ReGAL: Rule-Generative Active Learning for Model-in-the-Loop Weak Supervision

David Kartchner, Wendi Ren, Davi Nakajima An, Chao Zhang, Cassie Mitchell
Georgia Institute of Technology
Atlanta, Georgia, USA
{david.kartchner, wren44, dna, chaozhang}@gatech.edu
cassie.mitchell@bme.gatech.edu

Abstract

One of the main bottlenecks to extending deep learning systems to new domains is the prohibitive cost of acquiring sufficient training labels. While many previous works have sought to alleviate this problem with weak supervision and data programming, rule and label noise prevent them from approaching fully-supervised performance. This work-in-progress provides a principled, AI-guided approach to improve rule-based and weakly supervised text classification by performing active learning not on individual data instances, but on entire labeling functions. We argue that such a framework can guide users and subject matter experts to select labeling rules that expand label function coverage without sacrificing clarity. Our experiments show that our framework, ReGAL, is able to generate coherent labeling rules while simultaneously obtaining state-of-the-art performance in weakly supervised text classification.

1 Introduction

One of the primary criticisms that has been leveled against deep learning is the high volume of data needed to produce models that generalize well and avoid overfitting. Numerous approaches have been proposed to solve this problem, including transfer learning [1], active learning [2, 3], bootstrapping [4], distant supervision [5], rule-based supervision [6, 7], and crowd-sourced labeling [8].

While this task is present across all domains of natural language processing (NLP), it is significantly more difficult in the highly technical and low resource domains where labels must be curated by subject matter experts (SMEs). Low-resource languages often have a very small pool of potential annotators while technical domains such as scholarly articles contain such a wide breadth of information in literature means that a single SME cannot annotate a single corpus by himself or herself. These factors, combined with the high cost of SME expertise, motivates the development of systems that maximize both the speed and information content of annotation, thus allowing annotators to impart more valuable data in shorter time.

In this work, we develop a model called ReGAL for Rule-Generative Active Learning which accelerates data labeling by interactively soliciting human feedback on labeling functions instead of individual data points. This framework, depicted in Figure 1, allows our active learner to seek feedback on model weaknesses while simultaneously labeling large swaths of examples.
Figure 1. ReGAL model setup. ReGAL takes unlabeled documents and seed rules as input. It then
iteratively proposes new labeling

2 Preliminaries

2.1 Problem Formulation

We assume that we are given a set of documents \( \mathcal{D} = \{d_1, d_2, \ldots, d_{|\mathcal{D}|}\} \), each of which has a (possibly unknown) classification label \( c_i \in \mathcal{C} \). Each document \( d_i = [v_{i,1}, v_{i,2}, \ldots, v_{i,T}] \) represents a sequence of tokens from the vocabulary \( \mathcal{V} \), where tokens drawn from \( \mathcal{V} \) could be words, subwords, characters, etc.

We assume that we do not have access to ground-truth labels for the documents in the training set but that we have a small number of heuristic labeling functions (LFs; i.e. labeling rules) \( \mathcal{R} = \{r_1, r_2, \ldots, r_l\} \), where each \( r_j : \mathcal{D} \rightarrow \mathcal{C} \cup \{c_{\text{abstain}}\} \) is a function that maps documents to a class label in \( \mathcal{C} \) or abstains from labeling. This set of LFs induces a vector of noisy labels for each document, denoted \( \ell_i = [r_1(d_i), r_2(d_i), \ldots, r_l(d_i)]^T \). Because LFs act as rule-based labelers, we freely interchange the terms “labeling function” and “rule” throughout the paper.

2.2 Challenges

Weakly supervised text classification presents three main challenges: label noise, label incompleteness, and annotator effort.

Label Noise  Label noise is the problem of labeling functions generating incorrect labels for particular data instances. This problem generally occurs when a specified labeling function is too general and thus mislabels into the wrong class. The diversity of language presents an extremely large space of possible misapplications for a single labeling function, and enumerating them can be prohibitively expensive.

A number of recent works have sought to address this problem via numerous means, including generative label denoising \([6,9]\), self-training on synthetic examples generated with latent variable models \([10]\), using a neural network to identify improper applications of labeling functions using labeled rule exemplars \([11]\), and active learning on instances with conflicting labels \([12]\). While some of these methods have shown promising results, there is a distinct need for better methods of automatically solving labeling function conflicts. Our proposed methodology seeks to reduce label noise by automatically learning rules designed to differentiate between separate classes.

Label Incompleteness  A complementary problem to label noise is label incompleteness. Label incompleteness is the insufficiency of labeling functions to assign labels to particular slices of a dataset. This occurs when the syntactic and semantic patterns in a subset of examples do not lie within the scope of the given labeling functions. Label incompleteness is particularly pervasive in the
long tails of a dataset, which may contain more informative data for better model performance and
generalization. This can be especially manifested in low-resource or highly technical domains where
differences in nomenclature could lead to large labeling gaps.

Approaches to tackle this problem include differentiable soft-matching of labeling rules to unlabeled
instances [13], automatic rule generation using pre-specified rule patterns [13, 15], co-training a
rule-based labeling module with a deep learning module capable of matching unlabeled instances
[16, 10], and encouraging LF diversity by interactively soliciting LFs for unlabeled instances [12].

**Annotator Effort**  Many domains such as biomedical or legal NLP require substantial subject
matter expertise to annotate correctly. However, SMEs are costly and have limited time to for
annotation. This problem is strikingly evident when one compares the size of labeled datasets in
biomedicine [17, 18, 1] with those for general domain tasks. For example, the average training set in
the general domain GLUE benchmark [19] is over 25x larger than that of the average dataset in the
contemporaneous BLUE benchmark [1] for biomedical NLP. Accordingly, it is important to solicit
the most useful possible SME feedback to maximize label signal while minimizing annotator time
and effort. By presenting annotators with candidate labeling rules, ReGAL reduces the time necessary
to specify rules by hand and thus offers large potential gains to annotator efficiency.

2.3 Objectives

This paper develops a model, ReGAL, that addresses these challenges by automatically proposing
labeling rules designed to (1) disambiguate instances with conflicting LF-induced labels and (2)
extend coverage to unlabeled portions of the dataset. This is done interactively as annotators generate
labels, allowing ReGAL to adapt to new labeling needs as the set of labels expands.

3 Related Work

3.1 Data programming

Our work is closely related with recent works surrounding data programming and rule-based text
classification. One of the seminal works in this category proposed Snorkel [20, 6], which introduced
a rule-based labeling model where multiple, overlapping rules are proposed for each class and then are
denoised with a generative model to account for the inter-rule correlations. A number of alternative
denoising approaches have since been proposed to improve the quality of aggregated labels. Among
these, Ren et al. [16] propose a self-training discriminative denoising module, while Awasthi et al.
[11] propose learning appropriate bounds on rule coverage from a small labeled set of data. ReGAL
diffsers from this work by providing model-based assistance to define high-quality rules and augment
human knowledge with global data patterns in addition to providing a module to denoise labeling
rules.

Follow-up studies [12] on Snorkel have also shown that Snorkel users tended to select rules by
looking at individual labeled instances and seeking to create rules reflecting the patterns they observe
in the data. This study further demonstrated that label efficiency can be improved by showing users
currently unlabeled instances and instances with high label conflict, i.e. seeking labels in highly
valuable areas of the training data. ReGAL provides the possibility to extend this by automatically
proposing rules suited to conflicted or unlabeled data slices, thus reducing the manual effort involved.

3.2 Rule Learning

Another key area of related work is rule learning, in which models seek to automatically label datasets
by generating their own label rules. MetaPad [21] mined and consolidated information extraction
patterns from large text corpora to improve the rule-based extraction of specific types of named
entities and their attributes from raw text. REPEL [15], on the other hand, focused specifically on
rule-guided relation extraction. The authors began with small sets of labeled seed instances and
iteratively refined a set of potential candidate rules based on keywords in the shortest dependency
path between entities. While their approach has many similarities to our own, it is limited in that it
can only produce a single type of rule and does not allow for iterative refinement via human guidance.
Snuba [14] generates candidate rules using a collection of weak learner primitives (e.g. decision
stumps, k-nearest neighbors) and then synthesizes and prunes this set of rules to generate final rules for labeling. ReGAL combines elements of this framework to allow iterative communication between downstream classifiers and rule selectors to allow each to mutually enhance the other. ReGAL differs from these by soliciting user feedback on rules, thus enabling it complement model-generated LFs with annotator guidance.

4 Methods

Here we describe ReGAL, a model that interactively generates labeling functions via repeated rounds of user feedback. This model is composed of three modules: (1) A contextualized word and document encoder that produces global document embeddings and contextualized word embeddings; (2) a rule denoising module that aggregates labeling functions to produce coherent labels for matched and unmatched documents; and (3) a rule proposal module that generates candidate labeling functions. A depiction of our model is shown in Figure 2.

4.1 Contextualized Token Encoder

ReGAL begins with a token encoder. The purpose of the encoder is to produce expressive, contextualized token embeddings containing the semantic content necessary to identify potential keywords in generated LFs. These embeddings will also be aggregated by the rule denoiser into a cohesive document representation.

We use transformers [22] to construct our encoder because they allow each token to draw contextual information from every other token in its same input document. Specifically, we use a pretrained BERT-base [23] model as our encoder. We use the outputs \( h_{i,t} \) of the last layer as token embeddings:

\[
\left[ h_{i,1}, \ldots, h_{i,T} \right] = \text{enc}(\left[ v_{i,1}, \ldots, v_{i,T} \right])
\]

We will henceforth let \( H_i = \left[ h_{i,1}, \ldots, h_{i,T} \right] \) represent the sequence token embeddings from document \( d_i \).

In addition to initializing this module with a BERT-base, we encourage the encoder to learn contextual information about rules. We do so using masked language modeling (MLM), where our masking budget consists of all tokens used in keyword LFs as well as a random 10% of tokens from the sequence. We either mask or noise each token according to the strategy in [23] and require our
model to predict the correct token in each case. We optimize this using cross entropy loss over the masked/noised tokens, which we denote $\mathcal{L}_{MLM}$.

4.2 Rule Denoiser

We develop a rule denoiser module similar to [16] that allows our model to learn document-level applicability of matched LFs. This is co-trained with a neural prediction module that extends coverage to unmatched documents based on their learned semantic properties. The rule denoiser also learns class-specific token importance scores that will be used to generate new LFs. The rule denoiser has two submodules which we discuss in detail: a **token attention** module and a **rule attention** module.

4.2.1 Token attention

The purpose of the TokenAttention submodule is to learn the token and document level information necessary to generate expressive, class-specific labeling functions. Specifically, TokenAttention computes class-specific attention scores on each token, and the document’s probability of belonging to each class. These class probabilities also serve as suitability scores of how well-equipped a document is to generate LF keywords for a given class.

TokenAttention takes as inputs the token embeddings from the document encoder and produces class-specific token attention $a^{(c)}_{i,t}$, document embeddings $e_i$, and document-level class probabilities $p_i = [p_i^{(1)}, \ldots, p_i^{(C)}]$. We describe how each of these is computed as follows.

We begin by calculating class-specific attention scores for each token in our document. These will be used by our rule proposal network to generate new labeling rules. They are calculated as

$$a_{i,t} = W_{a}^2 \tanh \left( W_{a}^1 H_i \right)$$

where $W_{a}^1 \in \mathcal{R}^{m_2 \times m_1}$ and $W_{a}^2 \in \mathcal{R}^{c \times m_2}$. These scores are then used to calculate a class-specific document representation

$$\tilde{e}_i^{(c)} = \sum_{t=1}^{T} a_{i,t}^{(c)} h_i^{t}$$

These are in turn aggregated into an overall document representation with class weights $\eta_c$

$$e_i = \sum_{c=1}^{C} \eta_c \tilde{e}_i^{(c)}$$

This representation will be used by the rule attention submodule to estimate conditional LF reliability.

The class-specific embeddings $\tilde{e}_i^{(c)}$ are also used to compute the document’s class probabilities:

$$p_i = \text{softmax} \left( [\tilde{p}_i^{(1)}, \ldots, \tilde{p}_i^{(C)}] \right)$$

where $p_i^{(c)} = w_p^{(c)} \tilde{e}_i^{(c)}$ and $w_p^{(c)}$ is a weight vector corresponding to each class. In addition to serving as this submodule’s prediction of the document’s label, these probabilities can additionally serve as measures of the document’s suitability to generate keywords to be used in future LFs.

4.2.2 Rule attention

The RuleAttention submodule learns measures of labeling function validity conditioned on the encoded document representation $e_i$. It outputs document-specific LF reliability scores as well as soft labels to be used in co-training our neural classifier. We first learn conditional LF reliability scores $z_i$ for document $d_i$ as

$$z_i = \text{softmax} \left( W_{r}^2 \tanh \left( W_{r}^1 e_i \right) \right)$$

We weight our masked LFs by these reliability scores using a masked elementwise product $s_i = z_i \odot \ell_i$. This produces probabilistic class labels $\hat{y}_i$ for each labeled document. We use these soft labels and
the class probabilities \( p_i \) using probabilistic cross entropy loss:

\[
\mathcal{L}_{TOK} = - \sum_{c=1}^{C} y_i^{(c)} \log(p_i^{(c)})
\]

### 4.3 Rule Proposal Network

ReGAL’s rule proposal network (RPN) enables ReGAL to generate new rules given a set of seed rules. The RPN takes as inputs both the class-conditioned word level attention \( \alpha_{c,i,t} \) and document-level class probabilities \( p_i \) and outputs a score \( \tau_j^{(c)} \) for each \( v_j \in \mathcal{V} \) corresponding to how strongly \( v_j \) represents class \( c \). These scores are calculated as:

\[
\tau_j^{(c)} = \frac{1}{|v_j|} \sum_{i=1}^{\mathcal{|D|}} \sum_{t=1}^{T} 1_{v_{i,t} = v_j} p_i^{(c)} \alpha_{i,t}^{(c)}
\]

Here, \( \gamma \in [0, 1] \) is a parameter that controls how much our ReGAL’s RPN balances between the coverage of a token (i.e. how often it occurs) and its instance level importance. Low values of \( \gamma \) favor tokens with high coverage while high values of \( \gamma \) favor rules with high impact tokens without regard for coverage. Since the types of rules needed may differ as training progresses, we allow users to choose \( \gamma \) for each round of proposed rules. In practice, we find that \( \gamma \in [0.5, 0.9] \) tend to produce good rules.

After calculating token scores \( \tau_j^{(c)} \), the RPN proposes up to \( k \) new LFs for each class by choosing the top \( k \) scoring tokens \( \{v_1^{(c)},...,v_k^{(c)}\} \) that are not highly scoring for any other class \( c' \). These tokens each induce a labeling function of the form \( \text{HAS}(x, v_i^{(c)}) \rightarrow c \), where we assign the class label \( c \) to a text \( x \) if it contains the token \( v_i^{(c)} \). These proposed LFs are then presented to a human annotator, who can choose to accept or reject them.

### 4.4 Model optimization

We optimize our entire model by minimizing the unweighted sum of the loss functions of its components:

\[
\mathcal{L} = \mathcal{L}_{MLM} + \mathcal{L}_{TOK}
\]

### 5 Experiments and Discussion

We test the ability of ReGAL to generate meaningful, diverse rules to assist humans in efficient data annotation. In doing so, we follow many aspects of the experimental setup of [16] and specifically demonstrate how our model can be used to add to their carefully curated rule-based labeling LFs.

We note that this a work is in progress and thus models have not undergone hyperparameter optimization. However, even without this optimization we demonstrate that ReGAL is both able to generate coherent rules and adeptly denoise rule-induced labels, making it a powerful tool to assist annotators in generating LFs and disambiguate conflicted labels.

#### 5.1 Datasets and tasks

We evaluate our model’s performance on two well-known sentiment classification datasets and a topic classification dataset.

- **Yelp** is a collection of Yelp restaurant reviews classified according to their sentiment [24]
- **IMDB** is a set of movie reviews classified according to sentiment [25]
- **AGnews** is a news corpus for topic classification with four classes: sports, technology, politics, and business [26].

Basic summary statistics on our data are found in Table 1.
Table 1. Summary of Regal data and rule generation parameters. Update Iters refers to the number of training iterations between rounds of rule generation and Pool Proportion is the proportion of training instances sampled to generate new rules.

<table>
<thead>
<tr>
<th>Dataset</th>
<th># Classes</th>
<th>Train/Valid/Test</th>
<th>Update Iters</th>
<th>Pool Proportion</th>
<th>Rules per Update</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yelp</td>
<td>2</td>
<td>30.4k/7.8k/7.8k</td>
<td>100</td>
<td>1.0</td>
<td>20</td>
</tr>
<tr>
<td>IMDB</td>
<td>2</td>
<td>20k/2.5k/2.5k</td>
<td>100</td>
<td>1.0</td>
<td>20</td>
</tr>
<tr>
<td>AGNews</td>
<td>4</td>
<td>96k/12k/12k</td>
<td>250</td>
<td>0.167</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 2. Seed rules on text classification datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Rule</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yelp</td>
<td>HAS(x, ['best', 'excellent', 'awesome', 'pleasant', 'wonderful', 'amazing'])</td>
<td>POS</td>
</tr>
<tr>
<td>Yelp</td>
<td>HAS(x, ['bad', 'worst', 'horrible', 'awful', 'terrible', 'nasty'])</td>
<td>NEG</td>
</tr>
<tr>
<td>IMDB</td>
<td>HAS(x, ['masterpiece', 'excellent', 'awesome', 'fabulous', 'wonderful', 'amazing'])</td>
<td>POS</td>
</tr>
<tr>
<td>IMDB</td>
<td>HAS(x, ['bad', 'worst', 'horrible', 'awful', 'terrible', 'garbage'])</td>
<td>NEG</td>
</tr>
<tr>
<td>AG News</td>
<td>HAS(x, ['war', 'prime minister', 'president', 'commander', 'military', 'militant'])</td>
<td>Politics</td>
</tr>
<tr>
<td>AG News</td>
<td>HAS(x, ['baseball', 'basketball', 'soccer', 'football', 'world cup', 'olympics'])</td>
<td>Sports</td>
</tr>
<tr>
<td>AG News</td>
<td>HAS(x, ['sales', 'stock', 'market', 'shareholder', 'money', 'business'])</td>
<td>Business</td>
</tr>
<tr>
<td>AG News</td>
<td>HAS(x, ['tech', 'engineering', 'scientist', 'processor', 'cpu', 'compute'])</td>
<td>Technology</td>
</tr>
</tbody>
</table>

5.2 Baselines

We compare the predictive performance of our model to three strong baselines for text classification with weak supervision:

- **Majority voting** of the labeling functions, where ties are broken by choosing the most common class as specified by the labeling functions.
- **BERT** [23] is a transformer-based language model that has shown exceptional performance on a wide-range of NLP tasks. We train BERT with the majority vote labels from our LFs.
- **WeSTClass** [15] is a weakly supervised text classification that trains a classifier using documents generated using user-provided labeling rules.
- **Epoxy** [27] is a recent weak supervision paradigm that uses combined pretrained embeddings with anchored weakly-labeled examples to enable interactive model training.
- **MSWS** [16] is a denoising method for multi-source weak supervision that co-trains a rule denoiser with a neural classifier to learn optimal weightings for rules and label unmatched samples.
- **Fully Supervised BERT** We train a BERT model with a fully connected layer to provide baseline performance for a model trained with ground-truth labels.

5.3 Rule Denoising

Parsing signal from noise is critical to learning from weak and rule-based supervision. Accordingly, we compare ReGAL’s ability to that of our baselines in accurately classifying instances based on a set of seed rules, which are shown in Table 2. We provided a single seed rule for each class of Yelp and IMDB and six single-word LFs for each class of AG-News, where LFs contain keywords or phrases adapted from [16]. If any of these keywords is found in document $d_i$, the LF assigns its label; otherwise, it abstains from labeling $d_i$.

After providing seed rules, we trained ReGAL with learning rate 0.001 and batch size 64, generating new rules every 100 batches for Yelp and IMDB and every 250 batches for AGNews. We continued
Table 3. Accuracy of models with and without generated rules. All reported figures are percentages.

<table>
<thead>
<tr>
<th>Model</th>
<th>Seed</th>
<th>Seed + ReGAL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Yelp</td>
<td>IMDB</td>
</tr>
<tr>
<td>Majority voting</td>
<td>50.23</td>
<td>55.12</td>
</tr>
<tr>
<td>WeSTClass</td>
<td>76.90</td>
<td>71.87</td>
</tr>
<tr>
<td>Epoxy</td>
<td>73.03</td>
<td>74.00</td>
</tr>
<tr>
<td>BERT</td>
<td>79.08</td>
<td>72.76</td>
</tr>
<tr>
<td>MSWS</td>
<td>80.79</td>
<td>76.48</td>
</tr>
<tr>
<td>ReGAL</td>
<td>85.21</td>
<td>78.40</td>
</tr>
<tr>
<td>Fully Supervised BERT</td>
<td>91.1</td>
<td>90.7</td>
</tr>
</tbody>
</table>

Table 4. Proposed rules on Yelp dataset. ‘##’ indicates that a token is a subword unit that does not begin a word.

<table>
<thead>
<tr>
<th>Class</th>
<th>Rule</th>
<th>Accepted</th>
<th>Rule</th>
<th>Accepted</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEG</td>
<td>HAS(x, 'rude')</td>
<td>Y</td>
<td>HAS(x, 'positive')</td>
<td>N</td>
</tr>
<tr>
<td>NEG</td>
<td>HAS(x, 'poor')</td>
<td>Y</td>
<td>HAS(x, 'fine')</td>
<td>N</td>
</tr>
<tr>
<td>NEG</td>
<td>HAS(x, 'slow')</td>
<td>Y</td>
<td>HAS(x, 'decent')</td>
<td>N</td>
</tr>
<tr>
<td>POS</td>
<td>HAS(x, 'delicious')</td>
<td>Y</td>
<td>HAS(x, '##he')</td>
<td>N</td>
</tr>
<tr>
<td>POS</td>
<td>HAS(x, 'great')</td>
<td>Y</td>
<td>HAS(x, 'vegas')</td>
<td>N</td>
</tr>
<tr>
<td>POS</td>
<td>HAS(x, 'friendly')</td>
<td>Y</td>
<td>HAS(x, 'nt')</td>
<td>N</td>
</tr>
</tbody>
</table>

iterating until we saw multiple consecutive rounds with for which at least one class proposed no high-quality rules. We calculate rule proposal weights by aggregating word attention over the entire dataset for Yelp and IMDB and over a randomly sampled 1/6 of AGNews. A summary of important hyperparameter choices is given in Table 1. Proposed rules were adjudicated by university students to assess their utility and relevance.

One factor that majorly differentiates ReGAL from our baselines is its ability to interactively generate labeling functions to expand label coverage. This potentially gives ReGAL an advantage over our baselines which cannot generate new labeling rules. Accordingly, we show the performance of each of these models using (1) only seed LFs and (2) seed + generated LFs from ReGAL. The results are shown in Table 3. When rule generation persists for many rounds (as is the case with AGNews), ReGAL’s underlying model sometimes overfits to previously generated rules which results in performance degradation. We accordingly retrain the AGNews rule denoising module with probabilistic labels after solidifying labeling rules, which converges in just 250 batches.

Interestingly, although ReGAL generates coherent and intuitive rules (see section 5.4), these rules actually degrade ReGAL’s performance. This is particularly surprising considering that the added rules substantially boost the performance of each of the baselines. We suspect that this may be due to the masking of rule keywords in the encoder, which could inhibit the model from learning their signal. We will investigate this anomaly further as we continue to develop the model.

5.4 Evaluating Generated Rules

We evaluate the ability of our model to identify promising rules by providing it a small number of seed rules and qualitatively evaluating its ability to discover additional high-quality rules. In Table 4, we display the top 3 generated rules from the first two iterations of ReGAL for Yelp and in Table 5, we show rules from iterations 1, 2, and 5 on AG News.

From these results, one can see that on the first iteration, ReGAL generates a very high proportion of highly informative rules. For example, ReGAL’s first set of rules touches on multiple diverse aspects of a restaurant’s quality, including food quality, speed, and friendliness of staff. AGNews shows similar patterns, picking up on elections, stock trading, soccer leagues, and tech companies in the first iteration of rules.

Unfortunately, rule quality appears to degrade sharply after the first iteration. After a 90% acceptance rate in the first round of generation on Yelp, ReGAL failed to generate a single accepted rule in round 2. Rules generated on AGNews were slightly more promising, with coherent rules persisting into later rounds, though there is clear quality degradation for specific classes. One clear instance of this
Table 5. Proposed rules on AG News dataset. ‘##’ indicates that a token is a subword unit that does not begin a word. We omit the function notation \( \text{HAS}(x, \text{‘word’}) \) for brevity.

<table>
<thead>
<tr>
<th>Class</th>
<th>Rule</th>
<th>Accept</th>
<th>Rule</th>
<th>Accept</th>
<th>Rule</th>
<th>Accept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iteration 1</td>
<td></td>
<td></td>
<td>Iteration 2</td>
<td></td>
<td>Iteration 5</td>
<td></td>
</tr>
<tr>
<td>Politics</td>
<td>‘election’</td>
<td>Y</td>
<td>‘iraq’</td>
<td>Y</td>
<td>‘warming’</td>
<td>N</td>
</tr>
<tr>
<td>Politics</td>
<td>‘bush’</td>
<td>Y</td>
<td>‘iraqi’</td>
<td>Y</td>
<td>‘warm’</td>
<td>N</td>
</tr>
<tr>
<td>Politics</td>
<td>‘warned’</td>
<td>N</td>
<td>‘warning’</td>
<td>N</td>
<td>‘climate’</td>
<td>N</td>
</tr>
<tr>
<td>Sports</td>
<td>‘uefa’</td>
<td>N</td>
<td>‘coach’</td>
<td>N</td>
<td>‘hockey’</td>
<td>Y</td>
</tr>
<tr>
<td>Sports</td>
<td>‘league’</td>
<td>N</td>
<td>‘sports’</td>
<td>N</td>
<td>‘team’</td>
<td>Y</td>
</tr>
<tr>
<td>Sports</td>
<td>‘olympic’</td>
<td>Y</td>
<td>‘season’</td>
<td>N</td>
<td>‘night’</td>
<td>N</td>
</tr>
<tr>
<td>Business</td>
<td>‘shares’</td>
<td>Y</td>
<td>‘crude’</td>
<td>Y</td>
<td>‘futures’</td>
<td>Y</td>
</tr>
<tr>
<td>Business</td>
<td>‘prices’</td>
<td>Y</td>
<td>‘investors’</td>
<td>Y</td>
<td>‘economy’</td>
<td>Y</td>
</tr>
<tr>
<td>Business</td>
<td>‘oil’</td>
<td>Y</td>
<td>‘company’</td>
<td>Y</td>
<td>‘trading’</td>
<td>Y</td>
</tr>
<tr>
<td>Tech</td>
<td>‘apple’</td>
<td>Y</td>
<td>‘space’</td>
<td>N</td>
<td>‘shuttle’</td>
<td>Y</td>
</tr>
<tr>
<td>Tech</td>
<td>‘intel’</td>
<td>Y</td>
<td>‘microsoft’</td>
<td>N</td>
<td>‘enrichment’</td>
<td>N</td>
</tr>
<tr>
<td>Tech</td>
<td>‘internet’</td>
<td>Y</td>
<td>‘security’</td>
<td>N</td>
<td>‘spy’</td>
<td>N</td>
</tr>
</tbody>
</table>

is with the persistent proposal of rules starting with ‘war’ (one of the seed rules), which demonstrates overfitting to the keyword-based labeling functions. A similar pattern can be seen with nuclear arms development keywords (e.g. ‘enrichment’ in iteration 5) caused by the acceptance of ‘nuclear’ in the tech category at iteration 3.

Because ReGAL is an active learning framework, users are able to evaluate proposed rules and control how the model expands its labeling space. This makes ReGAL resilient to spurious generated rules and allows users to use it to quickly and meaningfully expand their rule space.

6 Future Work

As we continue to develop ReGAL, we intend to make improvements to both our model and evaluation. First, we will add a more diverse set of baselines and evaluation datasets that allow us to compare ReGAL to both traditional active learning frameworks as well as interactive data programming paradigms [12]. These, in turn, will be tested on a more diverse array of text classification tasks, especially those requiring domain expertise (e.g. biomedical). Second, we will expand ReGAL’s rule generating capacity to encompass multi-token sequences (e.g. for keywords that are split into multiple tokens) or non-contiguous textual patterns. Finally, we will conduct user tests to determine and optimize the quality of human interactions to make ReGAL more user-friendly and accurate.

One promising area for future work is adapting ReGAL to work with crowdsourced labeling. In such a framework, LFs could be adopted only after agreement from multiple annotators, which would lead to higher overall rule quality. We foresee that this could naturally lead to a two-stage crowdsourcing framework in which earlier annotators would focus on broadening coverage with new LFs while subsequent annotators could with higher LF uncertainty (e.g. due to conflicting LF matches). This would increase crowdsourcing efficiency by focusing annotators on providing instance-level supervision on difficult samples while still producing rule-induced labels on the remainder of the data.

7 Conclusion

In this work, we presented our work-in-progress ReGAL, a text classification model capable of both denoising multiple sources of noisy supervision and generating high-quality labeling functions capable of inductively labeling uncovered portions of dataset. In doing so, ReGAL offers the potential to substantially reduce human labeling efforts by helping them identify quality labeling rules they would otherwise miss. We demonstrate ReGAL’s utility by showing that it both generates coherent, reasonable rules and also that using the selected labeling rules improves the performance of each of our baseline models.
Broader Impact

This work has major potential to accelerate advancements in low-resource NLP domains by assisting subject matter experts in discovering patterns capable of quickly labeling their data. This would allow them to tackle more problems for which they previously lacked data while simultaneously reducing the cost of doing so.

One could observe similar benefits in NLP for scientific and biomedical texts. The breadth of even a single scientific discipline is so broad that a single SME may miss key semantic indicators that are not associated with his/her subfield. Thus, automatic generation of labeling rules can potentially increase the diversity of scientific and biomedical NLP tasks and thus accelerate the discovery of new insights derived from text mining.

There are two potential negative impacts of this work, both of which hinge on the cognitive biases of humans involved in data annotation. First, our model exploits patterns in seed rules for generating new labeling functions, which may lead it to miss important labeling signals that are independent from the given seed LFs. Second, the reduction of human effort in creating LFs may lead to annotators accepting proposed rule without sufficiently rigorous evaluation, thus degrading LF quality.

Acknowledgments and Disclosure of Funding

This work was supported by National Science Foundation CAREER grant 1944247, National Institute of Health grant R21-CA232249, Whitaker Biomedical Engineering Seed Grant for COVID-19, Alzheimer's Association Research Grant Award, Childrens Hospital of Atlanta Aflac Pilot Grant Award, National Science Foundation award III-2008334, an Amazon Faculty Award, and a Google Faculty Award.

References


