A Method of Moments Embedding Constraint for Generative Final Layers

Michael Majurski
Information Technology Lab, NIST
University of Maryland, Baltimore County
michael.majurski@nist.gov

Sumeet Menon
University of Maryland, Baltimore County
sumeet1@umbc.edu

Parniyan Farvardin
University of Miami
pxf291@miami.edu

David Chapman
University of Miami
dchapman@cs.miami.edu

Abstract

*Discriminative deep learning models with a linear+softmax final layer have a problem: the latent space only predicts the conditional probabilities $p(y|x)$ but not the full joint distribution $p(y, x)$, which necessitates a generative approach. The conditional probability cannot detect outliers, causing sensitivity to them in softmax networks. This exacerbates model over-confidence impacting many problems: from hallucinations, to confounding biases, and dependence on large datasets. We introduce a novel embedding constraint based on the Method of Moments (MoM). We investigate the use of polynomial moments ranging from 1st through 4th order hyper-covariance matrices. Furthermore, we use this embedding constraint to train an Axis-aligned Gaussian Mixture Model (AAGMM) final layer, which learns not only the conditional, but also the joint distribution of the latent space. We demonstrate our approach by extending FixMatch based semi-supervised image classification. We find our MoM constraint with the AAGMM layer is able to match or improve upon the reported FixMatch accuracy, while also modeling the joint distribution, thereby reducing outlier sensitivity. Future work explores potential applications for this layer and embedding constraint, and how/why this MoM technique can overcome theoretical limitations of other existing methods including the approximate KL-divergence constraint of variational autoencoders. Code is available at: https://github.com/*******.*

1. Introduction

[TODO: (majurski) anonymize before submission]

The majority of deep classifiers rely on a softmax final activation layer which predicts the conditional probability $p(y|x)$. When that layer receives input x, the model predicts a soft psuedo-distribution of labels y which argmax can convert into a hard label. If x is far from the decision boundary, then by definition, softmax assigns a prediction y with high confidence. This works well for inlier samples, well represented by the training distribution. However, when presented with an outlier x, it is likely x will be far from the decision boundary (**[TODO: Figure 1]**). This means that softmax perceptrons, by definition, over-

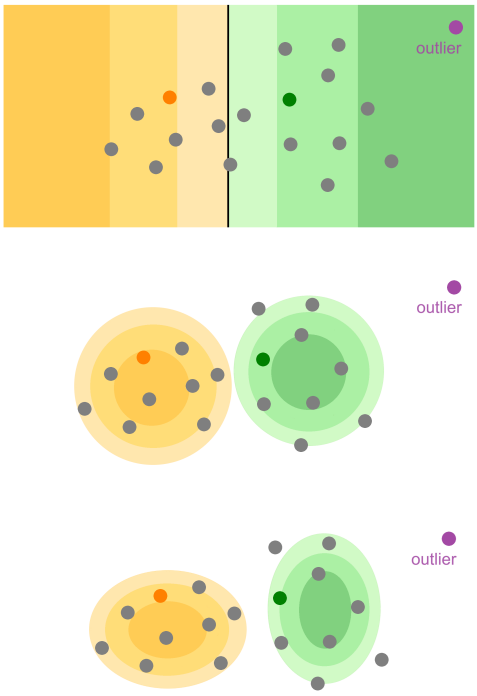

Figure 1. Schematic of the outlier problem, and how generative modeling of the joint probability can improve the situation. Prediction with (top) fully-superivsed softmax (middle) semi-supervised KMeans and (bottom) semi-supervised AAGMM.

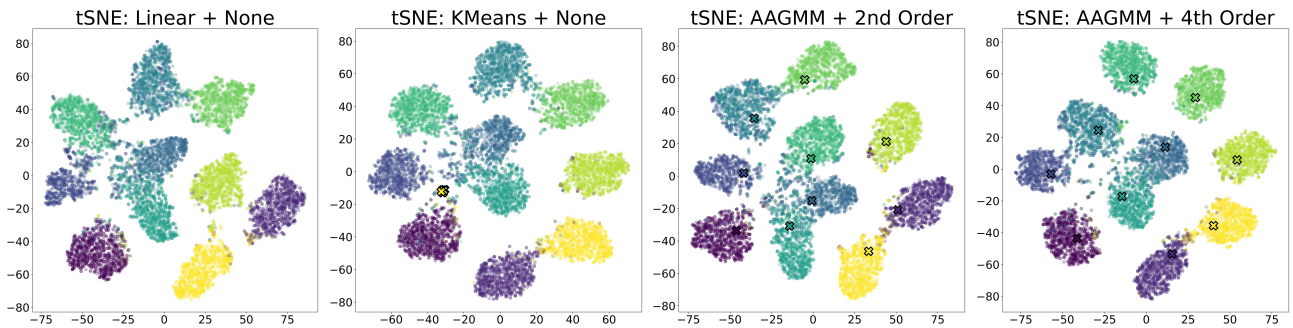

Figure 2. t-SNE[32] plot of the latent embedding space for various final layers with different MoM embedding constraints.

confidently hallucinate when given unexpected inputs [35]. Most deep classifiers use softmax without a safety net and thereby over-confidently predict y. Ideally, when input x is far from the decision boundary and training exemplars, the model should not be confident about the output class label y.

Replacing softmax with a generative method that models the joint probability $p(y, x)$ can improve the capability of deep classifiers. Models using a final layer capable of learning the joint probability $p(y, x)$ can infer the conditional $p(y|x)$. More importantly, such a layer can also infer the prior probability $p(x)$. Thus, if x is an unexpected input, then such layer can flag the input as a low-probability outlier, rather than confidently predicting a label.

Prior work has explored generative modeling for image classification [13, 14, 18]. Open questions remain of how to best train and utilize generative modeling within a deep learning context. The naive approach of minimizing cross entropy between y and y_pred will not work. Figure 2 (b) shows t-SNE plots for semi-supervised CIFAR-10 [15] image classification which illustrate why the naive approach will not work as intended. The t-SNE[32] plot of "KMeans + None" shows the latent space of a 93% accurate CIFAR-10 classification model. However, the explicitly modeled cluster centers (shown as X's) do not align with the underlying data. While that model has acceptable predictive performance, it does not accurately learn and represent the underlying training data. To construct a robust model, one cannot simply fit a decision boundary. The model needs to learn the full latent distribution. Figure 2 (d) demonstrates that with our proposed AAGMM final layer with 4th order MoM embedding constraints, the exact same model can achieve comparable (if not better accuracy), but with the added benefit of modeling the underlying data clusters in the latent space.

This work argues that Semi-supervised learning (SSL) pseudo-labeling methods can be improved with better calibration of the network logits used to filter reliable pseudo-labels from the unreliable ones. This is equivalent to improving the accuracy of class inlier determination. SSL is an excellent application to test generative final activation layers which model the joint probability $p(y, x)$, as pseudo-labeling methods are sensitive to outliers.

SSL leverages an abundance of unlabeled data to improve deep learning based model performance under limited training data regimes [11, 19, 44]. Contrastive learning methods leverage the intuition that similar instances should be close in the representation space, while different instances are farther apart [17, 38]. Consistency regularization borrows the intuition that modified views of the same instance should have similar representations and predictions [12, 16, 30, 41]. Pseudo-labeling methods like FixMatch [30] leverage the ideas of consistency regularization. This works contributes:

1. A replacement of the final linear+softmax final activation layer of the neural network with either an axis-aligned differentiable Gaussian Mixture Model (AAGMM) or an equal variance version named KMeans trained via back prop, both of which have explicit modeling of class cluster centroids.
2. A demonstration on the latent embedding space reacts to various constraints on how it should be structured by adding penalties if the per-class clustering does not conform to between 0 and 4 of the first gaussian moments being identity/zero [26].

We demonstrate this methodology using the standard CIFAR-10 benchmark dataset with 40 and 250 labels[15]. Finally, because the embedding constraint penalties are applied to all unlabeled data and not just the valid pseudo-labels, our method extracts training signal from every unlabeled data point, an improvement on baseline pseudo-labeling methods (like FixMatch [30]) which only learn from the valid pseudo-labels.

2. Related Work

Semi-Supervised learning has shown great progress in learning high quality models, in some cases matching fully

supervised performance for a number of benchmarks [41]. The goal of SSL is to produce a trained model of equivalent accuracy to fully supervised training, with vastly reduced data annotation requirements. Doing so relies on accurately characterizing inlier vs outlier unlabeled samples.

2.1. Pseudo-Labeling

Self-supervised learning was among the initial approaches employed in the context of semi-supervised learning to annotate unlabeled images. This technique involves the initial training of a classifier with a limited set of labeled samples and incorporates pseudo-labels into the gradient descent process, exceeding a predefined threshold [8, 20, 21, 23, 29, 39, 40]. A closely related method to self-training is co-training, where a given dataset is represented as two distinct feature sets [4]. These independent sample sets are subsequently trained separately using two distinct models, and the sample predictions surpassing predetermined thresholds are utilized in the final model training process [4, 27]. A notably advanced approach to pseudo-labeling is the Mean Teacher algorithm [31], which leverages exponential moving averages of model parameters to acquire a notably more stable target prediction. This refinement has a substantial impact on enhancing the convergence of the algorithm.

2.2. Consistency Regularization

Consistency regularization operates on the premise that when augmenting an unlabeled sample, its label should remain consistent. This approach implicitly enforces a smoothness assumption, promoting coherence between unlabeled samples and their basic augmentations [36]. In other words, the model should be able to predict the unlabeled sample x exactly the same way it predicts the class for $Augmented(x)$ [2, 3, 24, 30]. In addition to evaluating image-wise augmentations, recent research has demonstrated that incorporating class-wise and instance-based consistencies yields superior performance outcomes [17, 42]. Similarly, using consistencies between the predictions and low-dimensional embeddings from the unlabeled image strong and weak augmentations in a graph based setup demonstrates improvement over class-wise and instance-based consistencies [43]. Finally, pseudo-labeling filtering based on consistence between strongly augmented views, gaussian filtering and embedding based nearest neighbor filtering shows convergence improvement [12, 22].

2.3. Latent Embedding Constraints

Several papers have attempted to enhance the quality of pseudo-labels to either improve the final model accuracy, improve the rate of convergence, or avoid confirmation bias [1]. Rizve et al. [28] explores how uncertainty aware pseudo-label selection/filtering can be used to reduce the label noise. Incorrect pseudo-labels can be viewed as a network calibration issue [28] where better network logit calibration might improve results [37]. Improvements to the pseudo-labeling process have been demonstrated by imposing curriculum [41] or by including a class-aware contrastive term [38]. Leveraging the concept of explicit class cluster centers for conditioning semantic similarity improves final model accuracy [42]. Additionally, improvements have been found in extended purely clustering based methods like DINO [7] into semi-supervised methods [10].

3. Methodology

In this section, we explore our proposed replacement final activation layers and our embedding space constraints. FixMatch [30] is a simple, well performing SSL algorithm. As such, it serves as a good comparison point for exploring the effect of our contributions. Our methodology is based upon the published FixMatch [30] algorithm, with identical hyper-parameters unless otherwise stated. We extend FixMatch with a few minor training algorithm modifications explored in the Hyperparameters Section 4.1.

Both the linear layer replacements and the embedding constraints explored herein represent increasing levels of prescription about how the final latent embedding space should be arranged compared to a traditional linear layer. The idea of leveraging clusters in embedding space is not new [5, 6, 9], but we extend the core idea with a novel differentiable model with learned cluster centroids and MoM based constraints.

3.1. Alternate Final Layers

A limitation of traditional final activation layers such as linear+softmax is that they are fully discriminative; i.e. they estimate the posterior $p(Y|X)$, but do not attempt to model the sample distribution $p(X)$ or the joint probabilities $p(Y, X)$. To overcome this limitation, we present two semi-parametric final activation layers (a) the Axis Aligned GMM (AAGMM) layer, and (b) an equal variance version of AAGMM that we henceforth call the KMeans activation layer due to the similarity of the objective function with a gradient based KMeans.

These activation layers are fully differentiable and integrated into the neural network architecture as a module in the same way as a traditional final linear layer. As such, they do not require external training and do not use expectation maximization. They are drop in replacements for the final linear layer.

Importantly, these activation layers exhibit both discriminative and generative properties. The neural network model $F(X; \theta_F)$ transforms the data X into a latent space $Z = F(X; \theta_F)$, and the final activation layer estimates the probability densities $p(X)$, $p(Y; X)$ and $p(Y|X)$ by fitting a parametric model to the latent representation Z.

3.1.1 Axis Aligned Gaussian Mixture Model Layer

The AAGMM layer defines a set of K trainable clusters, one cluster per label category. Each cluster $k = 1 \dots K$ has a cluster center μ_k and cluster covariance Σ_k. The prior probability of any given sample X_i is defined by the mixture of cluster probability densities over the latent representation Z_i as follows,

$$p(X_i) = \sum_{k=1}^{K} \mathcal{N}(Z_i, \mu_k, \Sigma_k) \quad (1)$$
$$\textit{where} \quad Z_i = F(X_i, \theta_F)$$

Where $\mathcal{N}(Z_i, \mu_k, \Sigma_k)$ represents the multivariate gaussian pdf with centroid μ_k and covariance Σ_k. AAGMM is axis aligned because Σ_k is a diagonal matrix, as such the axis-aligned multivariate normal pdf simplifies to the marginal product of Gaussians along each of the D axes as follows,

$$\mathcal{N}(X_i, \mu_k, \Sigma_k) = \prod_{d=1}^{D} \frac{1}{\sigma_{k,d}\sqrt{2\pi}} exp\Big(\frac{Z_{i,d} - \mu_{k,d}}{\sigma_{k,d}}\Big)^2 \quad (2)$$
$$\textit{where} \quad \sigma^2_{k,d} = \Sigma_{k,d,d}$$

As there is one cluster per label category, the joint probability for sample i with label assignment k, $p(Y_{i,k}, X_i)$ is the given by the normal pdf of the k^{th} cluster,

$$p(Y_{i,k}, X_i) = \mathcal{N}(Z_i, \mu_k, \Sigma_k) \quad (3)$$

By Bayesian identity, the posterior probability $\hat{Y}_k = p(Y_k|X_i)$ can therefore be inferred from eq 1 and 3 as follows,

$$\hat{Y}_{i,k} = p(Y_{i,k}|X_i) = \frac{p(Y_{i,k}, X_i)}{p(X_i)} \quad (4)$$

The AAGMM layer is implemented as a normal PyTorch [25] module. It has two parameters updated by backprop. (1) the explicit cluster centers, a matrix $num\_classes \times embedding\_dim$ initialized randomly, and (2) the diagonal elements of the $Sigma$ matrix, randomly initialized in the range $[0.9, 1.1]$, which contains the diagonal elements of the GMM Sigma matrix for each cluster.

3.1.2 KMeans Layer

We also implement a KMeans final layer which is a more restrictive form of the AAGMM layer. The KMeans layer is additionally constrained such that the gaussian covariance matrix Σ_k for each cluster center k is the $[D \times D]$ identity matrix. This constraint yields spherical cluster centers; similar to how the traditional KMeans algorithm also assumes spherical clusters.

The KMeans layer is also implemented as a normal PyTorch [25] module. The explicit cluster centers is a learned parameter updated by backprop. See the published codebase for implementation details about the AAGMM and KMeans layers.

3.2. Method of Moments Embedding Constraints

We introduce and evaluate a series of embedding constraints based on the Method of Moments (MoM) [26] in order to fit the semi-parametric latent prior parameters. The latent prior $p(X_i)$ is calculated in equation 1 and then used to infer the posterior $p(Y_{i,k}|X_i)$. As usual, the posterior is trained using cross entropy loss. When embedding constraints are omitted, it is possible for the model to learn an accurate decision boundary for the posterior without modeling the latent prior.

Our novel last layer is semi-parametric, because the prior is a parametric model of the latent distribution Z which is the result of a neural network feature extraction $F(X; \theta)$. Therefore, attempting to fit the GMM directly to Z using Maximum Likelihood (ML) or simple Expectation Maximization (EM), is not appropriate, because doing so would fail to learn an appropriate feature space for discrimination. MoM solves these problems and is an appropriate strategy for semi-parametric models including ours.

The MoM relies on the use of *consistent estimators*, which asymptotically share sample and population statistics. Assume that z is a finite sample of n elements drawn from infinite population Z, then a series of P well-behaved sample statistics g_p should very closely approximate their k population statistic as follows,

$$\forall p = 1 \dots P \quad \frac{1}{n}\sum_{i=1}^{n} g_p(z_i) \approx E(g_p(Z)) \quad (5)$$

We can therefore constrain the latent representation of our model to approximate an independent joint Gaussian distribution. In the univariate gaussian case, the p^{th} order centralized moment constraint is the following.

$$E\left[(Z-\mu)^p\right] = \begin{cases} 0 & \text{if } p \text{ is odd} \\ \sigma^p(p-1)!! & \text{if } p \text{ is even} \end{cases} \quad (6)$$

By this formula, the univariate unit gaussian has mean 0, standard deviation 1, skew 0, and kurtosis 3.

In the joint multivariate case, each dimension is independent by definition. As such, if we redefine Z, μ, and p to be all D dimensional, then the centralized joint gaussian

moment can be defined as follows,

$$E\left[g_p(Z-\mu)\right] = E\left[\prod_{d=1}^{D}(Z_d-\mu_d)_d^p\right] \tag{7}$$

Due to independence of the axes, this moment can be represented as a product of univariate moments of the individual gaussians as follows,

$$E\left[\prod_{d=1}^{D}(Z_d-\mu_d)_d^p\right] = \prod_{d=1}^{D} E\left[(Z_d-\mu_d)_d^p\right] \tag{8}$$

The error (loss) term associated with the embedding constraint for any moment p is equal to the L2 difference between the sample and population statistics as follows,

$$\varepsilon_p = \left(\frac{1}{n}\sum_{i=1}^{n} g_p(z_i) - E(g_p(Z))\right)^2 \tag{9}$$

Some moments are more important than others, and must be weighted more heavily. First order moments are simply the sample mean, and should be given the greatest weight as an embedding constraint. The second order moments form a sample covariance matrix, which ideally should be equal to the identity matrix, but the diagonal terms should be given greater weight than the off-diagonal terms. This is because, in a $D \times D$ covariance matrix, there are $D(D-1)$ off diagonal terms, but only D, diagonal terms. The p^{th} order sample moments form a $p-1$ dimensional hyper-covariance matrix, with terms residing on the intersection of anywhere between 0 and $p-1$ hyper-diagonals. To prevent over-representation of off-diagonal terms and encourage representation of on-diagonal terms, the loss function we use for any given moment term is inversely proportional to the number moment terms that share the same number of hyper-diagonals. This heuristic weighting scheme ensures that the overall contribution of each moment order is not overly influenced by the off-diagonal terms, and that the error weighting is therefore diagonally dominant.

4. Experiments

We evaluate both AAGMM and KMeans linear layer replacements and the embedding space constraints using our modified FixMatch[30] on the common SSL benchmarks CIFAR-10 [15] at 40 and 250 labels (4 and 25 labels per class). We randomly selected 5 seed a priori for evaluation. For each algorithm configuration tested one model was trained per seed. During each run, the required number of labeled samples are drawn without replacement from the training population of the dataset in a deterministic manner (reproducible with the same seed). All data not in this labeled subset is used as unlabeled data (i.e. the labels are discarded).

AAGMM+None on CIFAR-10 at 40 Labels

Run Number	1	2	3	4	5
Test Accuracy %	94.6	92.8	92.8	89.9	86.4

Table 1. Test accuracy for showing the run-to-run variance depending on the quality of the 40 labels selected from the full population.

As prior work [30] has noted, the resulting model quality is highly variable when only 4 samples are selected per class, as the quality and usefulness of the specific 4 samples can vary drastically. Table 1 shows final test accuracy for the 5 AAGMM model runs with no embedding constraints, with the accuracy varying from 86% to 94%. Due to the potential for significant variance in the final model test accuracy, it can be informative to compare mean performance with max performance over the $N = 5$ runs. This explores both how well a method can be expected to do on average with random label sampling, vs how well it can potentially do with a more representative subset of labeled data.

4.1. Hyper-Parameters

All CIFAR-10 models were trained with the standard benchmark WideResNet28-2 architecture. This work leveraged the published FixMatch [30] hyper-parameters; using SGD with Nesterov momentum, $\lambda_u = 1$, $\beta = 0.9$, $\tau = 0.95$, $\mu = 7$, $B = 64$, and epoch size = 1024 batches regardless of the number of images in the labeled dataset. Model weights were updated as the moving average of the training weights with an exponential moving average (EMA) decay of 0.999. The training algorithm was modified from stock FixMatch to include an early-stopping condition when the model has not improved for 50 epochs. The $learning\_rate(\eta) = 0.01$. Replacing the fixed number of training steps with an early stopping criteria prevents the use of a cosine decay schedule. Therefore, it was replaced with a plateau learning rate scheduler which multiplies the learning rate by 0.2 every time the early stopping criteria is met (before being reset) for a max of 2 reductions. To reduce the training algorithm dependence on specific learning rate values, a cyclic learning rate scheduler was employed to vary the learning rate by a factor of ± 2.0 within each epoch. Additionally, due to the higher training instability of the AAGMM layers compared to a linear layer, if the training loss is greater than 1.0, the gradient norm was clipped to 1.0. Despite specific attention to computing the AAGMM and embedding constraints in a numerically stable manner, they are still less stable during backprop than a simple linear layer.

This work includes an exploration of how various latent embedding dimensionalities affects the generative linear layer replacement. As such, the model architecture was modified with a single additional linear layer before

CIFAR-10 Mean Test Accuracy

Last Layer	Emb Dim	40 Labels (5 trials)				
Embedding Constraint		None	$1st$ Order	$2nd$ Order	$3rd$ Order	$4th$ Order
Linear	128	82.19 ±6.54				
(i.e. FullyConnected)	8	88.40 ±3.54				
AAGMM	128	**91.23** ±2.89	89.23 ±2.50	90.22 ±3.42		
	8	88.34 ±2.34	82.39 ±9.96	87.97 ±3.34	82.51 ±9.42	82.57 ±6.94
KMeans	128	81.13 ±2.17	89.74 ±2.62	**89.89** ±2.79		
	8	78.28 ±9.87	73.27 ±12.1	75.80 ±10.69	80.96 ±5.94	80.01 ±7.53
Last Layer	Emb Dim	250 Labels (5 trials)				
Embedding Constraint		None	$1st$ Order	$2nd$ Order	$3rd$ Order	$4th$ Order
Linear	128	94.61 ±0.11				
(i.e. FullyConnected)	8	93.72 ±0.57				
AAGMM	128	94.09 ±0.34	94.20 ±0.62	94.42 ±0.14		
	8	94.17 ±0.57	94.01 ±0.73	94.33 ±0.37	93.77 ±0.90	94.22 ±0.50
KMeans	128	92.83 ±1.16	93.29 ±0.92	93.16 ±1.25		
	8	93.71 ±0.89	94.09 ±0.50	93.64 ±0.99	94.31 ±0.39	94.09 ±0.59

Table 2. Mean test accuracy % for CIFAR-10 SSL benchmark comparing various configurations of our method. The FixMatch results in the table is our reproduction of the published results, using our training pipeline modifications. For CIFAR-10 the WideResNet model used by FixMatch has an embedding size of 128 dimension. Due to exponential GPU memory requirements only the 8D embedding can operate with higher order MoM embedding constraints. Results for a given order of embedding constraint include all lower constraints.

the output to project the baseline model embedding dimension (128 for WideResNes28-2) down to a reduced 8 dimensional space. The AAGMM and KMeans replacement layers with and without this reduced embedding space, were evaluated to determine whether the generative capabilities improve when not fighting the curse of dimensionality. Results listed with an embedding dimensionality of 128 do not include the additional linear layer which reduces the latent dimensionality. Therefore, results with 128D embedding represents an unmodified network architecture.

Due to exponential GPU memory requirements with each successive MoM moment, only the 8D embedding can operate with higher order MoM embedding constraints. Results for any given order of embedding constraint include all lower constraints.

4.2. CIFAR-10

The CIFAR-10 SSL benchmark was used to explore the full configuration space of our method. While both 40 and 250 label counts were used, the 250 label case SOTA is close to fully supervised accuracy. We include 250 performance to document our result is approximately equivalent to SOTA. The 40 label case provides a far more challenging task, though recent results have demonstrated accuracies that nearly match fully-supervised performance on CIFAR-10 (similar to 250 label CIFAR-10).

Table 2 summarizes the relative performance of our various configurations for both 40 and 250 labels. We reproduced FixMatch [30] using our hyper-parameters and didn't quite matching the published performance at 40 labels (250 labels matched). Hyper-parameter selection for our training algorithm is likely sub-optimal for baseline FixMatch. The "Linear (i.e. FullyConnected)" rows in table 2 represent the baseline fully connected linear last layer without additional embedding dimensionality projection.

For CIFAR-10 with 250 labels, all last layers perform reasonably close to semi-supervised SOTA, which itself is almost identical to the fully supervised CIFAR-10 test accuracy of 95.38% [34].

In addition to average performance, it is informative to examine the max test accuracy over the $N = 5$ random trials to understand how well the algorithm can do, with samples that are representative of the larger dataset. Table 3 demonstrates that in the best case, the AAGMM can get within 1% of SOTA [43] performance.

The modeled cluster centers vary in quality between individual model runs of the AAGMM layer due to the stochasticity of the training process. Figure 3 (a & b) showcases degenerate cluster centers. The Figure 3 (c) AAGMM model learned cluster centers that are an ok approximation of the underlying data. However, the embedding constraints encourage cluster centers which are better aligned with the underlying data, Figure 3 (d). It is worth noting that we did not observed the KMeans layer learning non-degenerate cluster centers without an embedding constraint. In contrast, the AAGMM layer can, under some circumstances, learn viable cluster centers.

Table 4 puts these results in context with the current SSL

CIFAR-10 Max Test Accuracy

Last Layer	Emb Dim	40 Labels (5 trials)				
Embedding Constraint		None	$1st$ Order	$2nd$ Order	$3rd$ Order	$4th$ Order
Linear	128	91.01				
(i.e. FullyConnected)	8	92.10				
AAGMM	128	**94.64**	91.33	92.74		
	8	90.40	93.25	92.11	91.95	92.01
KMeans	128	84.08	91.62	92.22		
	8	**93.21**	91.22	89.35	85.96	92.59

Table 3. Max test accuracy (%) for CIFAR-10 SSL benchmark with 40 labels comparing various configurations. This table shows the best-case performance of our various methods; without the effect of poorly representative labels selected for each class.

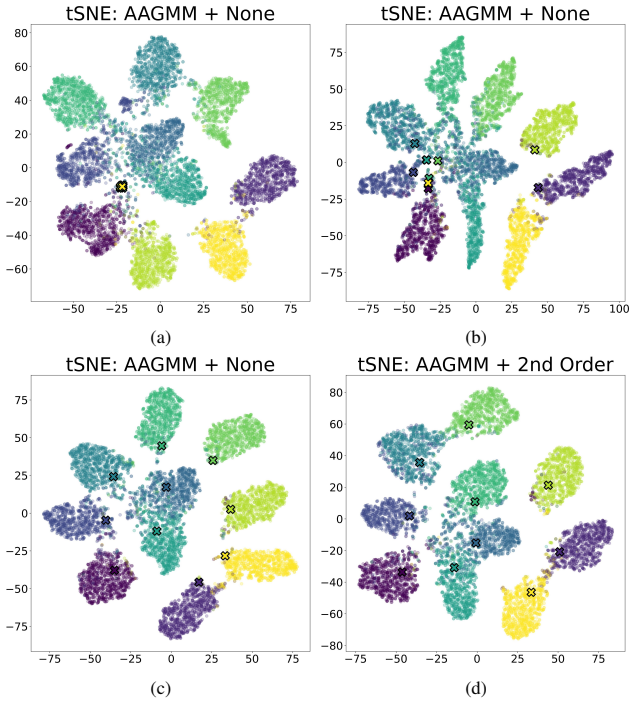

Figure 3. t-SNE plot of fully trained and reasonably accuracy AAGMM model's latent embedding space (with the learned cluster centers marked with X's). Depending on the run, the AAGMM cluster centers will be degenerate (top left), non-degenerate but still mis-aligned with the clusters (top right), acceptably aligned (bottom left), or well aligned with the underlying clusters when a 2nd order constraint is employed (bottom right).

SOTA for CIFAR-10 at 40 labels and demonstrates that this methodology still requires improvement before it is competitive with the latest methods.

5. Discussion

The proposed MoM embedding constraint is similar to the use of variational inference in deep learning based Variational Auto-Encoder models (VAE). VAE constrains

Method	CIFAR-10	
Label Count	40	250
FixMatch[30]	13.81 $\pm$3.37	5.07 $\pm$0.65
FlexMatch[41]	4.97 $\pm$0.06	4.98 $\pm$0.09
FreeMatch[34]	4.90 $\pm$0.29	4.98 $\pm$0.09
SimMatchV2[43]	4.90 $\pm$0.04	5.04 $\pm$0.09
Ours (AAGMM+None)	8.77 $\pm$2.89	5.91 $\pm$0.34
Ours (KMeans+2ndOrder)	10.11 $\pm$2.79	6.84 $\pm$1.25

Table 4. Error rate % for CIFAR-10 SSL benchmark comparing to state of the art results. Results for previously published methods are drawn from USB [33] except for FreeMatch[34] and SimMatchV2[43] publications.

the latent distribution using a KL-divergence penalty $D_KL(Q|P)$, where $Q(Z)$ is the variational distribution designed to approximate $p(Z|X)$, and P is the target distribution, typically the multivariate standard normal distribution. Although the use of KL-divergence is strongly rooted in information theory, true calculation of KL-divergence is intractable, and thus approximations are often employed. Therefore, the choice of variational distribution $Q(Z)$ can hide a great deal of approximation error, thereby limiting the VAE embedding constraint to penalize simple differences in the shape of the feature space.

In most VAE models the variational distribution $Q(Z)$ constrains the first order (mean) and second order diagonal moments (standard deviation). Second order off-diagonal terms are usually discarded by VAE models. Therefore, most practical VAE models cannot constrain the linear orthogonality of the feature space, because the cross-covariance terms are omitted to 'simplify' KL-divergence. In contrast, the proposed MoM latent embedding constraints not only encourage linear orthogonality, but also penalize more complicated non-linear shape descriptions including skew, kutorsis, and complex multivariate hyper-covariance descriptions of shape. By penalizing higher order off-diagonal terms, this method can produce feature spaces that are not only linearly orthogonal, but closely re-

semble true statistical independence.

The proposed MoM embedding constraint has one significant downside, it requires exponentially increasing amounts of GPU memory for each successive moment. This unreasonable GPU memory requirement limits the current practicality of MoM embedding constraints. Additional optimization and/or avoiding the explicit creation of both the nth order moment and its target value on device would likely improve the usability.

Semi-supervised learning is highly sensitive to both which samples are selected for the labeled population [30] and the stochasticity of the training process itself. Given identical starting random seeds, and identical labeled samples, training stochasticity will quickly cause models to diverge, resulting in vastly different final results. Anecdotally its appears worse in semi-supervised methods than fully supervised models. To characterize this variance, and hence how much one can trust the error bars for Table 2 and 3, we took a few final layer configurations and ran them $N = 5$ times with the same seed. Table 5 showcases the run-to-run variances for models that started out identical. Interestingly enough, the AAGMM models converge with much lower variance than the KMeans models.

Identical Seed Runs on CIFAR-10 at 40 Labels

	AAGMM +2nd Order	KMeans +2nd Order	AAGMM +None	KMeans +None
Run 1	87.5	86.9	71.2	86.5
Run 2	85.6	91.2	67.9	85.4
Run 3	84.8	77.7	75.8	86.0
Run 4	85.3	87.6	69.5	87.6
Run 5	85.2	83.7	$xx.y$	$xx.y$

Table 5. Test accuracy for independent training runs with the same random seed for various final layer configurations. All runs use the 128D (baseline) model latent embedding dimensionality. All runs use the same labeled samples.

Future work in this area should explore both accuracy improvements as well as implementation optimization to ensure the proposed novel final layers are not prohibitively memory expensive. Additionally, one should explore how to best take advantage of the better behaved latent embedding space to improve data efficiency for model training.

We demonstrate a novel fully differentiable Axis-Aligned Gaussian Mixture Model with Method of Moments based latent embedding space constraints to improve the generative inlier/outlier performance of image classification deep learning models. This preliminary work constructs those novel layers with the associated constraints, and demonstrates reasonable performance on challenging benchmark semi-supervised learning tasks.

References

[1] Eric Arazo, Diego Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020. 3

[2] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Advances in Neural Information Processing Systems*, 32, 2019. 3

[3] David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *International Conference on Learning Representations*, 2020. 3

[4] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100, 1998. 3

[5] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)*, pages 132–149, 2018. 3

[6] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924, 2020. 3

[7] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 3

[8] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009. 3

[9] Joseph Enguehard, Peter O'Halloran, and Ali Gholipour. Semi-supervised learning with deep embedded clustering for image classification and segmentation. *Ieee Access*, 7: 11093–11104, 2019. 3

[10] Enrico Fini, Pietro Astolfi, Karteek Alahari, Xavier Alameda-Pineda, Julien Mairal, Moin Nabi, and Elisa Ricci. Semi-supervised learning made simple with self-supervised clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3187–3197, 2023. 3

[11] Mohamed Farouk Abdel Hady and Friedhelm Schwenker. Semi-supervised learning. *Handbook on Neural Information Processing*, pages 215–239, 2013. 2

[12] Jiwon Kim, Youngjo Min, Daehwan Kim, Gyuseong Lee, Junyoung Seo, Kwangrok Ryoo, and Seungryong Kim. Conmatch: Semi-supervised learning with confidence-guided consistency regularization. In *European Conference on Computer Vision*, pages 674–690. Springer, 2022. 2, 3

[13] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[14] Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019. 2

[15] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, 2009. 2, 5

[16] Doyup Lee, Sungwoong Kim, Ildoo Kim, Yeongjae Cheon, Minsu Cho, and Wook-Shin Han. Contrastive regularization for semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3911–3920, 2022. 2

[17] Junnan Li, Caiming Xiong, and Steven CH Hoi. Comatch: Semi-supervised learning with contrastive graph regularization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9475–9484, 2021. 2, 3

[18] Yang Li, Quan Pan, Suhang Wang, Haiyun Peng, Tao Yang, and Erik Cambria. Disentangled variational auto-encoder for semi-supervised learning. *Information Sciences*, 482:73–85, 2019. 2

[19] Yu-Feng Li and De-Ming Liang. Safe semi-supervised learning: a brief introduction. *Frontiers of Computer Science*, 13: 669–676, 2019. 2

[20] Ioannis E Livieris, Konstantina Drakopoulou, Vassilis T Tampakas, Tassos A Mikropoulos, and Panagiotis Pintelas. Predicting secondary school students' performance utilizing a semi-supervised learning approach. *Journal of educational computing research*, 57(2):448–470, 2019. 3

[21] David McClosky, Eugene Charniak, and Mark Johnson. Reranking and self-training for parser adaptation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 337–344, 2006. 3

[22] Sumeet Menon. *Semi-Supervised Expectation Maximization with Contrastive Outlier Removal*. PhD thesis, 2022. AAI29167191. 3

[23] Sumeet Menon, David Chapman, Phuong Nguyen, Yelena Yesha, Michael Morris, and Babak Saboury. Deep expectation-maximization for semi-supervised lung cancer screening. 2019. 3

[24] Aamir Mustafa and Rafał K Mantiuk. Transformation consistency regularization–a semi-supervised paradigm for image-to-image translation. In *European Conference on Computer Vision*, pages 599–615. Springer, 2020. 3

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 4

[26] Karl Pearson. Method of moments and method of maximum likelihood. *Biometrika*, 28(1/2):34–59, 1936. 2, 4

[27] V Jothi Prakash and Dr LM Nithya. A survey on semi-supervised learning techniques. *arXiv preprint arXiv:1402.4645*, 2014. 3

[28] Mamshad Nayeem Rizve, Kevin Duarte, Yogesh S Rawat, and Mubarak Shah. In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. In *International Conference on Learning Representations*, 2021. 3

[29] Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. Semi-supervised self-training of object detection models. In *2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05) - Volume 1*, pages 29–36, 2005. 3

[30] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608, 2020. 2, 3, 5, 6, 7, 8

[31] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30, 2017. 3

[32] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9 (11), 2008. 2

[33] Yidong Wang, Hao Chen, Yue Fan, Wang Sun, Ran Tao, Wenxin Hou, Renjie Wang, Linyi Yang, Zhi Zhou, Lan-Zhe Guo, et al. Usb: A unified semi-supervised learning benchmark for classification. *Advances in Neural Information Processing Systems*, 35:3938–3961, 2022. 7

[34] Yidong Wang, Hao Chen, Qiang Heng, Wenxin Hou, Yue Fan, Zhen Wu, Jindong Wang, Marios Savvides, Takahiro Shinozaki, Bhiksha Raj, Bernt Schiele, and Xing Xie. Freematch: Self-adaptive thresholding for semi-supervised learning. In *The Eleventh International Conference on Learning Representations*, 2023. 6, 7

[35] Hongxin Wei, Renchunzi Xie, Hao Cheng, Lei Feng, Bo An, and Yixuan Li. Mitigating neural network overconfidence with logit normalization. In *International Conference on Machine Learning*, pages 23631–23644. PMLR, 2022. 2

[36] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. *Advances in neural information processing systems*, 33:6256–6268, 2020. 3

[37] Chen Xing, Sercan Arik, Zizhao Zhang, and Tomas Pfister. Distance-based learning from errors for confidence calibration. In *International Conference on Learning Representations*, 2020. 3

[38] Fan Yang, Kai Wu, Shuyi Zhang, Guannan Jiang, Yong Liu, Feng Zheng, Wei Zhang, Chengjie Wang, and Long Zeng. Class-aware contrastive semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14421–14430, 2022. 2, 3

[39] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd annual meeting of the association for computational linguistics*, pages 189–196, 1995. 3

[40] Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In

Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 1476–1485, 2019. 3

[41] Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *Advances in Neural Information Processing Systems*, 34:18408–18419, 2021. 2, 3, 7

[42] Mingkai Zheng, Shan You, Lang Huang, Fei Wang, Chen Qian, and Chang Xu. Simmatch: Semi-supervised learning with similarity matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14471–14481, 2022. 3

[43] Mingkai Zheng, Shan You, Lang Huang, Chen Luo, Fei Wang, Chen Qian, and Chang Xu. Simmatchv2: Semi-supervised learning with graph consistency. pages 16432–16442, 2023. 3, 6, 7

[44] Xiaojin Zhu and Andrew B Goldberg. *Introduction to semi-supervised learning*. Springer Nature, 2022. 2