# Appendix

**Organization of Appendix.** The Appendix is organized as follows. We first present the detailed active learning with re-sampling process and the corresponding pseudo code. We then provide complete proofs of major theorems. We present the detailed description of the datasets and experimental settings. Finally, we show some additional experimental results with detailed setting clarification, including both comparison with other existing baselines and more ablation study. The link to the source code is provided in the end of the Appendix.

## Active Learning with Re-Sampling Process and Key Assumptions

Figure 7 shows the process of active learning (AL) with re-sampling, which involves interactions between a machine $h$ and an annotator $h_\alpha$. Active sampling is controlled by the sampling function $f(\cdot)$, which selects a data instance from an unlabeled pool. Re-sampling is controlled by function $g(\cdot)$, which selects a data instances from the training set. Either way, the annotator would provide a label for the selected instance and the newly labeled instance would be used in model fitting for the next learning iteration. The key steps are highlighted in Algorithm 1.

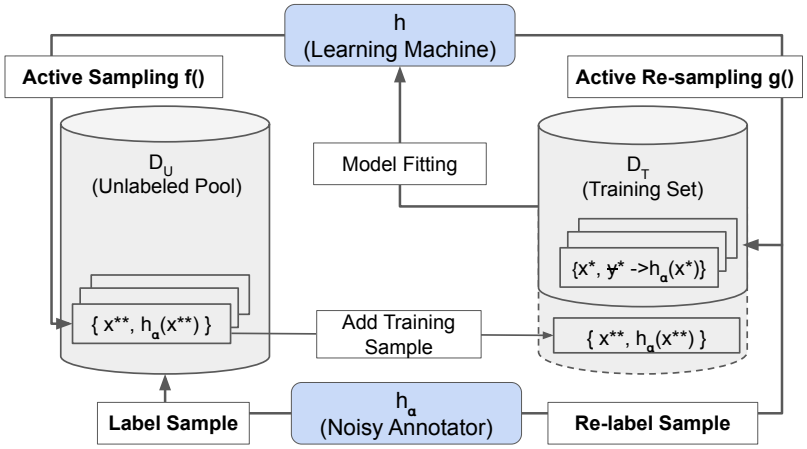



Figure 7: The process of active learning with re-sampling

Note that AL is the main process of the proposed work and re-sampling is triggered by some switch function. Both Figure 7 and Algorithm 1 only describe the non-trivial iteration where re-sample is triggered. The trivial iteration only involves active sampling thus is the same as the classical AL process. Intuitively, the switch function should consider inputs such as the expected annotation noise, the current stage (*i.e.,* early or late) of AL and statistics of the current model to reach a good balance of annotation between active sampling and re-sampling. The design of the switch function is one of the important direction in our future work but it is out of the scope of the present work. Since we fix the annotation noise in this work, we consider a periodic function as a suitable candidate for the switch function. In our experiments, re-sampling is triggered every four AL iterations.

Algorithm 1: Active Re-sampling

**input :** Training set: $\mathcal{D}_T$, Unlabeled pool: $\mathcal{D}_U$, active sampling function $f()$, active re-sample function $g()$ Hypothesis set: $\mathcal{H}$, Annotation budget:B,noisy annotator: $h_\alpha()$

**output :** Updated training set $\mathcal{D}_T$ and optimal hypothesis $h^*$

**while** $(B > 0)$ **do**

  $h^* = \arg\max_{h\in\mathcal{H}} \sum_{\{\mathbf{x}_n, y_n\}\in\mathcal{D}_T} \mathcal{L}_h(\mathbf{x}_n, y_n)$ `Model fitting using noisy data`

  $\mathbf{x}^{**} = \arg\max_{\mathbf{x}_m\in\mathcal{D}_U} f(\mathbf{x}_n, h^*)$ `Sampling`

  $B = B - 1$

  $\mathcal{D}_T = \mathcal{D}_T \cup \{\mathbf{x}^{**}, h_\alpha(\mathbf{x}^{**})\}$ `Labeling`

  $\{\mathbf{x}^*, y^*\} = \arg\max_{\{\mathbf{x}_n, y_n\}\in\mathcal{D}_T} g(\mathbf{x}_n, y_n, h^*)$ `Re-sampling`

  $B = B - 1$

  $\mathcal{D}_T = \mathcal{D}_T \backslash \{\mathbf{x}^*, y^*\} \cup \{\mathbf{x}^*, h_\alpha(\mathbf{x}^*)\}$ `relabeling`

**end**

The proposed framework also assumes that each label annotation is determined independently. This assumption is common in real-world active learning scenarios, such as clinical, in which multiple physicians with similar expertise levels (*i.e.,* the expected

error rate is $\alpha$) participate in the annotation task according to their availability. The annotation process can then be seen as done by one imaginary annotator with a fixed annotation error rate $\alpha$. This assumption impacts the learning process with re-sampling in two ways. First, from the machine's perspective, the annotators are non-identifiable. So we can not leverage the annotator's profile from prior knowledge or the observed labels to improve the model performance as some robust learning method in the context of crowd-sourced annotations (Zhao, Sukthankar, and Sukthankar 2011). We would like to clarify that the major goal of this work is not to train a noise-robust learner but to use an accurate re-sampling strategy and efficient annotation procedure to identify and purify as many wrongly labeled data as possible. To this end, neglecting the characteristics of different annotators does not hurt the contribution of our work. However, we recognize that knowing the annotator's profile has the potential to improve the annotation procedure in our work. For example, we could assign the data to an annotator that minimizes $p(\alpha|\mathbf{x})$ to make the re-labeling more efficient. This will be an interesting future direction.

Second, from the annotator's perspective, the decision-making is not influenced by other annotations they might have done previously. Suppose we relax the non-identifiable annotator's assumption; then, it may be possible to use a more sophisticated model to estimate each annotator's up-to-date error rate distribution conditioned on their previous annotation activities. Such an on-the-fly error rate estimation model, especially its uncertainty quantification, could help to better distribute data samples to the annotators (Khetan, Lipton, and Anandkumar 2017). However, in the active learning setting, the training data is usually highly scarce. As a result, it is typically insufficient to accurately learn a annotator profile accurately according to the very limited historical annotations.

## Proofs of Main Theorems

In this section, we provide the complete proofs for the major theorems presented in the main paper.

**Proof of Lemma 1** To prove Lemma 1, we first show a proposition:

**Proposition 1.** *For any classifier $h^*$ trained using a noisy training set $\mathcal{D}_T$ (i.e., $A(h_\alpha) > 0$) with a meaningful generalization performance that is better than random guess, i.e., $\mathbb{E}(\text{Error}[h^*]) = \epsilon < \frac{K-1}{K}$, then its true error rate is worse than being trained using the clean training set.*

*Proof.* Considering a dataset $\mathcal{D}_T$ with data noise $A(h_\alpha)$, the true error rate is given by:

$$\begin{aligned}
&\mathbb{E}[p(h^*(\mathbf{x}) \neq h_0(\mathbf{x}))] \\
&= 1 - p(y = h_0(\mathbf{x}))\mathbb{E}[p(h^*(\mathbf{x}) = y)] - \frac{1}{K-1}p(y \neq h_0(\mathbf{x}))\mathbb{E}[p(h^*(\mathbf{x}) \neq y)] \qquad (11) \\
&= 1 - (1 - A(h_\alpha))(1 - \epsilon) - \frac{1}{K-1}A(h_\alpha)\epsilon = A(h_\alpha) + \epsilon - \frac{K}{K-1}A(h_\alpha)\epsilon \geq \epsilon \qquad (12)
\end{aligned}$$

where the equality in (11) is due to the equiprobable assumption of the $K$ classes and the inequality in (12) is due to $\epsilon < \frac{K-1}{K}$ and $A(h_\alpha) > 0$. □

Lemma 1 is the direct extension of Proposition 1 using the law of large numbers. From the convergence we know that the gap is expected to increase.

**Proof of Theorem 1.** The proof is achieved by first showing the base case is correct and then through induction.

*Proof.* Using majority vote, the probability of a data instance $\mathbf{x}$'s final label remaining uncorrected $\alpha_R^{\text{maj}}$ after being relabeled $R$ times is is equivalent to the probability of the correct label occurring for no more than $\lfloor \frac{R}{2} \rfloor$ times:

$$\alpha_R^{\text{maj}} = F\left(\lfloor \frac{R}{2} \rfloor; R, 1-\alpha\right) = \sum_{n=0}^{\lfloor \frac{R}{2} \rfloor} \text{Bin}(n|R, 1-\alpha) = \sum_{n=0}^{\lfloor \frac{R}{2} \rfloor} \binom{R}{n} (1-\alpha)^n \alpha^{(R-n)}$$

To break the tie, we restrict $R$ to be odd numbers (it is easy to show that the same conclusion can be reached for even numbers as well if we assume when the numbers of correct and incorrect labels are the same the probability of a correct prediction is $50\%$ ). For $R = 3$,

$$\alpha_3^{\text{maj}} = \binom{3}{0}(1-\alpha)^0\alpha^3 + \binom{3}{1}(1-\alpha)^1\alpha^2 = 3\alpha^2 - 2\alpha^3 < \alpha \qquad (13)$$

We then show $\alpha_{R+2}^{\text{maj}} < \alpha_R^{\text{maj}}$ for $R \geq 3$:

$$\begin{aligned}
&\alpha_R^{\text{maj}} - \alpha_{R+2}^{\text{maj}} = F\left(\lfloor\frac{R}{2}\rfloor; R, 1-\alpha\right) - F\left(\lfloor\frac{R}{2}\rfloor + 1; R, 1-\alpha\right) \\
&= \left\{\sum_{n=0}^{\lfloor\frac{R}{2}\rfloor-1} [\binom{R}{n} - \binom{R+2}{n}\alpha^2]\text{bin}(R, \lfloor\frac{R}{2}\rfloor, 1-\alpha)\right\} \\
&+ \left\{\text{bin}(R, \lfloor\frac{R}{2}\rfloor, 1-\alpha) - \text{bin}(R+2, \lfloor\frac{R}{2}\rfloor, 1-\alpha) \ -\text{bin}(R+2, \lfloor\frac{R}{2}\rfloor + 1, 1-\alpha)\right\} \qquad (14)
\end{aligned}$$

The first term is greater than 0:

$$\begin{aligned}
&[\binom{R}{n} - \binom{R+2}{n}\alpha^2] = [1 - \alpha^2\frac{(R+2)(R+1)}{(R+2-n)(R+1-n)}]\binom{R}{n} \qquad (15) \\
&> 0, \text{ given } n \in [0, \lfloor\frac{R}{2}\rfloor - 1], \alpha < 0.5
\end{aligned}$$

The second term is also greater than 0:

$$\begin{aligned}
&\text{bin}(R, \lfloor\frac{R}{2}\rfloor, 1-\alpha) - \text{bin}(R+2, \lfloor\frac{R}{2}\rfloor, 1-\alpha) - \text{bin}(R+2, \lfloor\frac{R}{2}\rfloor + 1, 1-\alpha) \\
&= \frac{(R+2)!}{\lfloor\frac{R}{2}\rfloor!(\lceil\frac{R}{2}\rceil+2)!}(1-\alpha)^{\lfloor\frac{R}{2}\rfloor}\alpha^{\lceil\frac{R}{2}\rceil+2} - \frac{(R)!}{\lfloor\frac{R}{2}\rfloor!(\lceil\frac{R}{2}\rceil)!}(1-\alpha)^{\lfloor\frac{R}{2}\rfloor}\alpha^{\lceil\frac{R}{2}\rceil} \\
&- \frac{(R+2)!}{(\lfloor\frac{R}{2}\rfloor+1)!(\lceil\frac{R}{2}\rceil+1)!}(1-\alpha)^{\lfloor\frac{R}{2}\rfloor+1}\alpha^{\lceil\frac{R}{2}\rceil+1} \\
&= [1 - \alpha^2\frac{(R+2)(R+1)}{(\lceil\frac{R}{2}\rceil+2)(\lceil\frac{R}{2}\rceil+1)} - \alpha(1-\alpha)\frac{(R+2)(R+1)}{(\lfloor\frac{R}{2}\rfloor+1)(\lceil\frac{R}{2}\rceil+1)}] * \text{bin}(N_1, \lfloor\frac{R}{2}\rfloor, 1-\alpha) \\
&> 0 \text{ , given } n \in [0, \lfloor\frac{R}{2}\rfloor - 1], \alpha < 0.5 \qquad (16)
\end{aligned}$$

Thus, $\alpha_{R+2}^{\text{maj}} < \alpha_R^{\text{maj}}$. Since we also show $\alpha_3^{\text{maj}} < \alpha$, we complete the proof. □

**Proof of Theorem 2.** For Theorem 2, we consider an optimal re-sampling function $g^*$ that no longer samples a data instance once its label is correct. Given the noise rate at $\alpha$, the average number of steps to correct a label is $\lambda = \frac{1}{1-\alpha}$. Thus, the probability $\alpha_R^{poi}$ of the correct label not occurring after relabeling for $R$ times is modeled by the cumulative distribution function (CDF) of a Poisson distribution. The proof leverages the analytical form of the Poisson distribution and through induction.

*Proof.*

$$\alpha_R^{poi} = 1 - \text{CDF}_{\text{Pois}}(\lambda, R) = 1 - \sum_{n=0}^{R}\frac{\lambda^n e^{-\lambda}}{n!} \qquad (17)$$

For the initial case $R = 2$,

$$\alpha_2^{poi} = 1 - e^{-\lambda} - \lambda e^{-\lambda} - \frac{\lambda^2}{2}e^{-\lambda} = 1 - \frac{\alpha^2 - 3\alpha + 4}{(1-\alpha)^2}e^{-\frac{1}{1-\alpha}} \qquad (18)$$

To compare with $\alpha$, we evaluate $F(x) = x - \frac{x^2+x+2}{x^2}e^{-\frac{1}{x}}$, where $x = 1 - \alpha \in (0.5, 1)$. The derivative of $F(x)$ is:

$$F'(x) = 1 - \frac{2-3x}{x^4}e^{-\frac{1}{x}} \qquad (19)$$

Let $F'(x) = 0, x \in (0.5, 1)$, numerically we get $x = 0.50783$. Also, $F'(x) < 0, x < 0.50783$, and $F'(x) > 0, x > 0.50783$. Since at both bondaries $F(0.5) = -0.488688 < 0$ and $F(1) = -0.471578 < 0$, we have $F(x) < 0, \forall x \in (0.5, 1)$. Thus $\alpha_2^{poi} < \alpha, \forall \alpha \in (0, 0.5)$.

For $R > 2$, we have:

$$\alpha_R^{poi} - \alpha_{R+1}^{poi} = \text{CDF}_{\text{Pois}}(\lambda, R+1) - \text{CDF}_{\text{Pois}}(\lambda, R) = \frac{\lambda^{R+1}e^{-\lambda}}{(R+1)!} > 0 \qquad (20)$$

From induction, $\alpha_R^{poi} < \alpha, \forall R > 2, \forall \alpha \in (0, 0.5)$ □

**Proof of Theorem 3.** In order to prove the theorem, we first present a lemma that establishes the relationship between the change of the number of support vectors and correcting a high (hinge) loss.

**Lemma 2.** *When a support vector $\mathbf{x}_i$ with a hinge loss greater than 2 (hinge$_i > 2$) changes its label, it will become a non-support vector.*

*Proof.* Given the hinge loss of $\mathbf{x}_i$, we have

$$\text{hinge}_i = [1 - y_i h(\mathbf{x}_i)]_+ > 2 \tag{21}$$

Thus $\text{sign}(y_i)h(\mathbf{x}_i) < -1$ (using $\text{sign}(y_i)^2 = y_i^2 = 1$). If we flip the sign of $y_i$, $\text{sign}(y_i') = -\text{sign}(y_i)$, then

$$\text{sign}(y_i')h(\mathbf{x}_i) > 1 \tag{22}$$

We can then get $\text{hinge}_i' = 0$, and $y_i' h(\mathbf{x}_i) > 1$. Thus, $\mathbf{x}_i$ is then on the correct side of the decision boundary and outside of the margin, making it a non-support vector data instance. □

Now, we show our proof of the theorem using the result in Lemma 2 and the leave-one-out (LOO) error analysis of SVMs.

*Proof.* First, we use $h^{(T)}$ to represent the hypothesis after re-training with the $T^{th}$ instance, and $h^{(T-1)}_{\mathcal{D}_T \setminus \{\mathbf{x}_i, y_i\}}$ for the LOO hypothesis with $\mathbf{x}_i$ being left out. If $\mathbf{x}_i$ is not a support vector, then adding or removing $\mathbf{x}_i$ will not change the decision boundary, which means that $h^{(T-1)}_{\mathcal{D}_T \setminus \{\mathbf{x}_i, y_i\}}$ can correctly classify $\mathbf{x}_i$, thus $L[h^{(T-1)}_{D_T \setminus \{\mathbf{x}_i, y_i\}}, \mathbf{x}_i] = 0$. The worst case is that for all support vectors, $L[h^{(T-1)}_{D_T \setminus \{\mathbf{x}_i, y_i\}}, \mathbf{x}_i] > 0$. We define the LOO error as

$$\hat{L}^{(T)}_{LOO} = \frac{1}{T} \sum_{i=1}^{T} L[h^{(T-1)}_{\mathcal{D}_T \setminus \{\mathbf{x}_i, y_i\}}, \mathbf{x}_i] \tag{23}$$

According to the above analysis, we have $\hat{L}^{(T)}_{LOO} \leq \frac{N_{SV}}{T}$, where $N_{SV}$ denotes the number of support vectors. Since the average LOO error $\mathbb{E}[\hat{L}^{(T)}_{LOO}]$ for samples of size $T$ is an unbiased estimate of the average generalization error $\mathbb{E}[L(h^{(T-1)})]$ for samples of size $T-1$ (Mohri, Rostamizadeh, and Talwalkar 2018), we have

$$\mathbb{E}[L(h^{(T-1)})] \leq \frac{\mathbb{E}[N_{SV}]}{T} \tag{24}$$

For a noisy support vector with a high loss, it will become a non-support vector after being corrected according to Lemma 2, which reduces the number of support vectors. Thus, the error bound will be reduced due to (24). □

## Detailed Description of the Datasets and Experimental Setup

In this section, we provide additional details on the datasets and experimental setup.

**Datasets.** In the synthetic data experiments, $1,000$ data instances are evenly generated from two interleaving half-circles. We start AL by randomly picking 5 instances from each class. The rest data instances are put in the candidate pool. We use an SVM with a RBF kernel as the base learner. Both synthetic data generator and SVM follow the scikit-learn implementation (Pedregosa et al. 2011).

The choice of the real-world datasets intents to cover a wide range of AL tasks from diverse domains. We summarize key statistics of the five real-world datasets in Table 1. The two dermatology datasets include text narratives given by a physician when diagnosing skin diseases. The yeast dataset includes biology features for protein localization prediction. Penstroke includes stroke traces of English letters and a number of different writing styles. Autodrive includes readings from different sensors for failures of a running automobile. All features are pre-processed and properly normalized before model fitting. Since the size of Autodrive dataset is too large for AL experiments, we first down sample it to one-tenth of its size.

Table 1: Key Characteristics of Real Datasets

| Dataset | Instance | Feature | Class | Domain |
|---|---|---|---|---|
| Dermatology1 | 800 | 1,391 | 50 | medical |
| Dermatology2 | 868 | 1,554 | 30 | medical |
| Yeast | 1,484 | 8 | 10 | bioinformatic |
| Penstroke | 1,144 | 500 | 26 | image |
| Autodrive | 58,509 | 48 | 11 | automatic system |

**Experimental Settings.** For all experiments, we pick one random instance from each class to start the AL process. We preserve 50% (or at least 500 data samples) of the data for the unlabeled pool and use the rest for testing. To make the fair comparison between different re-sampling approaches, we use SVMs with the same set of hyper-parameters as the base learner. Specifically, we use a RBF kernel with the length scale as 1 and the tolerance factor $C = 10$. We use the same uncertainty based sampling strategy, BvSB (Joshi, Porikli, and Papanikolopoulos 2009), for active sampling.

## Additional Comparisons and Ablation Study

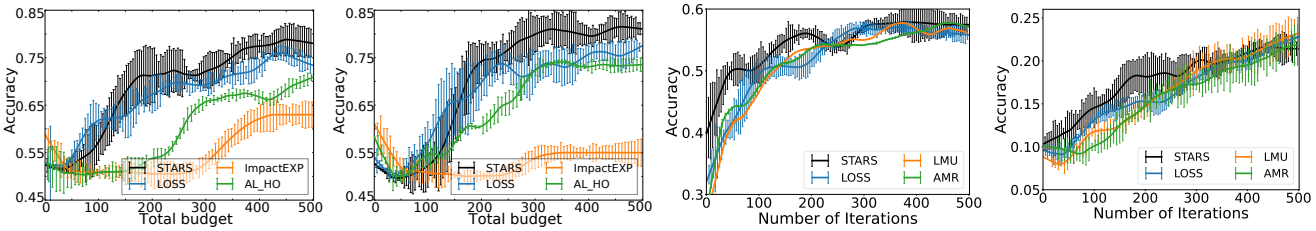
In this section, we provide additional comparisons with existing baselines. Before showing the comparison results, we first clarify the key differences in the problem settings between our approach and methods to be compared, including how the noisy labels are handled and the base models being used. In certain cases, while the settings are different, we are able to make adaptations to show the comparison results. In other cases, where more fundamental differences exist, we provide an in-depth discussion. After that, we show more ablation study results to justify some key design choices of our model. We provide some additional results on active re-sampling for deep learning models in the end.

### Additional Comparison Results

**Clarification of problem setting.** Most works that deal with the noisy data do not consider the learning process of AL. For example, Whitehill et al. (Whitehill et al. 2009) focuses on noisy crowd-sourcing data, but it requires sufficient training and the dataset is also not feasible for our setting. For others that do consider the active learning setting, we summarize the different problem settings in the following key aspects and discuss the related work accordingly:

- Majority vote with identifiable annotators: Sheng et al. (Sheng, Provost, and Ipeirotis 2008) consider the label and model uncertainty (LMU) using a majority voting strategy to tackle the quality issue with different annotators (*i.e.,* identifiable annotators). To make a fair comparison, we assume a uniform annotator quality by fixing the annotators' noise level to be $\alpha$. Besides the annotation method, LMU leverages the geometric average of label uncertainty and model uncertainty to achieve an active re-sampling function. The CDF determines the label uncertainty at the classification threshold of a Beta distribution, which is similar to the $g^{\text{DEC}}$ component in our proposed method. On the other hand, the model uncertainty is obtained from an ensemble method (*e.g.,* a random forest) trained independently from the main model. However, the random forest does not scale well to large datasets, so we implement model uncertainty in the same way as proposed in another baseline (*i.e.,* ARM by Zhao et al. (Zhao, Sukthankar, and Sukthankar 2011)).
- Majority vote with non-identifiable annotators: Zhao et al. (Zhao, Sukthankar, and Sukthankar 2011) develop an absolute majority relabeling (AMR) method, which is essentially a majority voting mechanism with non-identifiable annotators. We adopt the default setting of the work to set all annotator's error rates to $\alpha$. Once a data point is selected for relabeling, the algorithm would ask a sequence of $N \leq 7$ annotators to label the data independently. The label is either determined if over $\frac{N}{2}$ of the annotators reach the agreement or remains unknown otherwise. Except for the annotation method, the active re-sampling function proposed in AMR is also different from our method in two ways. First, the label inconsistency is computed using local label density estimation, whereas in our method, global label density estimation is adopted. Second, AMR does not leverage the temporal information to capture the drastic change of the learning model as we do. These differences are critical to an active learning setting because insufficient training data in active learning might not support a local density estimation and the model behavior changes drastically during the learning process. Mozafari et al. (Mozafari et al. 2014) also uses an uncertainty-based sampling, similar to LMU. However, it focuses on scaling up to very large datasets and would require a large number of labeled per sample, which is not feasible in our setting.
- Binary vs multi-class models: As mentioned in the main paper, one of the most related works, impactEXP (Lin, Mausam, and Weld 2015), is primarily designed for solving binary problems. In the main paper, we follow the authors' recommendation to simplify the re-sampling criterion of impactEXP, which allows us to test the approach on two datasets with a relatively small number of classes. Another related model, referred to as AL-HO, also focuses on the binary setting. However, for AL-HO, its computational cost is still too high to make it meaningful for practical usage in most multi-class AL problems. In order to still make a comparison with these relevant baselines, we choose two representative datasets, Dermatology 1 & 2, and convert them to binary problems by randomly merging half of the classes into a new class.
- Separating data cleaning from the active re-sampling loop: Some recent works, including Younesian et als, Zhao et al. (Younesian et al. 2021; Zhao et al. 2019, 2021), use a pre-trained model to select and correct data from a noisy labeled pool rather than from the training set. After correcting a certain number of instances, the entire pool would be used to update the learning model. This setting requires a large and purely labeled training set to pre-train the model, which is unrealistic in active learning tasks. Furthermore, those works ignore the annotation cost for obtaining the large and noisy labeled pool, making it difficult to have a fair comparison with our method that uses a fixed level of annotation budget.

**Comparison results.** Figures 8a and 8b show that the proposed STARS and LOSS maintain their advantage when the problem reduces to binary classification as compared with impactEXP and AL-HO. It is worth to note that the re-sampling method used in (Lin, Mausam, and Weld 2016) is the same as impactEXP, so we only draw one curve for this baseline. We observe that STARS and LOSS improve at a higher rate especially in the beginning. This is because they are able to leverage the annotations more

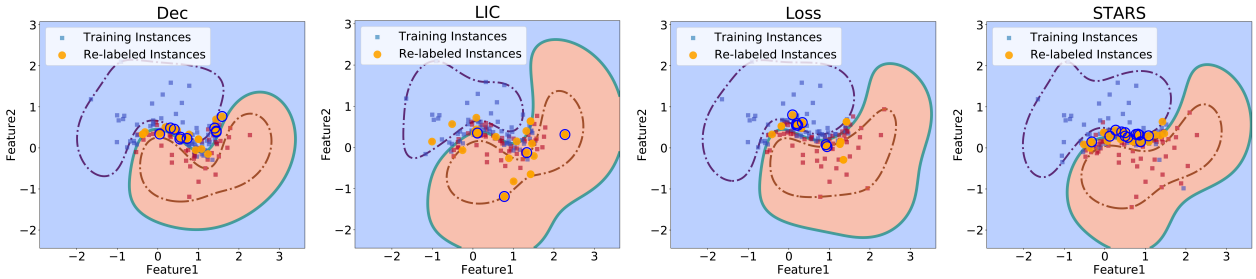



(a) Dermatology 1 (b) Dermatology 2 (c) Yeast (d) Penstroke

Figure 8: Other baselines comparison: (a)-(b): The proposed methods comparing with impactEXP and AL-HO on the binary degenerated datasets. (c)-(d): The proposed method comparing with AMR and LMU on yeast and penstroke datasets.

efficiently than the majority vote. Also, STARS is able to converge to a more accurate model mainly because its spatial-temporal balance guides the model to re-sample in the proper order so that the model is less likely to trap in a local optimal caused by re-sampling `Type-II` data too early.

Figures 8c and 8d show that majority vote methods are good at cleaning the noisy data. But at each re-sampling iteration, we adopt three (virtual) annotators to vote for the label of a given instance, which makes the annotation cost of both AMR and LMU three times higher than the proposed method. The proposed method thus shows its advantage for active learning problems where the annotation budget is strictly limited.




(a) (b) (c) (d)

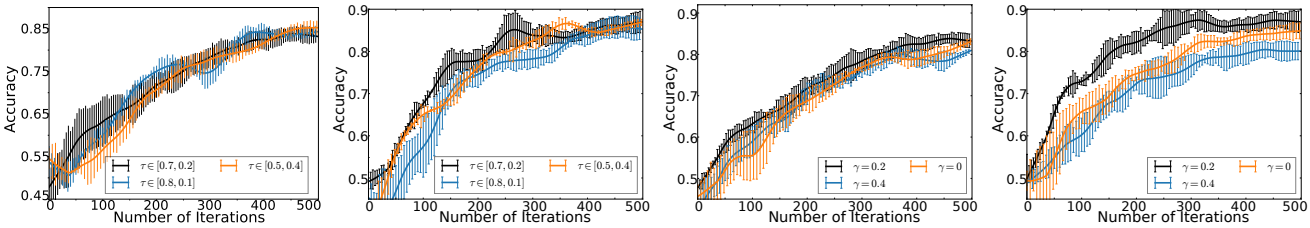
Figure 9: Results on synthetic data: (a)-(d) The snapshots at the early stage of AL showing the decision boundary of the model and the distribution of the re-labeled samples picked by different re-sampling strategies over the entire AL process;

## Additional Ablation Study

**Impact of Re-sampling Order.** Figure 9 shows how re-labeled instances selected by different approaches distributed at the end of the first 100 iterations of AL. Both DEC and STARS have samples locating near the decision boundary, which shows the 'exploitation' behavior as expected. Figures 9b and 9c confirm that early 'exploration' harms the model in two ways. First, we can observe that by revisiting samples far away from the decision boundary, the model only succeeds in correcting a few of them, which means a waste of annotation labor because many clean instances are re-sampled. Second, wrongly visiting a clean sample has a long term impact on the model as the noisy annotator still has probability of a non-zero chance to reverse the true label by mistake. Such negative effect starts to accumulate at the early stage of AL. As a result, even if the decision boundaries and margins of different re-sampling strategies shown in Figure 9 appear to be similar, LIC and LOSS end up with much more distorted decision boundaries as AL proceeds. In contrast, DEC and STARS have less false positive re-sampled instances compared with LIC and LOSS as they focus on exploiting the decision region at the early stage of AL. As a result, DEC and LOSS are less affected by the annotation noise during the process and better converge to the optimal decision boundary eventually.

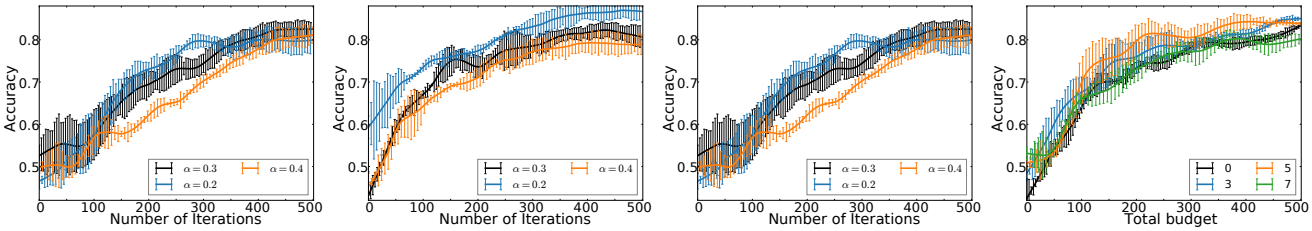
**Impact of hyperparameters.** We choose two representative datasets, Dermatology 1 & 2, to demonstrate the impacts of the two hyper-parameters of STARS: $\tau$ and $\gamma$. Other datasets exhibit a similar trend. Overall, we find the performance of STARTS is largely robust to its hyper-parameters, which is a good property of an AL model as extra annotation budget can be saved to build a validation set for fine tuning these parameters.

In Figures 10a and 10b, we show the impact of $\tau$, which balances two spatial components, DEC and LIC, in STARS. We compare the model performance with different ranges of $\tau$. Since both DEC and LIC are normalized, the most general interval would be $[0, 1]$ which means the model starts with being fully DEC based re-sampling then gradually reduces the weight to zero towards the end of the learning. We observe that STARS performs stable on large intervals close to $[0, 1]$, which means our model



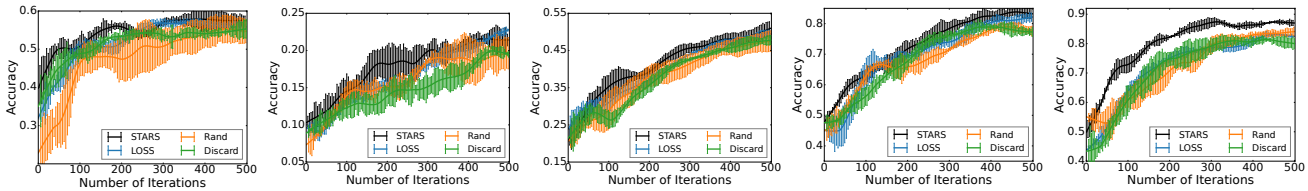

(a) Dermatology 1 (b) Dermatology 2 (c) Dermatology 1 (d) Dermatology 2

Figure 10: (a)-(b): Impact of different intervals of $\tau$ values. (c)-(d): Impact of $\gamma$.




(a) Dermatology 1 (b) Dermatology 2 (c) Dermatology 1 (d) Dermatology 2

Figure 11: (a)(b): Impact of annotation noise level $\alpha$. (c)(d): Impact of re-sampling frequency




(a) Yeast (b) Penstroke (c) Auto Drive (d) Dermatology 1 (e) Dermatology 2

Figure 12: Comparison with discarding the potential noisy samples (the Discard curves actually use 600 labels to make the comparison fair)

is less sensitive to this hyper-parameter. The results also confirm that the ideal re-sampling order of the entire AL process should be exploiting `Type-I` data first and gradually switching to exploring the `Type-II` data in the later phase. Meanwhile, the smallest interval $[0.4, 0.5]$, which indicates a close to random order of re-sampling between `Type-I` and `Type-II` producing the worst performance. This again demonstrates the importance of the re-sampling order.

In Figures 10c and 10d, we show the impact of $\gamma$, which controls the proportion of exponential moving average (EMA) in the re-sampling score. We observe that the EMA component does contribute to better re-sampling as $\gamma$ increase from 0 to 0.2. This shows that the consistency of the STARS score helps reveal the true re-sampling utility. However, since the model is supposed to be significantly evolved along the AL process, the general trend of the STARS score for a specific instance subjects to change inevitably. As a result, one should not put too much weight on the EMA component in case the model over-fits the general dynamics in AL and leads to poorly re-sampling as indicated by the curve with $\gamma = 0.4$.

In Figures 11a and 11b, we demonstrate the effectiveness of the proposed STARS at different annotation noise levels. We observe that as the noise level decreases, the performance of STARTS constantly improves. The proposed model maintains a meaningful learning curve while the noise level is approaching to its extreme (*i.e.,* 0.5). This makes STARS more suitable for challenging knowledge-rich domains, where the annotation noise may be high due to the complex nature of the domain knowledge and replacing/improving the current annotator is expensive.

In Figures 11c and 11d, we show the impact of re-sample frequency. Intuitively, the optimal re-sample frequency is positively related to the annotation noise. A higher annotation noise suggests more frequent re-sampling to clean the training data. We can see that no re-sampling (*i.e.,* zero re-sampling frequency) results in the worst performance because the model has to fit the annotation noise and generalizes poorly on the unseen data. However, a overly high re-sampling frequency hurts the model performance even as much as the no re-sample does. This is because the more frequent the model re-samples, the harder could it find the noise data in the future. When the model cleans up most of the annotation noise in the training dataset, it will start to revisit some clean data and introduce the annotation noise at the rate of $(1 - \alpha)$.

Table 2: Experiment Settings for DNN based Re-sampling

| Dataset | Initial Training | Batch Size | Re-sample Period |
|---|---|---|---|
| MNIST | 100 | 50 | 2 |
| FashionMNIST | 500 | 100 | 4 |
| CIFAR10 | 1000 | 100 | 4 |

**Re-sampling capacity.** The re-sampling capacity prevents the active learner from querying the same instance for too many times. Although previous work identified the issue, they still do not separate the active sampling of unlabeled instances and the re-sampling of labeled instances. We find that both AL-HO and impactEXP suffered from repeatedly re-labeling the same data. So we restrict the number of annotations of the same data instance (excluding the original label) to be 2. Relaxing the re-sampling capacity constraint will significantly hurt the performance of both baselines but STARS remains robust (see Figures 11c and 11d mainly due to its property to avoid labeling the instance with the corrected label as shown in Eqs. (3) and (4). However, we still enforce the same re-sampling capacity to STARS for a fair comparison.

**Re-labeling vs. discarding re-sampled instances.** To reduce the noise in the training data, another option is to directly discard a data instance once it has been re-sampled, instead of re-labeling it. However, as compared with correcting the label, removing the data sample may offer a slower process to make the training data clean. It may also lose some informative data samples that play a critical role in formulating a better decision boundary. Figure 12 confirms this and the model typically converges to a lower accuracy after training. Furthermore, for certain domains (e.g., medical), the unlabeled pool may not be significantly larger and in this case, the training data samples are more valuable.

## Active Re-sampling for Deep Learning Models

We train deep learning models on three benchmark datasets: Fashion MNIST, CIFAR10, and MNIST. The results for Fashion MNIST and CIFAR10 have been presented in the main paper and Figure 13 shows the result on MNIST. Table 2 summarizes the experiment settings. All the networks have simple architecture designs: two convolutional layers and two linear layers. We fix the dimension of the feature embedding $\phi(\mathbf{x})$ to be 50 and use the rectified linear unit as an activation function. We initialize the learning process with a small number of labeled samples. Then in each AL iteration, we update the model by annotating a new batch selected via the margin-based AL strategy. (Note the annotation noise also applies to the initial training set.) The re-sampling strategies are activated every few AL iterations with the re-sampling budget equal to the AL batch size. The coefficient $\tau$ is set to 0.3, and the results are averaged over ten runs. Note that the 'No re-sample' would consume less annotation budget for the same AL batch size than other methods. So we adjust the AL batch size at the rate of $1 + \frac{1}{\text{re-sample period}}$ to make sure a fair comparison between all the methods at the same annotation budget level.

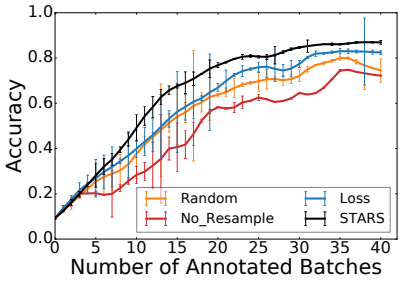



Figure 13: Result on MNIST

## Source code

The source code can be accessed through the following link: https://github.com/ritmininglab/STARS-Spatial-Temporal-Active-Re-Sampling-for-Label-Efficient-Learning-from-Noisy-Annotations.git