leads to weak labels for *many* examples at a time (i.e., for all the examples covered by the rule) and, as a result, a large weakly-labeled dataset can be created with a relatively small number of rules.

Interactive Learning with Rule Feedback. Our work is related to previous interactive methods that support expert queries on automatically generated rules from the n-gram family (Druck et al., 2008; Melville et al., 2009; Settles, 2011; Jagarlamudi et al., 2012; Poulis and Dasgupta, 2017; Dasgupta et al., 2018; Boecking et al., 2020; Kartchner et al., 2022). These methods extract simple n-gram based rules, which as we will show (e.g., in Figure 3) have limited effectiveness and different characteristics than expert-provided rules in R. As two exceptions, Sen et al. (2019) extract rules based on linguistic expressions via syntactic parsing and Zhang et al. (2022b) consider rules based on the output of pre-trained language models prompted with task-specific templates; both show that experts can successfully provide feedback on rules from the proposed families. Most of the above methods do not allow instance-labeling queries (i.e., these methods assume that $T_I = \infty$). In contrast, our method subsumes and generalizes existing work on rule labeling and active learning by querying an expert for both instances and automatically extracted rules from a new rule family with rich predicates.

3 Interactive Machine Teaching with Instance and Rule Feedback

This section describes our interactive machine teaching framework, which addresses the problem defined in Section 2.1. The core question is how to efficiently solicit expert feedback for machine teaching given a limited budget T. Our main idea is to balance the quality of instance labels with the efficiency of labeling rules under this low-resource setting. We propose a framework, INTERVAL, that supports efficient interaction by selecting which instances to label manually and by extracting candidate rules that, when accepted, can automatically generate many additional labels. INTERVAL can be used with several WSL methods and any learning model.

In the rest of this section, we describe the individual steps followed by INTERVAL on each iteration, namely, Teacher-Student co-training (Section 3.1), querying for instance feedback (Section 3.2), candidate rule extraction (Section 3.3), and querying for rule feedback (Section 3.4), and then we summarize the main ideas of our interactive machine teaching algorithm (Section 3.5).

3.1 Teacher-Student Co-Training

In the first step of each iteration, we use D_L, D_U, and R to train a model. This has been the main objective in non-interactive WSL. Our model training employs the Teacher-Student abstraction by Karamanolakis et al. (2021) to unify several WSL methods (Dawid and Skene, 1979; Ratner et al., 2016, 2019; Zhang et al., 2022a).

The teacher model $q_\phi(\cdot)$ considers D_L, D_U, and R, and predicts labels q_i for all examples $s_i \in D_U$ except for examples covered by no rules in R, which are then not covered by the Teacher either. The student model $p_\theta(\cdot)$ is the base learning model that is trained using D_L, D_U, and the teacher model by approximately solving the following optimization problem:

$$\begin{aligned}\min_\theta \; & \mathbb{E}_{s_l, y_l \in D_L}[-\log\, p_\theta(y_l \mid s_l)] + \\ & \lambda \mathbb{E}_{s \in D_U} \mathbb{E}_{y \sim q_{\phi^*}(y|s)}[-\log\, p_\theta(y \mid s)], \quad (1)\end{aligned}$$

where $\lambda \in \mathbb{R}$ is a hyper-parameter controlling the relative weight of the manually labeled data (first term) and the weakly labeled data (second term).

The same Teacher-Student abstraction appears across different WSL approaches (Zhang et al., 2022a), which differ in the teacher model design. For example, in simple majority voting, the Teacher aggregates the predictions of rules in R. In Snorkel (Ratner et al., 2017), the Teacher is a probabilistic graphical model that estimates weights for rules in R in an unsupervised way. In ASTRA (Karamanolakis et al., 2021), the Teacher is a rule-attention network that aggregates rule labels with instance-specific weights and is co-trained with the Student.

In our problem of focus, where the size of D_L is small and R contains just a small number of rules, the student model might have far less than satisfying accuracy for our target task. Next, we show how to exploit the interaction budget T.

3.2 Querying for Instance Feedback

After having trained the Student, INTERVAL queries the label y_i for an instance s_i from the unlabeled set D_U. To efficiently interact with an expert, we design a method that chooses which instance to query for feedback based on the Student's probabilities, as some instances might be more ''informative'' for the Student than others. INTERVAL identifies a diverse collection of unlabeled instances for which the Student's predicted probabilities have high entropy as explained next.

Instance Clustering. At the beginning of our algorithm, we construct a hierarchical clustering of the unlabeled instances in D_U. To achieve this, we implement agglomerative clustering using Ward's linkage method, which focuses on minimizing cluster variances. For cluster variances, we calculate the Euclidean distances between instances based on instance embeddings, which are computed via pre-trained BERT (Devlin et al., 2019). For implementation details see Section 4.

Instance Selection. To choose which instances to query, INTERVAL applies the Student $p_\theta(\cdot)$ on each unlabeled instance $s_i \in D_U$ to get soft labels $\mathbf{p}_i = (p_i^1, \ldots, p_i^K)$, where p_i^k represents the Student's predicted probability for assigning s_i into the target class $k \in K$. We use $D_{Student} = \{(s_i, \mathbf{p}_i)\}_{s_i \in D_U}$ to define the dataset that is soft-labeled by the Student. Then, INTERVAL selects sample instances s_i via the cluster-adaptive sampling algorithm of Dasgupta and Hsu (2008), which exploits the hierarchical structure of the data and evaluates cluster informativeness based on the entropy of the Student's predicted probabilities for s_i in D_S. Specifically, the algorithm chooses instances s_i from clusters characterized by low label ''purity'', or equivalently, high entropy based on the Student's probabilities $\mathbf{p}_i$. This selection is made under the premise that collecting expert labels for these instances will provide valuable information for the subsequent round of Student training. Once a cluster becomes ''pure,'' then the algorithm shifts its focus to another cluster with the goal to acquire a diverse collection of instances.

Instance Labeling. After selecting an instance s_i, the system queries the expert's label y_i at a cost of T_I. At the end of the iteration, the labeled pair (s_i, y_i) is added in D_L to train the Teacher and Student at the next iteration.

Name	**Prompt Template**
EXPERIENCE	Overall, the experience is [MASK]. [TEXT].
RECOMMEND	[TEXT]. Would I recommend it? The answer is [MASK].
ASKS_FOR	The following SMS message asks for [MASK]: [TEXT].
IS_ABOUT	The following SMS message is about [MASK]: [TEXT].

Table 1: Examples of templates used to prompt pre-trained models in Yelp (top) and SMS (bottom) for candidate rule extraction.

3.3 Candidate Rule Extraction

In contrast to WSL, where experts manually create rules with significant coverage on D_U, we propose to automatically extract candidate rules and hopefully reduce the cost of rule creation. After getting the label y_i for an instance s_i, we extract candidate rules r^j that predict the same label y_i for s_i and have non-trivial coverage in D_U. We first describe the types of rules and then how to extract them.

Rule Family. Most work on interactive learning with rule feedback has focused on extracting keyword-based labeling rules. These rules have limited expressiveness compared to expert-written rules, which include class-indicative keywords, regular expression patterns, and auxiliary classifiers (e.g., polarity and subjectivity classifiers for spam classification) (Zhang et al., 2022c). To improve expressiveness without sacrificing interpretability, our method extracts rules r^j whose predicates $v^j(s_i)$ are conjunctions of features that can have three different types: n-grams ($v^j(s_i)$ is true if a specific n-gram appears in s_i), linguistic features (e.g., part-of-speech tags and named entities), and prompt-based features. Specifically, to construct prompt-based rules, we prompt pre-trained models for s_i using templates from ''PromptSource'' (Bach et al., 2022). As an example, consider the sentence from Figure 1 (s_i: ''*I have been to this restaurant 3 times. I won't go back*''). We construct ''prompt-based'' predicates by prompting a pre-trained model to fill in the mask in the following template: ''<s_i>. *Overall, the experience is* [MASK]'' and extracting the top k tokens (e.g., ''terrible''). Table 1 shows more examples of prompt templates and Table 2 lists examples of rules extracted by our method using such templates (extraction details are discussed later). Our approach extracts common patterns across instances that might not even share any n-gram features, such as in tasks with short documents. As we will see in Section 5.2, the

Candidate Rules (predicate $\rightarrow$ label)
PMT-EXPERIENCE="terrible" $\rightarrow$ Negative
PMT-EXPERIENCE="fantastic"$\rightarrow$ Positive
PMT-RECOMMEND="certainly" $\rightarrow$ Positive
PMT-IS_ABOUT="prizes" $\rightarrow$ Spam
NGRAM="http" **AND** PMT-ASKS_FOR="donations" $\rightarrow$ Spam
NER="CARDINAL" **AND** PMT-ASKS_FOR="information"$\rightarrow$ Spam

Table 2: Examples of rules extracted by our method for Yelp (top) and SMS (bottom). "NGRAM $= a$" means that a appears as an n-gram in the text. "NER $= a$" means that at least one entity of type a exists in the text. "PMT-$b = a$" means that a appears among the top-k tokens predicted by the pre-trained model to fill in the [MASK] token for a prompt template b.

rules in our expanded family can be substantially more accurate than the simple n-gram rules considered in previous work, and yet they are nearly as interpretable. Note that, at test time, our method does not require access to the above resources as the student model predicts labels directly based on s_i.

Rule Extraction. We extract rules r from the above family, as long as (i) they cover at least t_{cov} examples in D_U *including* s_i and (ii) they have a precision of at least t_{prec} in D_L. Both t_{cov} and t_{prec} are hyper-parameters. Given the above coverage and precision constraints, we extract conjunctions of features using the Apriori algorithm (Agrawal et al., 1994). Specifically, we first exhaustively search all rules with a single feature from the above family and keep all rules that satisfy all constraints. (The constraint that all rules have to cover s_i with a label y_i is especially strong and allows efficient search.) Then, we create rules as conjunctions of two features selected before and pick just the resulting rules that satisfy all of the above constraints. Our method considers rules with conjunctions of up to t_{len} features, where t_{len} is another hyper-parameter. The set R_C contains all candidate rules that are extracted by our method and satisfy our constraints.

Automatically identifying a good rule is hard with limited labeled data D_L. For example, a candidate rule r^j with high coverage on D_U might have low coverage in D_L (D_L might contain just a few labeled examples), and therefore it is hard to estimate the true precision of r^j. Therefore, we rely on expert feedback for selected candidate rules from R_C, as discussed next.

Algorithm 1 Interactive Machine Teaching

```
**Input:** Small amount of labeled data $D_L$; task-specific unlabeled data $D_U$; small set of weak rules $R$; budget of $T$ cost units for interaction with a subject matter expert
**Output:** Student $p^*_\theta(\cdot)$, Teacher $q^*_\phi(\cdot)$, augmented labeled data $D'_L$, augmented set of weak rules $R'$
1: Cluster all data $s_i \in D_U$ into hierarchical clusters (agglomerative clustering; Ward's linkage; Euclidean distance of instance embeddings)
2: Initialize $D'_L = D_L$, $R' = R$
3: **Repeat until the budget $T$ runs out:**
    3.1: Train Teacher $q^*_\phi(\cdot)$ and Student $p_\theta(\cdot)$ using $D'_L$, $D_U$, $R'$
    3.2: Apply $p_\theta(\cdot)$ to $s \in D_U$ to obtain soft labels: $D_{Student} = \{(s_i, \mathbf{p}_i)\}_{s_i \in D_U}$
    3.3: Pick a candidate instance $s_i \in D_U$
    3.4: Query the label $y_i$ for $s_i$ (cost = $T_I$)
    3.5: Extract candidate rules $r^j$ that cover $s_i$
    3.6: Query the labels $z^j$ for $\beta_i$ rules $r^j$ (cost = $\beta_i \cdot T_R$)
    3.7: Update $D'_L = D'_L \cup \{(s_i, y_i)\}_{\beta_i}$, $R' = R' \cup \{r^j : (v^j(\cdot), z^j)\}$, $T = T - T_I - \beta_i \cdot T_R$
```

3.4 Querying for Rule Feedback

After having extracted the set of R_C candidate rules that cover s_i, we select up to β candidate rules r^j and query for their labels z^j_i, where β is a hyper-parameter. Specifically, we first select in R'_C all rules from R_C that predict a label $z^j_i = y_i$ (thus agreeing with the expert's label for s_i). Then, we select from R'_C the top β rules with the highest precision (computed on D_L). Note that R'_C might have fewer than β rules in total, thus we use $\beta_i \leq \beta$ to indicate the number of rules selected finally.

Next, we query the labels z^j_i for the β_i selected rules at a cost of $\beta_i \cdot T_R$. At the end of the iteration, the β_i labeled rules, which we denote as $\{(r^j, z^j)\}_{\beta_i}$, are added in R, where by design each rule r^j will predict the same label $z^j = z^j_i$ for all instances that it covers. Our method ignores rules labeled with $z^j_i = \perp$.

Throughout this interaction design, we assume that the domain expert can judge whether r^j provides the correct label for most of the examples that the rule covers, and is aware that (i) a rule r^j does not need to have perfect accuracy but

	YouTube	SMS	IMDB	Yelp	TREC	AGNews
Classification task	spam	spam	sentiment	sentiment	question type	topic
Domain	user comments	text messages	movies	reviews	web queries	news
# Classes (K)	2	2	2	2	6	4
Unlabeled size ($\|D_U\|$)	1546	4531	19,960	30,360	4845	95,920
Labeled train size ($\|D_L\|$)	40	40	40	40	120	80
Test size	250	500	2500	3800	500	12,000
# Prompt templates	5	5	15	12	6	9
# Expert-provided rules (R)	10	73	5	8	68	9

Table 3: Statistics for available datasets with expert-labeled rules.

rather represents a pattern that the expert intends to exploit to label examples more efficiently than manually; (ii) rule predictions will be aggregated to train a model in a noise-aware way. Similar to how expert-written rules are used for WSL, we assume that accepting a precise candidate rule for s_i could improve the Student in the next iteration. This is possible, by augmenting the Student's training data with all the unlabeled examples covered by the rule, and by increasing the overlap of accepted rules R on D_U, which provides useful signal for rule denoising, similar to inter-annotator agreement methods.

3.5 Interactive Machine Teaching Algorithm

These steps outlined in Sections 3.1–3.4 make up our interactive machine teaching method (Algorithm 1), which we recap as follows. First, our method clusters D_U into hierarchical clusters. In each interaction round: (1) we train the Teacher and Student using labeled data, unlabeled data, and expert-validated rules (line 3.1); (2) we apply the Student on unlabeled data to get soft labels (line 3.2); (3) we pick a candidate unlabeled instance (line 3.3) and obtain its instance label from an expert (line 3.4); (4) we extract candidate rules (line 3.5) and obtain the labels for β_i rules from an expert (line 3.6); and (5) we update the labeled dataset, expert-validated rules, and the remaining budget (line 3.7). In practice, we repeat Steps 3–6 (lines 3.3–3.6) in batches of 10 instances. We repeat the full procedure until the budget T runs out.

By associating r^j with a specific instance s_i, we give the expert extra context (e.g., the text of s_i) for deciding z^j. Also, we hypothesize that, in practice, reading the text of the instance can help reduce the cost T_R for deciding z^j. While some previous work assumes that labeling rules have no extra cost (Poulis and Dasgupta, 2017), we assume that $T_R > 0$. The hyper-parameter β_i controls how to distribute the budget T. Specifically, setting $\beta_i = 0$ reduces to standard active learning, as INTERVAL will perform $\lfloor \frac{T}{T_I} \rfloor$ queries on instances only. By setting $\beta_i \geq 1$, one can exploit feedback on rules that apply to s_i. As we will show, rule feedback leads to performance improvements relative to instance feedback only.

4 Experimental Settings

We now present our experimental setting for interactive machine teaching on several text classification datasets.

Datasets. For our analysis and to evaluate our framework, we consider six benchmark datasets from diverse domains: (1) spam classification of YouTube comments (Alberto et al., 2015); (2) spam classification of SMS messages (Almeida et al., 2011); (3) sentiment classification of IMDB movie reviews (Maas et al., 2011); (4) sentiment classification of Yelp reviews (Zhang et al., 2015); (5) question classification from TREC-6 (Li and Roth, 2002); and (6) topic classification in AG-News (Zhang et al., 2015). Table 3 reports dataset statistics. For each dataset, we use expert-made rules that are provided by Zhang et al. (2021) and prompt templates that are provided by Bach et al. (2022). For a fair comparison, we use exactly the same expert-written rules[3] as in previous work, which can have various types such as keywords, regular expression patterns, and lexicons.

Experimental Procedure. To simulate the low-resource setting, we split the training examples into D_L (labeled set) and D_U (unlabeled set) by sampling 20 labeled examples per class ($20 \cdot K$ in total) uniformly at random, which we use in D_L, while we use the rest in D_U. To be consistent with our low-resource assumptions, we downsample the validation set (used for training

[3]All rules are described at `https://github.com/JieyuZ2/wrench`.

Student via early stopping) to match the size of D_L. For interactive approaches, we consider the extreme low-resource setting where $R = \emptyset$. We simulate expert feedback for candidate instances s_i from D_U (Section 3.2) using the ground-truth labels of D_U (hidden to the main algorithm), which is common in active learning research (Zhang et al., 2022c). We simulate expert feedback for candidate automatic rules (Section 3.4) using all ground-truth labels in D_U: a candidate rule r^j is accepted if it correctly classifies more than t_{oracle} of the instances in D_U that it covers. We experiment with different values of t_{oracle}: 25%, 50%, 75%, 90%, and 100% and study their impact on the student's accuracy.

For a robust evaluation, for each method we run 10 different experiments with different random seeds, thus each run corresponds to a different version of D_L, D_U, and R. We report the average test performance over the 10 different runs. As evaluation metric, we use the macro-averaged F1 of the student model on the test set.

Model Configuration. For a fair comparison, we use exactly the same text pre-processing (tokenization, embedding) as in the WRENCH benchmark (Zhang et al., 2021). Following Zhang et al. (2021), we represent each text instance (s_i) as a vector using pre-trained BERT (Devlin et al., 2019), specifically as the output embedding of the [CLS] token of BERT-base.[4] For the hyper-parameters and search space for bag-of-words logistic regression, multilayer perceptron, and BERT, see Table 10 in Zhang et al. (2022c). For candidate rule extraction, we consider conjunctions (AND) of up to $t_{len} = 3$ features consisting of n-grams with $n = 1, 2, 3$; linguistic features (part-of-speech tags and named entities extracted using the spaCy library[5]); and prompt-based features as the top $k = 10$ tokens predicted by pre-trained RoBERTa (Liu et al., 2019) for each of the templates provided by Bach et al. (2022).[6] For our analysis of rule characteristics, we experiment with different values for the minimum rule coverage on D_U ($t_{cov} \in \{10, 100, 1000\}$) and the minimum rule precision based on D_L ($t_{prec} \in \{25\%, 50\%, 75\%, 100\%\}$). In INTERVAL, we use $t_{cov} = 100$ and $t_{prec} = 75\%$. For interaction, we study different relative values for β (maximum number of rules per instance), T_R (rule labeling cost) and T_I (instance labeling cost).

Model Comparison. For a robust evaluation of our approach, we compare several approaches that utilize different resources:

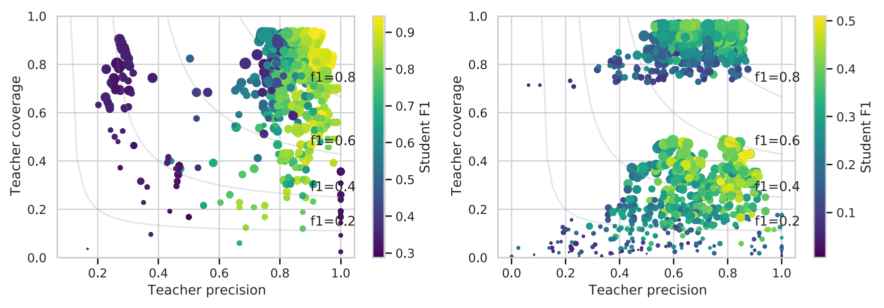
1. **"Fully supervised":** a model trained in the high-resource setting using *all* labeled data.
2. **"Low supervised":** a model trained in the low-resource setting using only D_L.
3. **"Semi supervised":** a model trained using D_L and D_U. We consider self-training (Nigam and Ghani, 2000; Lee, 2013) for up to 25 iterations with early stopping based on the validation performance.
4. **"WSL":** a model trained using D_L, D_U, and R. We experiment with different methods, including unweighted majority voting and weighted aggregation of rule predictions with majority voting, Snorkel (Ratner et al., 2017), Dawid-Skene (Dawid and Skene, 1979), FlyingSquid (Fu et al., 2020), MeTaL (Ratner et al., 2019), and ASTRA (Karamanolakis et al., 2021).
5. **"Active learning":** a model trained using D_L, D_U, and the interaction budget T. We experiment with standard active learning (performing $\lfloor \frac{T}{T_I} \rfloor$ queries on instances only) with different acquisition functions, including random instance selection, uncertainty-based sampling, hierarchical sampling (Dasgupta and Hsu, 2008), and contrastive active learning (Margatina et al., 2021). We also evaluate IWS (Boecking et al., 2020), which considers n-gram rule families and performs $\lfloor \frac{T}{T_R} \rfloor$ queries on rules only.[7]
6. **"INTERVAL":** a model trained using our interactive machine teaching method that uses D_L and D_U, and spends the interaction budget T to perform queries on both instances and rules.

[4]https://huggingface.co/google-bert/bert-base-cased.

[5]https://spacy.io/usage/linguistic-features/.

[6]Prompt templates are available at https://github.com/bigscience-workshop/promptsource.

[7]Unfortunately, the code repository for PRBoost (Zhang et al., 2022b), https://github.com/rz-zhang/PRBoost, does not contain any code as of August 9, 2024.



(a) YouTube. (b) TREC.

Figure 2: Precision-coverage scatterplots reporting the precision (x-axis) and coverage (y-axis) of the Teacher. Each data point corresponds to a different Teacher-Student pair and its color indicates the F1 score of the Student.

For a fair comparison, we use exactly the same modeling configuration across all methods (see paragraph ''Model configuration'' for details).

5 Experimental Results

We now present our analysis of expert-provided rules (Section 5.1), results on automatic rule extraction (Section 5.2), and our experiments for interactive machine teaching with queries on instances and rules (Section 5.3).

5.1 Analysis of Expert Rules

In this section, we analyze existing datasets with expert-labeled rules and simulate low-resource rule settings to understand the impact of Teacher properties on the performance of the Student.

Analysis of the Precision vs. Coverage Trade-off. In Section 1, we highlighted one challenging question: Should one prioritize rules that cover more examples but have a relatively lower precision or a few rules that have higher precision but lower coverage? To analyze the precision-coverage trade-off, we create different Teacher versions using different subsets of the expert-labeled rules and evaluate the performance of Student using each Teacher separately. For a robust analysis, we evaluate multiple Teacher types (majority voting, Snorkel (Ratner et al., 2016), Dawid-Skene (Dawid and Skene, 1979), MeTaL (Ratner et al., 2019), FlyingSquid (Ratner et al., 2019)), and multiple Student types (bag-of-words logistic regression, multilayer perceptron, BERT). See Section 4 for implementation details. For each Teacher type, we keep different randomly selected subsets of the rules in R ranging from 1% to 100%. For each Teacher-Student combination, we run 10 different experiments with different random seeds. This results in more than 1,000 Teacher-Student configurations for each dataset.

Figure 2 summarizes the results across all experiments for YouTube and TREC. While different datasets have Teacher-Student pairs with different characteristics, there are patterns that are prevalent across datasets. First, a more accurate Teacher does not necessarily lead to a more accurate Student. For example, in YouTube (Figure 2) some Teachers with F1 $\geq$ 0.6 train a Student with F1 $\geq$ 0.5, while other Teachers with F1 $\leq$ 0.2 train a Student with F1 $\geq$ 0.8. This result implies that naively optimizing the Teacher's performance (according to the standard ''data programming'' paradigm (Ratner et al., 2016)) might not lead to the best performing student model.

A second pattern that is prevalent across datasets is that the Teacher's precision is more important than coverage for training an accurate Student. In the scatterplots of Figure 2, most Teachers with high precision train high-quality Students, while many Teachers with high coverage train low-quality Students. To quantify this observation, we compute precision-coverage weights using the Teacher's precision and coverage to predict the Student's F1 score. Specifically, we compute the Student's F1 score as the weighted geometric average of the Teacher's precision and coverage, and we tune the corresponding weights using grid search. A higher weight thus indicates that the corresponding feature is more important for the prediction of the Student's F1 score. Table 4 shows the estimated precision and coverage weights for all datasets. Across all datasets, precision has higher weight than coverage: more precise Teachers lead to more accurate Students.

	YouTube	SMS	Yelp	IMDB	TREC	AGNews
Coverage weight	0.20	0.00	0.22	0.23	0.30	0.46
Precision weight	**0.80**	**1.00**	**0.78**	**0.77**	**0.70**	**0.54**

Table 4: Quantifying the relative importance of Teacher coverage and precision for training an accurate Student. Across all datasets, precision has a higher weight than coverage.

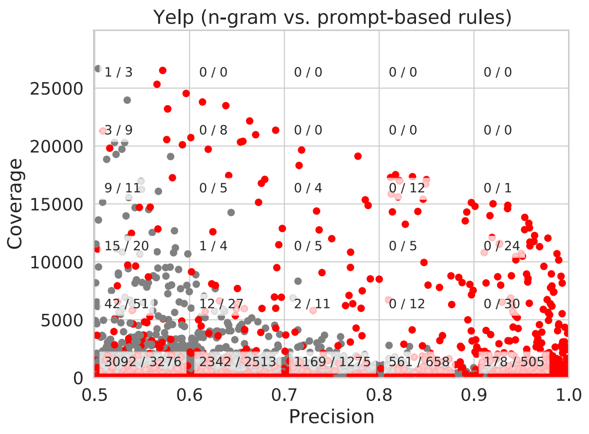


Figure 3: Precision-coverage scatterplots for automatically extracted n-grams (grey) and prompt-based rules (red). Grid numbers show the count of n-gram/prompt-based rules on the corresponding grid. Prompt-based rules can achieve relatively higher precision and coverage than n-gram rules.

Our observation that rule precision is more important than coverage explains recent design choices for WSL (Awasthi et al., 2020; Hsieh et al., 2022), such as the ‘‘contextualized LF modeling’’ component of Hsieh et al. (2022), which explicitly reduces rule coverage to improve rule precision. Moreover, our observation might inform guidelines for rule creation. In YouTube, for instance, if we reject all Teacher models with coverage lower than 0.5, then the precision's importance weight increases from 0.75 to 0.84, indicating that focusing on precision would be beneficial. Therefore, one potential guideline is that if the Teacher has a coverage higher than 50%, then the main focus should be on improving its precision.

5.2 Analysis of Automatic Rules

In this section, we compare our rule family to n-gram rules and expert rules. Figure 3 shows precision-coverage scatterplots for rules automatically extracted by our method. For this analysis, we have included all rules with precision higher than 0.5 and coverage higher than 0. Rules with high-level predicates (conjunctions of n-grams, named entities, and prompt-based features) can achieve relatively high precision and coverage compared to n-gram predicates and thus are promising to improve the overall performance of interactive machine teaching.

Table 5 reports the performance of ‘‘WSL’’ with automatically extracted rules extracted by our method using $t_{cov} = 100$ (minimum coverage) and $t_{prec} = 0.75$ (minimum precision). Across all datasets, our rule family is more effective than n-gram rules and could thus improve the effectiveness of automatic rule extraction. Also, across most datasets (except TREC and YouTube), our rule family is more effective than expert-provided rules: we effectively use D_U and D_L to discover high-quality rules. As an exception, TREC contains the highest number of manually crafted rules compared to the rest of the datasets. As we will show next, expert interaction can lead to further improvements.

5.3 Interactive Machine Teaching

Table 6 reports classification results of different methods for each dataset. For brevity, we report the best method under each category and list the average F1 across datasets (see AVG F1 column). In interactive methods, we assume $T_R = T_I$ and fix $\beta = 1$ (while we study different values later).

Non-interactive Approaches. Across non-interactive approaches, WSL ASTRA performs best: using both labeled instances and expert-provided rules is more effective than using just labeled instances (in Low Supervised or Semi Supervised), which agrees with conclusions from recent work (Karamanolakis et al., 2021). ASTRA outperformed other WSL methods, including majority voting (AVG F1= 74.1) and Snorkel (AVG F1 = 74.5).

Active Learning Approaches. Using the extra interaction budget T in Active Learning improves over Low Supervised: Labeling extra instances leads to important performance boosts,

Rule family	YouTube	SMS	IMDB	Yelp	TREC	AGNews	AVG F1
Expert	**90.0**	86.8	71.2	80.2	**57.0**	75.9	76.8
Automatic (n-gram; Boecking et al. (2020))	76.4	79.7	49.1	54.9	52.7	74.8	64.6
Automatic (ours)	82.7	**91.4**	**73.5**	**86.1**	53.3	**78.1**	**77.5**

Table 5: F1 score of the WSL method trained with expert rules and automatically extracted rules from two different families, namely, n-gram rules and high-level rules (conjunctions of n-grams, named entities, and prompt-based features). Our automatic rules lead to better performance than expert rules and n-gram rules. Performance differences for each dataset are statistically significant at $p < 0.05$ using the Student's t-test.

Method	$\|D_L\|$	D_U	R	T (T_I, T_R)	YouTube	SMS	IMDB	Yelp	TREC	AGNews	AVG F1
Fully Supervised	100%	–	–	–	94.0	95.6	79.6	87.5	90.3	80.7	88.0
Low Supervised	20·K	–	–	–	79.8	82.5	61.6	70.4	55.0	58.8	68.0
Semi Supervised	20·K	✓	–	–	80.7	83.2	63.4	72.0	55.0	60.7	69.2
WSL (ASTRA)	20·K	✓	✓	–	**90.0**	86.8	**71.2**	80.2	57.0	**75.9**	**76.8**
Active Learning (hierarchical)	20·K	✓	–	100 (100, 0)	85.3	**89.9**	67.6	**81.2**	**61.4**	71.4	76.1
INTERVAL	20·K	✓	–	100 (50, 50)	**91.4**	**94.8**	**79.3**	**86.2**	**66.6**	**78.8**	**82.8**

Table 6: F1 score reported for various methods on 6 datasets. Columns 2–5 describe the resource usage, specifically the size of labeled set D_L (where the number of classes K varies per dataset), the usage of the unlabeled set D_U and initial expert rules R, and the interaction budget for feedback on instances (T_I) and rules (T_R). We report the best performing method for each category. INTERVAL outperforms WSL and Active Learning across all datasets, where performance differences for each dataset are statistically significant at $p < 0.05$ using the Student's t-test.

as expected. Hierarchical sampling performs better than random sampling (AVG F1 = 75.0), uncertainty-based sampling (AVG F1 = 75.3), contrastive active learning (AVG F1 = 74.1), and IWS (AVG F1 = 75.3). For SMS, Yelp, and TREC, Active Learning with a budget of T = 100 outperforms ASTRA: Acquiring 100 extra instance labels is more effective than collecting expert rules for these datasets. However, for YouTube, IMDB, and AGNews, Active Learning (hierarchical) does not outperform ASTRA, which highlights that expert-provided rules are worth many examples. The above results suggest that there is no clear winner between Active Learning and WSL, and their relative performance varies across datasets.

Method	AVG F1
Fully Supervised	88.0
Low Supervised	68.0
Semi Supervised (self-training) (Lee, 2013)	69.2
WSL (majority voting)	74.0
WSL (Snorkel) (Ratner et al., 2017)	74.2
WSL (FlyingSquid) (Fu et al., 2020)	74.2
WSL (MeTaL) (Ratner et al., 2019)	74.7
WSL (ASTRA) (Karamanolakis et al., 2021)	76.8
Active Learning (random)	75.0
Active Learning (uncertainty)	75.3
Active Learning (contrastive) (Margatina et al., 2021)	75.4
Active Learning (hierarchical) (Dasgupta and Hsu, 2008)	76.1
Interactive Rule Labeling (IWS) (Boecking et al., 2020)	75.1
INTERVAL	**82.8**
INTERVAL w/o instance labeling	78.2 ↓6%
INTERVAL w/o rule labeling	76.1 ↓8%
INTERVAL w/o prompt-based rules	79.7 ↓4%
INTERVAL w/o n-gram rules	80.2 ↓3%

Table 7: Comparison of all methods (average F1 across datasets) and ablation experiments.

Interactive Learning with Queries on Rules and Instances. In Table 6, INTERVAL with a budget of T = 100 performs better than the best Active Learning (hierarchical) approach with the same budget: leveraging feedback on both instances and rules within a limited budget is more effective than feedback on instances only. Interestingly, even without using any expert-provided rules, INTERVAL outperforms ASTRA. This indicates that automatically-generated rules (analyzed in Section 5.2) are effective. While the ASTRA Student might capture implicit rules via self-training, many rules could be inaccurate, thus highlighting the importance of expert interaction.

Table 7 summarizes the results for all methods and ablation experiments. INTERVAL performs better than its ablations without instance labeling (by 6%) and without rule labeling (by 8%): feedback on both instances and rules is the most effective. Also, our rule family is more effective

Method	Budget (T) 10	50	100	150	200	250
Active Learning (rand.)	68.1	71.8	75.0	76.5	78.0	78.4
Active Learning (hier.)	68.4	73.9	76.1	78.3	79.3	79.9
INTERVAL	**76.2**	**81.1**	**82.8**	**84.3**	**85.5**	**86.2**

Table 8: Comparison of interactive methods (average F1) with different budget sizes (T). Performance differences for each budget size are statistically significant at $p < 0.05$ using the Student's t-test.

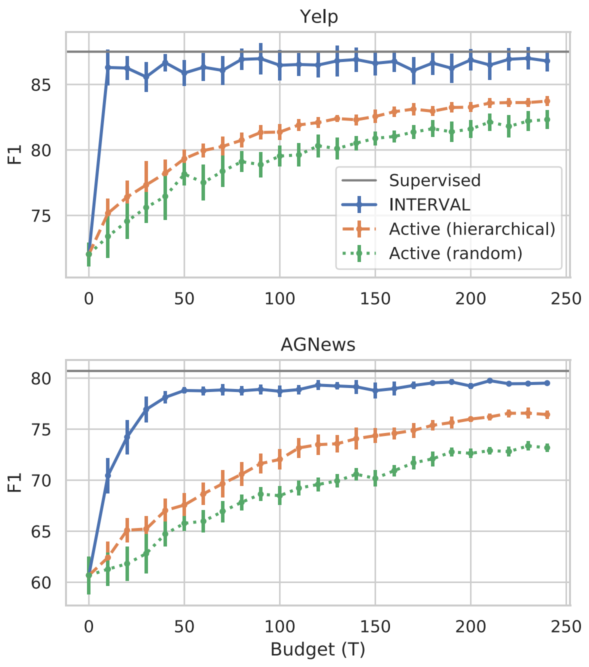


Figure 4: Performance of interactive methods on Yelp (top) and AGNews (bottom) as a function of budget (T). INTERVAL outperforms Active Learning with strongest improvements in low-budget settings (left).

than its ablations without n-gram rules (by 4%) and without prompt-based rules (by 3%). Performance differences on each dataset are statistically significant at $p < 0.05$ using the Student's t-test.

Performance with Different Budget Values. Table 8 reports the performance of interactive methods with different budget sizes ranging from 10 to 250. INTERVAL requires as few as $T = 10$ queries to reach F1 values that existing active learning methods cannot match even with $T = 100$ queries. Figure 4 shows the performance of INTERVAL compared to Active Learning approaches on Yelp and AGNews. INTERVAL leads to a big performance boost especially in low-budget settings where $T < 100$. Our results highlight that INTERVAL can effectively

	YouTube	SMS	IMDB	Yelp	TREC	AGNews
T_R	$4\,T_I$	$4\,T_I$	$8\,T_I$	$9\,T_I$	$2T_I$	$4\,T_I$

Table 9: Maximum value for T_R (cost of rule feedback) as a function of T_I (cost of instance feedback) so that feedback on rules and instances is more effective than just instance feedback.

β	YouTube	SMS	IMDB	Yelp	TREC	AGNews	AVG F1
0	85.3	89.9	67.6	81.2	61.4	71.4	76.1
1	91.4	**94.8**	**79.3**	86.2	**66.6**	78.8	82.8*
2	**91.9**	**94.8**	79.2*	**87.3**	65.0	**79.7**	**83.0**
5	91.0	94.7*	78.4	86.9	62.5	79.2	82.1

Table 10: F1 score of INTERVAL for each dataset by varying β (maximum rules labeled per instance). An asterisk (*) next to a number denotes that the difference is not statistically significant when compared to the bolded values, as determined by a p-value greater than 0.05 using the Student's t-test.

leverage feedback on both instances and automatic rules, and outperform previous interactive methods.

Evaluating the Relative Cost of Rules and Instances. So far, we have evaluated our method by assuming that $T_R = T_I$. Here, we experiment with different relative costs of labeling rules (T_R) and instances (T_I). We assume $T = 100 \cdot T_I$ (fixed total budget), $\beta = 1$ (labeling up to one rule per instance), and find the maximum value for T_R so INTERVAL ($T = \sum_i T_I + \beta_i \cdot T_R$) has an F1 score that is at least as high as the best Active Learning (hierarchical) method ($T = \sum_i T_I$). Table 9 reports the maximum T_R value for each dataset. On average across datasets, feedback on rules and instances is more effective than feedback on instances as long as $T_R \leq 5.2T_I$, though this value varies significantly per dataset and can be as high as $9T_I$ (for Yelp). In other words, our hybrid method for labeling rules and instances is highly effective even when labeling rules is 9 times (for Yelp) more expensive than labeling instances.

How Many Rules to Label per Instance. Table 10 shows the performance of INTERVAL by varying β (maximum number of rules to label per instance). Labeling up to one rule ($\beta = 1$) gives strong boosts compared to no rule labeling ($\beta = 0$) across datasets while labeling up to two rules ($\beta = 2$) gives further improvements in some tasks

Text instance s_i:
``*Prime Minister Manmohan Singh today said international environment for India's development was highly favourable...*''
Queries:
- Instance label: World
- Rule 1: NGRAM=``prime minister'' $\rightarrow$ World ✓
- Rule 2: PROMPT_IS_ABOUT=``politics'' $\rightarrow$ World ✓
- Rule 3: NGRAM=``international'' $\rightarrow$ World ✗
- Rule 4: –
- Rule 5: –

Table 11: Example from AGNews with $\beta = 5$. All classes are ``World,'' ``Sports,'' ``Business,'' and ``Sci/Tech.'' Out of the rules that were queried, 2 were accepted and 1 was rejected.

(YouTube, Yelp, AGNews). However, increasing β to values higher than 2 is less effective: when $\beta = 5$, then either less-accurate or redundant rules are queried, while this interaction budget could be used more effectively by labeling more instances (and the associated rules). Table 11 shows an example from AGNews (classes are ``World,'' ``Sports,'' ``Business,'' and ``Sci/Tech'') where INTERVAL is applied with $\beta = 5$. The candidate instance is labeled as ``World'' topic and out of the $\beta_i = 3$ rules that were queried (by satisfying the minimum precision and coverage thresholds), 2 were accepted and 1 was rejected as ``international'' also appears in other topics (e.g., ``Business''). Our analysis suggests that most performance benefits are realized by labeling up to 1 rule per instance, while future research could dynamically determine the threshold β, for example as a function of task characteristics and labeling costs.

6 Discussion and Future Work

Our framework and analysis demonstrates the advantages of soliciting feedback on both candidate rules and individual instances. We identify several areas for future research and discuss them next.

As future work, we will explore additional design choices for INTERVAL, including instance selection strategies (e.g., based on rule informativeness), rule extraction methods (e.g., based on rule diversity), and weak supervision techniques. While INTERVAL selects up to β candidate rules per instance (where β_i depends on how many rules satisfy the precision and coverage thresholds), we could further explore adaptive querying protocols, for example dynamically determining β or selectively skipping instance labeling based on dataset characteristics or labeling costs. We could also extend INTERVAL to support richer types of feedback, such as editing (rather than accepting or rejecting) candidate rules and prompt templates (rather than relying on fixed templates from Bach et al. (2022)). More research is required from a user perspective, for example on how to visualize rules (Lertvittayakumjorn et al., 2022) and effectively present a combination of rules and instances for expert labeling. INTERVAL supports prompting pre-trained models just for training data creation, and can work with any model for inference, thus enabling applications where deploying large language models might not be possible. We expect further gains by creating rules using more powerful pre-trained models such as InstructGPT (Ouyang et al., 2022); PaLM-T5 (Chung et al., 2022); LLaMA (Touvron et al., 2023a,b). We also expect performance improvements by replacing the Student using stronger pre-trained models and by representing instances using more recent text embedding techniques (He et al., 2020; Wang et al., 2023; Su et al., 2023; Muennighoff et al., 2024). INTERVAL could also be extended for multi-label classification by changing the Teacher-Student co-training objective (Section 3.1) and for other broader tasks by generating rules from more complex rule families using models such as Toolformer (Schick et al., 2023).

Our current experimental evaluation used simulated expert feedback, because a definitive evaluation involving actual subject matter experts would be too expensive. A potential stopgap is to use large language models (such as ChatGPT), which may be too expensive to query at test time, but are cheaper than subject matter experts to query at training time for selected instances.

7 Conclusions

In this paper, we presented an interactive machine teaching approach that queries experts for feedback on both instances and automatically generated rules. Our findings show that, even though rules are domain specific and have diverse characteristics, there are patterns that are prevalent across datasets. Specifically, a higher-F1 Teacher does not necessarily lead to a higher-F1 Student. We identified that the Teacher's precision is more important than coverage for training an accurate Student. These findings could potentially inform guidelines for rule creation. Our analysis demonstrates that automatic rules based on high-level predicates are more accurate than rules based on n-gram predicates. We additionally showed that by asking queries on both instances and automatically extracted rules, our method can be more effective than active learning methods.

Acknowledgments

We thank the reviewers and action editors for their constructive feedback. This material is based upon work supported by the National Science Foundation under grant no. IIS-15-63785.

References

Rakesh Agrawal and Ramakrishnan Srikant. 1994. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499.

Túlio C. Alberto, Johannes V. Lochter, and Tiago A. Almeida. 2015. Tubespam: Comment spam filtering on youtube. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. IEEE. https://doi.org/10.1109/ICMLA.2015.37

Tiago A. Almeida, José María G. Hidalgo, and Akebo Yamakami. 2011. Contributions to the study of SMS spam filtering: New collection and results. In *Proceedings of the 11th ACM Symposium on Document Engineering*, pages 259–262. https://doi.org/10.1145/2034691.2034742

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. 2019. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations*.

Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. 2016. Stance detection with bidirectional conditional encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 876–885. https://doi.org/10.18653/v1/D16-1084

Abhijeet Awasthi, Sabyasachi Ghosh, Rasna Goyal, and Sunita Sarawagi. 2020. Learning from rules generalizing labeled exemplars. In *International Conference on Learning Representations*.

Stephen Bach, Victor Sanh, Zheng Xin Yong, Albert Webson, Colin Raffel, Nihal V. Nayak, Abheesht Sharma, Taewoon Kim, M. Saiful Bari, Thibault Févry, Zaid Alyafeai, Manan Dey, Andrea Santilli, Zhiqing Sun, Srulik Ben-David, Canwen Xu, Gunjan Chhablani, Han Wang, Jason Alan Fries, Maged S. Al-shaibani, Shanya Sharma, Urmish Thakker, Khalid Almubarak, Xiangru Tang, Dragomir Radev, Mike Tian-Jian Jiang, and Alexander M. Rush. 2022. Promptsource: An integrated development environment and repository for natural language prompts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 93–104. https://doi.org/10.18653/v1/2022.acl-demo.9

Stephen H. Bach, Daniel Rodriguez, Yintao Liu, Chong Luo, Haidong Shao, Cassandra Xia, Souvik Sen, Alex Ratner, Braden Hancock, Houman Alborzi, Rahul Kuchhal, Christopher Ré, and Rob Malkin. 2019. Snorkel drybell: A case study in deploying weak supervision at industrial scale. In *Proceedings of the 2019 International Conference on Management of Data*, pages 362–375.

Sonia Badene, Kate Thompson, Jean-Pierre Lorré, and Nicholas Asher. 2019. Data programming for learning discourse structure. In *Association for Computational Linguistics (ACL)*. https://doi.org/10.18653/v1/P19-1061

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. 2022. A multitask, multilingual, multimodal evaluation of ChatGPT on reasoning, hallucination, and interactivity. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*. https://doi.org/10.18653/v1/2023.ijcnlp-main.45

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A. Raffel. 2019. Mixmatch: A holistic approach to semi-supervised learning. *Advances in Neural Information Processing Systems*, 32.

Alina Beygelzimer, Daniel J. Hsu, John Langford, and Tong Zhang. 2010. Agnostic active learning without constraints. In *Advances in Neural Information Processing Systems*, pages 199–207.

Benedikt Boecking, Willie Neiswanger, Eric Xing, and Artur Dubrawski. 2020. Interactive weak supervision: Learning useful heuristics for data labeling. In *International Conference on Learning Representations*.

Kianté Brantley, Hal Daumé III, and Amr Sharaf. 2020. Active imitation learning with noisy guidance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.189

Mary Elaine Califf and Raymond J. Mooney. 2003. Bottom-up relational learning of pattern matching rules for information extraction. *Journal of Machine Learning Research*, pages 177–210.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc Le. 2018. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. https://doi.org/10.18653/v1/D18-1217

David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. 1996. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145. https://doi.org/10.1613/jair.295

Sanjoy Dasgupta, Akansha Dey, Nicholas Roberts, and Sivan Sabato. 2018. Learning from discriminative feature feedback. In *Advances in Neural Information Processing Systems*, pages 3955–3963.

Sanjoy Dasgupta and Daniel Hsu. 2008. Hierarchical sampling for active learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 208–215. https://doi.org/10.1145/1390156.1390183

Sanjoy Dasgupta, Daniel J. Hsu, and Claire Monteleoni. 2007. A general agnostic active learning algorithm. *Advances in Neural information Processing Systems*, 20:353–360.

Alexander Philip Dawid and Allan M. Skene. 1979. Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28(1):20–28. https://doi.org/10.2307/2346806

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Liat Ein Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active learning for bert: An empirical study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. https://doi.org/10.18653/v1/2020.emnlp-main.638

Gregory Druck, Gideon Mann, and Andrew McCallum. 2008. Learning from labeled features using generalized expectation criteria. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 595–602. https://doi.org/10.1145/1390334.1390436

Daniel Fu, Mayee Chen, Frederic Sala, Sarah Hooper, Kayvon Fatahalian, and Christopher Ré. 2020. Fast and three-rious: Speeding up weak supervision with triplet methods. In *International Conference on Machine Learning*, pages 3280–3291. PMLR.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830. https://doi.org/10.18653/v1/2021.acl-long.295

Ariel Gera, Alon Halfon, Eyal Shnarch, Yotam Perlitz, Liat Ein Dor, and Noam Slonim. 2022. Zero-shot text classification with self-training. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1107–1119. https://doi.org/10.18653/v1/2022.emnlp-main.73

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. DEBERTA: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. 2011. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*.

Cheng-Yu Hsieh, Jieyu Zhang, and Alexander Ratner. 2022. Nemo: Guiding and contextualizing weak supervision for interactive data programming. *arXiv preprint arXiv:2203.01382*. https://doi.org/10.14778/3565838.3565859

Jagadeesh Jagarlamudi, Hal Daumé III, and Raghavendra Udupa. 2012. Incorporating lexical priors into topic models. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 204–213.

Nitin Jindal and Bing Liu. 2008. Opinion spam and analysis. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*. https://doi.org/10.1145/1341531.1341560

Giannis Karamanolakis, Daniel Hsu, and Luis Gravano. 2019. Leveraging just a few keywords for fine-grained aspect detection through weakly supervised co-training. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. https://doi.org/10.18653/v1/D19-1468

Giannis Karamanolakis, Subhabrata Mukherjee, Guoqing Zheng, and Ahmed Hassan Awadallah. 2021. Self-training with weak supervision. In *NAACL*. https://doi.org/10.18653/v1/2021.naacl-main.66

David Kartchner, Davi Nakajima An, Wendi Ren, Chao Zhang, and Cassie S. Mitchell. 2022. Rule-enhanced active learning for semi-automated weak supervision. *AI*, 3(1):211–228. https://doi.org/10.3390/ai3010013, PubMed: 35845102

Andreas Kirsch, Joost Van Amersfoort, and Yarin Gal. 2019. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. *Advances in Neural information Processing Systems*, 32.

Dong-Hyun Lee. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3.

Piyawat Lertvittayakumjorn, Leshem Choshen, Eyal Shnarch, and Francesca Toni. 2022. GrASP: A library for extracting and exploring human-interpretable textual patterns. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. European Language Resources Association.

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *SIGIR'94*, pages 3–12. Springer. https://doi.org/10.1007/978-1-4471-2099-5_1

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*. https://doi.org/10.3115/1072228.1072378

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*. https://doi.org/10.1145/3560815

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized

BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Andrew Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. 2021. Active learning by acquiring contrastive examples. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 650–663. https://doi.org/10.18653/v1/2021.emnlp-main.51

Prem Melville, Wojciech Gryc, and Richard D. Lawrence. 2009. Sentiment analysis of blogs by combining lexical knowledge with text classification. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1275–1284. https://doi.org/10.1145/1557019.1557156

Niklas Muennighoff, Hongjin Su, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. 2024. Generative representational instruction tuning. *arXiv preprint arXiv:2402.09906*.

Kamal Nigam and Rayid Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *Proceedings of the 9th International Conference on Information and Knowledge Management*, pages 86–93. https://doi.org/10.1145/354756.354805

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. *Advances in Neural Information Processing Systems*, 34:11054–11070.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. https://doi.org/10.18653/v1/N18-1202

Stefanos Poulis and Sanjoy Dasgupta. 2017. Learning with feature feedback: From theory to practice. In *Artificial Intelligence and Statistics*, pages 1104–1113.

Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, page 269. https://doi.org/10.14778/3157794.3157797, PubMed: 29770249

Alexander Ratner, Braden Hancock, Jared Dunnmon, Frederic Sala, Shreyash Pandey, and Christopher Ré. 2019. Training complex models with multi-task weak supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4763–4771. https://doi.org/10.1609/aaai.v33i01.33014763, PubMed: 31565535

Alexander J. Ratner, Christopher M. De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016. Data programming: Creating large training sets, quickly. In *Advances in Neural Information Processing Systems*.

Nicholas Roy and Andrew McCallum. 2001. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the 18th International Conference on Machine Learning*, pages 441–448.

Sebastian Ruder and Barbara Plank. 2018. Strong baselines for neural semi-supervised learning under domain shift. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. https://doi.org/10.18653/v1/P18-1096

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas

Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269. https://doi.org/10.18653/v1/2021.eacl-main.20

Matthias Seeger. 2006. A taxonomy for semi-supervised learning methods. Technical report, MIT Press. https://doi.org/10.7551/mitpress/6173.003.0005

Prithviraj Sen, Yunyao Li, Eser Kandogan, Yiwei Yang, and Walter Lasecki. 2019. Heidl: Learning linguistic expressions with deep learning and human-in-the-loop. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Burr Settles. 2009. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Burr Settles. 2011. Closing the loop: Fast, interactive semi-supervised annotation with queries on features and instances. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1467–1478.

Yanyao Shen, Hyokun Yun, Zachary C. Lipton, Yakov Kronrod, and Animashree Anandkumar. 2017. Deep active learning for named entity recognition. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 252–256. https://doi.org/10.18653/v1/W17-2630

Rion Snow, Daniel Jurafsky, and Andrew Ng. 2004. Learning syntactic patterns for automatic hypernym discovery. *Advances in Neural Information Processing Systems*, 17.

Ramakrishnan Srikant and Rakesh Agrawal. 1996. Mining sequential patterns: Generalizations and performance improvements. In *International Conference on Extending Database Technology*. https://doi.org/10.1007/BFb0014140

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2023. One embedder, any task: Instruction-finetuned text embeddings. In *Findings of the Association for Computational Linguistics: ACL 2023*. Association for Computational Linguistics.

Derek Tam, Rakesh R. Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. 2021. Improving and simplifying pattern exploiting training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4980–4991. https://doi.org/10.18653/v1/2021.emnlp-main.407

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Paroma Varma and Christopher Ré. 2018. Snuba: Automating weak supervision to label training data. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 12, page 223. https://doi.org/10.14778/3291264.3291268, PubMed: 31777681

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2023. Improving text embeddings with large language models. *arXiv preprint arXiv:2401.00368*.

Roman Yangarber, Ralph Grishman, and Pasi Tapanainen. 2000. Unsupervised discovery of scenario-level patterns for information extraction. In *Sixth Applied Natural Language Processing Conference*. https://doi.org/10.3115/974147.974186

Junjie Ye, Xuanting Chen, Nuo Xu, Can Zu, Zekai Shao, Shichun Liu, Yuhan Cui, Zeyang Zhou, Chao Gong, Yang Shen, Jie Zhou, Siming Chen, Tao Gui, Qi Zhang, and Xuanjing Huang. 2023. A comprehensive capability analysis of GPT-3 and GPT-3.5 series models.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3914–3923. https://doi.org/10.18653/v1/D19-1404

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. 2020. Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7935–7948. https://doi.org/10.18653/v1/2020.emnlp-main.637

Chicheng Zhang and Kamalika Chaudhuri. 2015. Active learning from weak and strong labelers. In *Advances in Neural Information Processing Systems*, pages 703–711.

Jieyu Zhang, Cheng-Yu Hsieh, Yue Yu, Chao Zhang, and Alexander Ratner. 2022a. A survey on programmatic weak supervision. *arXiv preprint arXiv:2202.05433*.

Jieyu Zhang, Yue Yu, Yinghao Li, Yujing Wang, Yaming Yang, Mao Yang, and Alexander Ratner. 2021. Wrench: A comprehensive benchmark for weak supervision. In *35th Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Rongzhi Zhang, Yue Yu, Pranav Shetty, Le Song, and Chao Zhang. 2022b. Prompt-based rule discovery and boosting for interactive weakly-supervised learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 745–758. https://doi.org/10.18653/v1/2022.acl-long.55

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657.

Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*.

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2022c. A survey of active learning for natural language processing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. https://doi.org/10.18653/v1/2022.emnlp-main.414

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR.