A APPENDIX

Algorithm 2 CCS: Coverage-centric Coreset Selection

Input: $\mathbb{S} = \{\text{NLI-Score}(x_i)\}_{i=1}^{n}$: dataset with the NLI-Score for each example;
α: data pruning rate;
β: hard cutoff rate ($\beta \leq 1 - \alpha$); k: the number of strata.
Output: Pruned data $\mathcal{S}^*$

$\mathbb{S}^{'} \leftarrow \mathbb{S} \backslash \{[n * \beta] hardestexamples\}$;
$\mathbb{B}^{'} \leftarrow \{\mathbb{B}_i \backslash \{[n * \beta] hardestexamples\}$;
$R_1, R_2, ...R_k \leftarrow$ Split scores in $\mathbb{S}^{'}$ into k ranges with an even range width ;
$\mathcal{B} \leftarrow \{\mathbb{B}_i, : \mathbb{B}_i$: consists of examples whose scores are in $R_i, i = 1...k\}$;
while $B \leq \varnothing$ **do**
  $R_1, R_2, ...R_k \leftarrow$ Split scores in $\mathbb{S}^{'}$ into k ranges with an even range width ;
  $\mathbb{B}_{min} \leftarrow \underset{\mathbb{B} \in B}{\text{argmin}} |\mathbb{B}|$;
  $m_B \leftarrow \min\{|\mathbb{B}_{min}|, |\frac{m}{|B|}\}$;
  $\mathbb{S}_B \leftarrow$ randomly sample m_B examples from $\mathbb{B}_{min}$;
  $\mathbb{S}_c \leftarrow \mathbb{S}_c \cup \mathbb{S}_B$
  $B \leftarrow B \varnothing \{\mathbb{B}_{min}\}$
  $m \leftarrow m - m_B$
end while
Return $\mathbb{S}^*$

A.1 DETAILS FOR COVERAGE SAMPLING

We leverage the density-based Coverage- centric Coreset Selection (CCS) Zheng et al. (2022b) to trade off the number of hard and easy samples, as outlined in Algorithm 2.. CCS first partitions the dataset into distinct, non-overlapping strata, with each stratum defined by a fixed-length range of NLI-Scores. Though the NLI-Score ranges are uniform across strata, the number of examples within each stratum may vary. CCS then sets an initial budget on the number of examples to be selected from each stratum, based on the desired pruning rate. However, if a particular stratum contains fewer examples than the allocated budget, the excess budget is evenly redistributed across the remaining strata.

A.2 DETAILS FOR SYNTHESIS IMAGENET-N

Since ImageNet-1K is a clean dataset with no known real label noise, we inject the synthetic label noise to construct ImageNet-N. Specifically, we inject asymmetric label noise to mimic real-world label noise following the prior noisy label literature. When targeting an $r\%$ noise ratio for ImageNet-N, we randomly select $r\%$ of the training examples from each class c in ImageNet-1K and then systematically flip their labels to the next consecutive class $c + 1$, i.e., class 0 into class 1, class 1 into class 2, and so on. This deliberate label flipping strategy is reasonable, as consecutive classes are often semantically related, belonging to the same high-level conceptual category. For the selected examples from the final class 1000, we uniquely flip their labels to class 0, completing the circular noise injection process. This holistic label corruption approach serves to recreate the complex, heterogeneous noise characteristics typically encountered in real-world visual recognition datasets, providing a more realistic test environment for our subsequent research endeavors.

A.3 LIMITATION AND SOCIAL IMPACT

A.3.1 LIMITATION

While the RoP has consistently demonstrated its effectiveness in tackling classification tasks involving real-world and synthetically introduced label noise, its applicability on datasets plagued by open-set noise or containing out-of-distribution examples remains to be validated. Moreover, we have not yet assessed the efficacy of RoP when applied to state-of-the-art deep learning models,

Table 7: Summary of the hyperparameters for training SOP+ on the CIFAR-10N/100N,Webvision, and ImageNet-N datasets.

Hyperparamters		CIFAR-10N	CIFAR-100N	WebVision	ImageNet-N
Training Configuration	architecture	PreAct PresNet18	PreAct PresNet18	InceptionResNetV2	ResNet50
	warm-up epoch	10	30	10	1
	training epoch	300	300	100	10
	batch size	128	128	32	32
	learning rate(lr)	0.02	0.02	0.02	0.02
	lr scheduler	Cosine Annealing	Cosine Annealing	MultiStep-50th	MultiStep-50th
SOP+	λ_C	0.9	0.9	0.1	0
	λ_B	0.1	0.1	0	0
	lr for u	10	1	0.1	0.1
	lr for v	100	100	1	1

such as large language models and vision-language architectures. Verifying the performance of RoP across this expanded range of datasets and model paradigms would be immensely valuable, as the need for robust data pruning strategies in the face of annotation noise is a ubiquitous challenge permeating a wide spectrum of real-world applications. Additionally, the Robustness to Perturbations approach has yet to be validated in other realistic data pruning scenarios, such as continual learning and neural architecture search, where the selective retention of informative examples is of paramount importance. We intend to address these crucial research gaps in our future work. By rigorously evaluating the versatility and generalizability of RoP across diverse datasets, model architectures, and application domains, we can further solidify its standing as a powerful and adaptable tool for mitigating the detrimental effects of label noise.

A.3.2 Social Impact

When it comes to preserving model performance while simultaneously reducing computational costs and energy consumption – which can lead to tangible benefits like lowering carbon dioxide emissions – we recognize the inherent challenges involved. However, we firmly believe that the techniques and approaches we explore in this work do not lend themselves to any nefarious or negative applications.

It is our conviction that by optimizing model performance and computational efficiency hand-in-hand, we can pave the way for wider adoption of AI technologies while minimizing their environmental footprint. This dual objective is a key driver behind our research, as we strive to create practical, ethical, and impactful solutions that benefit both the technical and the social realms. We remain steadfast in our commitment to responsible innovation, ensuring that our advancements in machine learning serve the greater good and do not give rise to any concerning social ramifications.

A.4 Experiment details

In Tab. 6, we provide a comprehensive summary of the configurations and hyperparameters employed during the training of the Re-labeling stage. The hyperparameters for the SOP+ method have been favorably configured in accordance with the original publication Liu et al. (2022a). SOP+ involves several key hyperparameters: λ_C for weighting the self-consistency loss, λ_B for weighting the class-balance objective, and learning rates for training its additional variables u and v. Specifically, for CIFAR-10N, we use λ_C = 0.9 and λ_B = 0.1, and set the learning rates of u and v to 10 and 100, respectively. For CIFAR-100N, the hyperparameters are set as λ_C = 0.9, λ_B = 0.1, the learning rates of u and v to 1 and 100, respectively. On the WebVision dataset, we employ λ_C = 0.1 and λ_B = 0, and the learning rates of u and v to 0.1 and 1, respectively. For the ImageNet-N dataset, the hyperparameters are $\lambda_C = 0$, $\lambda_B = 0$, and the learning rates of u and v are 0.1 and 1, respectively.

Furthermore, the hyperparameters of all compared data pruning methods are also favorably configured based on the recommendations from their respective prior works. Specifically, for CIFAR-10N and CIFAR-100N, A PreAct Resnet-18 is trained for 300 epochs using SGD with a momentum of 0.9, a weight decay of 0.0005, and a batch size of 128. The initial learning rate is 0.02, and it is

decayed with a cosine annealing scheduler. For WebVision, InceptionResNetV2 is trained for 100 epochs with a batch size of 32. For ImageNet-N, ResNet-50 model is trained for 50 epochs with a batch size of 64 and an initial learning rate of 0.02, also decayed with a cosine annealing scheduler. All methods are implemented with PyTorch 1.8.0 and executed on NVIDIA Tesla A100 GPUs.