Probable Human Knowledge Elicitation for Efficient Human-in-the-Loop Text Classification without Any Label-Related Data

Osamu Saisho
NTT / Tokyo, Japan
osamu.saisho.vm@hco.ntt.co.jp

Abstract

This paper presents a new human-in-the-loop text classification framework with no labeled data incorporating weak supervision and Bayesian active learning with a semantic clustering constraint. Labeling function (LF)-based weak supervision is a promising method to alleviate the huge labeling cost in supervised learning. However, most previous studies relied on the impractical assumption of sufficient and well-designed pre-built LFs, even though they also require a heavy workload to implement. The proposed framework is the first intended to substantially reduce this workload by a human-in-the-loop approach from scratch. Specifically, it interactively suggests data samples to help humans implement an LF considering the actual human workload. Its superiority is achieved by Bayesian active learning with a semantic clustering constraint for not only enhancing labeling utility but also reducing human workload. The results of the user studies with 16 volunteers demonstrate that the proposed method can achieve higher performance with fewer LFs and help humans implement more accurate LFs in a shorter work time than conventional methods.

1 Introduction

Supervised learning assuming the availability of massive labeled data has been put into practical use, but this assumption has also been the main obstacle to expanding its use. To overcome this obstacle, weak supervision (WS) has gained much attention as a promising method. It is incomplete, inexact, or inaccurate supervision as an alternative to strong supervision such as a fully labeled dataset (Zhou, 2017). Specifically, labeling function (LF) is a representative approach used in both academia and industry (Bach et al., 2019; Fries et al., 2019). LF is a human-defined function based on heuristics, collective knowledge, or implicit rules for indirect labeling instead of manual direct labeling. Its main strengths are its high performance and robustness to task changes (Ratner et al., 2016; Bach et al., 2017).

Even though LF-based WS alleviates the huge and redundant labeling cost, it usually depends on sufficient and well-designed pre-built rules and meta-rules for LFs. When implementing LFs in the wild from scratch, the difference in the required workload and the potential benefits on each LF must be considered. Thus, there are two issues left to be solved simultaneously for the practical LF-based WS framework. First, only high-performance LFs should be implemented. Second, humans should be able to implement more accurate LFs with a lighter workload, that is, in a shorter work time.

The most appropriate and pragmatic approach to solve both issues is a human-in-the-loop approach. Especially, active learning (AL) (Settles, 2009) is a key method to solve the first issue. A few recent studies have attempted to develop an AL-like framework for WS (Wang et al., 2019; Saisho et al., 2021). These methods try implementing effective LFs interactively, but they disregard the workload for each LF. In addition, their effectiveness has been verified only in simulation settings under unrealistic assumptions such as "selecting" the best one among the "pre-defined" candidates for each loop. From the

viewpoint of practical WS in the wild from scratch, the superiority in LF should be determined on the basis of the work time to design it, the voting coverage of the applicable data samples, and the accuracy of the voting, other than the labeling utility in AL.

The research objective is to reduce the substantial workload of humans while achieving high performance in the practical WS framework in the wild from scratch. The main contribution of this paper is the proposal and verification of a practical human-in-the-loop WS framework by Bayesian AL with a semantic clustering constraint. In each loop, this framework suggests data samples that not only have high utility for labeling but also elicit probable human knowledge to help humans evoke common labeling rules for a new LF with a lighter workload. Especially, the data samples are suggested on the basis of utility scores calculated by batchBALD acquisition function under the constraints of the batch being formed by data samples belonging to an identical semantic cluster. This constraint is the key to filling the gap in the practice of a human-in-the-loop WS. The results of the user studies with 16 volunteers verified the high performance of the classifier trained by using a small number of LFs, the short work time required to implement LFs, and the high accuracy of the LFs' votings. As an adjunct to the above contribution, this paper also shows an empirical validation method in human-in-the-loop WS that uses LFs as a substitute for ground truth, without any additional workload or implicit label leakage.

2 Related work

Towards the practical LF-based WS, the reduction of LF implementation cost has gained considerable attention in the past few years. Some researchers have proposed fully automated methods by transfer learning and few-shot learning approaches (Varma and Ré, 2018; Das et al., 2020). Their insights are significant, but they have also eliminated positive aspects of WS in practical use. They incur the uninterpretability because of the dependency on the implicit domain similarity and the distribution of the few-shot data (Raghu et al., 2019; Chen et al., 2019). On the other hand, a human-in-the-loop approach will work well with LF-based WS because of their analogous advantages if humans can implement LFs from their knowledge with light workload. Transparency and fairness are the primary reasons that human-in-the-loop is attracting attention (Holzinger, 2016; Li, 2017; Lertvittayakumjorn et al., 2020). Especially, AL is the representative human-in-the-loop approach toward reducing labeling costs.

Existing methods to combine WS and AL can be categorized into four types. The first type is "AL with WS" (Nashaat et al., 2018; Nashaat et al., 2020; Biegel et al., 2021). This type trains a classifier using "pre-built" WS and then improves its classification performance by direct labeling in AL. The second type is "WS with AL" (Qian et al., 2020; Gonsior et al., 2020). This type attaches labels directly by AL and augments the labeled dataset by "pre-built" WS. These two types can lighten somewhat humans' tasks in the conventional AL, but they cannot change the redundancy of the tasks. The third type is "AL for automated WS" (Kartchner et al., 2020; Boecking et al., 2021). This type is an AL framework that automatically generates WS on the basis of "pre-built" meta patterns and asks humans to judge whether to accept them or not. It can eliminate the redundant labeling cost, but their performance depends on the pre-built resources such as seed rules and structure information as the automated WS above. The most significant problem common to these three types in practice is that they all still depend on the existence of unrealistic pre-built resources. Even though they are human-in-the-loop approaches, preparing such well-designed meta resources without any support before the loops is difficult and unrealistic in the wild.

The last type is "AL for WS from scratch" (Wang et al., 2019; Saisho et al., 2021). This type is an AL framework that helps humans implement WS by suggesting a few data samples that have the highest labeling utility. Only this type is pragmatic in the wild because it does not require any presupposed resources and humans can implement WS in any format. It can reduce the number of LFs to be manually implemented, but this is still not enough for practical use in the wild. It is not human-friendly because it does not consider any characteristic of human-defined LFs. First, each LF requires a very different amount of workload to implement. Furthermore, both the coverage and the accuracy of LFs need to be enhanced in the case of WS, where there is no oracle

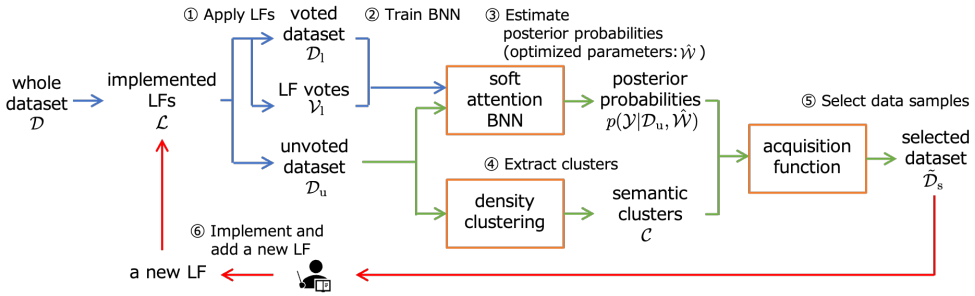

Figure 1: Overall framework: a few data samples are selected by batchBALD under the clustering constraint and then a human implements a new LF referring to them in each loop (performed in the order of blue, green and red).

that always attaches correct labels (Dasgupta, 2005; Ipeirotis et al., 2014). As a result, the superiority of the existing methods is verified only in simulation settings, i.e. human-in-the-loop without human settings. Therefore, the proposed framework is the first practical method of AL for WS from scratch intended to substantially reduce human workload.

In addition, AL itself has faced criticisms for practical use (Siddhant and Lipton, 2018; Atighehchian et al., 2020). The main concerns are inaccuracy of estimated uncertainty and implicit label leakage. To overcome the first concern, Bayesian AL such as Bayesian AL by Disagreement (BALD) (Houlsby et al., 2011) has often been used to estimate the uncertainty more accurately than the traditional uncertainty-based methods (Yuan et al., 2019). The proposed framework utilizes batchBALD with consistent Monte Carlo dropout (cMCdropout) (Kirsch et al., 2019) as an acquisition function in AL, which is an extension of BALD to select multiple data samples in a single loop. The model uncertainty is calculated from the variance in the output of multiple runs with different dropout masks (Gal and Ghahramani, 2016) but fixed during each epoch in cMCdropout. The maximum batchBALD score can be calculated by greedy approximation thanks to the submodularity of the mutual information. Label leakage, the second concern, means that fully labeled validation datasets are implicitly used for hyperparameter optimization or validation during training, even though they are unavailable in a practical setting. This concern can also be found in some WS studies. The proposed framework entirely avoids making any implicit label leakage by using LFs as a substitute for ground truth in the validation step.

3 Framework

The proposed framework is a new method of AL for WS from scratch, so the computer suggests a few prioritized data samples, and humans implement an LF in each loop. What is important is how to select the data samples to help humans design a more effective and probable LF with a lighter workload. To ensure the effectiveness, the proposed framework has a soft attention Bayesian neural network (BNN) derived from Ren et al. (2020) that both estimates Bayesian uncertainty via cMCdropout and reduces label noise of WS by attention mechanisms. In addition, to reduce the human workload, the proposed framework has density clustering to aggregate data samples having common semantic features and adds a clustering constraint to the acquisition function for considering the aggregation.

This section describes the proposed framework by dividing it into six steps in each loop as shown in Figure 1 ① - ⑥ for easy understanding. These six steps are executed in order in each loop. Note that the second and third step set and the fourth step can be executed in parallel because there is no dependency on their input data. The proposed framework is designed for cooperation between humans and computers, so each data sample $\boldsymbol{d}$ in the whole unlabeled dataset $\mathcal{D}$ is assumed to have two representation styles: (1) a human-recognizable representation such as a raw text; and (2) a feature vector representation by embedding or feature extraction. Also, let $\mathcal{Y}$ and $\mathrm{n}(\cdot)$ denote the class label set and the number of elements in a set, respectively.

The first step is applying LFs. Each LF in the set of implemented LFs $\mathcal{L}$ receives $\boldsymbol{d}$. The votes for $\boldsymbol{d}$ by all LFs in $\mathcal{L}$ are combined into a voting result vector $\boldsymbol{v} \in \{-1, 1, 2, \ldots, \mathrm{n}(\mathcal{Y})\}^{\mathrm{n}(\mathcal{L})}$. If the l-th compo-

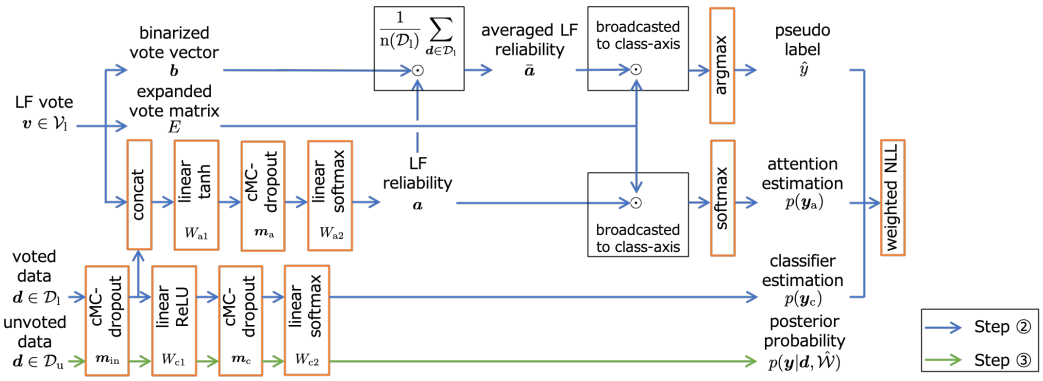

Figure 2: Structure of the soft attention BNN: the posterior probability is calculated with Bayesian uncertainty thanks to cMCdropout. To denoise the WS, the LF reliability is calculated for each data sample concatenated with the LF votes and then weighted majority voting is performed using the LF reliability.

nent of $\boldsymbol{v}$ is -1, it means that the l-th LF has abstained from voting. After the voting, $\mathcal{D}$ is divided into a voted dataset $\mathcal{D}_\mathrm{l}$ and an unvoted dataset $\mathcal{D}_\mathrm{u}$ depending on the voting results. If $\boldsymbol{d}$ has received at least one LF vote for any class, i.e., if not all components of the corresponding $\boldsymbol{v}$ are -1, $\boldsymbol{d}$ belongs to $\mathcal{D}_\mathrm{l}$, otherwise $\mathcal{D}_\mathrm{u}$. In addition, let $\mathcal{V}_\mathrm{l}$ be the set consisting of $\boldsymbol{v}$ corresponding to $\boldsymbol{d}$ belonging to $\mathcal{D}_\mathrm{l}$. For the following steps, let $E \in \mathbb{R}^{\mathrm{n}(\mathcal{Y}) \times \mathrm{n}(\mathcal{L})}$ be the expanded vote matrix whose (c, l)-th component e_{cl} is 1 if l-th LF votes for class c, otherwise 0, and $\boldsymbol{b} \in \mathbb{R}^{\mathrm{n}(\mathcal{L})}$ be the binarized vote vector whose l-th component is 1 if the l-th LF votes for any class, otherwise 0.

The second step is training and validating soft attention BNN. Figure 2 shows the BNN structure. It receives $\mathcal{D}_\mathrm{l}$ and $\mathcal{V}_\mathrm{l}$ for training. Let $\mathcal{W}=\{W_{\mathrm{c1}}, W_{\mathrm{c2}}, W_{\mathrm{a1}}, W_{\mathrm{a2}}\}$ denote the set of weight parameter matrices, $\{\boldsymbol{m}_\mathrm{in}, \boldsymbol{m}_\mathrm{c}, \boldsymbol{m}_\mathrm{a}\}$ denote the cMCdropout masks, and $\tilde{\boldsymbol{d}} = \boldsymbol{d} \odot \boldsymbol{m}_\mathrm{in}$, where $\odot$ is Hadamard product, i.e., element-wise multiplication. The classifier estimation $p(\boldsymbol{y}_\mathrm{c})$ is calculated by a two-layer feedforward neural network as

$$p(\boldsymbol{y}_\mathrm{c}) = \mathrm{softmax}(W_{\mathrm{c2}}^\mathsf{T}(\mathrm{ReLU}(W_{\mathrm{c1}}^\mathsf{T}\tilde{\boldsymbol{d}}) \odot \boldsymbol{m}_\mathrm{c})). \quad (1)$$

Let $\tilde{\boldsymbol{v}}$ be a concatenation of $\tilde{\boldsymbol{d}}$ and $\boldsymbol{v}$ corresponding to $\boldsymbol{d}$. To reduce the label noise of WS, the LF reliability $\boldsymbol{a} \in \mathbb{R}^{\mathrm{n}(\mathcal{L})}$ whose l-th component a_l represents the reliability of the l-th LF is calculated by the other two-layer feedforward neural network as

$$\boldsymbol{a} = \mathrm{softmax}(W_{\mathrm{a2}}^\mathsf{T}(\tanh(W_{\mathrm{a1}}^\mathsf{T}\tilde{\boldsymbol{v}}) \odot \boldsymbol{m}_\mathrm{a})). \quad (2)$$

Then, the c-th component of attention estimation $p(\boldsymbol{y}_\mathrm{a})$ is calculated by the LF voting results weighted by each LF reliability as

$$p(y_\mathrm{a}{=}c) = \frac{\exp\left(\sum_l e_{cl} a_l\right)}{\sum_{c'} \exp\left(\sum_l e_{c'l} a_l\right)}. \quad (3)$$

To obtain a pseudo label $\hat{y}$, the averaged LF reliability $\bar{\boldsymbol{a}}$ is calculated through $\mathcal{D}_\mathrm{l}$ as

$$\bar{\boldsymbol{a}} = \frac{1}{\mathrm{n}(\mathcal{D}_\mathrm{l})} \sum_{\boldsymbol{d} \in \mathcal{D}_\mathrm{l}} (\boldsymbol{b} \odot \boldsymbol{a}). \quad (4)$$

Then, $\hat{y}$ is decided by weighted majority voting as

$$\hat{y} = \operatorname*{argmax}_c \sum_l e_{cl} \bar{a}_l. \quad (5)$$

The training is performed with $\hat{y}$ as pseudo ground truth and the loss defined as the weighted sum of negative log likelihoods (NLL) from $(\hat{y}, p(\boldsymbol{y}_\mathrm{c}))$ and $(\hat{y}, p(\boldsymbol{y}_\mathrm{a}))$ for co-training the classifier and the WS denoiser. Let $\hat{\mathcal{W}}$ denote the optimized weight parameters through the training. Thanks to the pseudo labeling technique, validation can be performed with no label leakage. A part of the unlabeled dataset is separated as a validation dataset in advance. The validation is performed with the pseudo ground truth obtained by the denoised WS as in training.

The third step is estimating posterior probability of $\mathcal{D}_\mathrm{u}$ on the optimized classifier with $\hat{\mathcal{W}}$. The output posterior probability distribution of $\boldsymbol{d} \in \mathcal{D}_\mathrm{u}$ considering the uncertainty is estimated by repeated estimation with a cMCdropout as $p(\boldsymbol{y}|\boldsymbol{d}, \hat{\mathcal{W}})$.

The fourth step is extracting semantic clusters by density clustering. This step also receives $\mathcal{D}_\mathrm{u}$. Let $\mathcal{C}_k \subset \mathcal{D}_\mathrm{u}$ denote the set of data samples belonging to

the k-th cluster. Note that the maximum number of k changes depending on the clustering result. This step enables the following sampling to suggest data samples with semantic consistency to elicit probable labeling rules for humans to design LFs with lighter workload because the data in the identical cluster are more likely to have common semantic features.

The fifth step is sampling. Let $\mathcal{D}_\mathrm{s}$ and $q(\mathcal{D}_\mathrm{s})$ denote a small number of data samples selected from $\mathcal{D}_\mathrm{u}$ and its acquisition function calculated by batchBALD, respectively. Calculation of batchBALD involves using $p(\boldsymbol{y}|\boldsymbol{d}, \hat{\mathcal{W}})$ estimated in the third step. The difference from the original batchBALD is the clustering constraint added to make the sampling human-friendly. The batch is formed by only data samples belonging to an identical semantic cluster to ensure semantic consistency in the batch. Thus, in the implementation, batchBALD is applied for each $\mathcal{C}_k$ extracted in the fourth step. The candidate data samples $\tilde{\mathcal{D}}_{\mathrm{s}k}$ for each $\mathcal{C}_k$ are extracted as

$$\tilde{\mathcal{D}}_{\mathrm{s}k} = \underset{\mathcal{D}_\mathrm{s} \subset \mathcal{C}_k}{\operatorname{argmax}}\, q(\mathcal{D}_\mathrm{s}). \tag{6}$$

Among these candidates, the one with the largest batchBALD score is selected for the set of data samples $\tilde{\mathcal{D}}_\mathrm{s}$ to suggest for humans as

$$\tilde{\mathcal{D}}_\mathrm{s} = \underset{\tilde{\mathcal{D}}_{\mathrm{s}k}}{\operatorname{argmax}}\, q(\tilde{\mathcal{D}}_{\mathrm{s}k}). \tag{7}$$

BatchBALD with a clustering constraint is the key to reducing the number of LFs and the workload of LF designing simultaneously. Ratner et al. (2017) remarks that human defined LF tends to have low accuracy and large coverage. Trying to prevent it leads to make the coverage too small. The proposed framework can optimize the accuracy and the coverage by suggesting multiple data samples with common semantic features.

The last step is implementing a new LF. This is the only step performed by humans. Humans implement a new LF by referring to $\tilde{\mathcal{D}}_\mathrm{s}$ and add it to $\mathcal{L}$. The $\tilde{\mathcal{D}}_\mathrm{s}$ should have both semantic consistency and a large utility in classification performance when correctly labeled by the new LF. This process is an analogy for programming-by-examples in intuition-based human programming since the entity of LF is a simple program (Devlin et al., 2017). Thus, humans can implement an effective and probable LF with light workload. Thereafter, the loop consisting of these six steps is repeated the designated number of times while updating each variable associated with the addition of the new LF.

Table 1: Major parameter settings

The dimension of each hidden layer	128
learning rate	0.001
# of samples for cMCdropout	100
The maximum epoch for the training	1000
The patience of early stopping	100
The minimum cluster size	5
# of suggested samples in each loop	5

4 Experiment

The object of the user studies with 16 consenting volunteers in two text classification tasks is to verify the effectiveness of the proposed method. The tasks are SMS spam detection (Almeida et al., 2011) and TREC-6 question classification (Li and Roth, 2002), which were also used in recent WS studies (Awasthi et al., 2020). These tasks do not require specialized domain knowledge except for basic skills in English reading comprehension and Python programming. The volunteers were non-native English speaking software engineers who met the above requirements and offered from multiple groups. They were likely to take on the labeling task and understand the task but did not know about WS. This condition is desirable to verify the effectiveness of the method with many subjects, even though it would be ideal to verify the method on tasks in which specialized domain experts work well, such as medical data or financial data. Note that automatic experiments without humans do not make sense because the objective of the proposed method is to enable humans to design probable LFs with light workload.

4.1 Settings

The number of classes and data samples for training, validation, and test are 2, 4504, 500, 500 in SMS and 6, 4906, 546, 500 in TREC-6, respectively. Note that no data include any ground truth labels in either training or validation. Table 1 shows the major parameter settings of the framework used in the user study. Hyper-parameters are not pre-optimized to the tasks considering the constraints of AL for WS

from scratch in the wild. It is ideal to search for the best hyper-parameters with humans, but this is impractical in terms of human workload. Thus, it was also verified whether the hyper-parameters affect the superiority of the proposed method by recalculation using the implemented LFs. This setting is promising because the effectiveness of the implemented LFs is more important in this experimental design than the classification performance during the user study.

The implementation used sentence BERT (Reimers and Gurevych, 2019) provided by Hugging Face[1] for feature extraction, RAdam (Liu et al., 2020) for the optimizer, and HDBSCAN (Campello et al., 2013; McInnes and Healy, 2017) for the density clustering. The primary motivation for these choices is that they are robust to hyper-parameters and tolerant of overfitting in the state-of-the-art natural language processing (NLP) setting. Although it is common to use pre-trained BERT models for NLP tasks, the wider domains where the proposed method can be applied do not necessarily enable the use of prior knowledge. Thus, it was also verified whether the pre-training affects the superiority of the proposed method by recalculation using the features extracted by doc2vec (PV-DM) (Le and Mikolov, 2014), which is a representative method without pre-training.

To quantitatively verify the effectiveness of the method, the proposed method (Proposed) was compared to two other methods: batch extended Saisho et al. (2021) (Active) as the state-of-the-art AL for WS from scratch method and random sampling with a clustering constraint (Semantic) for reference. Note that the comparison with the other three types in the related work is meaningless because the prebuilt resources used in those types are the very targets or even more sophisticated ones expected to be implemented by subjects in the user study. "Semantic" randomly extracts data samples from the randomly extracted identical cluster. Simple random sampling (without the clustering constraint) should also be performed for an ablation study. However, it was excluded from the verification to prevent increasing the burden on subjects unnecessarily in the

[1] https://huggingface.co/sentence-transformers/paraphrase-distilroberta-base-v1

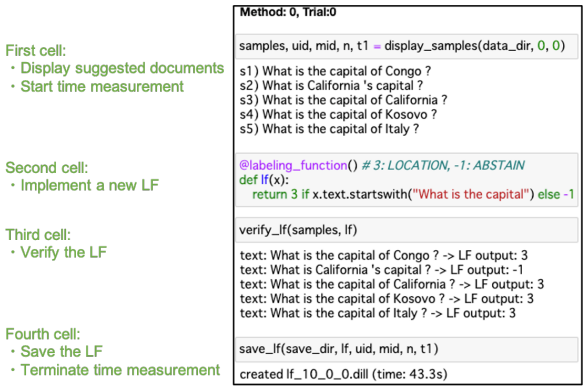

Figure 3: Four filled cells to implement a new LF in each loop: humans can display the suggested samples and then implement, verify, and save a new LF.

user study, considering that many existing studies have already shown AL is superior to random sampling. Similarly, to clarify in advance that the work time for each subject would fit into the daily working hours, the number of loops was set to 15, i.e., the total number of implemented LFs in each task is 45. The main evaluation metrics are the macro F-measure for classification performance, the time required to implement an LF, the accuracy of an LF voting, and the coverage of an LF voting.

Subjects were to work on the Jupyter notebooks (Kluyver et al., 2016) provided for each task as shown in Figure 3. When the first cell is run, the five data samples suggested by a method are displayed, and the time measurement starts. Subjects then implement an LF in a free format using Snorkel Python library (Ratner et al., 2017; Ratner et al., 2019) in the second cell, check its output by running the third cell, and repeat if they need to modify it. Finally, the fourth cell is run to send the new LF to a computer, and the time measurement is terminated. The computer then trains the classifier and selects the next suggested samples. Then, subjects can proceed to the next loop.

To evaluate the three methods equally, the user study devised two procedures. First, the subjects were told to perform the experiment procedure in the IMDB sentiment analysis (Maas et al., 2011) beforehand without performance measurement. This helps to reduce the effect on the results due to unfamiliarity with WS and the experimental procedure. Second, subjects were told to perform all three methods in parallel in a random order, without knowing the

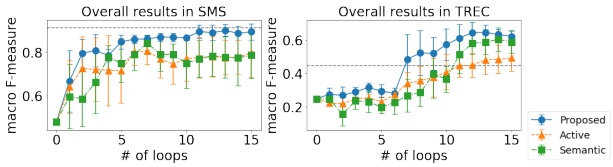

Figure 4: F-measures of each loop on each method: Proposed outperformed the other methods in terms of classification performance in all plots for both tasks.

Table 2: The mean time [sec.] and its relative value required to implement an LF: Proposed and Semantic outperformed Active thanks to the clustering constraint.

	SMS	TREC
Proposed	$\mathbf{156 \pm 143}$	$\mathbf{192 \pm 125}$
	$\mathbf{0.63 \pm 0.23}$	$\mathbf{0.75 \pm 0.05}$
Active	221 ± 166	258 ± 179
	1	1
Semantic	168 ± 137	197 ± 157
	0.78 ± 0.28	$\mathbf{0.75 \pm 0.10}$

differences between methods. This is also to prevent unfairness between methods due to mastery of WS and leakage of knowledge about the dataset, which may occur if the methods are conducted in series.

4.2 Results

Figure 4 shows the results of classification performance. Proposed outperformed the other methods in all plots in both tasks. These results quantitatively indicate that the proposed method enables humans to implement LFs with high utility. Slightly surprisingly, Semantic performed so well that its superiority to Active was intersected by plots. This viewpoint needs to be taken together with the performance of each LF, so it will be described in later paragraphs. In addition, the gray dashed lines show the results when applying all 73 LFs for SMS and 68 LFs for TREC from a recent WS study (Awasthi et al., 2020) to the classification model. The performance of Proposed is close to that for SMS and better than that for TREC. Note that the difference from the original results mainly comes from using some labeled data and a different evaluation metric in the original study. These results also indicate the effectiveness of the proposed method.

Table 2 shows the mean work times required to implement an LF. Their relative values on the basis of the mean of Active for each subject are also

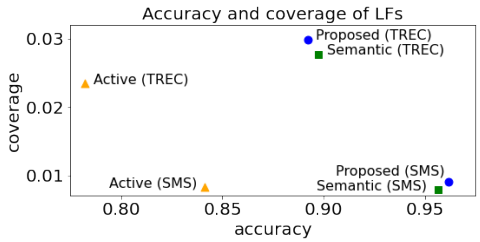

Figure 5: The mean accuracy and coverage of each LF: Proposed and Semantic outperformed Active in terms of helping humans to implement probable LFs.

listed since there are large individual differences in the overall work time, Proposed reduced the work times by 37% in SMS and 25% in TREC compared with Active, respectively. These results quantitatively indicate that Proposed and Semantic are superior to Active in suggesting samples that elicit labeling rules for designing a new LF with a lighter workload. Thus, they show that the clustering constraint works well to make the framework human-friendly.

Figure 5 shows the mean accuracy and mean coverage of each LF. Accuracy is the rate of votes that match the ground truth among the data samples in which LF votes. Coverage is the rate of data samples in the training dataset that were voted to any class by a single LF. In terms of coverage, there was no significant difference between the methods in SMS, so the superiority of the methods cannot be determined. On the other hand, there is a large difference in accuracy between Active and other methods, i.e., depending on whether the clustering constraint is added or not. This explains not only the reason Proposed is superior to other methods in the F-measure shown in Figure 4, but also the reason Semantic performs the same as or better than Active. These results indicate the importance of considering the accuracy of LF in a true human-in-the-loop situation in WS and the superiority of the proposed method to help humans design probable LFs.

For reference, Figure 6 shows the examples of data samples suggested at the same time by each method. Although it is difficult to explicitly visualize the difference in labeling utility by data samples, it should show the difference in the easiness of implementing a probable LF.

Figure 7 shows the results of hyper-parameter search by recalculations. The search spaces were $\{32, 64, 128, 256\}$ for the dimension of each hid-

Proposed

s1) What is the most populated city in the world ?
s2) What 's the most powerful country in the world ?
s3) "What is the most important nation in the world , historically ?"
s4) What are all the southern states of the United States ?
s5) What are all the southern states of the U.S. ?

Active

s1) What Sinatra hit did he dooby dooby do in ?
s2) What 2 statues did France give to other countries ?
s3) What part did Benjamin Franklin play in the development of the newspaper in America ?
s4) What stereo manufacturer is `` Slightly ahead of its time '' ?
s5) "What celestial body has a diameter of 864 , 000 miles ?"

Semantic

s1) What is a fear of failure ?
s2) What is a fear of drinking ?
s3) What is the fear of being loved ?
s4) What is a fear of parasites ?
s5) What is a fear of strong light ?

Figure 6: Examples of data samples suggested at the same time by each method in TREC task

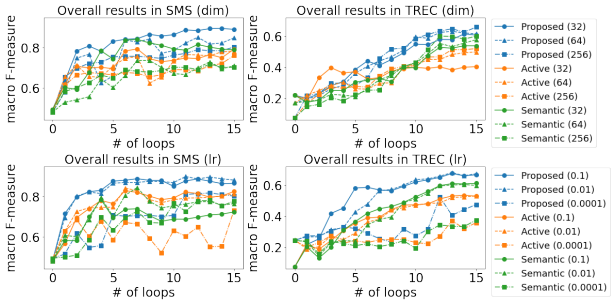

Figure 7: F-measures of each loop on each method with hyper-parameter search: the superiority of Proposed is not significantly affected under each condition although the superiority of a few plots is reversed. (Error bars are omitted in favour of visibility.)

den layer (dim) and $\{0.0001, 0.001, 0.01, 0.1\}$ for the learning rate (lr). When one hyper-parameter was searched for, the other was fixed to its default.

Lastly, Figure 8 shows the results obtained by recalculations with feature vectors extracted by doc2vec (PV-DM). In both recalculations, the overall trends do not differ significantly from the experimental results shown in Figure 4 under each condition although the superiority of a few plots is reversed and the overall performance changes. Thus, these results indicate that the superiority of the proposed method does not depend on hyper-parameters or the pre-trained embedding.

From the results of the user studies, the proposed method enables humans to implement more accurate LFs that can attach labels with greater utility for classification performance in a shorter work time.

4.3 Limitations and Future outlooks

In this section, three future outlooks are presented in accordance with the limitations of this paper. First,

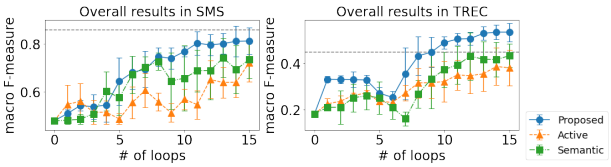

Figure 8: F-measures of each loop on each method without pre-trained embedding: the superiority of Proposed is not significantly affected although the superiority of a few plots is slightly reversed.

s1) What is the design of the ship Titanic ?
s2) What caused the Titanic to sink ?
s3) What was the name of the Titanic 's captain ?
s4) How many years ago did the ship Titanic sink ?
s5) What ocean did the Titanic sink in ?

Figure 9: Unsuitable example suggested by Proposed: "Titanic" is a common feature among these samples, but it is not a key in the classification. The samples actually belong to several classes.

the common features extracted by semantic clustering may not always be effective for classification tasks. In a few cases, the semantic clustering constraint did not work well and suggested samples unsuitable for the classification task as shown in Figure 9. This indicates another support framework is needed for learning the humans' viewpoint from the implemented LFs to make the semantic cluster constraint more directly related to LF implementation support. For example, a supervised attention approach is discussed also in a recent study (Sen et al., 2020). Incorporating explainable artificial intelligence (XAI) such as SHAP (Lundberg and Lee, 2017) should also be effective in eliciting more explicit rules for human-friendly support.

Second, errors can be reduced but not eliminated and will accumulate further. The classification performance was rarely degraded by adding an LF. This is because negative effects of noisy LFs are much larger than those of noisy labels in standard labeling. In AL, some solutions for noisy labels have been devised (Bouguelia et al., 2015; Bouguelia et al., 2018; Lin et al., 2016). The framework that extends these methods to noisy LFs should be very practical.

Lastly, some individual human differences in knowledge and skills cannot be adequately supported, and the experiment was performed by only software engineers. It is true that the LF implementation task depends more on individual skills than the traditional direct labeling task. Thus, methods

to help LF implementation without programming such as Hancock et al. (2018) should be effective in compensating for people's lack of skills. Beyond that, the development of tools to integrate the proposed method with autoML and no-code development tools will lead to the democratization of NLP as well as AI. From another point of view, task distribution including both LF implementation and direct labeling is also an alternative research direction when multiple humans perform a single task, especially as in crowdsourcing. Such frameworks can provide an environment where anyone can work in accordance with their skills.

5 Conclusion

This paper proposed a new method of practical active learning (AL) for weak supervision (WS) in the wild from scratch and demonstrated its superiority in helping users to design and implement labeling functions (LFs) in terms of classification performance, required work time, and LF accuracy in user studies with a real interactive system and no label leakage. Its superiority is achieved by Bayesian AL with a semantic clustering constraint for enhancing labeling utility, reducing human workload, and enhancing the effectiveness of each LF.. For future work, the model will be verified by other complicated tasks and larger-scale user studies, in addition to the future outlook.

References

T. Almeida, J. Hidalgo, and A. Yamakami. 2011. Contributions to the study of sms spam filtering: New collection and results. In *Proc. of DocEng*, pages 259–262.

P. Atighehchian, F. Charron, and A. Lacoste. 2020. Bayesian active learning for production, a systematic study and a reusable library. In *Proc. of ICML workshops*.

A. Awasthi, S. Ghosh, R. Goyal, and S. Sarawagi. 2020. Learning from rules generalizing labeled exemplars. In *Proc. of ICLR*.

S. Bach, B. He, A. Ratner, and C. Ré. 2017. Learning the structure of generative models without labeled data. In *Proc. of ICML*, pages 273–282.

S. Bach, D. Rodriguez, Y. Liu, C. Luo, H. Shao, C. Xia, S. Sen, A. Ratner, B. Hancock, H. Alborzi, R. Kuchhal, C. Ré, and R. Malkin. 2019. Snorkel drybell: A case study in deploying weak supervision at industrial scale. In *Proc. of SIGMOD*, pages 362–375.

S. Biegel, R. Khatib, L. Oliveira, M. Baak, and N. Aben. 2021. Active weasul: Improving weak supervision with active learning. In *Proc. of ICLR workshops*.

B. Boecking, W. Neiswanger, E. Xing, and A. Dubrawski. 2021. Interactive weak supervision: Learning useful heuristics for data labeling. In *Proc. of ICLR*.

M. Bouguelia, Y. Belaïd, and A. Belaïd. 2015. Identifying and mitigating labelling errors in active learning. In *Proc. of ICPRAM*, pages 35–51.

M. Bouguelia, S. Nowaczyk, K. Santosh, and A. Verikas. 2018. Agreeing to disagree: active learning with noisy labels without crowdsourcing. *Int. J. Mach. Learn. Cybern.*, 9:1307–1319.

R. Campello, D. Moulavi, and J. Sander. 2013. Density-based clustering based on hierarchical density estimates. In *Proc. of KDD*, pages 160–172.

W. Chen, Y. Liu, Z. Kira, Y. Wang, and J. Huang. 2019. A closer look at few-shot classification. In *Proc. of ICLR*.

N. Das, S. Chaba, R. Wu, S. Gandhi, D. Horng Chau, and X. Chu. 2020. Goggles: Automatic image labeling with affinity coding. In *Proc. of SIGMOD*, pages 1717–1732.

S. Dasgupta. 2005. Coarse sample complexity bounds for active learning. In *Proc. of NeuIPS*, pages 235–242.

J. Devlin, J. Uesato, S. Bhupatiraju, R. Singh, A. Mohamed, and P. Kohli. 2017. Robustfill: Neural program learning under noisy i/o. In *Proc. of ICML*, pages 990–998.

J. Fries, P. Varma, V. Chen, K. Xiao, H. Tejeda, P. Saha, J. Dunnmon, H. Chubb, S. Maskatia, M. Fiterau, S. Delp, E. Ashley, C. Re, and J. Priest. 2019. Weakly supervised classification of aortic valve malformations using unlabeled cardiac mri sequences. *Nature Communications*, 10(3111).

Y. Gal and Z. Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proc. of ICML*, volume 48, pages 1050–1059.

J. Gonsior, M. Thiele, and W. Lehner. 2020. Weakal: Combining active learning and weak supervision. In *Proc. of DS*, pages 34–49.

B. Hancock, P. Varma, S. Wang, M. Bringmann, P. Liang, and C. Ré. 2018. Training classifiers with natural language explanations. In *Proc. of ACL*, pages 1884–1895.

A. Holzinger. 2016. Interactive machine learning for health informatics: when do we need the human-in-the-loop? *Brain Informatics*, 3(2):119–131.

N. Houlsby, F. Huszar, Z. Ghahramani, and M. Lengyel. 2011. Bayesian active learning for classification and preference learning. *arXiv:1112.5745*.

P. Ipeirotis, F. Provost, V. Sheng, and J. Wang. 2014. Repeated labeling using multiple noisy labelers. *Data Min. Knowl. Discov.*, 28(2):402–441.

D. Kartchner, W. Ren, D. An, C. Zhang, and C. Mitchell. 2020. Regal: Rule-generative active learning for model-in-the-loop weak supervision. In *Proc. of NeurIPS workshops*.

A. Kirsch, J. van Amersfoort, and Y. Gal. 2019. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. In *Proc. of NeurIPS*, volume 32.

T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, C. Willing, and Jupyter development team. 2016. Jupyter notebooks - a publishing format for reproducible computational workflows. In *Proc. of ElPub*, pages 87–90.

Q. Le and T. Mikolov. 2014. Distributed representations of sentences and documents. In *Proc. of ICML*, pages 1188–1196.

P. Lertvittayakumjorn, L. Specia, and F. Toni. 2020. FIND: Human-in-the-Loop Debugging Deep Text Classifiers. In *Proc. of EMNLP*, pages 332–348.

X. Li and D. Roth. 2002. Learning question classifiers. In *Proc. of COLING*, pages 1–7.

G. Li. 2017. Human-in-the-loop data integration. *Proc. of VLDB Endow.*, 10(12):2006–2017.

C. Lin, Mausam, and D. Weld. 2016. Re-active learning: Active learning with relabeling. In *Proc. of AAAI*, pages 1845–1852.

L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han. 2020. On the variance of the adaptive learning rate and beyond. In *Proc. of ICLR*.

S. Lundberg and S. Lee. 2017. A unified approach to interpreting model predictions. In *Proc. of NeurIPS*.

A. Maas, R. Daly, P. Pham, D. Huang, A. Ng, and C. Potts. 2011. Learning word vectors for sentiment analysis. In *Proc. of ACL*, pages 142–150.

L. McInnes and J. Healy. 2017. Accelerated hierarchical density based clustering. In *Proc. of ICDM workshops*, pages 33–42.

M. Nashaat, A. Ghosh, J. Miller, S. Quader, C. Marston, and J. Puget. 2018. Hybridization of active learning and data programming for labeling large industrial datasets. In *Proc. of IEEE BigData*, pages 46–55.

M. Nashaat, A. Ghosh, J. Miller, and S. Quader. 2020. Asterisk: Generating large training datasets with automatic active supervision. *Trans. Data Sci.*, 1(2).

K. Qian, P. Raman, Y. Li, and L. Popa. 2020. Learning structured representations of entity names using ActiveLearning and weak supervision. In *Proc. of the EMNLP*, pages 6376–6383.

M. Raghu, C. Zhang, J. Kleinberg, and S. Bengio. 2019. Transfusion: Understanding transfer learning for medical imaging. In *Proc. of NeurIPS*, pages 3347–3357.

A. Ratner, C. Sa, S. Wu, D. Selsam, and C. Ré. 2016. Data programming: Creating large training sets, quickly. In *Proc. of NeurIPS*, pages 3567–3575.

A. Ratner, S. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *Proc. VLDB Endow.*, 11(3):269–282.

A. Ratner, B. Hancock, J. Dunnmon, F. Sala, S. Pandey, and C. Ré. 2019. Training complex models with multi-task weak supervision. In *Proc. of AAAI*, pages 4763–4771.

N. Reimers and I. Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proc. of EMNLP*, pages 3982–3992.

W. Ren, Y. Li, H. Su, D. Kartchner, C. Mitchell, and C. Zhang. 2020. Denoising multi-source weak supervision for neural text classification. In *Proc. of EMNLP*, pages 3739–3754.

O. Saisho, T. Ohguro, J. Sun, H. Imamura, S. Takeuchi, and D. Yokozeki. 2021. Human knowledge based efficient interactive data annotation via active weakly supervised learning. In *Proc. of PerCom workshops*, pages 332–335.

C. Sen, T. Hartvigsen, B. Yin, X. Kong, and E. Rundensteiner. 2020. Human attention maps for text classification: Do humans and neural networks focus on the same words? In *Proc. of ACL*, pages 4596–4608.

B. Settles. 2009. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison.

A. Siddhant and Z. Lipton. 2018. Deep Bayesian active learning for natural language processing: Results of a large-scale empirical study. In *Proc. of EMNLP*, pages 2904–2909.

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

B. Wang, A. Ratner, S. Mussmann, and C. Re. 2019. Interactive programmatic labeling for weak supervision. In *Proc. of KDD workshop*.

J. Yuan, X. Hou, Y. Xiao, D. Cao, W. Guan, and L. Nie. 2019. Multi-criteria active deep learning for image classification. *Knowledge-Based Systems*, 172:86–94.

Z. Zhou. 2017. A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53.