FAIR: Filtering of Automatically Induced Rules

Divya Jyoti Bajpai[1], Ayush Maheshwari[2*], Manjesh Kumar Hanawal[1], Ganesh Ramakrishnan[1]
[1] Indian Institute of Technology Bombay, India [2] Vizzhy Inc, Bengaluru, India
{divyajyoti.bajpai, mhanawal, ganramkr}@iitb.ac.in
ayush.maheshwari@vizzhy.com

Abstract

The availability of large annotated data can be a critical bottleneck in training machine learning algorithms successfully, especially when applied to diverse domains. Weak supervision offers a promising alternative by accelerating the creation of labeled training data using domain-specific rules. However, it requires users to write a diverse set of high-quality rules to assign labels to the unlabeled data. Automatic Rule Induction (ARI) approaches circumvent this problem by automatically creating rules from features on a small labeled set and filtering a final set of rules from them. In the ARI approach, the crucial step is to filter out a set of a high-quality useful subset of rules from the large set of automatically created rules. In this paper, we propose an algorithm FAIR (**F**iltering of **A**utomatically **I**nduced **R**ules) to filter rules from a large number of automatically induced rules using submodular objective functions that account for the collective precision, coverage, and conflicts of the rule set. We experiment with three ARI approaches and five text classification datasets to validate the superior performance of our algorithm with respect to several semi-supervised label aggregation approaches. Further, we show that FAIR achieves statistically significant results in comparison to existing rule-filtering approaches. The source code is available at `https://github.com/ayushbits/FAIR-LF-Induction`.

1 Introduction

Machine learning applications rely on large amounts of labeled training data to obtain state-of-the-art performance on downstream tasks such as text classification, machine translation, image captioning, *etc.* However, it is expensive to obtain high-quality labeled training data. Therefore, several methods such as crowdsourcing (Brabham, 2013), self-supervision (Asano et al., 2019), distant supervision (Mintz et al., 2009) and semi-supervision (Van Engelen and Hoos, 2020) techniques have been proposed to reduce the human annotation efforts. Another popular technique, *viz*, weak supervision, aims to quickly create labeled data by leveraging *expert* defined rules. These rules are generic patterns developed by assessing a few exemplars from the corpus. Typically, users encode supervision as rules in the form of labeling functions (LFs), where each rule assigns a noisy label to an instance. However, these different rules can assign different labels to an instance. Weak supervision approaches aggregate and resolves these conflicting rules to assign a weak label to an instance.

Although weak supervision methods reduce the data annotation effort, they still require human experts to frame and encode rules. Automatic rule induction (ARI) approaches circumvent this problem by automatically inducing rules from the data. ARI methods use a small labeled set to extract rules either by using decision tree approaches (Varma and Ré, 2018) or weights of a classifier (Pryzant et al., 2022). Other ARI approaches such as GRASP (Shnarch et al., 2017) extract rich linguistic patterns from a given set of positive and negative examples. These approaches initially find and filter a list of patterns to find the top-k patterns. These patterns are transformed into rules that yield noisy labels. The rules are then fed to the unsupervised (Bach et al., 2019; Chatterjee et al., 2020) or semi-supervised aggregation approaches to(Maheshwari et al., 2022; Karamanolakis et al., 2021) aggregate noisy labels.

Current ARI approaches select a final set of useful rules without considering explicit interdependencies between the rules. Classifier weights (Pryzant et al., 2022) and M-GRASP (Shnarch et al., 2017) greedily select top-k patterns having the highest weights assigned by the classifier. In this approach, any interdependence among LFs is not captured as LFs are based on the top-ranked features and not on their labeling properties.

[*] Outcome of research while pursuing PhD at IIT Bombay.

Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics Volume 1: Long Papers, pages 573–588

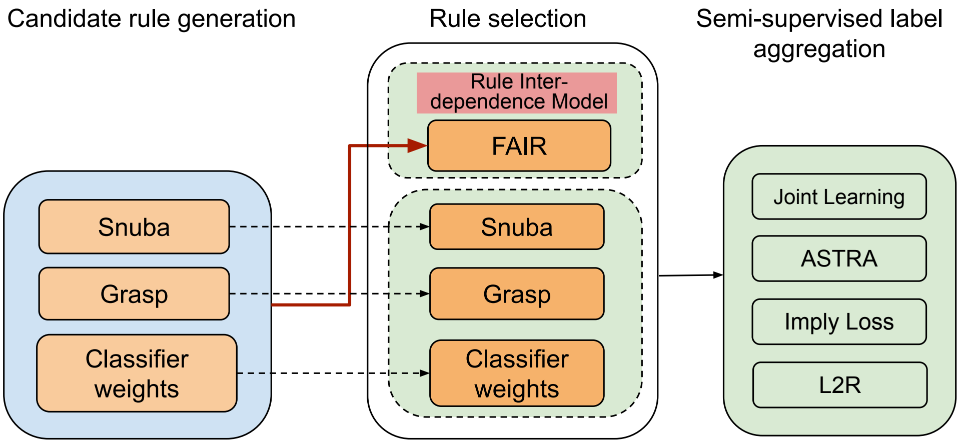

Figure 1: The flow of our approach. We first generate rules in the candidate rule generation block and then filter them using different respective approaches (such as with SNUBA (Varma and Ré, 2018), GRASP (Shnarch et al., 2017) and Classifier weights) as also with FAIR. The final committed rule set is passed on to the semi-supervised label aggregation approaches for the final performance on the downstream task.

SNUBA (Varma and Ré, 2018) chooses a rule in every iteration that maximizes the weighted sum of the F1 score on the labeled set and Jacard score. Then SNUBA reduces the labeled set size by removing instances labeled by added rule. Since LFs are generated only on the partially labeled set, the dependency of the rules set is not explicitly captured. Further, SNUBA is computationally involved since new LFs need to be generated iteratively depending upon the verifier's feedback. Also, this can lead to similar LFs getting added to the committed set, thereby causing instability.

As an illustration, for the question classification dataset which classifies a given question into five different classes, SNUBA selects the following rules: *how*, *how many*, and *many* for the class *Numeric* while FAIR (**F**iltering of **A**utomatically **I**nduced **R**ules) (Section 3.2) selects only the *how* rule in conjunction with other independent rules[1]. Clearly, derived rules for *how*, belonging to the same class, are dependent on the parent rule. Due to its iterative procedure of working with new induced rules from the uncovered labeled set, SNUBA selects LFs showing high overlap amongst themselves. Our algorithm captures explicit interdependence among LFs, thus selecting diverse and representative rules.

In this work, we propose a method, FAIR to select a subset of LFs (*committed set*) from a fixed set of automatically induced LFs (*candidate set*). Firstly, we consider a natural objective function defined as a weighted sum of precision, coverage, and agreement. Then, we optimize it over all possible subsets of sizes to obtain a final *committed* set. Though natural, this objective function is not submodular, and hence establishing its performance guarantees is not straightforward. We consider another objective function based on graph cut submodular function (Kothawade et al., 2021). Our algorithm, FAIR, maximizes the objective function based on a greedy approach to obtain the final committed set from a large set of noisy rules. The algorithm works iteratively by selecting patterns having the highest incremental precision and marginal coverage with smaller conflicts over the unlabeled set to determine the top-k rules.

Our setup favors the selection of a committed set in which the LFs do not mutually contradict while maintaining overall good accuracy and precision. Most label aggregation models built with such committed sets should consequently yield lower noise in the labeling assigned to the unlabeled set. FAIR can be used to filter rules produced by any rule generation approach and is robust to any label aggregation approach because it only uses the characteristics of rules. To the best of our knowledge, such an approach of selecting a committed set has not been addressed in erstwhile approaches.

We perform experiments with several ARI, pattern filtering, and label aggregation approaches as shown in Figure 1. For generating candidate rules, we use approaches such as decision tree (Varma and Ré, 2018), classifier weights (Pryzant et al., 2022), and a modified version of GRASP (Shnarch et al., 2017). Subsequently, we filter the large set of

[1] A class label is associated with a rule, *for eg*, if *how* appears in the text, the rule assigns a weak label as *numeric* for an instance.

rules using the corresponding candidate generation approaches and our algorithm FAIR. Finally, over the selected *committed set* of rules, we leverage semi-supervised label aggregation algorithms, *viz*, SPEAR (Maheshwari et al., 2021), ASTRA (Karamanolakis et al., 2021), ImplyLoss (Awasthi et al., 2020) and Learning to Reweight (Ren et al., 2018). We observe that our approach yields rules more refined than the other approaches by virtue of its analyzing (i) coverage and agreement over the unlabeled set and (ii) precision on the test set. We observe that rules filtered using FAIR perform better on label aggregation approaches, providing gains between 2 - 20% across different datasets. It implies that the filtering of rules is a crucial element that was not explored earlier.

2 Related Work

ARI methods have largely focused on using repetitive structures or patterns in the tasks involving text documents, *eg*, mentions of specific words or phrases (Varma and Ré, 2018; Shnarch et al., 2017). Prior work relies on this observation to learn first-order logic rules as a composition of semantic role attributes (Sen et al., 2020) or syntactic grammar rules (Sahay et al., 2021). Recently, Pryzant et al. (2022) proposed a heuristic generation method that trains a classifier on the small labeled set and uses features corresponding to the k largest values of weights as rules. Our proposed approach accounts for rule interdependence among a large set of generated heuristics and selects a useful subset of rules. Our work is closest to interactive weak supervision (Boecking et al., 2020) which uses the active learning paradigm to select a useful set of final rules from a large rule set. However, our approach does not require the additional step of human annotations.

Prior work has emphasized on LFs defined by experts based on observations in a few instances from the dataset. Unsupervised approaches like (Bach et al., 2019) use a generative model to determine the correct probability for labels in accord with noisy and conflicting labels assigned by LFs. Chatterjee et al. (2020) proposed a graphical model, CAGE, which extends to continuous LFs with scores obtained using cosine similarities of word vectors, TF-IDF score, the distance among entity pairs, *etc.* while semi-supervised approaches additionally use a small labeled set to guide the discovery of LFs for classification (Abhishek et al., 2022) and information extraction tasks (Singh et al., 2023). Recent methods (Maheshwari et al., 2022; Sivasubramanian et al., 2023) proposed a bi-level optimization wherein parameters are optimized on the validation set in a semi-supervised manner. We use semi-supervised aggregation methods in our experiments.

3 Background

3.1 Notations

Let the feature space be $\mathcal{X}$ and the label space be $\mathcal{Y} \in \{1 \dots K\}$ where K is the number of classes. We have M instances in the dataset out of which there are N labeled instances denoted by set $\mathcal{L} = \{(x_i, y_i), i = 1, 2, \dots, N\}$ and $M - N$ unlabeled set of instance denoted by set $\mathcal{U} = \{x_i : i = N+1, N+2, \dots, M\}$ where $M - N >> N$. The set of m automatically induced rules is denoted by $\mathcal{R} = (R_1, R_2 \dots R_m)$, where $R_i : \mathcal{X} \rightarrow \mathcal{Y} \cup \{0\}$ for all $i = 1, 2, \dots, m$. The label 0 corresponds to an abstain decision by a rule. Each rule may abstain on a different set of instances. We denote the final set of n filtered rules by $\mathcal{F} \subset \mathcal{R}$, where $n \leq m$.

3.2 Rule Induction Methods

We consider the following three methods for Automatic Rule Induction (ARI).

Decision Tree: SNUBA (Varma and Ré, 2018) presented an ARI approach by using a small labeled set and fitting a decision tree over n-grams of the input sentence. Initially, rules are generated as a basic component of propositions on the labeled set. A proposition could be a word, a phrase, a lemma, or an abstraction such as part of a speech tag. Each composed rule is in the form of a decision stump (1-depth decision tree). SNUBA is a three-step approach that (i) generates candidate rules using a labeled set, (ii) adds one rule based on the F1 score on the labeled set and Jacard score of the added rule, (iii) finds uncovered points or abstained points in the labeled set, and (iv) Removes the instances labeled in the labeled dataset by the added rule and repeat steps (i) - (iv) with updated labeled set. The process stops until the labeled set is completely covered or a limit on the number of iterations is reached.

Classifier Weights : Pryzant et al. (2022) trains a linear model classifier C on the small labeled set. Suppose for N instances in our dataset, each instance x_i is denoted by its feature matrix X_i of size K. The classifier model is $C(x_i) = \sigma(WX_i)$ where $W \in \mathcal{R}^{K\times N}$ is a weight matrix and σ represents an element-wise sigmoid function. Then, it finds P features corresponding to the largest weights in W which is obtained by learning the classifier and creates one rule from each feature with P largest weights. If weight $w_{i,k}$ is assigned to the i-th feature, then they create a rule associated with the i-th feature and the k-th label. Here, rule filtering is limited to choosing k rules having the largest weights in W.

M-GRASP: This is a modified version of GRASP (Shnarch et al., 2017) for automatically extracting the patterns that characterize subtle linguistic phenomena. M-GRASP augments each term of the text with multiple attributes such as lemma, hypernyms, NER, POS tags *etc.* to extract the rich set of generalizable patterns. The algorithm expects a considerable sized set of positive and negative samples to extract the discriminative patterns. In contrast to GRASP we have a small labeled set and a large unlabeled set with multiple classes. To make the core algorithm pertinent to our setting, we make two important modifications to GRASP. (i) The original GRASP algorithm uses the entire labeled set to generate an initial candidate set of patterns of length 1 but in M-GRASP we also employ the unlabeled set to generate these patterns. However, during the iterative process, we filter patterns using the information gain measure on the labeled set. (ii) While the original GRASP algorithm assumes binary classes, we extend the algorithm for the multi-class setting as well. For generating new patterns in M-GRASP we follow the same iterative process of the original GRASP algorithm other than the above-mentioned changes.

4 Problem Setup

We begin by defining key metrics used throughout the experiments. a) **Precision** of a rule on a labeled set is the ratio of the number of correctly assigned labels over the total assigned labels, b) **coverage** of a rule is the percentage of points covered over unlabeled set, c) **conflicts** between rules R_j and $R_i, j \neq i$, is the percentage of points that are assigned different labels by rule R_i and R_j and d) **agreement** between rules R_i and R_j is defined as the percentage of points that are assigned same labels by rules R_i and R_j. Note that, conflicts and agreements are related as both rules on a particular instance will either conflict or agree. We consider different submodular functions reflecting precision, coverage, and conflicts of the rules as objectives. We also explore their combinations as objective functions.

4.1 Precision Coverage Agreement Objective - f_{PCA}

Given a labeled set $\mathcal{L}$ and a candidate set of rules $\mathcal{R}$, our aim is to find a final set of rules $\mathcal{F} \subseteq \mathcal{R}$, which has high precision and coverage but fewer conflicts. Initially, we propose the following function:

$$f_{PCA}(\mathcal{F}) = w * \alpha(\mathcal{F})/|\mathcal{F}| + (1-w) * \beta(\mathcal{F}) + \gamma * \mu(\mathcal{F}) \quad (1)$$

where $\alpha(\mathcal{F}) = \sum_{R_i \in \mathcal{F}} \text{Precision}(R_i)$, $\beta(\mathcal{F})$ = coverage($\mathcal{F}$) and $\mu(\mathcal{F})$ as the percentage of instances over which all the rules in $\mathcal{F}$ provide non-conflicting labels. Given a maximum number of rules k, we define our objective as $\max_{|\mathcal{F}|\leq k} f_{PCA}(\mathcal{F})$. We observe that (1) does not satisfy the submodular properties of the function (Wei et al., 2015) (see Appendix A.1). Hence, we cannot secure theoretical guarantees for choosing the optimal rule subset. Since submodularity provides theoretical guarantees for the optimization problem, we substituted the objective as f_{GC} described in the next subsection. However, f_{PCA} still provides interpretable rules and competitive results. We perform a qualitative analysis of rules against the other variant in the Appendix 5.5. Below, we describe the algorithm for f_{PCA} in Algorithm 1. We define CovL as the function that outputs the coverage on the labeled set. Then, we add that rule to the committed set which maximizes the contribution i.e. $f_{PCA}\{\mathcal{F} \cup \{r\}\} - f_{PCA}\{\mathcal{F}\}$ as in line 5 of the algorithm.

Algorithm 1 FAIR Precision Coverage - f_{PCA}

1: **Input:** Candidate set of rules $\mathcal{R}$, Labeled set $\mathcal{L}$, Unlabeled set $\mathcal{U}$, final set of rules $\mathcal{F}$, Hyper-parameters : w, γ, k
2: Initialize $\mathcal{F} = argmax_i(f_{PCA}(R_i)) \;\; \forall i \in \mathcal{R}$
3: **while** $CovL(\mathcal{F}) < 1.0$ and $|\mathcal{F}| < k$ **do**
4: $\quad r^* \leftarrow argmax_{r\in\mathcal{R}-\mathcal{F}}(f_{PCA}\{\mathcal{F}\cup\{r\}\} - f_{PCA}\{\mathcal{F}\})$
5: $\quad \mathcal{F} \leftarrow \mathcal{F} \cup \{r^*\}$
6: **end while**
7: **Output:** $\mathcal{F}$

Termination Condition: In this variant, we have used a stopping criteria as if every instance in the labeled set got covered then we will stop. This condition states that if the labeled set is covered then we assume that rules in the committed set are diverse enough to label the unlabeled data.

4.2 Graph-Cut Submodular Objective - f_{GC}

Let Ω be a set of elements. A function $f : 2^{\Omega} \to \mathbb{R}$ is said to be submodular if it satisfies the property of diminishing returns i.e. for every $A \subseteq B \subseteq \Omega$ and $j \notin B$, $f(A \cup \{j\}) - f(A) \geq f(B \cup \{j\}) - f(B)$. A greedy algorithm provides an $\mathcal{O}(1)$-approximation to the optimal solution. Due to this algorithmic property, subset selection with submodular objective functions has found several applications in text summarization (Yao et al., 2017), video summarization (Kaushal et al., 2019a; Gygli et al., 2015), training speed up (Kaushal et al., 2019b), active learning (Settles, 2009), *etc.* Also, submodular functions have found wider acceptance due to their ability to naturally model the notion of representativeness, diversity, and coverage. Hence, we pose our rule selection problem within a submodular subset selection framework.

Kothawade et al. (2021) presents a wide array of submodular functions and their variants such as mutual information and their conditional gain counterparts. From that wide spectrum, we choose graph-cut (GC) that selects both representative and diverse instances from the ground set. We consider the candidate set of rules as the ground set and obtain a set of diverse and representative rules. For any $\mathcal{F} \subset \mathcal{R}$, the GC function is defined as follows:

$$f_{GC}(\mathcal{F}) = \sum_{i \in \mathcal{R}, j \in \mathcal{F}} s_{ij} - \lambda \sum_{i,j \in \mathcal{F}} s_{ij} \quad (2)$$

where $\lambda \in [0, 1]$ governs the trade-off between diversity and representation. Higher λ selects a diverse set of rules. s_{ij} is the similarity score for rule pair R_i and R_j. We propose a similarity score s_{ij} as:

$$s_{ij} = \alpha(R_i) + \alpha(R_j) + w * \beta(\{R_i, R_j\}) + \gamma * \mu(R_i, R_j) \quad (3)$$

where $\alpha(R_i)$ = Precision(R_i) and $\beta(\{R_i, R_j\})$ is the coverage function that gives the coverage of both the rules R_i and R_j, i.e., fraction of the unlabeled set that is labeled by at least one of the rule. $\mu(R_i, R_j)$ is the agreement between rule R_i and R_j. It is defined as the fraction of the unlabeled instances on which both rules provide the same labels. In Eq. 3, w and γ denotes the weights of β and μ respectively. While w regulates the weight given to the coverage component, γ regulates the weight given to the agreement component. The range of values of w and γ are chosen by carefully analyzing the statistics related to the coverage and precision.

Tuning w and γ We tune w and γ based on the validation dataset. We observed the quality of rules when w is $1 \leq w \leq 10$ and γ is $0 \leq \gamma \leq 1$. We found that when w is between 2 and 4, coverage component β has weightage comparable to the precision components. Intuitively, both components contribute equally while producing the final committed rule set. γ refers to the weighing factor for agreement between rule R_i and R_j. We found the best rule set when γ is between 0.2 and 0.5. The high agreement between various rules compensates for the lower values of γ.

In the GC function, $\lambda \in [0, 1]$ governs the trade-off between representation and diversity and we need a committed set that is diverse enough. In our experiments, we set λ to 0.7 so that the final candidate rules are diverse in nature. GC is a non-monotone submodular for $\lambda > 0.5$, hence in our case, f_{GC} is a non-monotone submodular function.

Given a cardinality budget constraint on the number of rules k, our objective function is, $\max_{|\mathcal{F}| \leq k} f_{GC}(\mathcal{F})$ which is a submodular function (Kothawade et al., 2021). We use a greedy algorithm for maximizing this function. We greedily choose the rule that maximizes the marginal utility i.e. $argmax_{i \in \{\mathcal{R} - \mathcal{F}\}} f_{GC}(\mathcal{F} \cup \{i\}) - f_{GC}(\mathcal{F})$. The greedy algorithm begins with an empty set and then iteratively adds a rule from the candidate set to the committed set by maximizing the marginal gain in every iteration until the budget constraint is met. The pseudo-code of our approach is given in Algorithm 2. For a candidate set of rules $\mathcal{R}$ induced from ARI approaches (Section 3.2), we compute precision on $\mathcal{L}$ using $findprecision$ and coverage over $\mathcal{U}$ using $findcoverage$ for each rule R_i. We calculate the agreement between two rules R_i, R_j using the $findagreement$ function. Then, we compute s_{ij}, $(i, j)^{th}$ entry of matrix S defined as in line 8. Finally, we find the set of committed rules $\mathcal{F}$ using Eq. 2 for a pre-specified budget of k rules. Note: In the following sections FAIR refers to the GC variant of FAIR, unless otherwise stated.

5 Experiments

We select five text classification datasets and compare rules induced by FAIR with the rules generated

Algorithm 2 FAIR Graph cut

Input: Candidate set of rules $\mathcal{R}$, Labeled set $\mathcal{L}$, Unlabeled set $\mathcal{U}$, final set of rules $\mathcal{F}$, w, maximum number of LFs k

Initialize $S = [0]_{|\mathcal{R}|\times|\mathcal{R}|}$

$\alpha(R_i) \leftarrow findprecision(R_i)$

$\beta(R_i + R_j) \leftarrow findcoverage(\{R_i \cup R_j\})$

$\mu(R_i, R_j) \leftarrow findagreement(R_i, R_j)$

$S(i,j) \leftarrow \alpha(R_i) + \alpha(R_j) + w * \beta(R_i + R_j) + \gamma * \mu(R_i, R_j)\ \forall i \neq j$

$\mathcal{F} \leftarrow GraphCut(|R|, S, k)$

Output: $\mathcal{F}$

by three ARI approaches (*c.f.*, Section 3.2). We measure the efficacy of these rules by aggregating the assigned labels using four semi-supervised label aggregation approaches. It takes around approximately 2 - 6 GPU hours to complete all experiments except SST which takes around 12 hours to complete. We performed all our experiments on a single Nvidia RTX 2070.

Datasets : (1) **TREC** (Li and Roth, 2002) is a multi-class question classification dataset consisting of open-domain, fact-based questions divided into broad semantic categories. The dataset has the following six labels: *Abbreviation, Description, Entities, Human, Locations* and *Numeric values*. (2) **YouTube Comment Classification** (Alberto et al., 2015) is a spam comment classification dataset. (3) **IMDB Genre Classification** [2] is a binary movie-genre classification dataset from a movie plot summary. The labels are *romantic* and *action*. (4) **SMS Spam Classification** (Almeida et al., 2011) is a binary classification dataset to classify a given sms as *spam* or *not spam*. (5) **Stanford Sentiment Treebank (SST)** (Socher et al., 2013) is a collection of written or spoken texts with fully labeled parse trees for a complete analysis of the compositional effects of sentiment in language. The output labels for this dataset are *negative, somewhat negative, neutral, somewhat positive* and *positive*.

5.1 Label aggregation methods

We use a small labeled set $\mathcal{L}$ to automatically induce rules using different ARIs. We form an Only-$\mathcal{L}$ baseline, where we train a supervised classifier on the small labeled set $\mathcal{L}$ using standard cross-entropy loss. The network architecture for Only-$\mathcal{L}$ is similar to Awasthi et al. (2020). We use the following semi-supervised label aggregation approaches:

Learn to Reweight (L2R) (Ren et al., 2018): This approach uses the noisy labels provided by rules as well and trains the classifier by meta-learning to reweight the loss on noisily labeled instances, and for performing this step clean labeled dataset $\mathcal{L}$ is utilized.

Imply Loss (Awasthi et al., 2020): It uses additional information in the form of labeled rule exemplars jointly denoises rules via latent coverage variables, and trains a model on soft implication loss over coverage and label variables.

SPEAR (Maheshwari et al., 2021) is a semi-supervised paradigm that jointly learns the parameters over features and labeling functions (rules) in a semi-supervised manner. It jointly learns a parameterized graphical model and a classifier model.

ASTRA (Karamanolakis et al., 2021) is a weak-supervision framework that uses all available data both labeled and unlabeled set in an iterative self-training framework. It trains a student model on unlabeled data that considers contextualized representations and predicts pseudo-labels for instances not covered by rules. Thereafter, it learns a rule attention network that learns to aggregate pseudo-labels assigned by the student model in conjunction with noisy labels assigned by rules. An iterative student-teacher model is trained with a semi-supervised objective.

5.2 Experimental Setting

We use 10% of the dataset as a labeled set to generate rules for our model. The 10% labeled set is split equally for the label aggregation stage. We reserve 5% of the total corpus as the labeled set and 5% as a validation set while the rest of the set is unlabeled. We performed a final evaluation on 500 instances for each dataset (refer Table 1 in Appendix). The remaining portion of the dataset was left unlabeled. We use SNUBA with raw count-based features with a decision tree to generate a candidate rule set. M-GRASP uses lemma, hypernym, text, and sentiment-based attributes to generate the rule set. The classifier weight approach uses logistic regression to train a classifier model for finding the top-*k* weights and associated features with these weights as rules. For label aggregation methods, we follow the same hyper-parameters as provided in the respective codebases. The features set is the same as followed in the SPEAR. It

[2] http://www.imdb.com/datasets

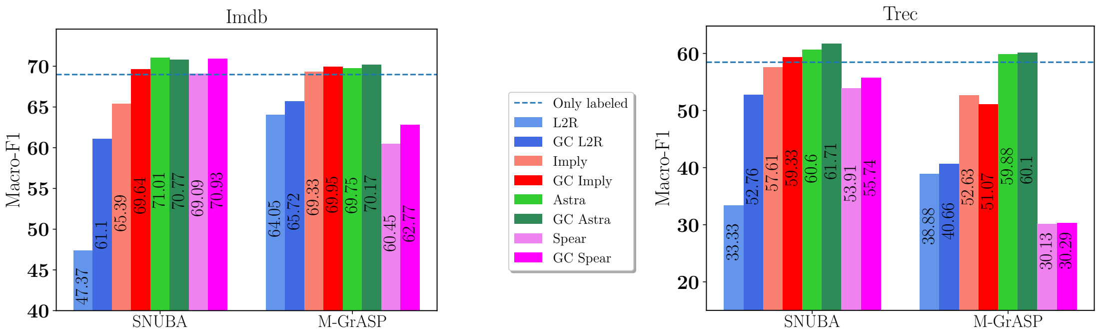

Figure 2: Results for IMDB and TREC dataset for FAIR GC against SNUBA and M-GRASP.

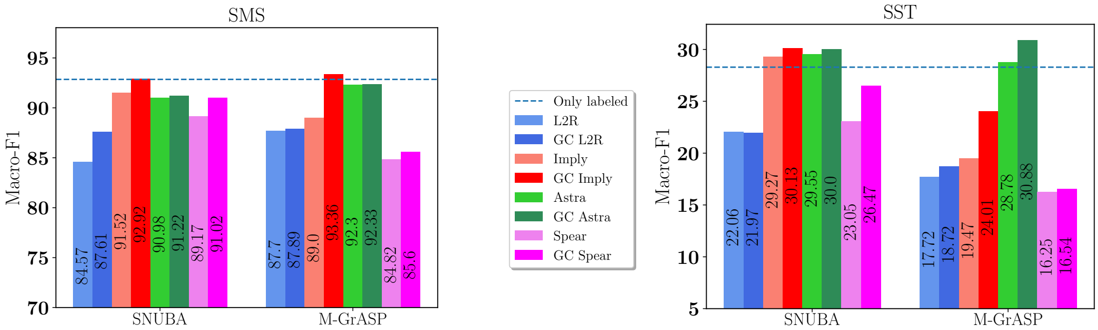

Figure 3: Comparison of FAIR against SNUBA and M-GRASP filtering over different label aggregation approaches, GC is FAIR GraphCut. The size of the final committed set is the same across all ARI approaches.

yields the best performance on the combination of first, third, fourth, and sixth loss combination. On all datasets, macro-F1 is employed as the evaluation criterion. Performance numbers for each experiment are obtained by averaging over five independent runs, each having a different random initialization.

Rule-level analysis We also performed analysis on the rules retrieved from FAIR and compared them with other filtering approaches. By qualitatively analyzing the rules, we found that FAIR returns a more interpretable and diverse collection of rules. By analyzing rules provided in the Appendix, we observe that FAIR is not prejudiced against any particular class. We see that different filtering techniques frequently favor a single class more than others. The diversity of the rules aids in covering more instances, leading to higher-end-model performance. One particular example from the SMS dataset could be *"www"* is a highly precise rule denoting the *spam* class was missed by the classifier's filtering method but covered by FAIR. Other than qualitative analysis, we also provide statistics of rules such as coverage, agreement, and precision on the test set of the committed set of rules. We compare them against different filtering approaches.

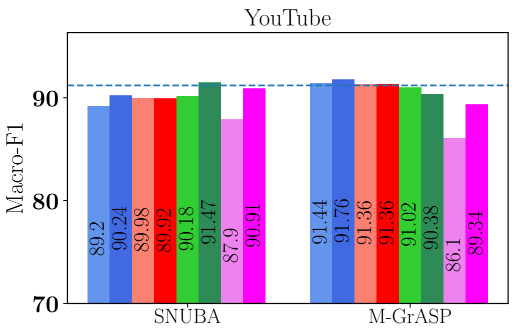

Figure 4: Results on YouTube dataset for FAIR GC

Due to the paucity of space, we present a full discussion in the Appendix section.

5.3 Results

In Figure 2, 3, 4 and 5, we present results for the four label aggregation methods for various datasets and rule filtering approaches (SNUBA GRASP Classifier weights and FAIR). For the SMS dataset, FAIR achieves better performance than rules induced by SNUBA, M-GRASP, and Classifier weights for all label aggregation approaches. We achieve maximum gains on L2R by up to 19 points and up to 5-point gains over ImplyLoss. Similarly, on ASTRA maximum gains are a bit lesser

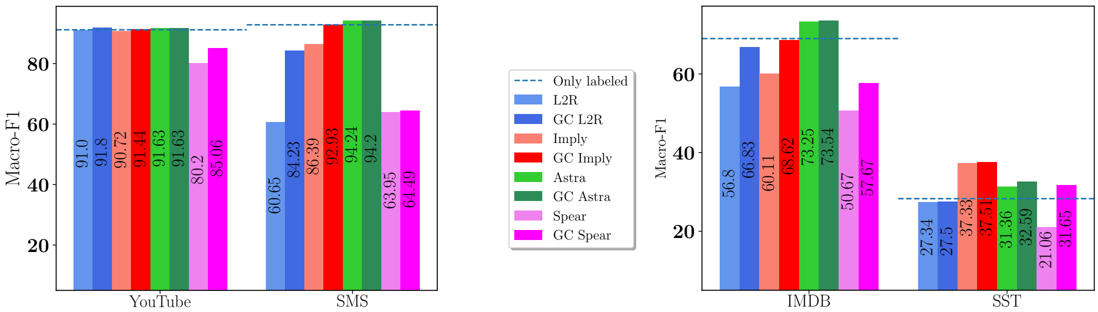

Figure 5: Comparison of FAIR GC against Classifier weights on different datasets.

of about 2 points, and on SPEAR maximum gains are 10 points. On the SST dataset, we observe a performance drop on L2R, however, gains are consistent for other approaches. This could be possible due to the aggregation scheme of L2R. FAIR consistently outperforms all the other filtering approaches across all label aggregation approaches and across all the datasets with a significant margin.

On SNUBA In comparison to SNUBA we observe better end-model performance using rules filtered by FAIR. In each iteration, SNUBA reduces the size of its candidate set thus missing out on important rules. FAIR produces the committed set of rules in a single iteration and chooses a diverse set of rules resulting in rules having higher coverage and competitive agreement.

On Classifier weights Since classifier weights choose top-K rules according to weights of the features, it does not model the coverage of a rule on an unlabeled dataset unlike FAIR. Further, it does not explicitly model the agreement between rules resulting in conflicting rules (See Table 6).

On M-GRASP: M-GRASP chooses the committed set according to the information gain of each rule. However, it does not model agreement as well as the diversity of rules in the committed set. In Table 5, we observe consistently better coverage and precision with a competitive agreement resulting in better performance for most of the datasets.

Observations on ASTRA: ASTRA uses self-supervision and uses a weighted sum of labels provided by rules and the student(classifier) model. While tuning the hyperparameter, we observed that the student model has a huge impact on the labels provided to unlabeled instances reducing the impact of rules on the final number. Even though rules have higher conflicts and low coverage in the committed set, ASTRA yields comparable numbers, unlike other aggregation approaches. This justifies lower gains over the ASTRA approach and even hits the YouTube dataset with M-GRASP rule generation.

5.4 Significance test

We employ the Wilcoxon signed-rank test (Wilcoxon, 1992) for statistical significance test. We chose the null hypothesis as there is no significant difference between FAIR and other filtering approaches and we successfully rejected it over all datasets. We had a value of n = 20 as there are 5 datasets and 4 aggregation approaches. FAIR is statistically significant at $p < 0.05$ than all other filtering methods across all label aggregation approaches. These results suggest that FAIR is robust to any label aggregation as well as any rule induction approach. p and z- values are reported in table 2.

5.5 Comparison between f_{GC} and f_{PCA}

Though theoretically not very promising, f_{PCA} implemented in algorithm 1, has reasonable interpretation and comparable performance. This section discusses the results obtained by implementing algorithm 1 on the datasets. In Figure 6, 7, and 8, we demonstrate the results for f_{PCA} and compare them against previous filtering approaches. We use the same data split as for FAIR GC. The numbers reported here are averaged over five independent runs over random generalizations. We tuned w and γ over a validation set. We note that we are getting constant gains using FAIR PCA, but the gains are small as compared to FAIR GC in most cases. In most of the datasets, initially, the rules chosen from the candidate set were the same for FAIR GC as well as FAIR PCA since the objective function performs similarly until the cardinality of the committed set is small. As more rules were added, we observed different rules being added in the committed set of rules as FAIR GC chooses more diverse

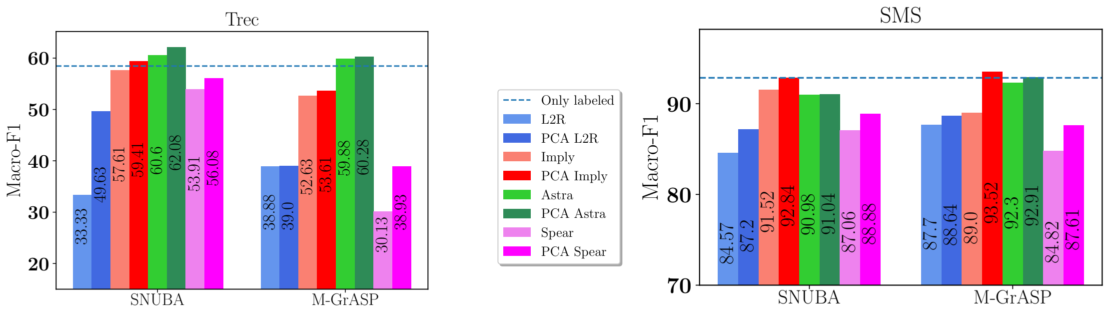

Figure 6: Comparison of Macro F1 score of SNUBA and M-GRASP on different aggregation approaches for TREC dataset and SMS dataset.

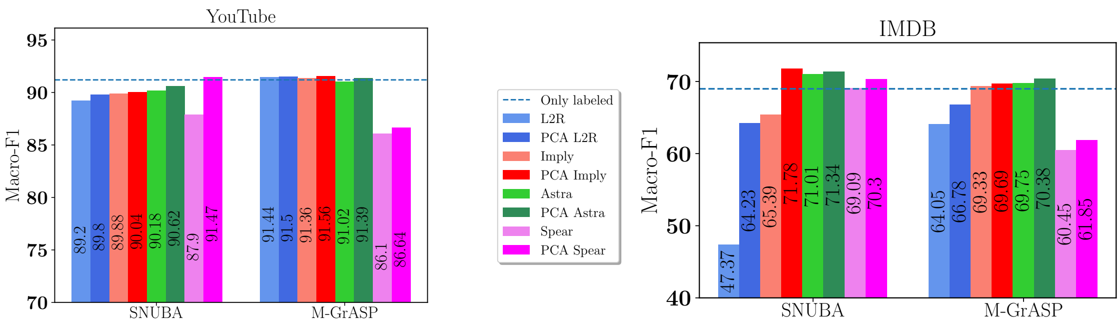

Figure 7: Comparison of Macro F1 score of SNUBA and M-GRASP on different aggregation approaches for YouTube and IMDB dataset.

rules using λ parameter. One such example could be found in Table 3 in the Appendix. In that table, each italic entry is a rule which assigns a weak label to an instance. Observe that GC rules make more sense than CW rules as *shuffle* can be reduced from the committed set of CW rules as it is not a likely word in a spam or ham comment. If we compare GC rules and PCA rules, we can observe that GC rules are more diverse covering rules from different classes. From Table 3, we observe PCA rules have only one rule in class *ham* while GC has three proving it to be more diverse than PCA.

6 Conclusion

We propose FAIR, a rule-filtering approach that selects a useful subset of rules from a given large candidate set of rules by leveraging implicit interdependencies among the rules. We introduce an objective function that maximizes precision, coverage, and agreement among rules and augments the function by designing a submodular function providing convergence guarantees. We conduct extensive experiments and demonstrate the importance of selecting high-quality and diverse rules with very few labeled instances. We qualitatively analyze the rules generated by FAIR and existing approaches. Further, we show that FAIR outperforms existing filtering approaches in terms of end-model performance using different label aggregation methods which makes FAIR robust to different aggregation as well as rule generation approaches.

7 Limitations

A key limitation is the performance of our approach on rule sets that are more noisier than current datasets. Our benchmark rule-filtering methods rely on generating and filtering via the same approach. An enhanced benchmark could encompass rule generation through one approach and subsequent filtering through a different method. The ARI approach is linked to the size of the labeled set. With an increase in the size of this set, the time required for rule generation and the rule-filtering method also correspondingly increases.

Acknowledgements

Divya Jyoti Bajpai is supported by the Prime Minister's Research Fellowship. Manjesh K. Hanawal thanks funding support from SERB, Govt. of India, through the Core Research Grant (CRG/2022/008807) and MATRICS grant (MTR/2021/000645), and DST-Inria Targeted Programme. Ganesh Ramakrishnan is grateful to the National Language Translation Mission (NLTM): Bhashini project by Government of India and IIT Bombay Institute Chair Professorship for their support and sponsorship.

References

Guttu Abhishek, Harshad Ingole, Parth Laturia, Vineeth Dorna, Ayush Maheshwari, Ganesh Ramakrishnan, and Rishabh Iyer. 2022. Spear: Semi-supervised data programming in python. In *Proceedings of the The 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 121–127.

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)*, pages 138–143. IEEE.

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.

YM Asano, C Rupprecht, and A Vedaldi. 2019. A critical analysis of self-supervision, or what we can learn from a single image. In *International Conference on Learning Representations*.

Abhijeet Awasthi, Sabyasachi Ghosh, Rasna Goyal, and Sunita Sarawagi. 2020. Learning from rules generalizing labeled exemplars. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Stephen H Bach, Daniel Rodriguez, Yintao Liu, Chong Luo, Haidong Shao, Cassandra Xia, Souvik Sen, Alex Ratner, Braden Hancock, Houman Alborzi, et al. 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.

Benedikt Boecking, Willie Neiswanger, Eric Xing, and Artur Dubrawski. 2020. Interactive weak supervision: Learning useful heuristics for data labeling. *arXiv preprint arXiv:2012.06046*.

Daren C Brabham. 2013. *Crowdsourcing*. Mit Press.

Oishik Chatterjee, Ganesh Ramakrishnan, and Sunita Sarawagi. 2020. Robust data programming with precision-guided labeling functions. In *AAAI*.

Michael Gygli, Helmut Grabner, and Luc Van Gool. 2015. Video summarization by learning submodular mixtures of objectives. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3090–3098.

Giannis Karamanolakis, Subhabrata Mukherjee, Guoqing Zheng, and Ahmed Hassan. 2021. Self-training with weak supervision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 845–863.

Vishal Kaushal, Rishabh Iyer, Khoshrav Doctor, Anurag Sahoo, Pratik Dubal, Suraj Kothawade, Rohan Mahadev, Kunal Dargan, and Ganesh Ramakrishnan. 2019a. Demystifying multi-faceted video summarization: tradeoff between diversity, representation, coverage and importance. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 452–461. IEEE.

Vishal Kaushal, Rishabh Iyer, Suraj Kothawade, Rohan Mahadev, Khoshrav Doctor, and Ganesh Ramakrishnan. 2019b. Learning from less data: A unified data subset selection and active learning framework for computer vision. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1289–1299. IEEE.

Suraj Kothawade, Vishal Kaushal, Ganesh Ramakrishnan, Jeff Bilmes, and Rishabh Iyer. 2021. Prism: A rich class of parameterized submodular information measures for guided subset selection. *arXiv preprint arXiv:2103.00128*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Ayush Maheshwari, Oishik Chatterjee, KrishnaTeja Killamsetty, Rishabh K. Iyer, and Ganesh Ramakrishnan. 2021. Data programming using semi-supervision and subset selection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*.

Ayush Maheshwari, Krishnateja Killamsetty, Ganesh Ramakrishnan, Rishabh Iyer, Marina Danilevsky, and Lucian Popa. 2022. Learning to robustly aggregate labeling functions for semi-supervised data programming. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1188–1202.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. 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*, pages 1003–1011.

Reid Pryzant, Ziyi Yang, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2022. Automatic rule induction for efficient semi-supervised learning. *arXiv preprint arXiv:2205.09067*.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning*, pages 4334–4343.

Atul Sahay, Anshul Nasery, Ayush Maheshwari, Ganesh Ramakrishnan, and Rishabh K Iyer. 2021. Rule augmented unsupervised constituency parsing. In *ACL/IJCNLP (Findings)*.

Prithviraj Sen, Marina Danilevsky, Yunyao Li, Siddhartha Brahma, Matthias Boehm, Laura Chiticariu, and Rajasekar Krishnamurthy. 2020. Learning explainable linguistic expressions with neural inductive logic programming for sentence classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4211–4221.

Burr Settles. 2009. Active learning literature survey.

Eyal Shnarch, Ran Levy, Vikas Raykar, and Noam Slonim. 2017. Grasp: Rich patterns for argumentation mining. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1345–1350.

Abhishek Singh, Venkatapathy Subramanian, Ayush Maheshwari, Pradeep Narayan, Devi Prasad Shetty, and Ganesh Ramakrishnan. 2023. Eigen: Expert-informed joint learning aggregation for high-fidelity information extraction from document images. In *Machine Learning for Health (ML4H)*, pages 559–573. PMLR.

Durga Sivasubramanian, Ayush Maheshwari, AP Prathosh, Pradeep Shenoy, and Ganesh Ramakrishnan. 2023. Adaptive mixing of auxiliary losses in supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 9855–9863.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Jesper E Van Engelen and Holger H Hoos. 2020. A survey on semi-supervised learning. *Machine Learning*, 109(2):373–440.

Paroma Varma and Christopher Ré. 2018. Snuba: Automating weak supervision to label training data. *Proc. VLDB Endow.*, 12(3):223–236.

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. NIH Public Access.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. 2015. Submodularity in data subset selection and active learning. In *International Conference on Machine Learning*, pages 1954–1963.

Frank Wilcoxon. 1992. Individual comparisons by ranking methods. In *Breakthroughs in statistics*, pages 196–202. Springer.

Jin-ge Yao, Xiaojun Wan, and Jianguo Xiao. 2017. Recent advances in document summarization. *Knowledge and Information Systems*, 53(2):297–336.

Dataset	$\|\mathcal{L}\|$	$\|\mathcal{U}\|$	|TEST|
IMDB	71	1278	500
YouTube	54	977	500
SMS	463	8335	500
TREC	273	4918	500
SST	568	10219	500

Table 1: More details on size of labeled set $\mathcal{L}$, unlabeled set $\mathcal{U}$ and size of Test data.

	Snuba	M-Grasp	Classifier
p-value	0.0018	0.0265	0.0180
z-value	2.9000	1.9300	2.0900

Table 2: p-value and z-value for statistical significance tests

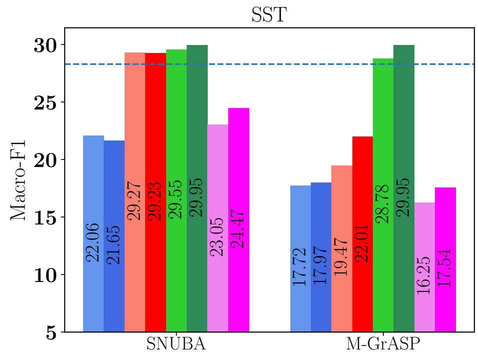

Figure 8: Results of f_{PCA} on SST dataset

Appendix

A Non-submodularity of f_{PCA}

We observed that f_{PCA} variant of FAIR does not follow the submodularity properties due to which we could not find any approximation guarantees for f_{PCA}. We illustrate this using the following example.

Example A.1. Consider sets S and T such that $S \subset T \subset \mathcal{R}$. Let us consider $S = \{R_1\}$ and $T = \{R_1, R_2\}$ and consider the rule $R_3 \in \mathcal{R} \neq R_1, R_2$ where $\alpha(R_1) = \alpha(R_2) = 1.0$ and $\alpha(R_3) = 0.5$ and all the rules covers same points with $\beta(R_1) = \beta(R_2) = \beta(R_3) = 0.1$ and $\mu(R_1) = 1.0, \mu(R_1 \cup R_2) = 0.5, \mu(R_1 \cup R_3) = 0.5$ and $\mu(R_1 \cup R_2 \cup R_3) = 0.5$ then f_{PCA} violates the property of diminishing marginal returns.

We first calculate $f_{PCA}(S) = 1.0 + 0.1 + 1.0 = 2.1$ and $f_{PCA}(S \cup R_3) = (1.0 + 0.5)/2 + 0.1 + 0.5 = 0.90$. Then we calculate $f_{PCA}(T) = (1.0 + 1.0)/2 + 0.1 + 0.5 = 1.6$. $f_{PCA}(T \cup R_3) = (1 + 1 + 0.5)/3 + 0.1 + 0.5 = 1.43$.

Now observe that $f_{PCA}(S \cup R_3) - f_{PCA}(S) = -1.2 \leq f_{PCA}(T \cup R_3) - f_{PCA}(T) = -0.2$. Hence f_{PCA} does not follows the property of diminishing returns on this example. Hence we conclude that f_{PCA} is not submodular.

B Rule-related statistics

In tables 4, 5 and 6, we provide the rule-related statistics i.e. Precision of the rules on the test set, Coverage of the rule set as well as the Agreement of the rule set. We observe from the tables that if FAIR has a gain in the coverage then it has a low agreement which is intuitive as more coverage will surely cause more conflicts. The committed set of rules is taken as generated by the respective ARIs. For a fair comparison, we have taken the same number of rules in the final committed set of all filtering approaches.

Observe that although there are mixed results for coverage and agreement as if FAIR GC does better in terms of Agreement, it gets a hit in coverage and vice-versa. However, FAIR GC performs consistently better in terms of precision on the test set across all the rule induction approaches.

C Qualitative Analysis

We qualitatively analyse the rules generated by different ARI approaches for various dataset in tables 7, 8, 9, 10, 11 and 12. We provide rules in the committed set of different rule filtering approaches. We observe in table 7 that FAIR GC provides more diverse rules hence covers both the classes equally in this case while for classifier weights rules are more biased towards the *"ham"* class. Observe that rules make more sense when filtered using the FAIR GC. Similar observations are made in table (9, 10) as well as table (11, 12).

Table 3: Final committed set of rules generated using Classifier weights for the YouTube dataset.

	ARI: CW, Dataset: YouTube					
Filtering	CW		GC		PCA	
Class	Spam	Ham	Spam	Ham	Spam	Ham
Rules	*check*	*song*	*check*	*song*	*check*	*song*
	subscribe	*views*	*subscribe*	*good*	*subscribe*	_
	com	*shuffle*	*http*	*years*	*http*	_
	channel	_	*channel*	_	*channel*	_
	_	_	_	_	*watch*	_
	_	_	_	_	*com*	_

Table 4: Rule statistics for Snuba and GC variant of FAIR. #RulesCS is the number of rules in the candidate set, and #Rules are the number of rules in the committed set. Coverage is the percentage of points covered in the unlabeled set, Agreement is the percentage of points on which rules have non-conflicting labels and Precision is the micro-precision of rules on the test set. Number of rules in the final set was the same for Snuba and FAIR.

Dataset	**#RulesCS**	**#Rules**	**Coverage**		**Agreement**		**Precision**	
			Snuba	GC	Snuba	GC	Snuba	GC
YouTube	99	7	55.8	54.3	95.8	96.0	94.3	94.5
IMDB	143	16	41.2	43.6	95.3	90.2	76.5	77.3
Trec	930	15	62.2	71.9	89.7	87.8	70.1	75.8
SMS	137	18	46.6	41.0	96.6	100	93.8	96.3
SST	2057	70	42.5	40.0	86.5	93.4	35.9	36.1

Table 5: Rule statistics for M-Grasp and GC variant of FAIR.

Dataset	**#RulesCS**	**#Rules**	**Coverage**		**Agreement**		**Precision**	
			M-Grasp	GC	M-Grasp	GC	M-Grasp	GC
YouTube	200	7	60.1	61.9	88.2	85.2	88.9	91.0
IMDB	200	16	49.3	77.9	89.4	70.3	68.5	69.3
Trec	200	15	47.2	48.9	97.7	99.1	45.8	46.3
SMS	200	18	65.5	71.3	95.5	95.2	84.9	85.1
SST	200	70	90.6	92.1	0.01	0.01	31.2	32.4

Table 6: Rule statistics for Classifier Weights (CW) and GC variant of FAIR.

Dataset	**#RulesCS**	**#Rules**	**Coverage**		**Agreement**		**Precision**	
			CW	GC	CW	GC	CW	GC
YouTube	50	7	63.6	63.4	91.0	93.7	93.3	94.2
IMDB	50	16	60.2	54.3	92.6	89.6	71.9	78.4
Trec	50	15	22.1	13.9	93.2	98.8	76.4	81.4
SMS	50	18	65.7	59.6	73.8	69.5	47.8	48.6
SST	100	70	24.4	15.0	89.5	94.1	29.1	29.2

Table 7: Final set of rules generated by CW and FAIR-GC for YouTube dataset.

Filtering	CW		GC	
Class	Spam	Ham	Spam	Ham
Rules	*txt*	*yo*	*txt*	*ll*
	voucher	*wat*	*mobile*	*wat*
	150p	*oh*	*claim*	*oh*
	nokia	*ll*	*www*	*said*
	ringtones	*lol*	*service*	*later*
	500	*yup*	*uk*	*town*
	_	*haha*	*text*	*class*
	_	*ve*	*urgent*	*didn*
	_	*aight*	*ringtone*	*aight*
	_	*said*	*orange*	*gonna*
	_	*sure*	_	_
	_	*fine*	_	_
	_	*sir*	_	_
	_	*later*	_	_

Table 8: Final set of rules generated by Snuba and FAIR-GC for IMDB dataset.

Filtering	Snuba		GC	
Class	Action	Romance	Action	Romance
Rules	*love*	*team*	*york*	*world*
	man	*government*	*new york*	*year*
	boyfriend	*agent*	*girl*	*american*
	discovers	*race*	*story*	*war*
	friend	*home*	*falls*	*agent*
	town	*cop*	*friend*	*time*
	friendship	*earth*	*best*	*team*
	story	_	*meets*	_
	falls	_	*young man*	_

Table 9: Final set of rules generated by Classifier Weights (CW) approach on SST Dataset

Filtering	CW				
Class	Negative	Somewhat negative	Neutral	Somewhat positive	Positive
Rules	*money*	*heavy*	*appealing*	*era*	*excellent*
	bad	*popcorn*	*conciousness*	*reality*	*emotionally*
	assed	*confusing*	*glib*	*presents*	*mesmerizing*
	dull	*sappy*	*insomnia*	*urban*	*stuck*
	flaccid	*feel good*	*movie work*	*issues*	*roles*
	town	_	*just like*	*riveting*	*frailty*
	_	_	*film ca*	*liked*	*performances*
	_	_	*igby*	*reality*	_
	_	_	_	*heart*	_
	_	_	_	*moved*	_
	_	_	_	*life*	_
	_	_	_	*tasty*	_
	_	_	_	*look*	_
	_	_	_	*odds*	_
	_	_	_	*actor*	_
	_	_	_	*spice*	_
	_	_	_	*new*	_
	_	_	_	*diversion*	_
	_	_	_	*voices*	_
	_	_	_	*strong performances*	_
	_	_	_	*answers*	_
	_	_	_	*works*	_

Table 10: Final set of rules generated by FAIR-GC for SST dataset

Filtering	GC				
Class	Negative	Somewhat negative	Neutral	Somewhat positive	Positive
Rules	*assed*	*heavy*	*appealing*	*presents*	
	really bad	*popcorn*	*just like*	*issues*	*mesmerizing*
	assed film	*popcorn film*	*movie work*	*tasty*	*stuck*
	nonexistent	*numbers*	*work better*	*spice*	*graet films*
	week	*away*	*like igby*	*diversion*	*tremedous piece*
	dull	*felt*	*igby*	*odds*	*leads*
	hours	_	*happy*	*moved*	*thoughtful*
	bad	_	*doing*	*answers*	_
	_	_	*flawed*	*simone*	_
	_	_	_	*perfectly*	_
	_	_	_	*enjoyable*	_
	_	_	_	*quirky*	_
	_	_	_	*step*	_
	_	_	_	*heart*	_
	_	_	_	*french*	_
	_	_	_	*works*	_
	_	_	_	*actor*	_
	_	_	_	*leave*	_

Table 11: Final set of rules generated by Snuba for SST dataset. We display few selected rules out of total 70 rules.

Filtering	GC				
Class	Negative	Somewhat negative	Neutral	Somewhat positive	Positive
Rules	*worse*	*like it*	*always*	*beautiful*	*theme*
	week	*getting*	*at time*	*occasionaly*	*epic*
	just	*no*	*appealing*	*tale of*	*not be*
	imagine	*by it*	*work better*	*psychological*	*excellent*
	should have	_	_	*works*	*music*
	_	_	_	*rock*	_
	_	_	_	*ernest*	_
	_	_	_	*the actor*	_
	_	_	_	*story that*	_
	_	_	_	*during*	_

Table 12: Final set of rules generated by FAIR-GC for SST dataset over Snuba generated candidate rule set. We display few selected rules out of total 70 rules.

Filtering	Snuba				
Class	Negative	Somewhat negative	Neutral	Somewhat positive	Positive
Rules	*bad*	*too*	*see the*	*who*	*comedy with*
	week	*even*	*slow*	*compelling*	*music*
	just	*where*	_	*both*	_
	_	*only*	_	.	_
	_	*no*	_	*era*	_
	_	*better*	_	*there are*	_
	_	*out*	_	*his own*	_
	_	*book*	_	_	_
	_	*it*	_	_	_
	_	*away*	_	_	_
	_	*but ultimately*	_	_	_
	_	*movie to*	_	_	_
	_	*the same*	_	_	_
	_	*sit*	_	_	_
	_	*of an*	_	_	_
	_	*all it*	_	_	_
	_	*try*	_	_	_
	_	*next*	_	_	_