Problem

Having noisy labels is often inevitable during data collection and it greatly degrades the performance of trained neural networks. In this test case, your task is to learn with noisy labels on the CIFAR-10 dataset with the provided model to test a specific loss function and report your results and analysis. We provide following information and instructions to help you get started.

1 Data preprocessing

1.1 dataset

CIFAR-10 dataset consists of 60,000 32×32 colour images in 10 classes, with 6,000 images per class. There are 50,000 training images and 10,000 test images. The dataset is divided into five training batches and one test batch, each with 10,000 images. The test batch contains exactly 1,000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5,000 images from each class.

1.2 noisy label generation

Uniform (symmetric) noise is designated for this task as it is a basic form of noise and easy to implement. Uniform noisy labels are generated by flipping the labels of a given proportion of training samples to one of the other class labels uniformly. Further, noise rate $\eta \in [0, 1)$ is defined as the proportion of the number of corrupted (noisy) labels amongst those of all data points. If you need more detailed information, please read [1].

```
Hint: In numpy, you can use np.random.choice for a quick implementation
of the uniform noisy label assignment.
```

1.3 data augmentation

Data augmentation methods are at your own choice.

2 Model and hyperparameters

2.1 network architecture

The pre-defined network architecture shown in Table. 1 should be used as the backbone in the project. It is a 8-layer CNN with 6 convolutional layers followed by 2 fully connected layers.

2.2 loss function

Given a dataset $\mathcal{D} = \{(x, y)^i\}_{i=1}^n$, with $x \in \mathcal{X}$ denoting the input samples and $y \in \mathcal{Y}$ its label. For each sample x, a classifier $f(x)$ computes its probability belonging to a certain

Table 1: A 8-layer network used for the test case.

layer name	output size	operations
conv1	32×32	$\begin{bmatrix} 3 \times 3, 64, batchnorm, relu \\ 3 \times 3, 64, batchnorm, relu \end{bmatrix} \times 2$
pool1	16×16	2×2 max pool, stride 2
conv2	16×16	$\begin{bmatrix} 3 \times 3, 128, batchnorm, relu \\ 3 \times 3, 128, batchnorm, relu \end{bmatrix} \times 2$
pool2	8×8	2×2 max pool, stride 2
conv3	8×8	$\begin{bmatrix} 3 \times 3, 196, batchnorm, relu \\ 3 \times 3, 196, batchnorm, relu \end{bmatrix} \times 2$
pool3	4×4	2×2 max pool, stride 2
fc1	256	flatten, fc [3136, 256], batchnorm, relu
fc2	1×1	fc [256, 10], softmax

label $k \in \{1, ..., K\} : p(k|x) = \frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}}$, where z_j are the logits. Conventionally, the ground-truth distribution over labels for sample x is denoted by $q(k|x)$, and $\sum_{k=1}^{K} q(k|x) = 1$. With above definitions, we have the cross entropy loss for sample x written as:

$$l_{ce} = -\sum_{k=1}^{K} q(k|x) \log p(k|x) \tag{1}$$

In this test case, we introduce a reversed formulation of cross entropy loss to complement the learning, written as:

$$l_{ces} = -\sum_{k=1}^{K} p(k|x) \log q(k|x) \tag{2}$$

Note that $q(k|x)$ can be 0 inside the logarithm, which can be bypassed by setting $\log 0 = A$, where $A < 0$ is some constant.

In all, we have the final loss defined as:

$$l = \alpha l_{ce} + \beta l_{ces} \tag{3}$$

where α and β are two tradeoff parameters for a more effective and robust learning.

2.3 hyperparameters

Network shall be trained using SGD with momentum 0.9, weight decay 10^{-4} and an initial learning rate of 0.01 which is divided by 10 after 40 and 80 epochs (120 epochs in total). The parameters α, β, η and A are set to 0.1, 1, 0.6 and -4 respectively.

```
Info: There are no restrictions regarding the framework/language that
you can use for developing your model.
```

3 Task

Write a report up to 2 pages about this task and present your results which includes:

1. run cross-entropy loss only as the baseline and report the result.
2. report the test accuracy on CIFAR-10 with given model and hyperparameters. Note that results need to be the average accuracy and standard deviation of 5 random runs.
3. vary η from 0 to 0.8 with step 0.2 and report results (preferably with a line chart).
4. some discussion or comments on above observed results.
5. optional: vary α and β to make more observations and disscusion.
6. optional: you can propose your own solutions combating the noise.

References

[1] Aritra Ghosh, Himanshu Kumar, and PS Sastry. Robust loss functions under label noise for deep neural networks. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.