Semi-Supervised Image Classification (Team 20, abc123)

Di He[1] **CongYun Jin**[1] **Colin Wan**[1]

Abstract

Semi-supervised learning incorporates both labeled and unlabeled data points in the training process and aims to use leverage the unlabeled data to learn features that would support modeling process when using the training data for specific tasks. The community has gained popularity as the amount of data required and cost of obtaining human labeled data increased over the years. This paper explains the modeling and thought process our team had when tackling the given task, and achieved 50% accuracy on the test set with CoMatch.

1. Introduction

Since the huge success of semi/self supervised learning in NLP, the machine learning community began trying to transfer the success into other fields such as computer vision and obtained encouraging results in several problem setups(Bachman et al., 2019)(He et al., 2020).

The task given is similar to those in the industry: train a model for a specific task (in our case classification) with vast amount of unlabeled data and a small percentage of labeled data (5%). Although several well established models has achieved impressive results on benchmark datasets such as CIFAR-10, STL-10 and ImageNet with 1%, 10% labels, the results are hard to reproduce when applied to a new dataset. The mentioned datasets are well massaged for machine learning tasks: the image are cropped/scaled to enhance the object of interest and reduce as much undesired noise as it can. In our task, however, the image qualities are not as good: some image are near impossible for human eyes to understand due to the low contrast between background and object and/or the ambiguous object of interest.

In addition, among the models that preformed well in benchmark settings, most of them required either large batch size, large memory storage, hundreds of hours of training time, or all of them. With the given limited computation power and resource, our team had to leverage the characteristics of different models to optimize the training process.

2. Related Work

2.1. Contrastive Learning

Contrastive learning framework is arguably the most popular architecture in the self-supervised learning community(Chen et al., 2020)(Chen & He, 2020)(Zbontar et al., 2021). The core idea is to encourage the model to attract similar samples and push apart different ones. For a given image the simplest way to define similar and different samples is through augmenting the same image and sample other images from the batch.
In practice, this approach faces two problems:

- If the model leverages the divergence of data within one batch, then the model's performance is hugely influenced by the batch size.
- If the model instead keeps a dynamic memory bank of samples to contrast new samples against, then the model's performance is hugely influenced by the amount and quality information stored.

Both approaches requires a large amount of computation power to achieve satisfying results, which may not be feasible for practitioners.

2.2. Clustering Learning

Clustering learning methods alternates between learning features and cluster assignments of input data(Caron et al., 2020). Unlike contrastive learning, clustering learning methods do not require positive and negative samples; such role is replaced with the centers for different clusters. However, clustering learning methods still require either large batches or memory bank to maintain the validity of clustering process.

2.3. Redundancy Reduction

Redundancy reduction method will make the representation vectors of distorted versions of an image to be similar, while minimizing the redundancy between the components of these vectors.
Barlow Twins use its objective function to naturally avoid collapsed solutions by measuring the cross-correlation matrix between the outputs of two identical networks fed with distorted versions of a sample, and making it as close to the

identity matrix as possible. Unlike most current contrastive methods, Barlow Twins does not require large batches nor asymmetry between the network twins such as a predictor network, gradient stopping, or a moving average on the weight update(Zbontar et al., 2021).

2.4. Pseudo Labeling

Unlike the two methods described above, pseudo labeling aims to increase the consistency of the generated pseudo label for unlabeled dataset(Sohn et al., 2020)(Berthelot et al., 2019). Typically, the modeling process includes three parts

- Modeling labeled data
- Modeling strongly augmented unlabeled data
- Modeling weakly augmented unlabeled data

The model aims to reduce the divergence between

- Prediction of label data and actual label
- Prediction of strongly augmented unlabeled data and prediction of weakly augmented unlabeled data

The architecture can be thought of as a adversarial network: the unlabeled data loss pushes the model to converge to the degenerate solution, and the labeled data loss pulls the model back and tries to learn the features of the input distribution.

3. Methodology and Results

3.1. Framework

In this section, we introduce the architecture we used in this competition (Li et al., 2020). Different from most existing semi-supervised learning methods, CoMatch jointly learns the encoder $f(\cdot)$, the classification head $h(\cdot)$, and the projection head $g(\cdot)$ and jointly optimizes three losses: a supervised classification loss on labeled data $\mathcal{L}_x$, which is defined as the cross-entropy between the truth labels and predictions, an unsupervised classification loss on unlabeled data $\mathcal{L}_u^{cls}$, which is defined as the cross-entropy between the pseudo-labels q and the predictions, and a graph-based contrastive loss on unlabeled data $\mathcal{L}_u^{ctr}$

The overall training objective is:

$$\mathcal{L} = \mathcal{L}_x + \lambda_{cls}\mathcal{L}_u^{cls} + \lambda_{ctr}\mathcal{L}_u^{ctr}$$

where λ_{cls} and λ_{ctr} are the weight of unsupervised losses. In CoMatch the high-dimensional feature of each sampleis transformed to class probability p and its normalized low-dimensional embedding z. It contains four main steps:

- Given a batch B of unlabeled images $\{(x_k, y_k)\}_{k=1}^{N}$, CoMatch first perform memory-smoothed pseudo-labeling on weak augmentations $Aug_w(x_k)$ to produce pseudo-labels. The model's class predictions are smoothed by neighboring samples in the embedding space. Specifically, given prediction and embedding of data in the memory bank, $\{p_k, z_k\}_k$, and current sample prediction and embedding (p_0, z_0), the smoothed pseudo label is defined as minimizer of:

  $$J(q_0) = (1-\alpha)\sum_{k=1}^{K} a_k \|q_0 - p_k\|_2^2 + \alpha \|q_b - p_b\|_2^2$$

  where α is the similarity score between the current sample and points in the memory bank

  $$a_k = \frac{\exp(z_0 \cdot z_k/t)}{\sum_{k=1}^{K}\exp(z_0 \cdot z_k/t)}$$

- The pseudo-labels graph W^q, which defines the similarity of samples in the label space, are used as targets to train the classifier, using strongly-augmented images as inputs. Note the graph is computed within a batch. Formally we have

  $$W_{bj}^{q} = \begin{cases} 1 & \text{if } b = j \\ q_b \cdot q_j & \text{if } b \neq j \text{ and } q_b \cdot q_j \geq T \\ 0 & \text{otherwise} \end{cases}$$

- CoMatch construct a embedding-label graph W^z, which measures the similarity of strongly-augmented samples $Aug_s(\mathcal{U})$ in the embedding space. The embedding-label graph contains self-loops as self-supervision. Similar to W^q, this graph is computed within a batch. Formally,

  $$W_{bj}^{z} = \begin{cases} \exp(z_b \cdot z_b'/t) & \text{if } b = j \\ \exp(z_b \cdot z_j/t) & \text{if } b \neq j \end{cases}$$

  where z_b' is the embedding of another strongly augmented version of the data.
- The paper also proposes one can store a momentum queue with size K of the prediction and embedding of previous data points and compute the graphs between current batch and the queue. Therefore W^q and W^z instead of having dimension of $B \times B$, where B is the batch size, it would have a dimension of $B \times K$.
- The pseudo-label graph is used as the target to train an embedding graph with contrastive learning, such that images with similar pseudo-labels are encouraged to have similar embeddings.

An illustration of CoMatch is shown in Fig 1. In order to build a meaningful pseudo-label graph, the unlabeled batch of samples should contain a sufficient number of samples from each class. It is less likely to satisfied for our dataset which contains 800 classes, since a large unlabeled batch would exceed the memory capacity under limited resources (1 GPU). To improve the performance of CoMatch on large-scale datasets, an EMA model whose parameters $\bar{\theta}$ are the

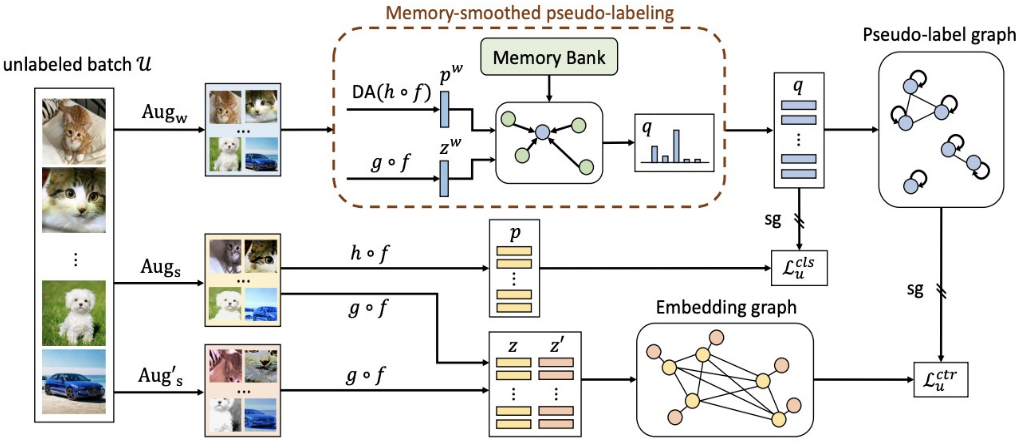

Figure 1. Framework of CoMatch

moving-average of the original model's parameters $\theta : \bar{\theta} \leftarrow m\bar{\theta} + (1 - m)\theta$, where m is the momentum parameter controlling the model to evolve smoothly.

Additionally, when the unlabeled data contains CoMatch shows its advantage. The smoothness constraint gives these samples low-confidence pseudo-labels. Therefore, they are less connected to in-distribution samples, and will be pushed further away from in-distribution samples by $\mathcal{L}_u^{ctr}$.

4. Experiment and Results

First, we conduct experiments for different models on 5% labeled dataset which contains 25,600 labeled images of size 96 × 96 from 800 classes and 512,000 unlabeled images. Then, we submitted labeling request for extra 12,800 images based on Barlow Twins, which is our best model before the second ledearboard. Lastly, CoMatch and Barlow Twins were selected to be continuously trained based on extra dataset. And CoMatch finally got best performance.

4.1. CoMatch

Implementation details For our dataset, we use a ResNet-50 model as the encoder. We train the model using SGD with a momentum of 0.9 and a weight decay of 0.0001. The learning rate is 0.1, which follows a cosine decay schedule for 400 epochs. Different from the original paper, we use different set of hyper-parameters due to a larger percentage of labeled images in our dataset, shown in Table 1. **Augmentations** CoMatch uses one weak augmentation Aug_w, which is the standard horizontal flip, and two strong augmentations Aug_s and Aug'_s, which are random color jittering and grayscale conversion.

Dataset	B	μ	λ_{cls}	α	K	t	r	T	λ_{ctr}
5% labels	64	4	10	0.9	30000	0.1	0.55	0.25	5
7.5% labels							0.52	0.22	3

Table 1. Hyperparameters for CoMatch in two datasets

Extra Labeling Request We used the uncertainty sampling method to choose the image set that would best improve our model performance. Specifically, given a trained classification model and unlabeled dataset, the entropy confidence was computed for the prediction of each unlabeled image. And 12,800 images with the highest entropy, which our model is more uncertain of, are selected. Together with extra labels, we can conduct our experiments n 7.5 % labeled dataset. Because we haven't gotten outstanding CoMatch results at the labeling requesting time, Barlow Twins was used instead to select extra labeling images, leading only a tiny performance improvement for CoMatch with extra labels. Table 2 shows that the accuracy of Barlow Twins increased by 2% with extra labels, while the accuracy of CoMatch just increased 0.2%.

Results We train CoMatch for only 400 epochs to demonstrate its efficiency in learning for both 5% labeled dataset and 7.5% labeled dataset. The training progresses on both two dataset are shown in Fig2. Tabel 2 shows the the results of different models we have tried. CoMatch obtains the highest accuracy of **50.8%** on 5% labels, and **51%** on 7.5% labels

Model		Epochs	Acc.	Acc. Ext.
Contrastive	SimCLR	300	21%	N/A
	SimSiam	200	23%	N/A
	BarlowTwins	100	24%	26%
Auto-Encoder	AE	200	18%	N/A
Pseudo Label	FixMatch	300	29%	N/A
	CoMatch	400	**50.8%**	**51%**

Table 2. Models Accuracy based on original labels and extra labels

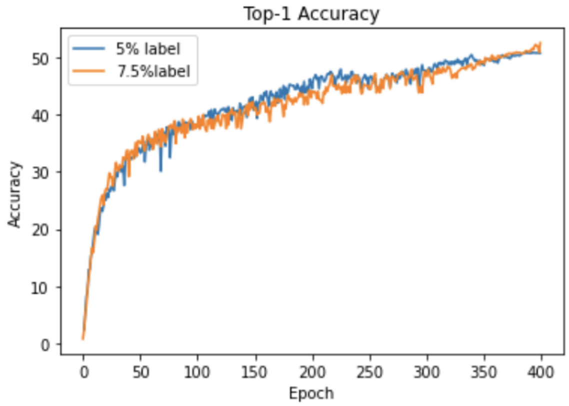

Figure 2. Training progresses on 5% labels and 7.5% labels

4.2. Comparative Models

SimCLR(Chen et al., 2020): SimCLR has demonstrated outstanding performance in large data settings such as ImageNet. We were only able to train SimCLR with batch size 512 which limited the model performance and converged at a validation accuracy of 21%.

SimSiam(Chen & He, 2020): SimSiam shares a similar structure with the incorporation f another prediction network. It suffers from the same limitation as SimCLR, and our best SimSiam model reached a validation accuracy of 23%.

Auto-Encoder: Although our Auto-Encoder was able to reconstruct the original image very well, the features the model learned was not useful for prediction. After fine-tuning the encoder network on labeled data, the model reached a highest validation accuracy of 18%.

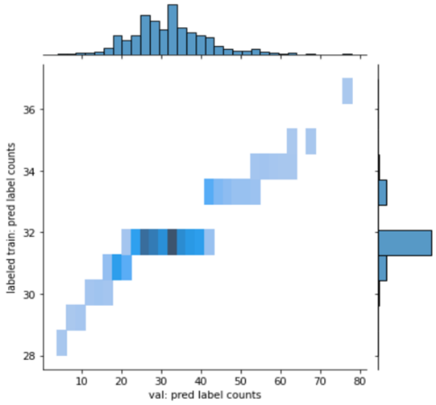

Figure 3. Validation vs Training: Distribution of Image Counts group by Predicted Labels

Barlow Twins(Zbontar et al., 2021): Barlow Twins does not require large batch sizes, which makes it to be very practical in our simple GPU sets. Due to time and GPU limits, the model was just trained for 100 epochs. Our experiments showed that, with accuracy of 24%, Barlow Twins outperformed other contrastive models. It has lots of potentials.

FixMatch(Sohn et al., 2020): Belonging to the same family of models, FixMatch has a simple architecture: it uses the prediction of weak augmented version as the label for its strong augmented counterpart. This model has a lot of potential, however we couldn't spend too much resource on tuning the training and hyper parameters. The model achieved 29% on the validation set.

5. Discussion and Visualization

5.1. Predicted Labels Distribution

Figure 3 shows most of 800 classes were predicted to contain about 32 images, which implies the predicted labels of the training set is very balanced. However, for the validation set, the predicted labels are not very balanced. There are some under-classified labels at the bottom-left, and some over-classified labels at the upper-right. Further, Figure 3 also shows that if a class was under-classified in the training set, it is likely to be also under-classified in the validation set, vice versa.

5.2. Visualization of Network

Given the butterfly image, Fig 4 visualizes the some intermediate layers of CoMatch encoder. From those layers, the model captured some features to differentiate the butterfly from other classes. In the beginning layers, the model captured features like the shape of wings, the texture of wings and background flowers. With the deeper of layers, the features it detected become more specific. It can detect not only the wings, but also left wing, right wing, fore wing and hind wing. All those useful features will help the final layer to classify this image correctly.

5.3. Error Analysis

Based on the distribution analysis of Fig 3, we will dig into those under-classified labels as well as over-classified ones to analyze the reasons.

Under-Classified Classes. The top half of Fig 5 shows sample images of two under-classified classes. The shield and hand sanitizer both have challenging characteristics, like intra-class variation, scale variation and viewpoint variation. Those characteristics will make the features of one image be very specific for this image rather than the whole class, which increases the difficulty to correctly classify those images.

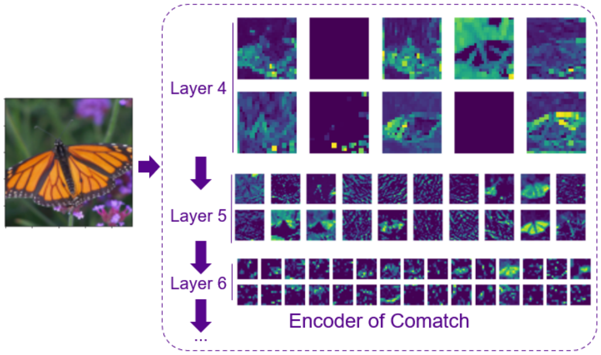

Figure 4. Visualization of Intermediate Layers of CoMach Encoder

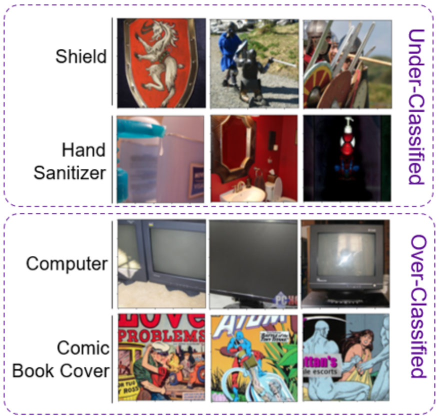

Figure 5. Examples of Under-Classified and Over-Classified Classes

Over-Classified Classes. The bottom half of Fig 5 shows sample images of two over-classified classes. The shape of computers is just like a rectangle, and the color and texture is also very simple. The features of computers are so general, which could be also included in other classes. Therefore, images of other classes could be incorrectly classified as computers. The covers of comic books are very flexible, they can be any object. The model could learn many various features from this class. And images from other classes with the same features could be incorrectly classified as comic book covers.

6. Conclusion

Through this project, we learnt a lot and left some thoughts and experiments for future explorations. Firstly, it is important to find the right model architecture suitable for the target problem and related constraints. Even some contrastive models, like SimCLR, got state-of-the-art performance on ImageNet, they can hardly achieve high accuracy on our smaller dataset with limited computation resources. For our problem, Pseudo Labeling methods and Barlow Twins are better choices. Secondly, hyper-parameters, like learning rate, batch size, epochs, are crucial for model performance. It is worthwhile to put more time searching for good hyper-parameters. Given more time and resources, we will conduct more experiments on Barlow Twins to achieve a higher accuracy and try to combine it with the CoMatch model. Thirdly, extra labeling requests could be a smart investment with proper selection methods. If our labeling request was based on CoMatch, the improvement would be much more significant. Given a more flexible timeline, we might try more creative selection methods based on CoMatch and get higher accuracy.

References

Bachman, P., Hjelm, R. D., and Buchwalter, W. Learning representations by maximizing mutual information across views. *arXiv preprint arXiv:1906.00910*, 2019.

Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., and Raffel, C. Mixmatch: A holistic approach to semi-supervised learning. *arXiv preprint arXiv:1905.02249*, 2019.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Chen, X. and He, K. Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*, 2020.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

Li, J., Xiong, C., and Hoi, S. C. Semi-supervised learning with contrastive graph regularization. *arXiv preprint arXiv:2011.11183*, 2020.

Sohn, K., Berthelot, D., Li, C.-L., Zhang, Z., Carlini, N., Cubuk, E. D., Kurakin, A., Zhang, H., and Raffel, C. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685*, 2020.

Zbontar, J., Jing, L., Misra, I., LeCun, Y., and Deny, S. Barlow twins: Self-supervised learning via redundancy reduction. *arXiv preprint arXiv:2103.03230*, 2021.