ProSelfLC: Progressive Self Label Correction for Training Robust Deep Neural Networks

Xinshao Wang *
Anyvision Research Team
Queen's University Belfast
xwang39 at qub.ac.uk

Yang Hua
Queen's University Belfast
y.hua at qub.ac.uk

Elyor Kodirov
Anyvision Research Team
elyor at anyvision.co

Neil M. Robertson
Anyvision Research Team
Queen's University Belfast
n.robertson at qub.ac.uk

Abstract

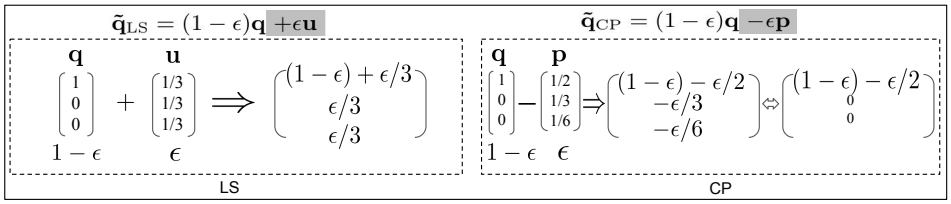
We systematically study popular target modification approaches in supervised learning. We show that they can be connected mathematically through entropy and KL divergence. This uncovers that some methods penalise while the others reward low entropy. Additionally, some of them are suboptimal because they do not leverage the knowledge of a model itself; some rely on extra learners or stage-wise training that may require a human intervention thus being difficult to optimise; most importantly, there does not exist an automatic way to decide how much we trust a predicted label distribution, let alone exploiting it. To resolve these issues, taking two well-accepted expertise: deep neural networks learn meaningful patterns before fitting noise [1] and minimum entropy regularisation principle [2], we propose a simple end-to-end method named ProSelfLC, which is endorsed by long learning time and high prediction confidence. Specifically, given a data point, we progressively trust more its predicted label distribution than its annotated one if a model has been trained for a long time and outputs a highly confident prediction (low entropy). By extensive experiments, we show: (1) ProSelfLC can revise an example's one-hot label distribution by adding the perceptual similarity structure information so that its learning target becomes structured and soft; (2) When being applied to noisy labels, it can correct their semantic classes; (3) It outperforms existing methods with the lowest entropy, which indicates it is right for a learner to be confident in correct patterns.

1 Introduction

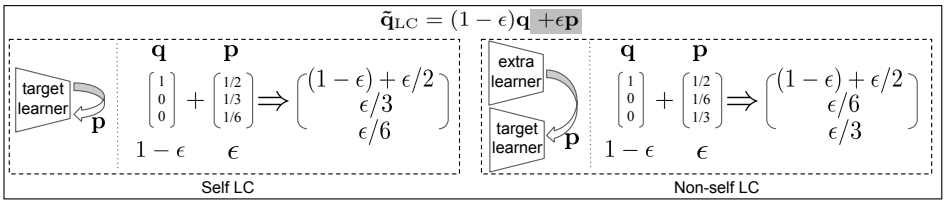
There exist many target (label) modification approaches. They are proposed for different purposes and can be roughly divided into two groups: (1) Output regularisation (OR). It is proposed to penalise overconfident predictions for regularising deep neural networks. It includes label smoothing (LS) [3, 4] and confidence penalty (CP) [5]; (2) Label correction (LC). On the one hand, LC regularises neural networks by adding the perceptual similarity structure information over training classes into

*For any specific discussion or potential future collaboration, please feel free to contact me. As a young researcher, your interest and star (citation) will mean a lot for me and my collaborators. For source codes, we are happy to provide if there is a request conditioned on academic use only and kindness to cite our work.

Preprint. Open discussion: we show it's fine for a learner to be confident towards a correct low-entropy status. Then more future research attention should be paid to the definition of correct knowledge, as in general we accept, human annotations used for learning supervision may be biased, subjective, and wrong.



(a) OR includes LS [3] and CP [5]. LS softens a learning target by adding a uniform label distribution. CP changes the probability 1 to a smaller value $1-\epsilon$ in the one-hot target. The double-ended arrow means conceptual equivalence, because an output probability is definitely non-negative after a softmax layer.



(b) LC contains Self LC [7–9] and Non-self LC [6]. The convex combination parameter ϵ defines how much a predicted label distribution is trusted.

Figure 1: Target modification includes OR (LS and CP), and LC (Self LC and Non-self LC). Assume there are three training classes. $\mathbf{q}$ is the one-hot target. $\mathbf{u}$ is a uniform label distribution. $\mathbf{p}$ denotes a predicted label distribution. The target combination parameter is $\epsilon \in [0, 1]$. Our mathematical study of them is presented in Section 3 and Table 1.

one-hot label distributions so that the learning targets become *structured and soft*. On the other hand, it can *correct the semantic classes* of noisy label distributions. LC can be further divided into two subgroups: Non-self LC and Self LC. The former requires extra learners, while the latter relies on the model itself. A typical approach of Non-self LC is knowledge distillation (KD) since it exploits the predictions of other model(s), usually termed teacher(s) [6]. Popular methods for Self LC include Pseudo-Label [7], bootstrapping (Boot-soft and Boot-hard) [8], and Joint Optimisation (Joint-soft and Joint-hard) [9]. The differences over these groups are shown in Figure 1.

In this paper, we find that they can be unified mathematically from the angle of target modification. Specifically, in all methods, the learning target of a data point is defined by combining *a one-hot label distribution and its corresponding prediction or a predefined distribution*. This finding leads us to uncover that OR-based methods penalise low entropy while LC-based ones reward low entropy. Furthermore, we reveal the drawbacks of existing approaches: (1) OR-based methods naively penalise confident outputs without leveraging easily accessible knowledge from other learners or itself, which makes them suboptimal (Figure 1a); (2) Non-self LC relies on 'good' auxiliary models to generate predictions (Figure 1b). (3) Self LC is the most appealing because it requires no extra learners to revise learning targets. However, there is a core question that is not well answered:

In Self LC, how much do we trust a learner to leverage its knowledge?

As shown in Figure 1b, in Self LC, for a data point, we have two labels: a predefined one-hot $\mathbf{q}$ and a predicted structured $\mathbf{p}$. Its learning target is $(1-\epsilon)\mathbf{q} + \epsilon\mathbf{p}$, i.e., a trade-off between $\mathbf{q}$ and $\mathbf{p}$. Therefore, ϵ defines the trust score of a learner. In existing methods, when setting ϵ, it is not considered that a model's knowledge grows as the training time goes, so that they are suboptimal. For example, in bootstrapping [8], ϵ is fixed throughout the whole training process. Recently, joint optimisation [9] fully trusts a learner by setting $\epsilon = 1$, but use stage-wise training to gradually train the model. Although it shows better performance than bootstrapping, stage-wise training requires a significant human intervention and is time-consuming in practice.

Based on the analysis and aforementioned issues, we propose a novel method named Progressive Self Label Correction (ProSelfLC). ProSelfLC is robust and end-to-end trainable. It can refine the learning targets such that they capture the similarity structure of training classes without an extra learner. Most importantly, it can decide how to modify the target progressively and adaptively as training goes. *The underlying principle of ProSelfLC is*: (1) When a learner starts to learn, it trusts the supervision from human annotations. This idea is inspired by the paradigm that deep models

learn simple meaningful patterns before fitting noise, even when severe label noise exists in human annotations [1]. (2) As a learner attains confident knowledge as time goes, we leverage its confident knowledge to correct labels. This is surrounded by minimum entropy regularisation, which has been widely evaluated in unsupervised and semi-supervised scenarios [10, 2].

Finally, we summarise our main contributions:

- We provide a comprehensive mathematical study on popular target modification methods, and show that they can be unified under a single framework.
- We reveal the drawbacks of existing methods. As a solution, we propose ProSelfLC, a method to trust a learner progressively and adaptively. Furthermore, ProSelfLC is appealing in that it can: (1) add and correct the perceptual similarity structure information over training classes so that the learning targets become structured thus more informative; (2) correct the semantic classes of potential noisy label distributions.
- We conduct extensive experiments demonstrating the effectiveness of ProSelfLC on several tasks: standard image classification and robust image classification when noisy labels exist.

2 Related Work

In addition to adding the similarity structure information, target modification is demonstrated to be effective in coping with label noise. Besides, addressing label noise is connected to the semi-supervised learning [10, 11, 7], where only a subset of training examples are annotated, leading to *missing labels*. Then the key to semi-supervised training is to reliably fill them. *When these missing labels are incorrectly filled, the challenge of semi-supervised learning changes to noisy labels.* For a further comparison, in semi-supervised learning, the annotated set is clean and trustful, so that the label noise only exists in the unannotated set. While in our general setting, we are not given the information whether an example is trusted or not, thus being even more challenging. We summarise existing approaches for solving label noise: (1) Loss correction, where we are given or we need to estimate a noise-transition matrix, which defines the distribution of noise labels [12–19]. A noise-transition matrix is difficult and complex to estimate in practice; (2) Exploiting an auxiliary trusted training set to differentiate examples [20–22]. This requires extra annotation cost; (3) Co-training strategies, which train two or more learners [23–28] and exploit their 'disagreement' information to differentiate data points; (4) Label engineering methods [29, 7–9, 16], which relate to our focus in this work. Their idea is to annotate unlabelled samples and correct noisy labels.

Intrinsically, Self LC can be interpreted as self knowledge distillation [5], which exploits the knowledge of a model itself. We use the term self label correction instead of self knowledge distillation for two reasons: (1) self label correction is more descriptive since we modify labels; (2) the scope of self knowledge distillation now becomes beyond label modification, e.g., it means forcing two inputs of the same class to have similar output distributions in [30, 31], while making shallow classifiers approximate the deepest classifier in [32].

3 Mathematical analysis of existing methods

Let $\mathbf{X} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ represent N training examples, where $(\mathbf{x}_i, y_i)$ denotes $i-$th sample with input $\mathbf{x}_i \in \mathbb{R}^D$ and label $y_i \in \{1, 2, ..., C\}$. C is the number of classes. A deep neural network z consists of an embedding network $f(\cdot) : \mathbb{R}^D \rightarrow \mathbb{R}^K$ and a linear classifier $g(\cdot) : \mathbb{R}^K \rightarrow \mathbb{R}^C$, i.e., $\mathbf{z}_i = z(\mathbf{x}_i) = g(f(\mathbf{x}_i)) : \mathbb{R}^D \rightarrow \mathbb{R}^C$. For the brevity of analysis, we take a data point and omit its subscript so that it is denoted by $(\mathbf{x}, y)$. The linear classifier is usually the last fully-connected layer. Its output is named logit vector $\mathbf{z} \in \mathbb{R}^C$. We produce its classification probabilities $\mathbf{p}$ by normalising the logits using a softmax function:

$$p(j|\mathbf{x}) = \exp(\mathbf{z}_j) / \sum\nolimits_{m=1}^{C} \exp(\mathbf{z}_m), \tag{1}$$

where $p(j|\mathbf{x})$ is the probability of $\mathbf{x}$ belonging to class j. Its corresponding ground-truth is usually denoted by a one-hot representation $\mathbf{q}$: $q(j|\mathbf{x}) = 1$ if $j = y$, $q(j|\mathbf{x}) = 0$ otherwise.

3.1 Analysis from the perspective of label modification and entropy

CCE with one-hot label representations. For any input $(\mathbf{x}, y)$, the minimisation objective of CE is:

$$L_{\mathrm{CCE}}(\mathbf{q}, \mathbf{p}) = \mathrm{H}(\mathbf{q}, \mathbf{p}) = \mathrm{E}_{\mathbf{q}}(-\log\ \mathbf{p}), \tag{2}$$

where $\mathrm{E}_{\mathbf{q}}(-\log\ \mathbf{p})$ denotes the expectation of negative log-likelihood, and $\mathbf{q}$ is the probability mass function, $\mathrm{H}(\cdot, \cdot)$ represents the cross entropy.

Label smoothing. In LS [3, 6, 4], we soften one-hot targets by adding a uniform distribution: $\tilde{\mathbf{q}}_{\mathrm{LS}} = (1-\epsilon)\mathbf{q} + \epsilon\mathbf{u}$, $\mathbf{u} \in \mathbb{R}^C$, and $\forall j, \mathbf{u}_j = \frac{1}{C}$. The minimisation objective of $(\mathbf{x}, y)$ becomes:

$$L_{\mathrm{CCE+LS}}(\mathbf{q}, \mathbf{p}; \epsilon) = \mathrm{H}(\tilde{\mathbf{q}}_{\mathrm{LS}}, \mathbf{p}) = \mathrm{E}_{\tilde{\mathbf{q}}_{\mathrm{LS}}}(-\log\ \mathbf{p}) = (1-\epsilon)\mathrm{H}(\mathbf{q}, \mathbf{p}) + \epsilon\mathrm{H}(\mathbf{u}, \mathbf{p}). \tag{3}$$

Confidence penalty. CP [5] penalises highly confident predictions, and we derive it to a target revising format:

$$L_{\mathrm{CCE+CP}}(\mathbf{q}, \mathbf{p}; \epsilon) = (1-\epsilon)\mathrm{H}(\mathbf{q}, \mathbf{p}) - \epsilon\mathrm{H}(\mathbf{p}, \mathbf{p}) = \mathrm{E}_{(\mathbf{1}-\epsilon)\mathbf{q}-\epsilon\mathbf{p}}(-\log\ \mathbf{p}). \tag{4}$$

We see that CP modifies its target to be: $\tilde{\mathbf{q}}_{\mathrm{CP}} = (1-\epsilon)\mathbf{q} - \epsilon\mathbf{p}$. *Confidence penalty was not understood and interpreted from the perspective of target modification.*

Label correction. As illustrated in Figure 1, LC is a family of algorithms, where a one-hot label distribution is modified to a convex combination of itself and its predicted label distribution:

$$\tilde{\mathbf{q}}_{\mathrm{LC}} = (1-\epsilon)\mathbf{q} + \epsilon\mathbf{p} \ \Rightarrow\ L_{\mathrm{CCE+LC}}(\mathbf{q}, \mathbf{p}; \epsilon) = \mathrm{H}(\tilde{\mathbf{q}}_{\mathrm{LC}}, \mathbf{p}) = (1-\epsilon)\mathrm{H}(\mathbf{q}, \mathbf{p}) + \epsilon\mathrm{H}(\mathbf{p}, \mathbf{p}). \tag{5}$$

3.2 Analysis from the perspective of KL Divergence

We can rewrite CCE, LS, CP, and LC from the viewpoint of KL divergence [33], according to $\mathrm{KL}(\mathbf{q}||\mathbf{p}) = \mathrm{H}(\mathbf{q}, \mathbf{p}) - \mathrm{H}(\mathbf{q}, \mathbf{q})$, $\mathrm{KL}(\cdot||\cdot)$ denotes the KL divergence. We rewrite CCE in Eq (2):

$$L_{\mathrm{CCE}}(\mathbf{q}, \mathbf{p}) = \mathrm{H}(\mathbf{q}, \mathbf{p}) = \mathrm{KL}(\mathbf{q}||\mathbf{p}) + \mathrm{H}(\mathbf{q}, \mathbf{q}) = \mathrm{KL}(\mathbf{q}||\mathbf{p}). \tag{6}$$

Note that we have $\mathrm{H}(\mathbf{q}, \mathbf{q}) = 0$ because $\mathbf{q}$ is a one-hot distribution. We rewrite LS in Eq (3):

$$\begin{aligned} L_{\mathrm{CCE+LS}}(\mathbf{q}, \mathbf{p}; \epsilon) &= (1-\epsilon)\mathrm{KL}(\mathbf{q}||\mathbf{p}) + \epsilon\mathrm{KL}(\mathbf{u}||\mathbf{p}) + \epsilon\mathrm{H}(\mathbf{u}, \mathbf{u}) \\ &= (1-\epsilon)\mathrm{KL}(\mathbf{q}||\mathbf{p}) + \epsilon\mathrm{KL}(\mathbf{u}||\mathbf{p}) + \epsilon \cdot \text{constant}. \end{aligned} \tag{7}$$

$\mathrm{H}(\mathbf{u}, \mathbf{u})$ is a constant. Similarly, we rewrite CP in Eq (4):

$$\begin{aligned} L_{\mathrm{CCE+CP}}(\mathbf{q}, \mathbf{p}; \epsilon) &= (1-\epsilon)\mathrm{KL}(\mathbf{q}||\mathbf{p}) - \epsilon(\mathrm{H}(\mathbf{p}, \mathbf{u}) - \mathrm{KL}(\mathbf{p}||\mathbf{u})) \\ &= (1-\epsilon)\mathrm{KL}(\mathbf{q}||\mathbf{p}) + \epsilon\mathrm{KL}(\mathbf{p}||\mathbf{u}) - \epsilon \cdot \text{constant}. \end{aligned} \tag{8}$$

$\mathrm{H}(\mathbf{p}, \mathbf{u}) = \mathrm{H}(\mathbf{u}, \mathbf{u}) =$ constant. Therefore, LC in Eq (5) can also be rewritten:

$$L_{\mathrm{CCE+LC}}(\mathbf{q}, \mathbf{p}; \epsilon) = (1-\epsilon)\mathrm{KL}(\mathbf{q}||\mathbf{p}) - \epsilon\mathrm{KL}(\mathbf{p}||\mathbf{u}) + \epsilon \cdot \text{constant}. \tag{9}$$

3.3 Comparison and discussion

We summarise CCE, LS, CP and LC in Table 1, so that we can easily see their mathematical differences. The constant in KL divergence is ignored. Based on the summary, we observe that:

Remark 1 *(LS and CP behave differently in terms of target modification)*. As highlighted in Table 1, LS softens a learning target by adding a non-meaningful uniform distribution. While in CP, the target becomes a one-hot distribution subtracts its corresponding prediction. From the perspective of label definition, *CP is against intuition because these zero-value positions in CCE are filled with negative values in CP.* However, we can further interpret them based on the illustration in Figure 1: (a) LS changes every probability in the label vector, i.e., enlarges zero values while makes one value smaller; (2) CP only makes one value smaller.

Remark 2 *(LS and CP are consistent in terms of KL divergence)*. Both are proposed to avoid over-confident predictions [5]. LS adds $\mathrm{KL}(\mathbf{u}||\mathbf{p})$ while CP adds $\mathrm{KL}(\mathbf{p}||\mathbf{u})$ for regularisation.

Remark 3 *(Only LC exploits informative information and has the ability to correct labels, while LS and CP only relax the hard targets)*. By correcting labels, we mean: (1) $\mathbf{p}$ provides meaningful information about an example's relative probabilities of being different training classes; (2) If ϵ is large, and $\mathbf{p}$ is confident in predicting a different class, i.e., $\arg\max_j \mathbf{p}(j|\mathbf{x}) \neq \arg\max_j \mathbf{q}(j|\mathbf{x})$, $\tilde{\mathbf{q}}_{\mathrm{LS}}$ defines a different semantic class from $\mathbf{q}$.

Table 1: Summary of CCE, LS, CP and LC from the angle of target modification and KL divergence.

	CCE	LS	CP	LC
Learning Target	$\mathbf{q}$	$\tilde{\mathbf{q}}_{\mathrm{LS}} = (1-\epsilon)\mathbf{q} + \epsilon\mathbf{u}$	$\tilde{\mathbf{q}}_{\mathrm{CP}} = (1-\epsilon)\mathbf{q} - \epsilon\mathbf{p}$	$\tilde{\mathbf{q}}_{\mathrm{LC}} = (1-\epsilon)\mathbf{q} + \epsilon\mathbf{p}$
Cross Entropy	$\mathrm{E}_{\mathbf{q}}(-\log\ \mathbf{p})$	$\mathrm{E}_{\tilde{\mathbf{q}}_{\mathrm{LS}}}(-\log\ \mathbf{p})$	$\mathrm{E}_{\tilde{\mathbf{q}}_{\mathrm{CP}}}(-\log\ \mathbf{p})$	$\mathrm{E}_{\tilde{\mathbf{q}}_{\mathrm{LC}}}(-\log\ \mathbf{p})$
KL Divergence	$\mathrm{KL}(\mathbf{q}\|\|\mathbf{p})$	$(1-\epsilon)\mathrm{KL}(\mathbf{q}\|\|\mathbf{p}) + \epsilon\mathrm{KL}(\mathbf{u}\|\|\mathbf{p})$	$(1-\epsilon)\mathrm{KL}(\mathbf{q}\|\|\mathbf{p}) + \epsilon\mathrm{KL}(\mathbf{p}\|\|\mathbf{u})$	$(1-\epsilon)\mathrm{KL}(\mathbf{q}\|\|\mathbf{p}) - \epsilon\mathrm{KL}(\mathbf{p}\|\|\mathbf{u})$

4 ProSelfLC: Progressive and Adaptive Label Correction Accredited by Long Learning Time and Low Entropy

4.1 Beyond semantic class: the similarity structure defined by a label distribution

Definition 1 (*Semantic Class*). Given a target label distribution $\tilde{\mathbf{q}}(\mathbf{x}) \in \mathbb{R}^C$, the semantic class is defined by $\arg\max_j \tilde{\mathbf{q}}(j|\mathbf{x})$, i.e., the class whose probability is the largest.
In LS, the target is $\tilde{\mathbf{q}}_{\mathrm{LS}} = (1-\epsilon)\mathbf{q} + \epsilon\mathbf{u}$. For any $0 \leq \epsilon < 1$, the semantic class is not changed, because $1 - \epsilon + \epsilon * \frac{1}{C} > \epsilon * \frac{1}{C}$. CP does not change the semantic class either.
Definition 2 (*Similarity Structure*). In CCE, LS and CP, a data point has an identical probability of belonging to other classes except for the semantic class. Instead, in soft versions of LC, a target label distribution captures the probability difference of an example being predicted to every class. We define it to be the similarity structure of one example belonging to different training classes.

In the literature and popular practice, i.e., CCE, LS and CP, the semantic class is considered while the similarity structure is ignored. The reason is simply because it is difficult to annotate the similarity structure of every data point, especially when the number of classes is large. However, recent progress demonstrates there are some effective approaches to define the similarity structure of data points without human annotation: (1) In KD, a teacher model, e.g., a pre-trained model or a mixture of experts, can provide a student model the similarity structure over all training classes [6, 4]; (2) In Self LC, e.g., Boot-soft, a model helps itself by exploiting the knowledge it has learned so far. We focus on improving the training of a single model, so that we choose to improve Self LC in this work.

4.2 Human annotations and predicted label distributions, which should we trust more?

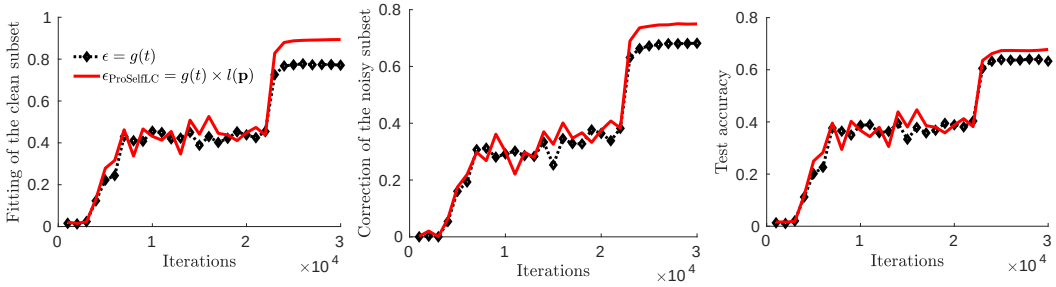
In Self LC, ϵ indicates how much a predicted label distribution is trusted. In ProSelfLC, it is set adaptively according to learning time t and prediction entropy $\mathrm{H}(\mathbf{p})$. For any $\mathbf{x}$, we summarise:

$$\begin{aligned} & L_(\tilde{\mathbf{q}}_{\mathrm{ProSelfLC}}, \mathbf{p}; \epsilon_{\mathrm{ProSelfLC}}) = \mathrm{H}(\tilde{\mathbf{q}}_{\mathrm{ProSelfLC}}, \mathbf{p}) = \mathrm{E}_{\tilde{\mathbf{q}}_{\mathrm{ProSelfLC}}}(-\log\ \mathbf{p}); \\ & \tilde{\mathbf{q}}_{\mathrm{ProSelfLC}} = (1-\epsilon_{\mathrm{ProSelfLC}})\mathbf{q} + \epsilon_{\mathrm{ProSelfLC}}\mathbf{p}, \qquad \epsilon_{\mathrm{ProSelfLC}} = g(t) \times l(\mathbf{p}); \\ & g(t) = h(t/\Gamma - 0.5) \in (0,1), \qquad l(\mathbf{p}) = 1 - \mathrm{H}(\mathbf{p})/\mathrm{H}(\mathbf{u}) \in (0,1). \end{aligned} \tag{10}$$

t and Γ are the iteration (time) counter and the number of total iterations, respectively. $h(\cdot)$ is a logistic function. We explain $g(t)$ and $l(\mathbf{p})$ as follows:
The global trust score $g(t)$: it denotes how much we trust a learner and is decided by the learning time t, being independent of specific data points. $g(t)$ grows as t rises.
The local trust score $l(\mathbf{p})$: $\mathrm{H}(\mathbf{p})$ represents a learner's confidence in its prediction of a concrete example. A lower entropy represents a higher confidence. The uniform distribution $\mathbf{u}$ has the highest entropy so that we use $\mathrm{H}(\mathbf{u})$ for normalisation. $l(\mathbf{p})$ rises as $\mathrm{H}(\mathbf{p})$ becomes lower.
Design reason. (1) Regarding $g(t)$, in the earlier learning phase, i.e., $t < \Gamma/2$, $g(t) < 0.5 \Rightarrow \epsilon_{\mathrm{ProSelfLC}} < 0.5, \forall \mathbf{p}$, so that the human annotations dominate and ProSelfLC only modifies the similarity structure. This is because when a learner does not see the training data for enough times, we assume it is not trained well, which is the most elementary concept in deep learning. *Most importantly, more randomness exists at the earlier phase, as a result, the learner may output a wrong confident prediction.* In our design, $\epsilon_{\mathrm{ProSelfLC}} < 0.5, \forall \mathbf{p}$ can assuage the bad impact of such unexpected cases. When it comes to the later learning phase, i.e., $t > \Gamma/2$, we have $g(t) > 0.5$, which means overall we give enough credits to a learner as it has been trained for more than the half of total iterations. (2) Regarding $l(\mathbf{p})$, we discuss its effect in the later learning phase when it becomes more meaningful. If $\mathbf{p}$ is not confident, $l(\mathbf{p})$ will be large, then $\epsilon_{\mathrm{ProSelfLC}}$ will be small, which means *we choose to trust a one-hot annotation more when its prediction is of high entropy, so that we can further reduce the entropy of output distributions*. In this case, ProSelfLC only modifies the similarity structure.

Table 2: Case analysis of ProSelfLC throughout the learning process. We show the $\epsilon_{\text{ProSelfLC}}$ of six cases. Here, we use concrete values, i.e., 0.1 and 0.9, for more concise interpretation. We bold the special case, where an output distribution $\mathbf{p}$ is confident but inconsistent with $\mathbf{q}$.

		$l(\mathbf{p})$: Consistency is defined by whether $\mathbf{p}$ and $\mathbf{q}$ share the semantic class or not.		
	$g(t)$	0.1(non-confident)	0.9(confident+consistent)	0.9(confident+inconsistent)
The earlier phase	0.1	0.01	0.09	0.09
The later phase	0.9	0.09	0.81	**0.81**(correct the semantic class)



(a) Correct fitting. (b) Semantic class correction (c) Generalisation.

Figure 2: Ablation study of the design ϵ on CIFAR-100 with asymmetric label noise $r = 0.4$. For data-dependent items, mean results are reported.

Beyond, when $\mathbf{p}$ is highly confident, there are two fine cases: If $\mathbf{p}$ is consistent with $\mathbf{q}$ in the semantic class, ProSelfLC only modifies the similarity structure too; If they are inconsistent, ProSelfLC further corrects the semantic class of a human annotation. Ablation study of our design is in Figure 2, where $\epsilon_{\text{ProSelfLC}}$ consistently performs the best when multiple metrics are reported.

We conduct the case analysis on ProSelfLC in Table 2 and summarise its core tactics as follows:
Correct the similarity structure for every data point in all cases. Given any data point $\mathbf{x}$, by a convex combination of $\mathbf{p}$ and $\mathbf{q}$, we add the information about its relative probabilities of being different training classes using the knowledge of a learner itself.
Revise the semantic class of an example only when the learning time is long and its prediction is confidently inconsistent. As highlighted in Table 2, only when two conditions are met, we have $\epsilon_{\text{ProSelfLC}} > 0.5$ and $\arg\max_j \mathbf{p}(j|\mathbf{x}) \neq \arg\max_j \mathbf{q}(j|\mathbf{x})$, then the semantic class in $\tilde{\mathbf{q}}_{\text{ProSelfLC}}$ is changed to be determined by $\mathbf{p}$. For example, we can deduce $\mathbf{p} = [0.95, 0.01, 0.04], \mathbf{q} = [0, 0, 1], \epsilon_{\text{ProSelfLC}} = 0.8 \Rightarrow \tilde{\mathbf{q}}_{\text{ProSelfLC}} = (1 - \epsilon_{\text{ProSelfLC}})\mathbf{q} + \epsilon_{\text{ProSelfLC}}\mathbf{p} = [0.76, 0.008, 0.232]$. Theoretically, ProSelfLC also becomes robust against long time being exposed to the training data, so that early stopping is not required.

5 Experiments

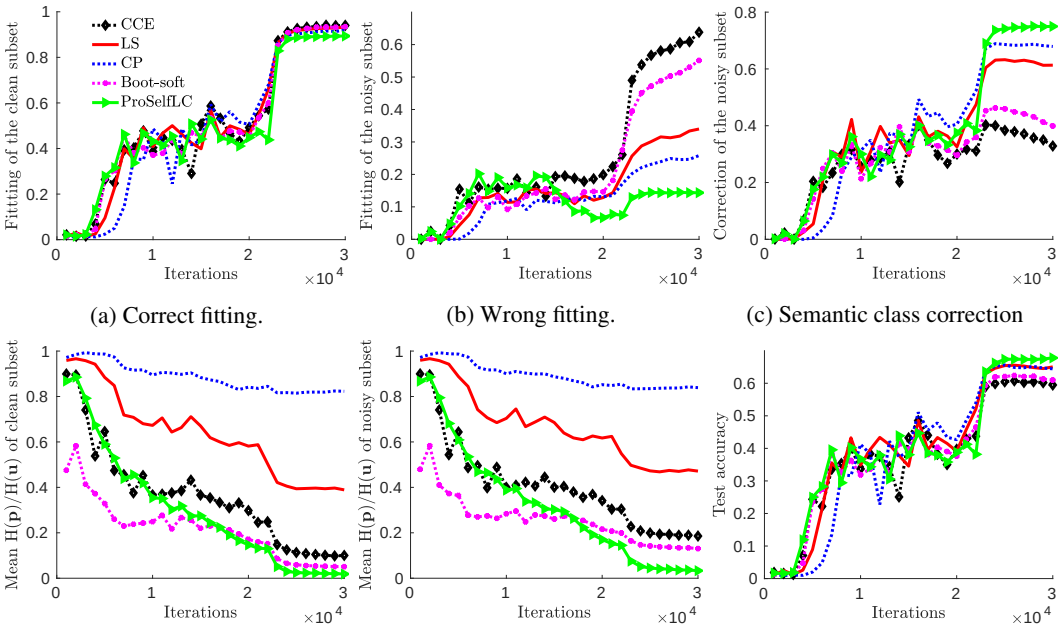
In our experiments, we re-implement LS and CP. Regarding Self LC methods, we re-implement Boot-soft [8], where ϵ is fixed throughout training. We do not re-implement stage-wise Self LC methods, e.g., joint optimisation [9], because time-consuming tuning is required. In addition, we fix the random seed and do not use any random accelerator for an exact reproducibility purpose.

5.1 Correcting the similarity structure in standard image classification

Datasets and training details. (1) CIFAR-100 [34] has 20 coarse classes. Each of them contains 5 fine ones. There are 500 and 100 images per class in the training and testing sets, respectively. The image size is 32×32. We apply simple standard data augmentation [35], i.e., we pad 4 pixels on every side of the image, and then randomly crop it with a size of 32×32. Finally, this crop is horizontally flipped with a probability of 0.5. We choose SGD with its settings as: (a) a learning rate of 0.1; (b) a momentum of 0.9; (c) a weight decay of $5e-4$; (d) the batch size is 256 and the number of training iterations is 30k. We divide the learning rate by 10 at 15k and 22k iterations, respectively. (2) We train ResNet-50 [35] on ImageNet 2012 classification dataset, which has 1k classes and 50k images in the test set [36]. We use SGD with a start learning rate of $2e-3$. A polynomial learning

Table 3: Test accuracy (%) in the standard setting. For LS, CP and Boot-soft, we try two settings of ϵ.

Dataset	CCE	LS		CP		Boot-soft		ProSelfLC
	$\epsilon = 0$	$\epsilon = 0.25$	$\epsilon = 0.50$	$\epsilon = 0.25$	$\epsilon = 0.50$	$\epsilon = 0.25$	$\epsilon = 0.50$	$\epsilon_{\text{ProSelfLC}}$
CIFAR-100	69.0	69.6	68.4	69.3	68.7	69.1	69.1	**70.3**
ImageNet 2012	75.5	75.2	74.9	74.8	74.6	75.8	75.8	**76.0**



(a) Correct fitting. (b) Wrong fitting. (c) Semantic class correction

(d) Entropy of clean subset. (e) Entropy of noisy subset. (f) Generalisation.

Figure 3: Comprehensive learning dynamics on CIFAR-100 with asymmetric label noise $r = 0.4$. For data-dependent items, mean results are reported. We remark ***at training, a learner is not given whether an example is clean or not.*** We store intermediate models and analyse them when the learning ends.

rate decay with a power of 2 is used. We set the momentum to 0.95 and the weight decay to $1e-4$. We train on a single V100 GPU and the batch size is 64. We report the final test accuracy when the training ends at 500k iterations. We use the standard data augmentation: an original image is warped to 256×256, followed by a random crop of 224×224. This crop is also randomly flipped. We fix common settings to fairly compare CCE, LS, CP, Boot-soft and ProSelfLC.

Result analysis. Our results are shown in Table 3, by which we observe the superiority of ProSelfLC in standard setting without considering semantic label noise. Being probably surprising, LS and CP reduce the performance consistently as ϵ increases on ImageNet. Instead, Boot-soft and ProSelfLC improve versus CCE. We remark that both test sets are large so that their differences are noticeable.

5.2 Correcting the similarity structure and semantic class under synthetic label noise

Label noise generation. (1) Symmetric label noise: the original label of an image is uniformly changed to one of the other classes with a probability of r; (2) Asymmetric label noise: we follow [37] to generate asymmetric label noise to fairly compare with their reported results. Within each coarse class, we randomly select two fine classes A and B. Then we flip $r \times 100\%$ labels of A to B, and $r \times 100\%$ labels of B to A. We remark the overall label noise rate is smaller than r.

Baselines.[2] We compare with the results reported recently in SL [37]. Forward is a loss correction approach which uses a noise-transition matrix [18]. D2L monitors the subspace dimensionality change at training [39]. GCE denotes generalised cross entropy [40] and SL is symmetric cross entropy [37]. They are robust losses for solving label noise.

[2]We do not consider DisturbLabel [38], which flips labels randomly and is counter-intuitive. Generally, the performance drops as the uniform label noise increases, which proves that DisturbLabel hurts the generalisation.

Table 4: Accuracy (%) on the CIFAR-100 clean test set. A test set has to be clean, otherwise we cannot evaluate whether a model's predictions are correct or not. All compared methods use ResNet-44. The best results are in bold.

		Asymmetric Noisy Labels			Symmetric Noisy Labels		
	Method	r=0.2	r=0.3	r=0.4	r=0.2	r=0.4	r=0.6
Results From SL [41]	Boot-hard	63.4	63.2	62.1	57.9	48.2	12.3
	Forward	64.1	64.0	60.9	59.8	53.1	24.7
	D2L	62.4	63.2	61.4	59.2	52.0	35.3
	GCE	63.0	63.2	61.7	59.1	53.3	36.2
	SL	65.6	65.1	63.1	60.0	53.7	41.5
Our Trained Results	CCE	66.6	63.4	59.5	58.0	50.1	37.9
	LS	67.9	66.4	65.0	63.8	57.2	46.5
	CP	67.7	66.0	64.4	64.0	56.8	44.1
	Boot-soft	66.9	65.3	61.0	63.2	59.0	44.8
	ProSelfLC	**68.7**	**68.0**	**67.8**	**64.9**	**59.3**	**47.7**

Training details. To make LS, CP and Boot-soft more competitive, we train them using three ϵ, i.e., $0.125, 0.25$ and 0.5, and report the best one. Other details have been presented in Section 5.1.
Result analysis. We do not select the best model according to using a validation set. Instead, we directly report the final results of all methods when the training terminates. *This is important in that we test the robustness of a model against not only label noise, but also a long time being exposed to the training data.* The results are displayed in Table 4. We observe that: (1) On symmetric and asymmetric label noise, ProSelfLC is the best over all baselines; (2) In our re-implementation, LS, CP and Boot-soft are competitive with other reported baselines. Overall, Boot-soft is worse than LS and CP. However, our proposed advanced variant, ProSelfLC, outperforms them significantly. Comprehensive learning dynamics are shown in Figure 3, which demonstrates ProSelfLC learns meaningful patterns under severe noise. The dynamics along with label noise rate are in the supplementary material.
Revising the semantic class and perceptual similarity structure. Generally, the semantic class of an example is defined according to its perceptual similarities with training classes, and is chosen to be the most similar class. In Figure 3b and 3c, we show a learner's behaviours on without fitting wrong labels and correcting them in different approaches. We remark that ProSelfLC performs the best.
To reward or penalise low entropy? LS and CP are proposed to penalise low entropy. On the one hand, we observe that LS and CP work, being consistent with prior evidence. As shown in Figure 3d and 3e, the entropies of both clean and noisy subset are the largest in LS and CP, and correspondingly their generalisation performance is the best except for ProSelfLC in Figure 3f. On the other hand, our ProSelfLC has the lowest low entropy while performs the best, which demonstrates *it does not hurt for a learner to be confident*. However, *a learning model needs to be careful about what to be confident in*. Let us look at Figure 3b and 3c, ProSelfLC has the least wrong fitting while the highest semantic class correction rate, which denotes it is confident in learning meaningful patterns.

5.3 Correcting the similarity structure and semantic class under real-world label noise

Clothing 1M [19] has around 38.46% semantic noise. The noise type is agnostic. There are around 1 million images of 14 classes from shopping websites. We only use its noisy training data.
Baselines. For loss correction and estimating the noise-transition matrix, S-adaption [13] uses an extra softmax layer, while Masking [17] exploits human cognition. MD-DYR-SH [42] is a combination of three techniques, i.e., dynamic mixup (MD), dynamic bootstrapping together with label regularisation (DYR) and soft to hard (SH). All other baselines have already been introduced.
Training details. We follow [9] to train ResNet-50 and initialise it by a trained model on ImageNet. We follow Section 5.1 with small changes: the initial learning rate is 0.01 and we train 10k iterations.
Result analysis. We show the results in Table 5. Analogous to CIFAR-100, we report our trained results of CCE, LS, CP, Boot-soft and ProSelfLC for a completely fair comparison. We observe that the performance of ProSelfLC is the best of all.

Table 5: Test accuracy (%) on real-world noisy dataset Clothing 1M. The best result is in bold.

Boot-hard	Forward	D2L	GCE	SL	S-adaptation	Masking	MD-DYR-SH	Joint-soft	Our Trained Results				
									CCE	LS	CP	Boot-soft	ProSelfLC
68.9	69.8	69.5	69.8	71.0	70.3	71.1	71.0	72.2	71.8	72.6	72.4	72.3	**73.4**

6 Conclusion

In this work, we present a comprehensive mathematical study on popular target modification techniques and show their relationships and limitations. Based on extensive comparative experiments, we can conclude that rewarding low entropy (towards a meaningful status) leads to better generalisation than penalising low entropy. Further, our proposed ProSelfLC achieves the best performance compared with the existing methods under both standard and noisy settings.

Rethinking Open Core Research Questions in Machine Learning

Our study presents a fundamental analysis about deep/machine learning.

1. Should we trust and exploit a learner's knowledge as training goes, or always trust human annotations?
2. Should we optimise a learner towards a correct low-entropy status, or penalise a low-entropy status?

References

[1] Arpit, D., Jastrzębski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M.S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., Lacoste-Julien, S.: A closer look at memorization in deep networks. In: ICML. (2017)

[2] Grandvalet, Y., Bengio, Y.: Entropy regularization. Semi-supervised learning (2006) 151–168

[3] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR. (2016)

[4] Müller, R., Kornblith, S., Hinton, G.E.: When does label smoothing help? In: NeurIPS. (2019)

[5] Pereyra, G., Tucker, G., Chorowski, J., Kaiser, Ł., Hinton, G.: Regularizing neural networks by penalizing confident output distributions. In: ICLR Workshop. (2017)

[6] Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: NeurIPS Deep Learning and Representation Learning Workshop. (2015)

[7] Lee, D.H.: Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. (2013)

[8] Reed, S., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., Rabinovich, A.: Training deep neural networks on noisy labels with bootstrapping. In: ICLR Workshop. (2015)

[9] Tanaka, D., Ikami, D., Yamasaki, T., Aizawa, K.: Joint optimization framework for learning with noisy labels. In: CVPR. (2018)

[10] Grandvalet, Y., Bengio, Y.: Semi-supervised learning by entropy minimization. In: NeurIPS. (2005)

[11] Weston, J., Ratle, F., Mobahi, H., Collobert, R.: Deep learning via semi-supervised embedding. In: Neural networks: Tricks of the trade. Springer (2012) 639–655

[12] Li, Y., Yang, J., Song, Y., Cao, L., Luo, J., Li, L.J.: Learning from noisy labels with distillation. In: ICCV. (2017)

[13] Goldberger, J., Ben-Reuven, E.: Training deep neural-networks using a noise adaptation layer. In: ICLR. (2017)

[14] Sukhbaatar, S., Fergus, R.: Learning from noisy labels with deep neural networks. arXiv preprint arXiv:1406.2080 (2014)

[15] Vahdat, A.: Toward robustness against label noise in training deep discriminative neural networks. In: NeurIPS. (2017)

[16] Yao, J., Wu, H., Zhang, Y., Tsang, I.W., Sun, J.: Safeguarded dynamic label regression for noisy supervision. In: AAAI. (2019)

[17] Han, B., Yao, J., Niu, G., Zhou, M., Tsang, I., Zhang, Y., Sugiyama, M.: Masking: A new perspective of noisy supervision. In: NeurIPS. (2018)

[18] Patrini, G., Rozza, A., Menon, A.K., Nock, R., Qu, L.: Making deep neural networks robust to label noise: A loss correction approach. In: CVPR. (2017)

[19] Xiao, T., Xia, T., Yang, Y., Huang, C., Wang, X.: Learning from massive noisy labeled data for image classification. In: CVPR. (2015)

[20] Veit, A., Alldrin, N., Chechik, G., Krasin, I., Gupta, A., Belongie, S.: Learning from noisy large-scale datasets with minimal supervision. In: CVPR. (2017)

[21] Lee, K.H., He, X., Zhang, L., Yang, L.: Cleannet: Transfer learning for scalable image classifier training with label noise. In: CVPR. (2018)

[22] Hendrycks, D., Mazeika, M., Wilson, D., Gimpel, K.: Using trusted data to train deep networks on labels corrupted by severe noise. In: NeurIPS. (2018)

[23] Malach, E., Shalev-Shwartz, S.: Decoupling "when to update" from "how to update". In: NeurIPS. (2017)

[24] Jiang, L., Zhou, Z., Leung, T., Li, L.J., Fei-Fei, L.: Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In: ICML. (2018)

[25] Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., Sugiyama, M.: Co-teaching: Robust training of deep neural networks with extremely noisy labels. In: NeurIPS. (2018)

[26] Yu, X., Han, B., Yao, J., Niu, G., Tsang, I.W., Sugiyama, M.: How does disagreement help generalization against label corruption? In: ICML. (2019)

[27] Wei, H., Feng, L., Chen, X., An, B.: Combating noisy labels by agreement: A joint training method with co-regularization. In: CVPR. (2020)

[28] Qiao, S., Shen, W., Zhang, Z., Wang, B., Yuille, A.: Deep co-training for semi-supervised image recognition. In: ECCV. (2018)

[29] Song, H., Kim, M., Lee, J.G.: Selfie: Refurbishing unclean samples for robust deep learning. In: ICML. (2019)

[30] Xu, T.B., Liu, C.L.: Data-distortion guided self-distillation for deep neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. Volume 33. (2019) 5565–5572

[31] Yun, S., Park, J., Lee, K., Shin, J.: Regularizing class-wise predictions via self-knowledge distillation. arXiv preprint arXiv:2003.13964 (2020)

[32] Zhang, L., Song, J., Gao, A., Chen, J., Bao, C., Ma, K.: Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In: Proceedings of the IEEE International Conference on Computer Vision. (2019) 3713–3722

[33] Kullback, S., Leibler, R.A.: On information and sufficiency. The annals of mathematical statistics (1951) 79–86

[34] Krizhevsky, A.: Learning multiple layers of features from tiny images. (2009)

[35] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. (2016)

[36] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision (2015) 211–252

[37] Wang, Y., Ma, X., Chen, Z., Luo, Y., Yi, J., Bailey, J.: Symmetric cross entropy for robust learning with noisy labels. In: ICCV. (2019)

[38] Xie, L., Wang, J., Wei, Z., Wang, M., Tian, Q.: Disturblabel: Regularizing cnn on the loss layer. In: CVPR. (2016)

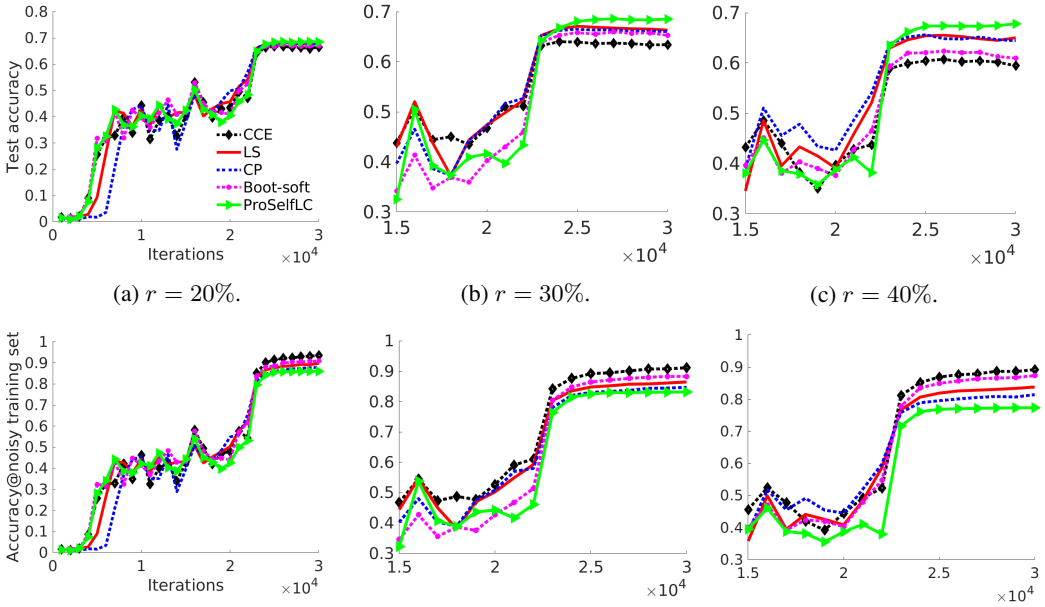
[39] Ma, X., Wang, Y., Houle, M.E., Zhou, S., Erfani, S.M., Xia, S.T., Wijewickrema, S., Bailey, J.: Dimensionality-driven learning with noisy labels. In: ICML. (2018)

[40] Zhang, Z., Sabuncu, M.R.: Generalized cross entropy loss for training deep neural networks with noisy labels. In: NeurIPS. (2018)

[41] Wang, X., Kodirov, E., Hua, Y., Robertson, N.M.: Derivative manipulation for general example weighting. arXiv preprint arXiv:1905.11233 (2019)

[42] Arazo, E., Ortego, D., Albert, P., O'Connor, N., Mcguinness, K.: Unsupervised label noise modeling and loss correction. In: ICML. (2019)

Supplementary Material of ProSelfLC



(a) $r = 20\%$. (b) $r = 30\%$. (c) $r = 40\%$.

(d) $r = 20\%$. (e) $r = 30\%$. (f) $r = 40\%$.

Figure 4: Learning dynamics on CIFAR-100 under asymmetric noisy labels. We show all iterations only in (a) and (d). In the others, we show the second half iterations, which are of higher interest. As the noise rate increases, the superiority of ProSelfLC becomes more significant, i.e., alleviating fitting noise in the 2nd row and leading to better generalisation in the 1st row.