Algorithm 2 Greedy Balanced Neighborhood Confidence (Prune4ReL)

**Input:** \( \tilde{D} \): training set, \( \tilde{D}_j (\subset \tilde{D}) \): set of training examples with a \( j \)-th class, \( s \): target subset size, and \( C(x) \): confidence of \( x \) calculated from a warm-up classifier

1: Initialize \( S \leftarrow \emptyset; \forall x \in \tilde{D}, \hat{C}_N(x) = 0 \)
2: while \( |S| < s \) do
3: \hspace{1em} for \( j = 1 \) to \( c \) do
4: \hspace{2em} \( x = \text{argmax}_{x \in \tilde{D}_j \setminus S} \sigma(\hat{C}_N(x) + C(x)) - \sigma(\hat{C}_N(x)) \)
5: \hspace{1em} \( S \leftarrow S \cup \{x\} \)
6: \hspace{1em} for all \( v \in \tilde{D} \) do
7: \hspace{2em} \( \hat{C}_N(v) \leftarrow \hat{C}_N(v) + \mathbb{1}_{[\text{sim}(x,v) \geq \tau]} \cdot \text{sim}(x,v) \cdot C(x) \)
8: \hspace{2em} if \( |S| = s \) then
9: \hspace{3em} return \( S \)
10: end

**Output:** Final selected subset \( S \)
Algorithm 2 describes the class-balanced version of our greedy algorithm. We first divide the entire training set into $c$ groups according to the noisy label of each example, under the assumption that the number of correctly labeled examples is much larger than that of incorrectly labeled examples in practice [7]. Similar to Algorithm 1 in Section 3.2, we begin with an empty set $S$ and initialize the reduced neighborhood confidence $\bar{C}_N$ to 0 for each training example (Line 1). Then, by iterating class $j$, we select an example $x$ that maximizes the marginal benefit $\sigma(\bar{C}_N(x) + C(x)) - \sigma(\bar{C}_N(x))$ within the set $\tilde{D}_j (\subset \tilde{D})$ and add it to the selected subset $S$ (Lines 3–5). Next, we update the reduced neighborhood confidence $\bar{C}_N$ of each example in the entire training set by using the confidence and the similarity score to the selected example $x$ (Lines 6–7). We repeat this procedure until the size of the selected subset $S$ meets the target size $s$ (Lines 8–9).

C Complete Proof of Theorem 3.5

We complete Theorem 3.5 by proving the monotonicity and submodularity of Eq. (6) in Lemmas C.1 and C.2, under the widely proven fact that the monotonicity and submodularity of a combinatorial objective guarantee the greedy selection to get an objective value within $(1-1/e)$ of the optimum [54].

Lemma C.1. (MONOTONICITY). Our data pruning objective in Eq. (6), denoted as OBJ, is monotonic. Formally,

$$\forall S \subset S', \ OBJ(S) \leq OBJ(S').$$

Proof. For notational simplicity, let $x_i$ be i, $x_j$ be j, and $1_{[\sim \text{sim}(x_i, x_j) \geq \tau]} \cdot \sim \text{sim}(x_i, x_j) \cdot C(x_j)$ be $C_{ij}$. Then, Eq. (13) can be represented as

$$\sum_{i \in \tilde{D}} \sigma\left(\sum_{j \in S} C_{ij} + C_{ix}\right) - \sum_{i \in \tilde{D}} \sigma\left(\sum_{j \in S} C_{ij}\right) \geq \sum_{i \in \tilde{D}} \sigma\left(\sum_{j \in S'} C_{ij} + C_{ix}\right) - \sum_{i \in \tilde{D}} \sigma\left(\sum_{j \in S'} C_{ij}\right).$$

Proving Eq. (14) is equivalent to proving the decomposed inequality for each example $x_i \in \tilde{D}$,

$$\sigma\left(\sum_{j \in S} C_{ij} + C_{ix}\right) - \sigma\left(\sum_{j \in S} C_{ij}\right) \geq \sigma\left(\sum_{j \in S'} C_{ij} + C_{ix}\right) - \sigma\left(\sum_{j \in S'} C_{ij}\right).$$

Since $S, S' \cap S, \text{ and } \{x\}$ do not intersect each other, we can further simplify Eq. (15) with independent scalar variables such that

$$\sigma(a + \epsilon) - \sigma(a) \geq \sigma(a + b + \epsilon) - \sigma(a + b),$$

where $a = \sum_{j \in S} C_{ij}, b = \sum_{j \in S' \cap S} C_{ij}, \text{ and } \epsilon = C_{ix}$.

Since the utility function $\sigma$ is concave, by the definition of concavity,

$$\frac{\sigma(a + \epsilon) - \sigma(a)}{(a + \epsilon - a)} \geq \frac{\sigma(a + b + \epsilon) - \sigma(a + b)}{(a + b + \epsilon - (a + b))}.$$  

The denominators of both sides of the inequality become $\epsilon$, and Eq. (17) can be transformed to Eq. (16). Therefore, Eq. (16) should hold, and $OBJ(S \cup \{x\}) - OBJ(S) \geq OBJ(S' \cup \{x\}) - OBJ(S')$. 

16
Table 7: Summary of the hyperparameters for training SOP+ and DivideMix on the CIFAR-10N/100N, Webvision, and Clothing-1M datasets.

<table>
<thead>
<tr>
<th>Hyperparameters</th>
<th>CIFAR-10N</th>
<th>CIFAR-100N</th>
<th>WebVision</th>
<th>Clothing-1M</th>
</tr>
</thead>
<tbody>
<tr>
<td>architecture</td>
<td>PreAct PresNet18</td>
<td>PreAct PresNet18</td>
<td>InceptionResNetV2</td>
<td>ResNet-50 (pretrained)</td>
</tr>
<tr>
<td>warm-up epoch</td>
<td>10</td>
<td>30</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>training epoch</td>
<td>300</td>
<td>300</td>
<td>100</td>
<td>10</td>
</tr>
<tr>
<td>batch size</td>
<td>128</td>
<td>128</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>learning rate (lr)</td>
<td>0.02</td>
<td>0.02</td>
<td>0.02</td>
<td>0.002</td>
</tr>
<tr>
<td>lr scheduler</td>
<td>Cosine Annealing</td>
<td>Cosine Annealing</td>
<td>MultiStep-50th</td>
<td>MultiStep-5th</td>
</tr>
<tr>
<td>weight decay</td>
<td>$5 \times 10^{-4}$</td>
<td>$5 \times 10^{-4}$</td>
<td>$5 \times 10^{-4}$</td>
<td>$5 \times 10^{-4}$</td>
</tr>
</tbody>
</table>

DivideMix

- $\lambda_U$: 1
- $\kappa$: 0.5
- $T$: 0.5
- $\gamma$: 4
- M: 2

SOP+

- $\lambda_C$: 0.9
- $\lambda_B$: 0.1
- $\lambda$ for $u$: 100
- $\lambda$ for $v$: 1

By Lemmas C.1 and C.2, the monotonicity and submodularity of Eq. (6) hold. Therefore, Eq. (7) naturally holds, and this concludes the proof of Theorem 3.5.

D Details for Constructing 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 [10]. When a target noise ratio of ImageNet-N is $r\%$, we randomly select $r\%$ of the training examples for each class $c$ in ImageNet-1K and then flip their label into class $c + 1$, i.e., class 0 into class 1, class 1 into class 2, and so on. This flipping is reasonable because consecutive classes likely belong to the same high-level category. For the selected examples with the last class 1000, we flip their label into class 0.

E Implementation Details

Table 7 summarizes the overall training configurations and hyperparameters used to train the two Relabeling models, DivideMix and SOP+. The hyperparameters for DivideMix and SOP+ are favorably configured following the original papers. DivideMix [13] has multiple hyperparameters: $\lambda_U$ for weighting the self-consistency loss, $\kappa$ for selecting confidence examples, $T$ for sharpening prediction probabilities, $\gamma$ for controlling the Beta distribution, and $M$ for the number of augmentations. For both CIFAR-10N and CIFAR-100N, we use $\lambda_U = 1$, $\kappa = 0.5$, $T = 0.5$, $\gamma = 4$, and $M = 2$. For Clothing-1M, we use $\lambda_U = 0.1$, $\kappa = 0.5$, $T = 0.5$, $\gamma = 0.5$, and $M = 2$. SOP+ [33] also involves several hyperparameters: $\lambda_C$ for weighting the self-consistency loss, $\lambda_B$ for weighting the class-balance, and learning rates for training its additional variables $u$ and $v$. For CIFAR-10N, we use $\lambda_C = 0.9$ and $\lambda_B = 0.1$, and set the learning rates of $u$ and $v$ to 10 and 100, respectively. For CIFAR-100N, we use $\lambda_C = 0.9$ and $\lambda_B = 0.1$, and set the learning rates of $u$ and $v$ to 1 and 100, respectively. For WebVision, we use $\lambda_C = 0.1$ and $\lambda_B = 0$, and set the learning rates of $u$ and $v$ to 0.1 and 1, respectively.

Besides, the hyperparameters for all data pruning algorithms are also favorably configured following the original papers. For Forgetting [14], we calculate the forgetting event of each example throughout the warm-up training epochs in each dataset. For GraNd [15], we train ten different warm-up classifiers and calculate the per-sample average of the norms of the gradient vectors obtained from the ten classifiers.
F  Limitation and Potential Negative Societal Impact

Limitation. Although Prune4ReL has demonstrated consistent effectiveness in the classification task with real and synthetic label noises, we have not validated its applicability on datasets with open-set noise or out-of-distribution examples [55, 56]. Also, we have not validated its applicability to state-of-the-art deep learning models, such as large language models [3] and vision-language models [4]. This verification would be valuable because the need for data pruning in the face of annotation noise is consistently high across a wide range of real-world tasks. In addition, Prune4ReL has not been validated in other realistic applications of data pruning, such as continual learning [57] and neural architecture search [58]. In these scenarios, selecting informative examples is very important, and we leave them for future research.

Potential Negative Societal Impact. We consider how to preserve the model performance while reducing the computation costs, which can even reduce substantial energy consumption, e.g., CO₂ emission. Hence, it is hard to apply to any negative applications, and there is no discussion of potential negative social impact.