A ALGORITHM

Algorithm 1: MutexMatch algorithm

Input: batch of labeled data $\mathcal{X} = \{(x_b^{lb}, y_b^{lb})\}_{b=1}^{B}$, batch of unlabeled data $\mathcal{U} = \{x_b^{ulb}\}_{b=1}^{\mu B}$, feature extractor θ, TNC $\mathcal{P}$, TNC $\mathcal{N}$

for *iteration* t **do**
- $\mathcal{L}_{sup} = \frac{1}{B}\sum_{n=1}^{B} H(y_n^{lb}, \mathcal{P}(x_n^{lb}))$ *// Supervised loss for* x^{lb}
- **for** *iteration* $b = 1$ *to* μB **do**
  - $p^w = \mathcal{P}(\theta(\alpha_w(x^{ulb})))$ *// Compute TPC's prediction for weak augmentation of* x^{ulb}
  - $p^s = \mathcal{P}(\theta(\alpha_s(x^{ulb})))$ *// Compute TPC's prediction for strong augmentation of* x^{ulb}
  - $r^w = \mathcal{N}(\theta(\alpha_w(x^{ulb})))$ *// Compute TNC's prediction for weak augmentation of* x^{ulb}
  - $r^s = \mathcal{N}(\theta(\alpha_s(x^{ulb})))$ *// Compute TNC's prediction for strong augmentation of* x^{ulb}
  - $\hat{p}^w = \arg\max(p^w)$ *// Select pseudo-labels for* x^{ulb}
  - $\hat{q}^w = \arg\min(p^w)$ *// Select complementary pseudo-labels for* x^{ulb}
- **end**
- $\mathcal{L}_{sep} = \frac{1}{\mu B}\sum_{n=1}^{\mu B} H(\hat{q}_n^w, \mathcal{N}(\hat{\theta}(x_n^w)))$ *// Stop back-propagating gradients on* θ
- $\mathcal{L}_p = \frac{1}{\mu B}\sum_{n=1}^{\mu B} \mathbb{1}(\max(p_n^w) \geq \tau) H(\hat{p}_n^w, p_n^s)$ *// Positive consistency loss for* x^{ulb}
- $\mathcal{L}_n = \frac{1}{\mu B}\sum_{n=1}^{\mu B} \mathbb{1}(\max(p_n^w) < \tau) H(r_n^w, r_n^s)$ *// Negative consistency loss for* x^{ulb}
- update $\theta, \mathcal{P}, \mathcal{N}$ by SGD to optimise $\mathcal{L}_{sup} + \lambda_{sep}\mathcal{L}_{sep} + \lambda_p\mathcal{L}_p + \lambda_n\mathcal{L}_n$

end

B BARELY SUPERVISED LEARNING

The experimental protocol of barely supervised learning (BSL) described in Sohn et al. (2020) assume a limited availability (*e.g.*, 1 or 5) of labeled data from categories of interest. In order to test the performance of our method in extreme cases, we conduct experiments on CIFAR-10 with only one label per class, and considered developing a simple method to use our TNC in the test phase. As shown in Table 4, we use five different random seeds to extract one label of each class from CIFAR-10, and use MutexMatch to achieve test accuracy reaching between 65.30% and 93.07% with a mean of 78.73%. Compared with FixMatch (Sohn et al., 2020) reaching between 48.58% and 85.32%, the performance of MutexMatch is more superior. Then we consider using TNC to complete the test phase under this setting to obtain the test accuracy. We assume that in the ideal case, according to Equation (3), for test data x, the prediction of TNC $r_x = \mathcal{N}(x)$ and the prediction of TPC $p_x = \mathcal{P}(x)$ should satisfy $\arg\max(r_x) = \arg\min(p_x)$.

According to negative learning proposed in Kim et al. (2019), we hypothesis TNC is trained to classify what input image does not belong to its complementary label, so that we can use $\hat{r}_x = \arg\min(r_x)$ to classify an input image x. Compared with TPC, TNC may learn less error information when the label is extremely scarce, so as to obtain better test performance. In order to verify this idea, we used TNC to participate in the test phase showed in Figure 8. For test sample x, we set a confidence threshold T, if $p_x > T$ we uses TPC to predict, if $p_x < T$ uses TNC instead, that is, the leftmost point ($T = 0$) in the figure represents only TNC for test, and the rightmost point ($T = 1$) represents only TNC for test. Taking 20 labels as the dividing line, we can see that using TNC for prediction has more advantages in the case of fewer labels.

Table 4: Accuracy of MutexMatch on a single 1-label split of CIFAR-10 with different random seeds. Results are ordered by accuracy.

Fold	1	2	3	4	5
Accuracy	65.30	71.12	77.83	86.33	93.07

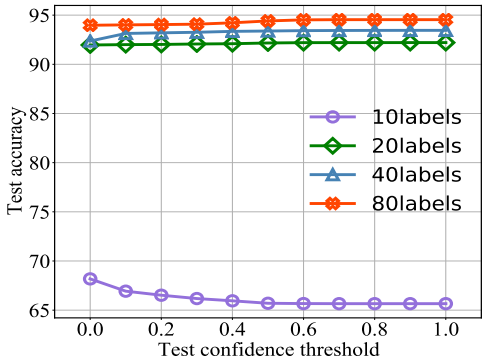

Figure 8: Test accuracy on CIFAR-10 in single run with various amount of labels using TNC to participate test phase. The x-axis represents confidence threshold T and y-axis represents test accuracy.

C Additional Experimental Results

C.1 Ablation Study on Learning Rate and Learning Rate Schedule

We note that learning rate and learning rate schedule are very important for MutexMatch. In this section, we use the experiment setting in Section 4.1 to conduct additional ablation experiments for both. Following Loshchilov & Hutter (2017), recent work(Sohn et al., 2020; Li et al., 2020) use a cosine learning rate decay and achieve best performance. However, as shown in Table 5, we found that MutexMatch achieve better results using no decay on CIFAR-10, outperforming cosine learning rate decay by 0.32%. When there are many labels, the pseudo-labels output by TPC are more likely to have high-confidence and remain stable. It is necessary for MutexMatch to use cosine learning rate decay to jump out of the local optimum.

Table 5: Ablation study on learning rate and learning rate schedule. Results are reported on CIFAR-10 varying number of labels.

Decay Schedule	Learning Rate	Labels	Backbone	Accuracy
No Decay	0.03	40	WRN-28-2	**93.54**
No Decay	0.07	40	WRN-28-2	93.02
No Decay	0.10	40	WRN-28-2	92.89
Cosine Decay	0.03	40	WRN-28-2	93.22
Cosine Decay	0.07	40	WRN-28-2	93.20
Cosine Decay	0.10	40	WRN-28-2	92.59
No Decay	0.03	80	WRN-28-2	93.95
Cosine Decay	0.03	80	WRN-28-2	**94.53**
No Decay	0.03	1000	CNN-13	91.57
Cosine Decay	0.03	1000	CNN-13	**93.46**
No Decay	0.03	4000	CNN-13	92.75
Cosine Decay	0.03	4000	CNN-13	**94.41**

C.2 Hyperparameters

For MutexMatch, the choice of τ needs to be very cautious, because different τ will lead to the division of high and low-confidence portions, which will affect the impact of the mutex-based consistency regularization on the model. We use the identical setting of experiments in Section 4.1 for MutexMatch and vary τ to verify the sensitivity of MutexMatch to this hyperparameter. As shown in Table 6, MutexMatch needs to select appropriate τ to divide confidence portions. We note that when there are many labels, τ has a greater impact on performance. The more labels are available,

the less confirmation bias will be when using TPC directly for classification, so the portion of TPC in mutex-based consistency regularization can be used directly for learning. Therefore, we guess that in general, we should choose a smaller τ to make more pseudo-labels participate in the training of TPC when the number of labels increases.

At the same time, showed in Figure 9, we vary the weight λ_{sep} of the separate training loss for TNC $\mathcal{L}_{sep}$ and λ_n of the negative consistency loss $\mathcal{L}_n$. Choosing the appropriate weight of loss is very important for MutexMatch. Larger λ_{sep} ensures the accuracy of complementary pseudo-labels, which helps TNC better participate in training. Appropriate λ_m weighs the contribution of TNC and TPC in mutex-based consistency regularization, so that the model can achieve better performance.

Table 6: Ablation study on confidence threshold τ. Results are reported on CIFAR-10 varying number of labels.

τ	Labels	Backbone	Accuracy
0.5	40	WRN-28-2	93.52
0.75	40	WRN-28-2	93.44
0.85	40	WRN-28-2	93.28
0.95	40	WRN-28-2	**93.54**
0.99	40	WRN-28-2	92.17
0.5	80	WRN-28-2	**94.53**
0.95	80	WRN-28-2	93.64
0.5	1000	CNN-13	**93.46**
0.95	1000	CNN-13	92.07
0.5	4000	CNN-13	**94.41**
0.95	4000	CNN-13	92.94

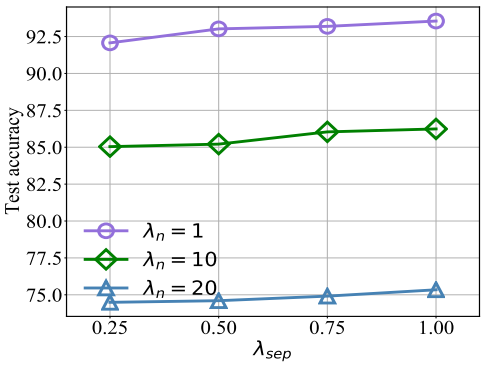

Figure 9: Testing accuracy of MutexMatch on CIFAR-10 with various λ_{sep} and λ_n.

D SEMI-SUPERVISED LEARNING WITH NOISY LABELS

To evaluate the robustness of MutexMatch, we conduct our experiments following settings of semi-supervised learning with noisy labels on CIFAR-10. Semi-supervised learning and noise labels are challenging problems, and semi-supervised with noise labels is much more, because the ability of the model to resist noise labels will be greatly weakened when there is only a small amount of labeled data.

Setting. Following Kim et al. (2019); Patrini et al. (2017), we applied three different types of noise in experiments:
(1) *Symmetric-inc* noise is created by randomly selecting the label from all classes.

(2) *Symmetric-exc* noise is created by randomly selecting the label from all classes without ground truth label.
(3) *Asymmetric* noise is generated by mapping TRUCK $\rightarrow$ AUTOMOBILE, BIRD $\rightarrow$ PLANE, DEER $\rightarrow$ HORSE, and CAT $\leftrightarrow$ DOG for CIFAR-10.
We evaluate MutexMatch and baselines with noisy labels mentioned above using the same settings as Section 4.1. All experiments use 40 labeled data for training, varying radio of noisy labels in labeled data (25%&50%).

Results. Table 7 shows the accuracy comparison between MutexMatch and baselines. All the results are reported by averaging the same labeled data randomly selected five times. Experiments show the robustness of MutexMatch under this setting. For example, with 2 labels and 2 noisy labels (Symmetric-inc) per class, MutexMatch achieves 88.72$\pm$3.51% accuracy, while training of FixMatch collapse reaching a lower 77.80$\pm$17.57% accuracy. MutexMatch contains the idea of negative learning. Learning from the perspective of complementary pseudo-label can prevents model from overfitting to noisy data (Kim et al., 2019) so that MutexMatch achieves superior performance in SSL with noisy labels.

Table 7: Accuracy for CIFAR-10 with noisy labels averaged on 5 different folds. All experiments were based on 40 labeled data with varying radio of noisy labels.

Method	Symmetric-inc		Symmetric-exc		Asymmetric	
	25%noisy	50%noisy	25%noisy	50%noisy	25%noisy	50%noisy
FixMatch	77.80$\pm$17.57	**81.54$\pm$18.47**	80.05$\pm$5.80	75.11$\pm$14.66	84.58$\pm$5.90	72.91$\pm$19.30
MutexMatch	**88.72$\pm$8.51**	77.18$\pm$10.55	**89.37$\pm$6.10**	**81.85$\pm$8.00**	**89.51$\pm$5.14**	**78.28$\pm$15.44**