COMBINING CLASSIFICATION AND CLUSTERING TASKS TO CATEGORIZE KNOWN AND UNKNOWN CLASSES

A Dissertation
Presented to
The Academic Faculty

By

Javeria Shabbir

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in the
College of Computing

Georgia Institute of Technology

December 2022

COMBINING CLASSIFICATION AND CLUSTERING TASKS TO CATEGORIZE KNOWN AND UNKNOWN CLASSES

Approved by:

Dr. Zsolt Kira, Advisor
School of Interactive Computing
Georgia Institute of Technology

Dr. Patricio A. Vela
School of Electrical & Computer Engineering
Georgia Institute of Technology

Dr. Judy Hoffman
School of Interactive Computing
Georgia Institute of Technology

Date approved: 12/12/2022

ACKNOWLEDGMENTS

First and foremost, I would like to praise Allah the Almighty, the Supreme, for the strengths and opportunities He has bestowed on me for this thesis.

I am highly grateful and deeply indebted to my MS supervisor, Prof. Zsolt Kira, who has provided me with constant supervision based on his broad research knowledge and the opportunity for independent thinking, to explore new research areas, which collectively helped me in flourishing my research ability through the MS duration.

Finally, the MS work would not have been possible without my husband's support, Muhammad Umair Mukati, who cooperated with me in sharing my personal responsibilities and helped me with some technical challenges due to his engineering background.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

ARI Adjusted Rand Index

cls cannot-links

CNN convolutional neural network

DEC Deep Embedded Clustering

JULE Joint Unsupervised Learning

KCL Kullback–Leibler divergence based contrastive loss

KNN k-nearest neighbors

MCL Meta Classification Likelihood

mls must-links

NMI Normalized Mutual Information

SCAN Semantic Clustering by Adopting Nearest neighbors

SUMMARY

Due to the human effort required for obtaining annotations for visual data (i.e. images), this research is focused on making use of unlabeled data. Under this setting, neural networks are currently trained using unsupervised and semi-supervised learning mechanisms. However, unsupervised learning methods involve no hint about the underlying structure of input data while semi-supervised learning methods provide information about the relationship between data points. In this thesis, we specifically look at *constrained clustering-based semi-supervised learning* where some limited additional information is provided as 1/-1 labels where 1 (must-link) indicates that images in given image pair belong to the same class and -1 (cannot-link) otherwise. Such labels can be obatined from category labels, though it is a weaker form of supervision and can also be obtained in various other ways as well (e.g. users comparing images of unknown labels, temporal tracking in videos to generate positive constraints, etc.). Currently, Semantic Clustering by Adopting Nearest neighbors (SCAN) [1] is the state-of-the-art algorithm in the area of unsupervised learning. This thesis investigates methods to improve the performance of the SCAN [1] algorithm using the additional constraints. Specifically, the SCAN algorithm consists of three steps: 1) representation learning through pretext tasks, 2) mining of positive pairs through nearest neighbor search, and 3) self-labelling. This thesis integrates semi-supervised learning in the second step of the algorithm, both by proposing to add the additional constraints to the nearest-neighbor based ones, but also proposing to use several additional constraint-based loss functions useful during clustering. Unlike the original SCAN algorithm, we propose these additional losses to fully utilize both the must-link and cannot-link constraints effec-

tively. The performance of the new setup was therefore analyzed by varying the number of must-link constraints (mls) and cannot-link constraints (cls) provided. Additionally, the clustering loss in step 2 was combined with Kullback–Leibler divergence based contrastive loss (KCL) Loss and Meta Classification Likelihood (MCL) Loss. All of the experiments outperformed the baseline performance of SCAN [1], and we show that the addition of the new loss functions further improves performance. Providing must-links proved to be more useful than compared to cannot-links. Very high accuracy is obtained in three cases: when clustering loss is combined with MCL Loss, secondly, when KCL Loss is combined with clustering loss in presence of supervision from both must-links (mls) and cannot-links (cls) and lastly, when a large number of mls are integrated into the second step of SCAN [1].

CHAPTER 1
INTRODUCTION AND BACKGROUND

1.1 Motivation

An immense amount of study is present in the Machine Learning, and specifically Computer Vision, literature for image classification and clustering tasks using a supervised learning paradigm. Supervised learning is the learning method in which the model is provided with images along with their annotations for training and at test time the model is asked to predict a cluster or label for the provided test image. A major drawback of this type of learning is the tremendous amount of labeled data required for the model's training. The availability of data annotations for a large dataset is not always possible as it requires much human effort to manually label each image. Therefore, in order to tackle a real-world problem where data is provided without any labels and the model can cover a wide range of classes, recent research has been directed towards self-supervised, semi-supervised, and unsupervised learning frameworks.

In the self-supervised learning technique, first, the model learns from unlabelled data via pretext tasks for which labels can be derived, and then it is transformed for a downstream task like clustering or regression using a small amount of labeled data in less amount of time. Since we have a large amount of unlabelled data at hand and it is increasing exponentially, the training of the model in a self-supervised way is not a problem. Besides, labels for model training on the pretext task are produced from the data itself and a range of such tasks are possible. For example, we have a significant number of text sentences. Similarly, in computer vision one can rotate the images and perform a pretext rotation prediction task [2]. We can mask some words of a sentence and train the model to predict the masked words. In this scenario, the masked words will act as a label for the model.

Semi-supervised learning refers to the type of training in which the model is provided with a combination of labelled and unlabelled data. Usually, the amount of unlabelled data is much higher as compared to labelled data. The model learns from labelled data and makes predictions on the unlabelled data; confident predictions can then be used to augment the original training labels. This learning technique is highly useful in scenarios where obtaining labelled data is much more costly. Combining a small amount of labelled data with a large amount of unlabelled data has shown significant improvement in the performance of a model as compared to the setting where only labelled or only unlabelled data is provided.

In an unsupervised learning setting, the model is trained with only unlabelled data. Here the model has to find underlying structures in different images and group them by learning the differences and patterns in the given input images. This learning framework is not less accurate due to many limitations, such as requiring a good feature space to cluster.

If we review the literature of the last few years, different algorithms in self-supervised, semi-supervised and unsupervised domains have been developed with assumptions that the model has information about the number of clusters, class names etc. whereas in real-world scenarios it is not always possible to have such information as a prior. Therefore, the SCAN [1] algorithm was presented which can perform clustering tasks using only unlabelled data when no prior information is provided.

1.2 Problem Statement

Images can be clustered using Representation learning where the steps of feature learning and clustering are decoupled. First, the model learns the features from images which are provided without annotations by performing a pretext task. A number of pretext tasks have been proposed like patch context prediction [3, 4], predicting noisy version of image [5], instance discrimination task [6, 7, 8, 9, 10], image colourization [11, 12], assigning the task of solving a puzzle to the network [13, 14], finding a rotated version of the given image [2],

generating a similar image for the given image [15] and many others.

The model weights are learned by minimizing the loss function of a given pretext task. After the model's training is finished, clustering is performed. This method of image clustering is not optimal and the generated clusters may not align with actual class groups.

Methods which combine feature learning and clustering have also been explored and are known as end-to-end learning. Complex convolutional neural network (CNN) trained on a large dataset is used as a prior. The weights of CNN are then refined by performing a clustering task on the target dataset. This technique showed performance improvement but there were some downsides. Using a trained network as a prior makes this method sensitive to initialization and hence it learns low-level features which are not suitable for clustering tasks.

SCAN [1] utilizes the positive points of representation learning and end-to-end learning and overcome their shortcomings. SCAN [1] performs feature learning and clustering as a two-step approach like representation learning. Firstly, it learns meaningful features from given input images by performing a pretext task. Then, unlike representation learning which performs k-means clustering, SCAN [1] finds the top 20 nearest neighbours for each image using the k-nearest neighbors (KNN) algorithm. Later, the image and each of its neighbours is assigned to the same cluster by maximizing the dot product between each image and its neighbour. In contrast to end-to-end learning, SCAN [1] relies on neighbours obtained by utilizing high-level feature similarity knowledge and not the network architecture.

In upcoming sections, we explain the details of the SCAN [1] algorithm in detail. The first step of the algorithm is pretext task-based learning. Due to the absence of labels, SCAN [1] requires some idea about which images are similar and which are not. For this purpose, the network is assigned a pretext task. The network learns its weights by reducing the objective function of the assigned task. After the completion of the network's training, the knowledge is utilized to obtain the top 20 neighbours for each image. Most of these

neighbours belong to the same category as the image but there are some false neighbours too. This image-neighbour information is utilized in step 2 of SCAN [1]. In the second step, the network is trained to assign the image and its neighbour to the same cluster. Degradation of performance due to incorrect neighbours obtained in step 1 is removed in step 3. During step 3, highly confident data points serve as a prototype for clusters and labels for less confident images are obtained.

1.2.1 Pretext Task

In the Supervised learning scenario, we have a ground truth label for each data point but in the current situation of unsupervised learning, we don't have access to labels. There should be some way that the network can learn to group the images. One of the popular methods to give prior information to the network in case of an unsupervised learning situation is assigning the network a pretext task. Various pretext tasks including image colorization [11, 12], rotation prediction [2], noise prediction [5], instance discrimination [6, 7, 8, 9, 10] have been explored. By minimizing the loss function of a given pretext task, the network learns weights which gives the network idea to distinguish similar and dissimilar images.

End-to-end Learning utilizes CNN as a prior. The convolutional neural network is given a large labelled dataset and tasked to perform clustering on given input images. The dataset given to the CNN for initial training is different from the dataset on which we need to perform clustering; in essense the goal is to transfer the learned clustering network from the auxiliary labelled dataset to the unlabelled dataset. After completion of training, the network learns to group the target dataset. Another way of training a similarity-based CNN is to train it such that the network assigns the image and its augmented image to the same cluster. Both of these approaches share a downside in that the network used for the target dataset is sensitive to initialization. Moreover, when the network starts learning from the target dataset, it first learns low-level features like colour and pattern. Before the training reaches the point where the network extracts high level features, it uses information like

contrast, texture and colour for grouping images into different clusters which results in improper image classification.

Representation Learning overcomes the downside of end-to-end learning and, therefore, it is used as the first step in SCAN [1]. The network learns weights θ by minimizing the objective function of a given pretext task in a self-supervised manner. Hence, it maps given input images to feature learning function Φ_θ. As there has been an emphasis on unsupervised and semi-supervised learning for a while different methods have been explored for network training in a scenario of data without labels. The pretext task is one of the successful methods through which a network can learn about a given unlabelled dataset itself. Different options for pretext tasks are image generation, predicting rotated image for the input image, predicting noisy image for the input image, and training the network by assigning the task to solve a given jigsaw puzzle. Any pretext task that does not depend on the specific transformation of the image can be chosen because we want the feature learning function to be independent of certain image transformations. For example different affine transformation of the same image results in a different output from the feature learning function. Although Φ_θ should predict the same cluster for different transformations of an image but it can be transformation dependent if the pretext task used to learn Φ_θ relies on specific transformation. To mitigate this problem, the objective function is designed in a way that assigns the same cluster to the image and its augmented images. It can be shown mathematically as:

$$\min_{\theta} d(\Phi_\theta(X_i), \Phi_\theta(T[X_i])) \tag{1.1}$$

In the above equation, X_i is an image and $T[X_i]$ represents a transformed image.

To illustrate, instance discrimination task and predicting rotated image for a given image both can be used as pretext tasks but the instance discrimination task satisfies Equation 1.1 and reduces the distance between the image and its transformed version. On the other hand, the rotation prediction task does not consider the relation between the image

and the augmented images. Therefore, the accuracy of 83.5% and 87.6% is obtained using two different variants of instance discrimination task [6, 8]. On the contrary, an accuracy of 74.3% is obtained when the rotation prediction task is used as a pretext task. The results demonstrates that the pretext task which classifies the image and the respective augmented image in one cluster is likely better suited for the SCAN [1] algorithm.

To understand why Φ_θ can classify the image and the transformed image into one cluster, we need to understand two points. First is that Φ_θ learns features from a given input image. Secondly, it is impossible for Φ_θ to understand similarities between image and transformed image if it only relies on low-level features like colour or texture. Hence, it is clear that Φ_θ learns high-level features which helps it to assign the same group to the image and its transformed images.

The knowledge learned through pretext task helps the network to gain insights about the given dataset but accuracy obtained at this stage is not good enough. Therefore, the pretext task is only used as an initial step in the pipeline of algorithm. Now, the obtained knowledge will be used as prior for the next step which will improve the performance of network. Figure 1.1 shows overall working of this step.

1.2.2 Clustering

This step consists of two sub-steps. The first step is to mine positive pairs (pseudo-constraints) via the nearest neighbours for each data point and the second step is to cluster the image and the respective neighbours in the same group. Working of this step is illustrated in Figure 1.2.

Mining top nearest neighbors: Through different experimental setups, it has been shown that the pretext task is a good option to extract image features but simply applying k-means for clustering on the extracted features results in imperfect clustering, e.g. fewer number of clusters than there are actual classes. Therefore, to avoid this problem SCAN [1] mines

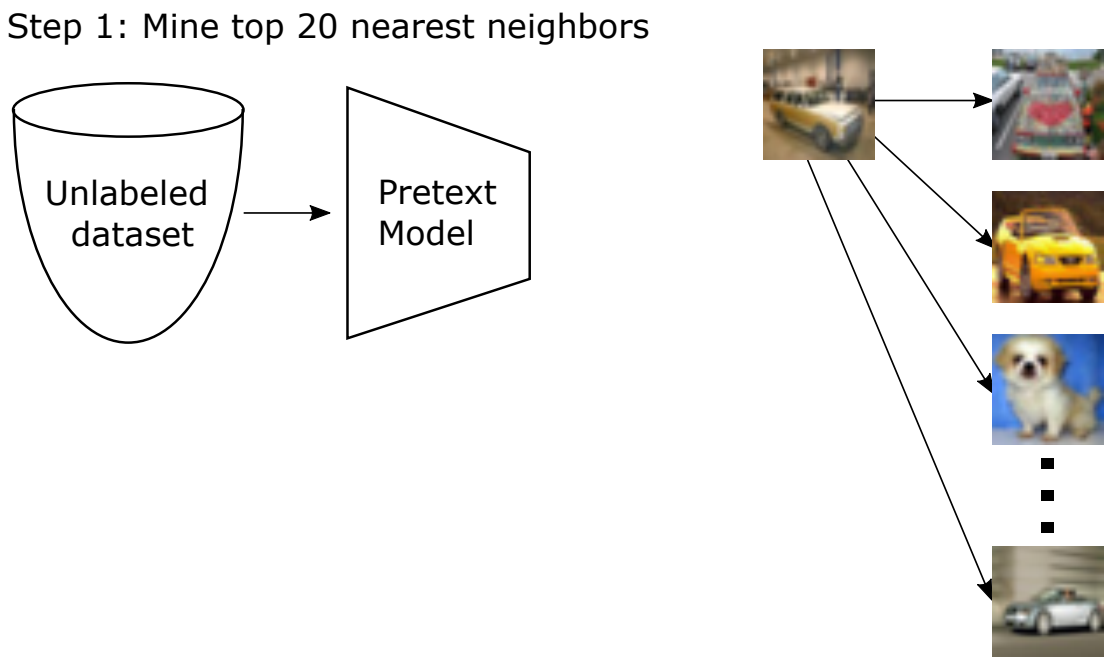

Figure 1.1: Working of first step of SCAN i.e. pretext task

positive pairs based on nearest neighbours for each image, utilizing the learned features to determine the neighbors. These are called pseudo-constraints.

Given unlabeled dataset $\mathcal{D}$, first, the network learns image features by performing a pretext task. The network's weights are updated by minimizing the objective function of the given pretext task. Then for each image X_i in dataset $\mathcal{D}$, we mine top K neighbours represented by N_{X_i} in the learned embedding space Φ_θ using the KNN algorithm. SCAN [1] finds top 20 nearest neighbors for each image. This information about the image and its neighbours is integrated into the next step where we actually group the images.

Clustering Loss: Next, a new network Φ_η with parameters denoted by η is trained given the samples and mined neighbours. The network learns by assigning the same cluster to the image and its neighbour. One of the neighbors from 20 neighbours is selected randomly. The last layer of the network is Softmax which produces the probability for the image belonging to each possible cluster represented by $C = 1, ..., C$. Mathematically, the loss function is represented as:

$$\Lambda = -\frac{1}{|\mathcal{D}|}\sum_{X\in\mathcal{D}}\sum_{k\in N_X} log\big\langle \Phi_\eta(X), \Phi_\eta(k)\big\rangle + \lambda\sum_{c\in C}\Phi_\eta^{'c}\, log\, \Phi_\eta^{'c} \tag{1.2}$$

$\big\langle . \big\rangle$ denotes the dot product in the above equation. The probability that image X_i belongs to some cluster c is indicated by $\Phi_\eta^c(X_i)$. The mathematical definition for $\Phi_\eta^{'c}$ is:

$$\Phi_\eta^{'c} = \frac{1}{|\mathcal{D}|}\sum_{X\in\mathcal{D}}\Phi_\eta^c \tag{1.3}$$

There are two terms in Equation 1.2. The first half of the equation calculates the dot product between the image and its neighbours to assign same group to both. The second term is about entropy and its purpose is to avoid assigning all images to the same cluster.

SCAN [1] assumes that the number of clusters is known and hence it is set equal to the actual number of classes. This is done to make performance evaluation possible. Otherwise, it is also possible to make a rough assumption about a possible number of groups and do over-clustering.

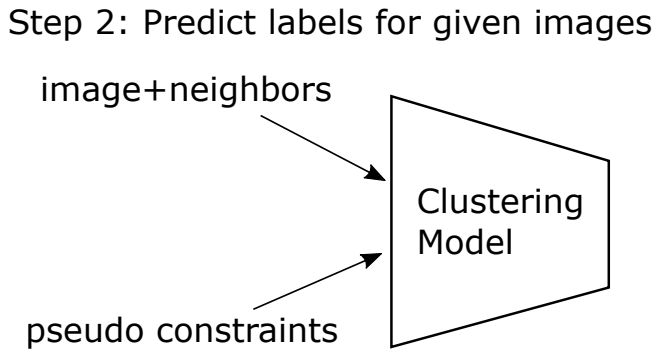

Figure 1.2: Working of second step of SCAN i.e. clustering

1.2.3 Self-labeling

In the previous step, the network learned by assigning the same cluster to the image and its neighbour. This neighbouring image is selected randomly from 20 mined neighbours. However, all of these 20 neighbours may not be true positive pairs. Due to false positives, the performance of the network is not as good as it should be. On investigating the probability for each image made by the network, it was observed that cluster assignment is correct for the images for which the network is highly confident. Therefore, SCAN [1] uses highly confident samples as prototypes for clusters and finds correct cluster prediction for low-confident samples. Figure 1.3 demonstrates the working of self-labeling step.

In the self-labelling step, SCAN [1] discards the predictions for samples for which the network is not highly confident while the rest are used as representatives of their respective clusters. During this step, the network re-calculates probabilities for the low-confident samples again to correct its previous mistakes. The threshold value used by SCAN [1] is 0.99. Cross-entropy loss is used on strongly transformed versions of confident samples to circumvent over-fitting. As the network finds correct predictions for less confident samples, it adds them to the prototype group. Hence, the network becomes more precise and hence the overall accuracy improves.

1.3 Thesis Structure

In this thesis, we adapt the SCAN algorithm to utilize a small number of ground truth constraints, investigating how to incorporate them into a semi-supervised version. The following is the flow of the thesis: Chapter 2 reviews relevant work about clustering and unsupervised learning. Chapter 3 explains how pseudo constraints were extracted from unlabeled dataset, hyper-parameter values, and the type of settings in which SCAN was tested. Finally, in Chapter 4 we present the results obtained through various variations of SCAN algorithm. The last Chapter 5 concludes the whole research.

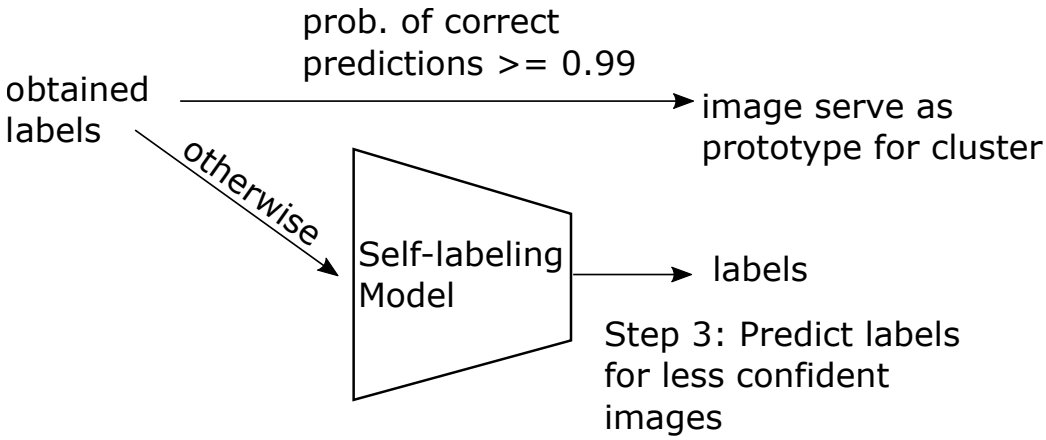

Figure 1.3: Working of third step of SCAN i.e. self-labeling

CHAPTER 2
RELATED WORK

2.1 Clustering

Caron et al. [16] proposed a clustering approach based on contrastive instance learning. Like our method, this technique is a pure unsupervised or self-supervised learning approach but the difference is that it is an online clustering algorithm and does not require going through the dataset multiple times.

The unsupervised learning method proposed by Yuki Asano [17] uses Representation Learning for clustering similar to SCAN [1]. To obtain a prior about labels for the given unlabelled dataset, it uses a self-labelling algorithm which yields labels for input images.

Yan et al. [18] introduced an algorithm called ClusterFit. In this technique, first, a pre-trained network extracts features which are clustered using k-means. Then a new network is trained from scratch on the same dataset using clustering labels obtained from the previous step as pseudo labels. ClusterFit uses a pre-trained network to extract features while SCAN [1] uses pretext tasks to achieve the same goal. Both of the methods use information obtained from the first step as prior in the second step. ClusterFit uses clustering assignment information obtained in the initial step while SCAN [1] finds the top nearest neighbours and integrate the obtained information in the second step.

An unsupervised clustering approach DeepCluster [19] works in a two-step. In the initial step, the network extract features from the input dataset and produces pseudo labels by grouping the features using the k-means algorithm. In the second step, the network updates its parameters by making a prediction for the labels produced in the first step. Unlike SCAN [1], both label generation and clustering are performed simultaneously. Also based on extracted features, DeepCluster groups input images in clusters using the K-means

algorithm while SCAN [1] utilizes KNN to find neighbouring images for each given image.

Due to the availability of a large amount of non-curated data, Caron et al. [20] emphasized making use of such data as it is easily available and therefore, proposed a method known as DeeperCluster. DeeperCluster combines clustering and self-supervision to improve the performance of the previously proposed algorithm called DeepCluster. DeeperCluster first trains the network to extract features of the image dataset and then produces target labels by clustering the extracted features. To obtain image features, DeeperCluster uses pretext task just like SCAN [1] but the main difference is the pretext task being used by it. DeeperCluster uses Image Rotation as a pretext task while SCAN [1] does not give good performance on the image rotation prediction task.

Joint Unsupervised Learning (JULE) [21] algorithm combines feature learning and clustering into one process unlike SCAN [1] which performs the mentioned tasks in two separate steps. The model performs clustering in the forward pass while learning features in the backward pass. Hence image groups and learned features both are updated in each epoch. Image clusters help the model to learn image features while improved image representation guides the model to group the images more correctly.

Xie et al. [22] proposed Deep Embedded Clustering (DEC) which transforms input data points to image features which are low dimensional as compared to the given input. The transformation of input data to low-dimensional space and clustering is performed simultaneously by the network. Similar to SCAN [1], this algorithm uses highly confident data points as prototypes for clusters to make a correct prediction for the rest. Ultimately SCAN has been able to obtain better performance than these prior methods, achieving state of art results across a number of datasets.

2.2 Unsupervised Learning

K-means has proved to be an effective algorithm for feature learning if applied properly. Therefore, Coates and Ng [23] proposed a method to learn features from unlabelled images

by training a network where layers are trained sequentially.

Bojanowski and Joulin [5] introduced a method to train networks for representation learning without any supervision. Here, the target representation is fixed and then the network is trained in a manner that features learned by the network align with target representation. SCAN [1] on the other hand, has no target representation and learns by performing pretext tasks.

Unlike SCAN [1], [24] performs feature learning and clustering jointly. Surrogate classes are formed by applying transformations to image patches. The network is trained to differentiate between the surrogate classes.

The method proposed by Liao et al. [25] learns from unlabeled data by optimizing objective functions based on k-means. As opposed to SCAN [1], it combines clustering and representation learning in a single step.

CHAPTER 3
PROPOSED METHOD

3.1 Adding positive and negative constraints

The neighbours of images obtained through the pretext task in step 1 of SCAN [1] can be called positive constraints. The goal of this thesis was to adapt the SCAN method to semi-supervised learning, where additional constraints are available from labelled data. Specifically, we tried to analyze the performance of SCAN [1] by supplying it with more positive constraints from the constrained dataset in addition to the ones obtained through the pretext task. The rows in a constrained dataset with value 1 in column ”constraint” as positive constraints. We also analyzed the performance by varying the number of positive constraints.

We also tested the impact of negative constraints on the performance of SCAN [1] to check if the algorithm learns anything from negative constraints. The images with different labels in the constrained dataset are referred to as negative constraints.

A key question is how such constraints can be incorporated into the SCAN algorithm. Below, we describe our proposal to integrate two loss functions, based on KL-divergence, that utilize these additional constraints. In the experimental section we show that this addition can significantly improve performance, beyond just naively adding the constraints alongside the mined nearest-neighbor based constraints in the original algorithm.

3.2 Integrating KCL Loss Function

Hsu et al. [26, 27] proposed a loss function called KCL to utilize learned weights for different domains and different tasks. To achieve this purpose, this objective function utilizes pseudo pairwise constraints which are called must-link or similar pairs if the two images

belong to the same class/category. If the paired images belong to different classes then the constraint is called a cannot-link or dissimilar pair. We have used the notation of 1 and -1 for must-link and cannot-link constraints respectively.

We therefore propose to utilize this loss function in order to leverage the additional supervised constraints. This loss function is easy to integrate with any neural network and is independent of the number of pairwise constraints. The probability distribution of both images from a pair is obtained from the neural network by applying the Softmax layer at the end. The probability distribution of the images in a pair is similar if the images forming a pair belong to the same cluster otherwise different. KL-divergence is used to calculate the distance between obtained distributions.

If we consider x_p, x_q are the two images from a pair and $P = f(x_p)$, $Q = f(x_q)$ are the respective probabilistic output, the following equation is used for the must-link pair.

$$L(x_p, x_q)^+ = D_{KL}(P^*||Q) + D_{KL}(Q^*||P) \tag{3.1}$$

Here, D_{KL} is defined as

$$D_{KL}(P^*||Q) = \sum_{c=1}^{k} p_c \, log(\frac{p_c}{q_c}) \tag{3.2}$$

Hinge Loss L_h is used in the case if the images in a pair are from different classes. The objective function in this case is defined as the following:

$$L(x_p, x_q)^- = L_h(D_{KL}(P^*||Q), \sigma) + L_h(D_{KL}(Q^*||P), \sigma) \tag{3.3}$$

The mathematical definition for Hinge Loss is:

$$L_h(e, \sigma) = \max(0, \sigma - e) \tag{3.4}$$

If $G_(x_p, x_q)$ defines pairwise similarity in binary fashion i.e. 1 for similar pair and 0 for

an otherwise then the overall mathematical definition of KCL Loss is defined as:

$$L(x_p, x_q) = G(x_p, x_q)L(x_p, x_q)^+ + (1 - G(x_p, x_q))L(x_p, x_q)^- \tag{3.5}$$

As we are giving information about pseudo constraints in addition to image-neighbour pairs, we carried out experiments in which we integrated the KCL loss function in addition to the loss function of SCAN [1] algorithm. We also studied the usefulness of similar and dissimilar pair information and which one is more important for the learning purpose of neural networks.

3.3 Integrating MCL Loss Function

Later Hsu et al. [28] introduced a new loss function for multi-class classification known as MCL Loss. To train a neural network using MCL Loss, no labels are required instead pairwise similarity information S_{ij} is needed. Such pairwise similarity information can be collected through various ways like using cross-transfer tasks or in a supervised or semi-supervised fashion.

If two samples x_i and x_j have the same labels Y_i and Y_j then S_{ij} is equal to 1 otherwise 0. First, a network is trained for binary classification. Given the pairwise similarity, the network learns to differentiate between similar and dissimilar images. The binary classifier is present only during the training of the network and is helpful to train the network for multi-class classification tasks. MCL Loss is represented as follows:

$$L_{meta} = -\sum_{i,j} s_{ij}\, log\, \hat{s}_{ij} + (1 - s_{ij})\, log\, (1 - \hat{s}_{ij}). \tag{3.6}$$

In the above equation, s_{ij} indicates actual similarity information between two samples while $\hat{s}_{ij}$ is the one predicted by the network.

CHAPTER 4

RESULT AND ANALYSIS

4.1 Dataset

For all of the experiments, the unlabeled CIFAR10 [29] dataset is used for performing pretext tasks which consist of images from 10 different classes. For integrating semi-supervised learning into the second step of SCAN [1], we extract a set of constraints from a subset of the labelled data. These pseudo constraints were extracted from the CIFAR10 dataset. Table 4.1 shows the resulting constraints for data shown in Figure 4.1

Figure 4.1: Random figures from CIFAR10 dataset

For each image in the dataset, a pairing image is picked randomly. If the label of both images is the same then such an image pair is assigned a constraint value of 1 otherwise

Table 4.1: Pseudo constraints extracted from CIFAR10 dataset to be used for semi-supervision in second step of SCAN [1]

image id1	image id2	label1	label2	constraint
1	2	dog	dog	1
2	5	dog	horse	-1
1	6	dog	ship	-1
10	12	cat	cat	1
4	5	horse	horse	1
5	8	horse	car	-1
6	5	ship	horse	-1
4	7	horse	truck	-1

-1. As the second image is selected randomly therefore it is highly likely that the selected image belongs to a different class. Therefore, the produced constrained dataset contains only 10% data with a constraint value of 1. 90% of data comprises image pairs in which paired images belong to different classes. The network in step 2 of SCAN [1] algorithm is supplied with the produced constrained dataset in addition to image-neighbour pairs produced through the pretext task.

4.2 Experimental Setup

All of the hyper-parameter values including the number of epochs and learning rate are kept the same as used in the original SCAN [1] experiment setting for fair comparison except the threshold value used in step 3. The SCAN [1] algorithm uses threshold value of 0.99 for all datasets. We have fine-tuned this parameter for each experiment.

Another major difference is in the split of the train/validation/test set. SCAN [1] uses a test set for validation and a training set for training the neural network. This could result in over-fitting to the test set during development and model selection. To use the test set solely for testing purposes, we have divided the training set into 90:10 ratios i.e. 90% of data will be used for training while 10% for validation.

4.3 Experimental Results

In this section, we discuss different modifications we made to the basic SCAN [1] algorithm including integrating constrained dataset in step 2, integrating KCL/MCL loss with clustering loss, and division of dataset into train/test/validation. We discuss the performance of each case concerning accuracy in comparison with basic SCAN [1]. All of the settings performed better compared to SCAN [1]. Table 4.2, Table 4.3 and Table 4.4 shows performance for step 2 of SCAN [1] in terms of accuracy, Adjusted Rand Index (ARI) and Normalized Mutual Information (NMI) respectively while Table 4.5, Table 4.6 and Table 4.7 represents performance for step 3 of SCAN [1] in terms of accuracy, ARI and NMI respectively .

Table 4.2: Result table showing accuracy for test set and validation set for step 2 of SCAN

Setup	constraints	clust. loss	add. loss	Val. Accuracy	Test Accuracy
SCAN	-	-	-	76.06	75.21
SCAN+DL	10k ml	ml	-	86.36	86.13
SCAN+DL	10k ml+cl	ml	-	80.48	79.57
SCAN+DL	10k ml+cl	ml+cl	-	81.88	81.4
SCAN+DL+KCL	10k ml+cl	ml	ml	82.39	82.63
SCAN+DL+KCL	10k ml+cl	ml	ml+cl	83.06	82.41
SCAN+DL+MCL	10k ml+cl	ml	ml+cl	83.5	82.72

Table 4.3: Result table showing ARI for test set and validation set for step 2 of SCAN

Setup	constraints	clust. loss	add. loss	Val. ARI	Test ARI
SCAN	-	-	-	0.5775	0.5625
SCAN+DL	10k ml	ml	-	0.7331	0.7282
SCAN+DL	10k ml+cl	ml	-	0.6434	0.6252
SCAN+DL	10k ml+cl	ml+cl	-	0.6663	0.6542
SCAN+DL+KCL	10k ml+cl	ml	ml	0.6725	0.6744
SCAN+DL+KCL	10k ml+cl	ml	ml+cl	0.6839	0.6706
SCAN+DL+MCL	10k ml+cl	ml	ml+cl	0.6908	0.6752

Table 4.4: Result table showing NMI for test set and validation set for step 2 of SCAN

Setup	constraints	clust. loss	add. loss	Val. NMI	Test NMI
SCAN	-	-	-	0.6423	0.6261
SCAN+DL	10k ml	ml	-	0.7592	0.7479
SCAN+DL	10k ml+cl	ml	-	0.6912	0.6714
SCAN+DL	10k ml+cl	ml+cl	-	0.6922	0.7058
SCAN+DL+KCL	10k ml+cl	ml	ml	0.7083	0.7063
SCAN+DL+KCL	10k ml+cl	ml	ml+cl	0.7202	0.7035
SCAN+DL+MCL	10k ml+cl	ml	ml+cl	0.7252	0.7076

Table 4.5: Result table showing accuracy for test set and validation set for step 3 of SCAN

Setup	constraints	clust. loss	add. loss	Val. Accuracy	Test Accuracy
SCAN	-	-	-	86.66	86.03
SCAN+DL	10k ml	ml	-	91.6	90.93
SCAN+DL	10k ml+cl	ml	-	87.28	86.63
SCAN+DL	10k ml+cl	ml+cl	-	88.28	88.42
SCAN+DL+KCL	10k ml+cl	ml	ml	88.68	88.38
SCAN+DL+KCL	10k ml+cl	ml	ml+cl	89.16	88.89
SCAN+DL+MCL	10k ml+cl	ml	ml+cl	90.08	90.08

4.4 Performance of SCAN with new division of dataset

As mentioned in Section 4.1, the split of the dataset for training, validation and testing purpose was changed as used by SCAN [1] algorithm. Previously, training data was used for training and part of the dataset for testing was utilized for validation by the SCAN [1] algorithm. this could potentially lead to over-fitting on the test set if significant model selection is done. To get a pure test set, we changed the dataset split. Now, the test set is used for testing while 90% of training data is used for training of neural network while the rest of 10% is being used for validating the network's performance.

For a fair comparison of SCAN [1]'s performance with different settings, we calculated its performance on the new division of the dataset. We used the same threshold value of 0.99 as used by SCAN [1] algorithm. Hence, we got a test accuracy of 75.21% for step 2 which increased up to an accuracy of 86.03% in step 3.

Table 4.6: Result table showing ARI for test set and validation set for step 3 of SCAN

Setup	constraints	clust. loss	add. loss	Val. ARI	Test ARI
SCAN	-	-	-	0.7448	0.7315
SCAN+DL	10k ml	ml	-	0.8271	0.8158
SCAN+DL	10k ml+cl	ml	-	0.7558	0.7412
SCAN+DL	10k ml+cl	ml+cl	-	0.7687	0.7703
SCAN+DL+KCL	10k ml+cl	ml	ml	0.7771	0.7700
SCAN+DL+KCL	10k ml+cl	ml	ml+cl	0.7861	0.7780
SCAN+DL+MCL	10k ml+cl	ml	ml+cl	0.8008	0.8009

Table 4.7: Result table showing NMI for test set and validation set for step 3 of SCAN

Setup	constraints	clust. loss	add. loss	Val. NMI	Test NMI
SCAN	-	-	-	0.7742	0.7601
SCAN+DL	10k ml	ml	-	0.8324	0.8179
SCAN+DL	10k ml+cl	ml	-	0.7829	0.7676
SCAN+DL	10k ml+cl	ml+cl	-	0.7860	0.7824
SCAN+DL+KCL	10k ml+cl	ml	ml	0.7927	0.7843
SCAN+DL+KCL	10k ml+cl	ml	ml+cl	0.8037	0.7917
SCAN+DL+MCL	10k ml+cl	ml	ml+cl	0.8108	0.8050

4.5 Performance of SCAN with 10k mls from 10k ml constrained dataset

For this scenario, we integrated the constrained dataset obtained from a small amount of labelled data (leading to a semi-supervised setting) in step 2 of SCAN [1]. Now, SCAN [1] gets additional information from the constrained dataset in addition to image-neighbour pairs obtained from step 1. Here, the constrained dataset consists of 10k must-links and no cannot-links. The information of neighbours obtained from step 1 has some errors but the neighbouring information provided by the constrained dataset consists of only true and correct neighbours. So, the network was trained using neighbouring information from step 1 and constrained dataset together. Therefore, the loss function of SCAN [1] was calculated on both data sources. The optimal value of the threshold was found to be 0.99.

The addition of 10k correct pairwise constraints improved the performance of SCAN [1] from 75.21% to 86.13% for step 2 while accuracy for step 3 increased from 86.03% to

90.93%. This proves that SCAN [1] algorithm requires more pairwise information for the training of the neural network and can benefit from more reliable constraints.

4.6 Performance of SCAN with mls from 10k ml+cl constrained dataset

The constrained dataset was integrated into step 2 of SCAN [1] algorithm. This additional dataset consists of 10k pairwise constraints out of which 90% pairs are cannot-links and 10% are must-links. This represents a more realistic case, as in reality ground truth information will be imbalanced and therefore the mls that we use in the algorithm will be fewer. Note that we are providing only must-links for the training of the network and for the calculation of loss. We are not using cannot-links at all in this setup. A threshold value of 0.97 was used in step 3.

Accuracy for step 2 and step 3 turned out to be 79.57% and 86.63% respectively. We can observe that both of the obtained accuracy values are low as compared to the ones obtained in Section 4.5. It proves that the number of must-link information provided to SCAN [1] has a big impact on its performance. SCAN [1] performs better when provided with more pairwise information. As we used 10k additional must-link pairs in Section 4.5 as compared to 1k in this case, therefore, we got improved accuracy in the former experiment.

4.7 Performance of SCAN with mls+cls from 10k ml+cl constrained dataset

Here, the the clustering loss in step 2 of SCAN [1] is calculated on images and mined neighbours along with a constrained dataset which consists of *both* must-links and cannot-links. Originally, the objective function in step 2 only accepts similar images either in form of an image-neighbour pair or must-link but we modified it to accept cannot-links too. 0.99 was used as a threshold value.

In terms of performance, in this setting, we obtained a test accuracy of 81.4% for step 2 which is 6% improved accuracy as compared to basic SCAN [1]'s performance. For step 3, we got an accuracy of 88.42% which is a 2% improvement.

4.8 Performance of SCAN with KCL Loss utilizing mls from 10k ml+cl constrained dataset

Here, we integrated the constrained dataset to step 2 of SCAN [1]. This dataset includes 90% of dissimilar pairs while 10% similar pairs which we have used.

Besides, we integrated KCL loss to SCAN [1] so total loss now is the summation of clustering loss and KCL loss. Loss function in this case is the summation of Equation 1.2 and Equation 3.5:

$$Loss = \Lambda + \lambda * L(x_p, x_q) \tag{4.1}$$

In the above equation, the optimal value of λ in this setting was found to be 1.

Semantic clustering loss is calculated using the network's output for pairwise data from step 1 and must-link pairs' output while KCL Loss is calculated for the network's output for similar pairs from the constrained dataset. The optimal value for the threshold was found to be 0.97.

Integrating KCL loss and 1k must-links proved to be useful and accuracy increased from 75.21% to 82.63% for step 2. The accuracy enhanced from 86.03% to 88.38% for step 3. The performance is low as compared to the accuracy obtained by integrating 10k must-links (which is 10x more labelled constraint data) but much improved if compared with base SCAN [1] performance.

4.9 Performance of SCAN with KCL Loss utilizing mls+cls from 10k ml+cl constrained dataset

For the current experiment, the network in step 2 of SCAN [1] learns from image-neighbour pairs and from the constrained dataset which consists of 10k similar and dissimilar image pairs. 10% of these images are must-links while the rest are cannot-links. Here, the network learns by minimizing not only clustering loss but also KCL loss. Clustering loss is

calculated on image-neighbour pairs and must-links from a constrained dataset while KCL loss is evaluated on similar and dissimilar images from a constrained dataset. Equation 4.2 is used here for calculating loss in step 2.

After fine-tuning, an optimal value of lambda was found to be 1 while the threshold value was set to 0.99. In terms of performance, the accuracy improved by much margin in comparison to the base SCAN [1] algorithm. For step 2, we obtained an accuracy of 82.41% while an accuracy of 88.89% was obtained for step 3.

4.10 Performance of SCAN with MCL Loss utilizing mls+cls from 10k ml+cl constrained dataset

In this case, loss in step 2 is the summation of MCL loss and clustering loss where the former is calculated using similar and dissimilar images from 10k must-link and cannot-link images while the latter is evaluated on image-neighbour pairs and must-links from the constrained dataset.

Here, loss function is the summation of Equation 1.2 and Equation 3.6:

$$Loss = \Lambda + \lambda * L_{meta} \tag{4.2}$$

We found the fine-tuned value of lambda to be 1.5 and the threshold to be 0.97. The performance, in this case, is again further improved over the previous experiment using KCL loss evaluated on both must-links and cannot-links. We found an accuracy of 82.72% on the test set for step 2 and an accuracy of 90.08% for step 3. This is the best performance that we obtained across comparable conditions (with about 1k labelled mls), demonstrating that our proposed method is effective.

CHAPTER 5
CONCLUSION

Since the past few years research has been directed towards the training of neural networks using unlabeled data or pairwise pseudo constraints known as unsupervised learning and semi-supervised learning respectively. In this thesis, we explored several methods to improve the performance of the current state-of-the-art algorithm for unsupervised learning called SCAN [1] using semi-supervision from pairwise pseudo constraints. The overall working of the SCAN [1] algorithm can be broken down into three steps: the first step is a pre-text task, the second is clustering and the third is self-labelling. We proposed to integrate semi-supervision in the second step of SCAN [1]. Experiments were carried out to evaluate performance by providing must-links and cannot-links. The more we increase the number of must-links, the more accuracy of the algorithm improves. One downside of adding the small amount of labelled constraints into SCAN is that this does not effectively utilize both must-link and cannot-link constraints into an integrated loss function. We therefore also proposed to utilize the additional labelled constraints by combining the clustering loss with the KCL Loss and MCL loss functions. The results show that KCL Loss performs better than the original SCAN algorithm in presence of both mls and cls. Working of SCAN [1] was evaluated by combining KCL and MCL Losses and the accuracy shows that MCL is better suited for SCAN [1]. Overall, the best accuracy was obtained when a large number of mls were provided to step 2 of SCAN [1] with our proposed usage of the MCL loss.

REFERENCES

[1] W. Van Gansbeke, S. Vandenhende, S. Georgoulis, M. Proesmans, and L. Van Gool, "Scan: Learning to classify images without labels," in *European conference on computer vision*, Springer, 2020, pp. 268–285.

[2] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," *arXiv preprint arXiv:1803.07728*, 2018.

[3] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1422–1430.

[4] T. N. Mundhenk, D. Ho, and B. Y. Chen, "Improvements to context based self-supervised learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9339–9348.

[5] P. Bojanowski and A. Joulin, "Unsupervised learning by predicting noise," in *International Conference on Machine Learning*, PMLR, 2017, pp. 517–526.

[6] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3733–3742.

[7] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9729–9738.

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

[9] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," in *European conference on computer vision*, Springer, 2020, pp. 776–794.

[10] I. Misra and L. v. d. Maaten, "Self-supervised learning of pretext-invariant representations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6707–6717.

[11] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *European conference on computer vision*, Springer, 2016, pp. 649–666.

[12] G. Larsson, M. Maire, and G. Shakhnarovich, "Colorization as a proxy task for visual understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 6874–6883.

[13] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *European conference on computer vision*, Springer, 2016, pp. 69–84.

[14] M. Noroozi, A. Vinjimoor, P. Favaro, and H. Pirsiavash, "Boosting self-supervised learning via knowledge transfer," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9359–9367.

[15] Z. Ren and Y. J. Lee, "Cross-domain self-supervised multi-task feature learning using synthetic imagery," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 762–771.

[16] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9912–9924, 2020.

[17] Y. M. Asano, C. Rupprecht, and A. Vedaldi, "Self-labelling via simultaneous clustering and representation learning," *arXiv preprint arXiv:1911.05371*, 2019.

[18] X. Yan, I. Misra, A. Gupta, D. Ghadiyaram, and D. Mahajan, "Clusterfit: Improving generalization of visual representations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6509–6518.

[19] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 132–149.

[20] M. Caron, P. Bojanowski, J. Mairal, and A. Joulin, "Unsupervised pre-training of image features on non-curated data," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2959–2968.

[21] J. Yang, D. Parikh, and D. Batra, "Joint unsupervised learning of deep representations and image clusters," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5147–5156.

[22] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *International conference on machine learning*, PMLR, 2016, pp. 478–487.

[23] A. Coates and A. Y. Ng, "Learning feature representations with k-means," in *Neural networks: Tricks of the trade*, Springer, 2012, pp. 561–580.

[24] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with convolutional neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[25] R. Liao, A. Schwing, R. Zemel, and R. Urtasun, "Learning deep parsimonious representations," *Advances in neural information processing systems*, vol. 29, 2016.

[26] Y.-C. Hsu and Z. Kira, "Neural network-based clustering using pairwise constraints," *arXiv preprint arXiv:1511.06321*, 2015.

[27] Y.-C. Hsu, Z. Lv, and Z. Kira, "Learning to cluster in order to transfer across domains and tasks," *arXiv preprint arXiv:1711.10125*, 2017.

[28] Y.-C. Hsu, Z. Lv, J. Schlosser, P. Odom, and Z. Kira, "Multi-class classification without multi-class labels," *arXiv preprint arXiv:1901.00544*, 2019.

[29] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.