Clustering Latent Representations for Semi-Supervised Learning

Serkan Karakulak
Center for Data Science
NYU
serkan@nyu.edu

Aakriti Gupta
Tandon
NYU
ag6817@nyu.edu

David Martuscello
Courant
NYU
dm4350@nyu.edu

Abstract

In this work, we propose using clustering in the latent vector space to impose regularity in the presence of unlabeled data points. Our algorithm uses k-means clustering and learns network parameters simultaneously with the latent cluster centroids in different layers of the network. By making the loss value a function of the deviations from latent cluster centroid, we impose consistent latent representations between the labeled and the unlabeled data to reduce overfitting.

1 Introduction

The existence of valuable annotated large datasets like ImageNet has fueled advances in supervised learning and there are many successful use cases where the pretrained weights are transferred and used successfully. However it still remains a challenge to learn efficiently from data that has only small number of annotated samples.

Vinyals et al. [4] proposed matching networks, which uses an attention mechanism over a learned embedding of the labeled set of examples to predict classes for the unlabeled points. Ravi and Larochelle [2] introduces the idea of learning the exact optimization algorithm used to train another learner neural network in the few-shot regime. Tarvainen and Valpola [3] proposes to store exponential moving average of the model parameters

One study that showed clustering might be a powerful tool to utilize for learning from the unlabeled features was Deep Clustering from Caron et al. [1] which is an algorithm that alternates between clustering of the image descriptors and updating the weights of the network by predicting cluster assignments. Cluster assignments are used as pseudo-labels to train the unlabeled data and the centroid matrix itself was not being used.

Our approach is based on the observation that at the higher levels of the network hierarchy the latent representations are invariant to local information of the inputs. Hence they can be grouped into a limited number of clusters - which correspond to higher level information about the data, without decreasing the input's information content. If both the labeled and the unlabeled examples are coming from the same unobserved latent distribution, we would expect our neural network to map both the labeled and the unlabeled samples to similar regions in the latent space and produce latent representations that are consistent between these two groups. Hence we can add a penalty term for large deviations from those clusters and regularize the parameters in a way to produce consistent representations between both the labeled and unlabeled data. To that end, we propose a simple algorithm that simultaneously learns cluster centroids and the network parameter during the training.

Our method can be applied to any given neural network, with minor modifications. For our experiments we have used a resnet18 architecture, which consists of 18 convolutional layers that are built by 4 so-called resnet blocks. We apply clustering after the second and third resnet blocks, as well as after the pooling layer that follows the final resnet block. For our experiments, we used k-means clustering

where the cluster centroids are parametric. The choice of clustering algorithm is also flexible and we can use other centroid based methods such as expectation algorithm with little modification.

2 Method

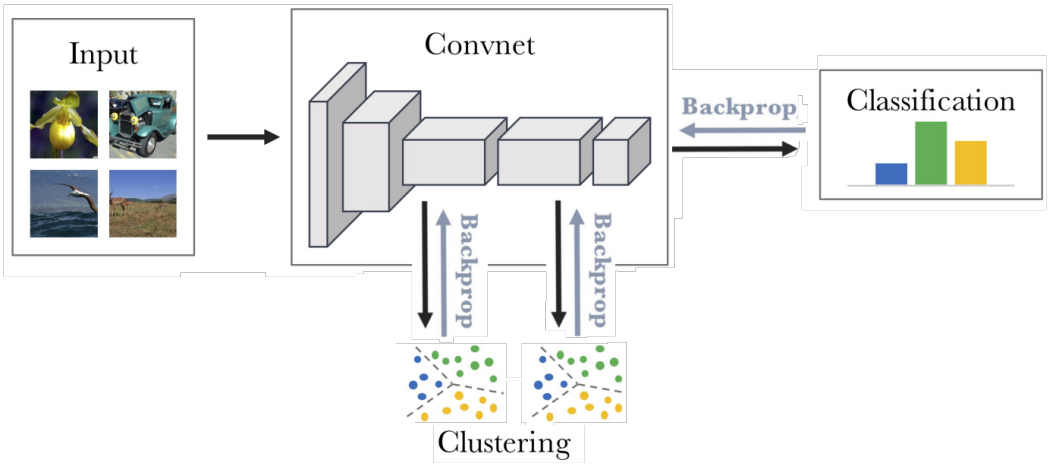


Figure 1: Model architecture.

2.1 Model Architecture

Given the labeled and unlabeled training sets $X^L = \{x_1^L, x_2^L, \ldots, x_N^L\}$, $X^U = \{x_1^U, x_2^U, \ldots, x_M^U\}$, we want to find the parameter θ^* such that the mapping f_{θ^*} would generate good and generalizable features. Let each labeled image x_k^L have a label $y_k \in \{0, 1\}$. Also, let f_θ^i denote the latent representation at the ith layer. If the data generating distributions of X^L and X^U are the same, then we can expect the latent representations $f_\theta^i(X^L)$ and $f_\theta^i(X^U)$ for the labeled and unlabeled dataset to have similar distributions.

In the computer vision domain the inputs themselves are very high dimensional, have lots of local variations, and the neighborhood in terms of the distances between raw inputs do not tell us anything about the content. Hence enforcing distance based dimension reduction at inputs would make us lose information about the content. However latent representations represent high level information that is extracted from the input and hence we can impose a neighborhood without losing information about the high level content.

Let J_i be the number of cluster centroids that we have for the latent representations generated by f_θ^i, and let $\mathcal{C}_{ij}$ be the jth cluster centroid for f_θ^i. Note that $\mathcal{C}_{ij} \in \mathbf{R}^d$, where d is the number of features that we have at f_θ^i. Then we can formulate the problem as below:

$$\ell_c(x) = \sum_i \min_{\mathcal{C}_{ij}} (f_\theta^i(x) - \mathcal{C}_{ij})^2$$

$$\mathcal{L} = \min_\theta \frac{1}{N} \sum_{k=1}^{N} \left(\ell(f_\theta(x_k^L), y_k) + \beta \ell_c(x_k^L)\right) + \frac{1}{M} \sum_{k=1}^{M} \beta \ell_c(x_k^U)$$

where ℓ is the multinomial logistic loss. Since the clustering component is also differentiable, we can use backpropagation to compute the gradient for both the parameter θ and the cluster centroids $\mathcal{C}$. At each step of the gradient descent the cluster centroids would move towards the mean point of the latent representations, for which it is the closest centroid. The minimum operation over the $\mathcal{C}_{ij}$ makes this formulation equivalent to k-means.

2.2 Implementation Details

One crucial aspect of k-means is the initialization of the clusters. We may have the case where some of the clusters remain empty and others get all the data points. There is also a risk of assigning

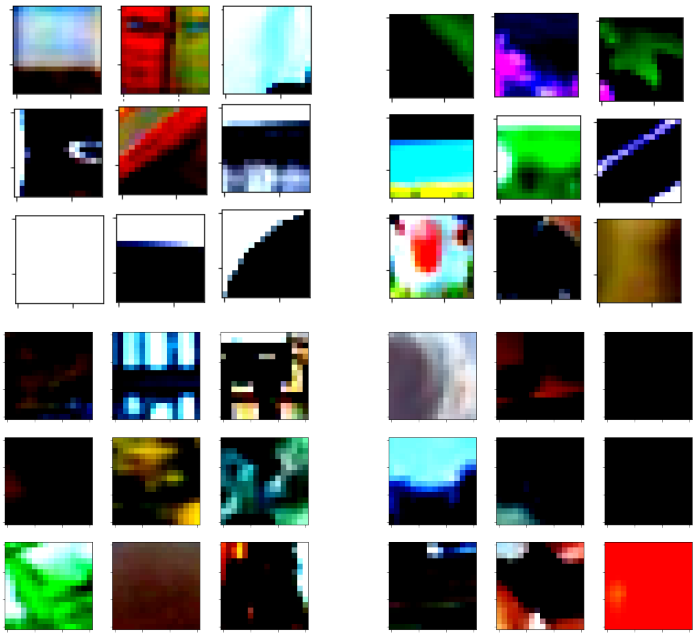


Figure 2: Sampled images from 4 different clusters.

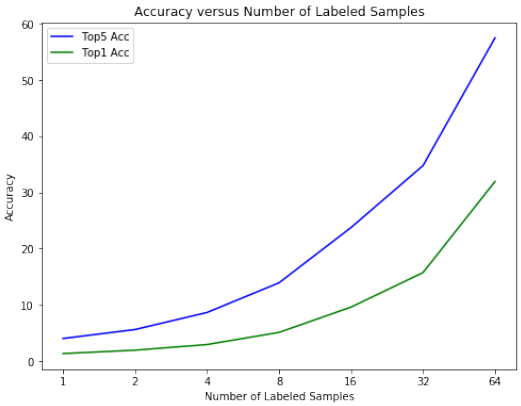
and enforcing clusters very early in the training, before we allow sufficient time for the network to extract higher level information from the inputs and construct a mapping to a meaningful latent representation. To avoid this, we have set the β parameter to zero in the beginning of the training, and increased it gradually afterwards. This however does not remove the risk of bad initializations. In order to avoid empty clusters and other problems, we initialized the cluster centroids among the latent representations of datapoints randomly while distributing them evenly among all classes. In addition, at the beginning of each epoch we reinitialized the cluster centroids using this methodology for a group of the epochs early on in the training.

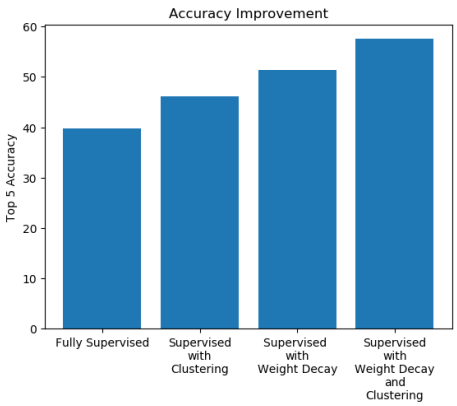
2.3 Experiments

As a side note, we have also experimented with other settings, where there was a KL-divergence term applied at clusters which forced each of them to be gaussians with unit variance. This allowed us to do soft cluster assignments and apply expectation maximization algorithm instead of k-means. But we did not observe a performance boost and hence kept the k-means formulation for simplicity. We have also experimented with using conditional generative adversarial networks, where we generated training samples from this mixture of gaussians. In that setup the discriminator is also tasked with discriminating images from different clusters in addition to separating generated images from the real ones. But the generator did not get good enough in the given time and data. We can speculate that the data augmentation with this conditional GAN method might also be more useful on larger datasets.

3 Performance

The main metrics we used was top1 and top5 accuracy in the given dataset. While a fully supervised neural net obtained 39.7% top5 accuracy, adding cluster distances as regularization has increased this score to 46.2%. This alone, however, does not provide enough information that doing clustering of the latent representations add value over the other regularization variables. It may be the case that this dataset is prone to overfitting and any kind of regularization does improve model performance even without using information from the unlabeled samples. To test this hypothesis, we applied weight decay and finetuned the model to archive optimum performance without using clustering loss. In addition, we observed that applying dropout to the raw input images (with probablity 0.05) and to the outputs of the average pooling layer (with probability 0.33) improved performance of the resnet18





model on this dataset. Using these regularization we were able to improve model's top5 accuracy to 51.3%. When we added cluster distances of the labeled and unlabeled data to our loss value, we observed that this value increased to 57.5%, showing that regularizing with the latent representation of the unlabeled data adds value even in existence of other regularizers.

We have also investigated model's performances while using different number of labeled samples. If we limit ourselves to 1 data point for each class, our method produces 4% top5 accuracy on the validation set. Having a second sample for each class increases this value to 5.6% and using 4 samples per class produces 8.6% top5 accuracy. Beyond that, the performance increases quite rapidly, reaching 13.9% with 8 samples, 23.8% with 16 samples, 34.8% with 32 samples and finally 57.5% with using the full 64 labeled samples.

References

[1] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features. 2018.

[2] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. 2017. URL `https://openreview.net/forum?id=rJY0-Kcll`.

[3] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. 2017. URL `https://arxiv.org/abs/1703.01780`.

[4] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra. Matching networks for one shot learning. 2016. URL `https://arxiv.org/abs/1606.04080`.