Appendix A. Experiment Details

In Table 5, we show the complete set of hyperparameters in our semi-supervised learning experiments.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>B</th>
<th>μ</th>
<th>λcls</th>
<th>α</th>
<th>K</th>
<th>t</th>
<th>τ</th>
<th>T</th>
<th>λctr</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIFAR-10</td>
<td>64</td>
<td>7</td>
<td>1</td>
<td>0.9</td>
<td>2560</td>
<td>0.2</td>
<td>0.95</td>
<td>0.8</td>
<td>1</td>
</tr>
<tr>
<td>STL-10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset</th>
<th>B</th>
<th>μ</th>
<th>λcls</th>
<th>α</th>
<th>K</th>
<th>t</th>
<th>τ</th>
<th>T</th>
<th>λctr</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageNet 1% labels</td>
<td>160</td>
<td>4</td>
<td>10</td>
<td>0.9</td>
<td>30000</td>
<td>0.1</td>
<td>0.6</td>
<td>0.3</td>
<td>10</td>
</tr>
<tr>
<td>ImageNet 10% labels</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
</tbody>
</table>

Table 5: Hyperparameters for CoMatch in the semi-supervised learning experiments.

The strong augmentation Aug on ImageNet unlabeled data uses color distortion in addition to the standard crop-and-flip. A pseudo-code for the color distortion in PyTorch is as follows:

```python
from torchvision import transforms as T
color_jitter = T.ColorJitter(0.4, 0.4, 0.4, 0.1)
transforms.Compose([
    T.RandomApply([color_jitter, p=0.8]),
    T.RandomGrayscale(p=0.2)]
```

Appendix B. MB and MQ in CoMatch

Figure 5 illustrates how the EMA model is utilized in CoMatch to construct the memory bank (MB) and the momentum queue (MQ). The memory bank contains the class probability and the low-dimensional embeddings for both weakly-augmented labeled samples and weakly-augmented unlabeled samples. The momentum queue contains the pseudo-labels for the unlabeled samples and their strongly-augmented embeddings.

Algorithm 1: Pseudo-code of CoMatch

```python
Algorithm 1: Pseudo-code of CoMatch (one iteration).
1 Input: labeled batch \( \mathcal{X} = \{(x_b, y_b)\}_{b=1}^B \), unlabeled batch \( \mathcal{U} = \{u_b\}_{b=1}^{\mu B} \), encoder \( f \), classifier \( h \), projection head \( g \), memory bank \( \text{MB} = \{(p_w^b, z_w^b)\}_{b=1}^K \).
2 for \( b \in \{1, \ldots, \mu B\} \) do
   // class probability prediction
   3 \( p_w^b = h \circ f(\text{Aug}_w(u_b)) \)
   // distribution alignment
   4 \( p_b^w = \text{DA}(p_w^b) \)
   // weakly-augmented embedding
   5 \( z_w^b = g \circ f(\text{Aug}_w(u_b)) \)
   // memory-smoothed pseudo-labeling
   6 for \( k \in \{1, \ldots, K\} \) do
      7 \( a_k = \frac{\exp(z_w^b - z_k^w / t)}{\sum_{k=1}^K \exp(z_w^b - z_k^w / t)} \) // affinity
   8 end
   9 \( q_b = \alpha p_w^b + (1 - \alpha) \sum_{k=1}^K a_k p_k^w \)
      // strongly-augmented embeddings
   10 \( z_b = g \circ f(\text{Aug}_w(u_b)) \)
   11 \( z_b' = g \circ f(\text{Aug}_w(u_b)) \)
   12 end
   13 for \( b \in \{1, \ldots, B\} \) do
      // pseudo-label graph
      14 for \( j \in \{1, \ldots, B\} \) do
         15 \( W_{b,j} = \begin{cases} 
         1 & \text{if } b = j \\
         q_b \cdot q_j & \text{if } b \neq j \text{ and } q_b \cdot q_j \geq T \\
         0 & \text{otherwise}
         \end{cases} \)
         // embedding graph
         16 \( W_{b,j}^z = \begin{cases} 
         \exp(z_b \cdot z_j / t) & \text{if } b = j \\
         \exp(z_b \cdot z_j / t) & \text{if } b \neq j
         \end{cases} \)
      17 end
      18 \( W^q = \text{Normalize}(W^q) \)
      19 \( W^z = \text{Normalize}(W^z) \)
   20 end
   // losses
   21 \( L_e = \frac{1}{B} \sum_{b=1}^B H(y_b, p(y|\text{Aug}_w(x_b))) \)
   22 \( L^{cls}_u = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{1}(\max_{q_b} \geq T) H(q_b, p(y|\text{Aug}_w(u_b))) \)
   23 \( L^{ctr}_u = \frac{1}{\mu B} \sum_{b=1}^{\mu B} H(W^q_b, W^z_b) \)
   24 \( L = L_e + \lambda^{cls} L^{cls}_u + \lambda^{ctr} L^{ctr}_u \)
   25 update \( f, h, g \) with SGD to minimize \( L \).
```

Figure 5: Illustration of the memory bank and the momentum queue. \( \mathcal{U} \) is the batch of unlabeled data, \( \mathcal{X} \) is the batch of labeled data. \( f, h, \) and \( g \) refer to the EMA version of the encoder, the classification head, and the projection head, respectively.