Semi-supervised Learning via Triplet Network Based Active Learning

Student Name: Divyanshu Sundriyal

IIIT-D-MTech-CS-MT18096

June 2020

Indraprastha Institute of Information Technology
New Delhi

Thesis Advisors
Dr. Mayank Vatsa
Dr. Richa Singh

Submitted in partial fulfilment of the requirements for the Degree of M.Tech. in Computer Science

Certificate

This is to certify that the thesis titled **"Semi-supervised Learning via Triplet Network Based Active Learning"** submitted by **Divyanshu Sundriyal** for the partial fulfillment of the requirements for the degree of *Master of Technology* in *Computer Science & Engineering* is a record of the bonafide work carried out by him under our guidance and supervision at Indraprastha Institute of Information Technology, Delhi. This work has not been submitted anywhere else for the reward of any other degree.

Dr. Mayank Vatsa
Indian Institute of Technology Jodhpur

Dr. Richa Singh
Indian Institute of Technology Jodhpur

Abstract

Deep learning systems require a large amount of labelled training dataset. However large amount of labelled data is not available in many cases as it requires considerable human effort to label each sample correctly. In many cases like medical imaging, there is a small amount of labelled dataset along with large amount of unlabelled samples. In this research, we implement an Active learning algorithm which can help in increasing performance of deep learning models by using large amount of available unlabelled dataset. We propose a novel Active learning algorithm (Triplet AL) which uses a triplet network to select samples from unlabelled set for training classification model. Past active learning methods rely on classification model's final prediction scores as a measure of confidence for an unlabelled sample. We propose a more reliable confidence measure called Top-Two-Margin which is given by Triplet Network. We used STL-10 and CIFAR-10 dataset to test proposed algorithm. To test architectural independence of proposed algorithm, we tested proposed algorithm by using different model architectures for classification model. We compared results obtained using proposed method with past active learning methods. Proposed algorithm outperforms other active learning approaches we used to compare in our research.

Keywords : Active Learning, CNN, Triplet loss, Semi-Supervised learning

Acknowledgments

I sincerely thank my advisors Dr. Mayank Vatsa and Dr. Richa Singh for their support they provided during my thesis work under their guidance. They conducted regular meetings for discussion about thesis work and continuously guided me throughout my duration of thesis. I would also express my sincere gratitude to my mentor Soumyadeep Ghosh. Whenever I asked him for some guidance, he was always available to help me. Finally I would like to thank my family for their love and support which helped me to achieve my goals.

Contents

List of Algorithms

List of Tables

List of Figures

Chapter 1

Introduction

Deep learning systems require a large amount of data to train. For training these deep learning systems, the data should be labelled i.e. for each training example we need to have its corresponding label. The labels for each training example are generated using human annotators who examine each sample and assign a label to it. Sometimes this labelling is cheap e.g. in digit recognition an annotator requires very little effort to label a sample to its corresponding label. However there are many cases in which labelling a sample is expensive in terms of time or cost or both [22] . For example-

1. CIFAR-10 dataset we have only 50,000 labelled images of car, bird, horse, ship etc. However in internet there is a huge amount of such images present but these are unlabelled. Labelling these images to their correct class requires huge time and effort.

2. Labelling X rays- Labelling a sample of X-ray to contain a specific disease require an expert radiologist to label. Labelling it precisely requires time and effort making it an expensive process.

3. Speech Recognition- Expert linguistics are required to label each and every word of a speech correctly. It is highly time consuming as understanding each and every word of a speech utterance is a difficult task. If we require a speech recognition system for rare language it is very difficult to find an expert linguistic who can label the utterance correctly.

4. Labelling defect in concrete structures- It requires good civil engineers and high precision to correctly label the defect in concrete structures as mistakes in such cases can incur huge loss of money or life.

Active learning comes to rescue when there is a problem of expensive labelling. In Active Learning, we actively (smartly) choose samples from a large pool of unlabelled samples. We choose samples such that the chosen samples can lead to good training of deep learning model. After samples are chosen, we give them to expert to label. The expert in Active Learning literature is termed as 'Oracle'. We then use the chosen examples along with their labels to train the deep learning model.

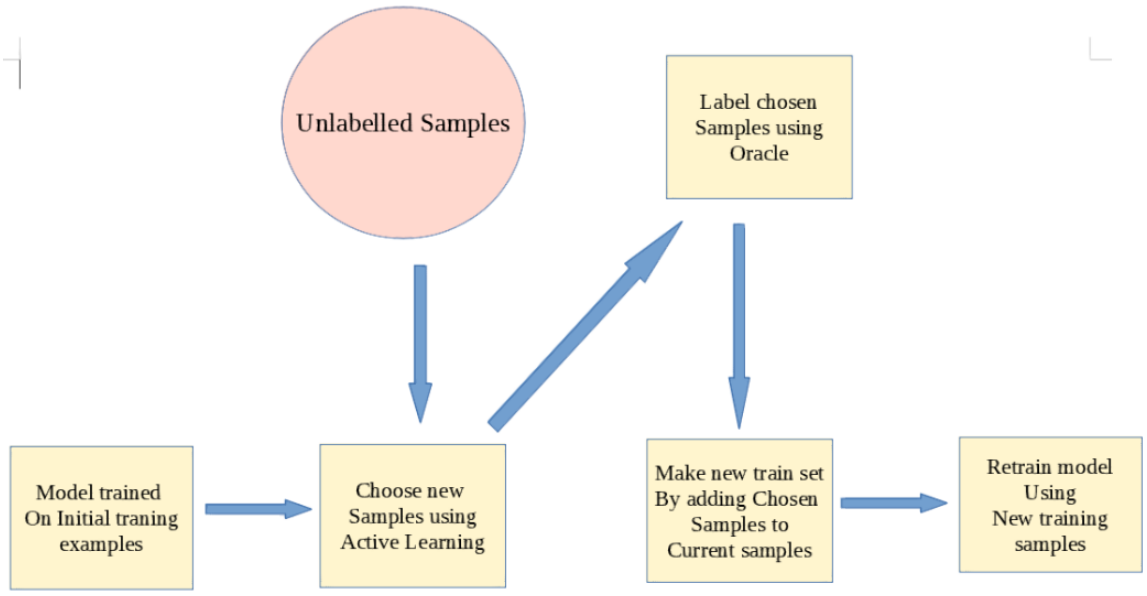


Figure 1.1: Single Iteration of Active Learning

1.1 Problem Statement and Motivation

In this research, our objective is to implement an Active Learning algorithm which can be used for Semi-Supervised learning. In Semi-Supervised learning, we have a small labelled set and a large unlabelled set and our goal is to increase model performance using unlabelled set.

There are many cases in real world where we encounter a situation where we have small number of labelled samples and large number of unlabelled samples. For example in CIFAR-10 dataset we have only 50,000 labelled images of car, bird, horse, ship etc. However in internet there is a huge amount of such images present but these are unlabelled. Our proposed algorithm can select samples from the unlabelled set in iterative manner and pseudo label the selected samples such that these pseudo labelled images when added to current initial labelled set can increase accuracy of classification model.

1.2 Research Contribution

We proposed a novel Triplet Network based Active learning algorithm **Tripet AL** to help deep learning models improve performance using unlabelled dataset. Figure 1.2 shows a graphical abstract of single iteration of proposed Triplet AL algorithm. Some key points about proposed active learning algorithm are following-

- In proposed algorithm, we use two models M1 and M2. M1 is a classification model and M2 is a Triplet Network. We aim to increase accuracy of model M1 using unlabelled dataset.
- We develop a Teacher-Student framework, where Teacher model M2 chooses samples for Student model M1, so that M1 can increase it's performance.
- We propose a term Top-Two-Margin as a measure to decide confidence score of model M2 on an unlabelled sample
- We compared proposed algorithm with conventional and two recent active learning algorithms. We achieved better increase in model performance when we used proposed algorithm than other active learning methods used.

- We also present a detailed analysis on why proposed method works better than other methods used in this research.

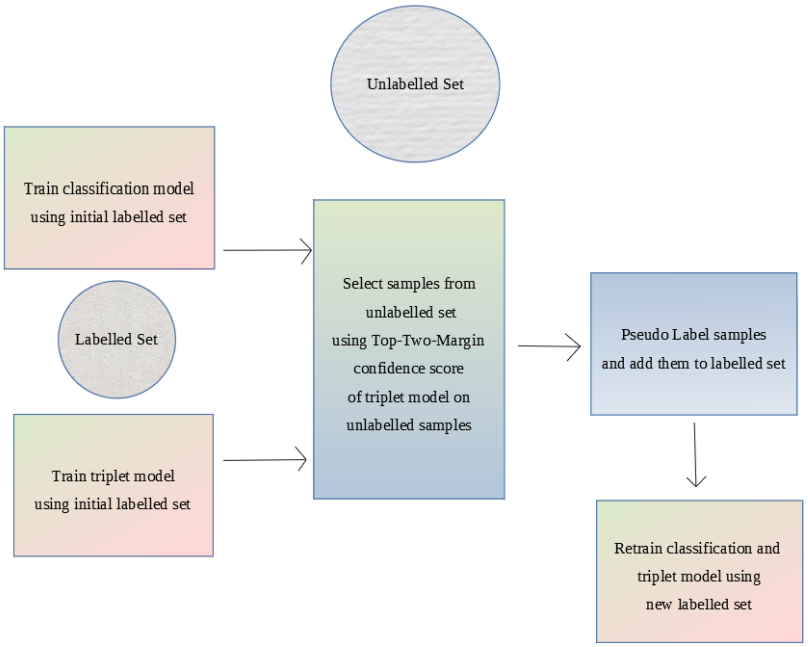


Figure 1.2: Single Iteration of Proposed Triplet AL Algorithm

1.3 Organization of Thesis Report

Chapter 2 discusses about conventional and recent Active learning methods used in previous works. In Chapter 3 we present proposed Active learning algorithm. Chapter 4 contains experiments we carried out in this research and their results. We show the results obtained in graphical and tabular formats. We also do a detailed analysis of comparison of proposed active learning method with other active learning methods used. We then discuss about why proposed method is effective. In chapter 5 we discuss about conclusions and future work.

Chapter 2

Related Work

In past many active learning strategies are proposed. These strategies differ in how we get the unlabelled data i.e. we get all unlabelled samples at once or we get them one by one and how the unlabelled samples are selected [22].

2.1 Active Learning Problem Settings

Active Learning can be used in different scenarios. The scenarios depend on how we get the new unlabelled data.

2.1.1 Stream Based Selective Sampling

In stream based selective sampling it is assumed that we get an unlabelled instance with no associated cost and cost encountered is only while labelling it. We get unlabelled samples one by one.

We then check whether it is beneficial to label the sample based on how informative the sample is. Informativeness of a sample can be drawn from methods like Uncertainty based method [3]. If sample is an informative sample, we ask the Oracle i.e. annotator to label the sample else we discard it.

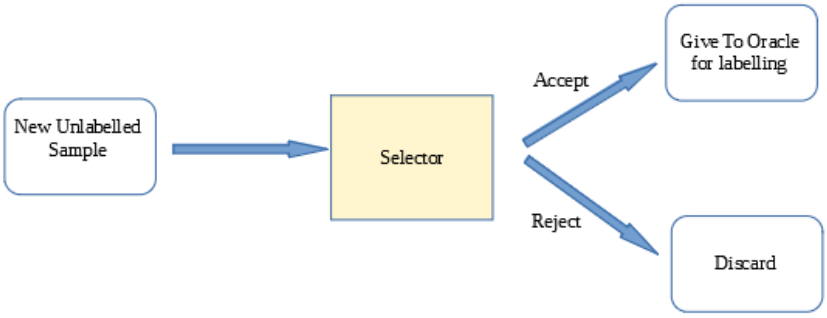


Figure 2.1: Stream Based Selective Sampling

2.1.2 Pool Based Sampling

In pool based sampling [17], we get all the unlabelled samples at once. We then select the most informative sample i.e. the samples which can help in better model training from the large pool of unlabelled samples. The chosen samples are then labelled by Oracle and then these are fed to the model to train.

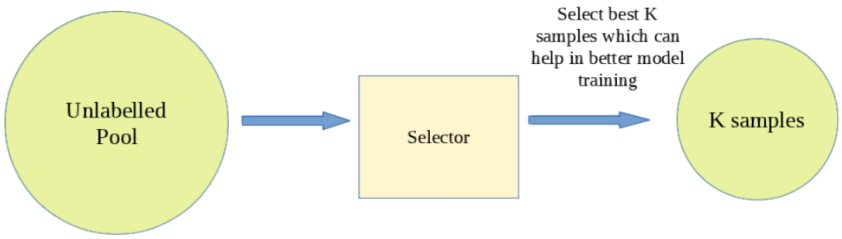


Figure 2.2: Pool Based Active Learning [17]

2.1.3 Membership Query Synthesis

In this setting unlabelled instances are synthesized by the machine. These synthesized unlabelled instances can sometimes be difficult to be labelled by a human annotator as the synthesized sample may not be having a meaning understandable by human [1]. So this setting is useful when there is an automatic annotator present along with the synthesizer [13].

2.2 Active Learning Methods

Active Learning methods select samples which can help in better training of model in each iteration. These samples are then labelled using Oracle and added to initial labelled set.

2.2.1 Classical Active Learning Methods

- Random Selection - In this query strategy, the samples whose label has to be asked from Oracle are chosen randomly. In stream based setting, we randomly accept or reject a new unlabelled instance and in pool based setting, we take say K samples from unlabelled pool randomly.
- Least Certainty Based Sampling - In this Uncertainity based sampling [17], those samples are chosen in which the model trained on initial samples is least certain i.e. the samples on which highest softmax probability score is least or the model predicts a category with a low probability score. For two class classification problem, a sample is uncertain if it's posterior probability of belonging to one of the class is near to 0.5 [16], [17].

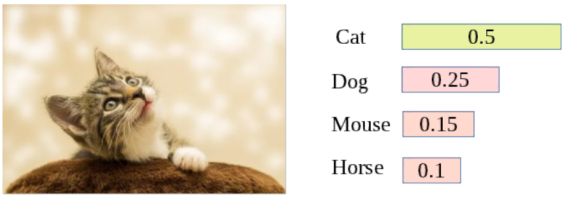


Figure 2.3: Uncertainty Based Active Learning [16]

In Figure 2.3, the model predicts that this image is cat by confidence of 0.5 only. So for this image, the model is uncertain. If we give such low confidence samples along with their correct labels, model can be trained faster.

- Most Certainty Based sampling - This is reverse of Least Certainty Based sampling where we select those samples on which model trained on initial samples is most certain. We require such samples when we are not given with the correct labels and we have to pseudo label them.

- Minimum Margin Based Sampling - In Minimum Margin based sampling [19], those samples are selected whose difference in top 2 class predictions is minimum.

Figure 2.4: Margin Based Active Learning [19]

In Figure 2.4, the cat which also nearly looks like a dog, top 2 prediction scores are 0.5 and 0.45 for cat and dog class respectively. The margin between top 2 classes is just 0.05 i.e. the model is unsure whether it is a cat or dog. So feeding this example with a correct label can help model to train faster.

- Maximum Margin Based Sampling - In Maximum Margin based sampling, those samples are selected whose difference in top 2 class predictions is maximum.

- Query By Committee - In Query by committee [23] strategy, we select those samples on which a committee of classifiers are in disagreement. We train say N classifiers using initial training data. Now the samples for which many classifiers disagrees with each other i.e. predicts different classes for that image, it means that models are unsure about the new image and giving this image along with its corresponding correct label can help to train model better.

2.2.2 Limitations of Classical Active Learning Methods

The classical Active learning methods like Uncertainty sampling etc. works very well in a machine learning setting or in shallow networks but they do not work well in Deep learning models. There are many limitations of conventional Active learning methods when they are used in Deep learning models.

- According to Wang et al. [26], Active learning methods assume that feature representation is fixed. However in CNNs along with task learning, features are also learned.

- According to Gal et al. [6], deep neural networks are dependent on large amount of data but in Active learning we generally talk of using small amount of data to train the model. Also in Active learning we talk about how uncertain model is but there is no representation of uncertainty for deep neural networks.
- Deep neural networks are composed of feature learning phase and task learning (Classification or Regression) phase. According to Tao He et al. [9], the uncertainty evaluated from the results of final layer indicates uncertainty in Task learning phase only. However since the overall learning consist of feature learning phase also, it is not good practice to consider uncertainty in feature learning phase only.
- According to Duffofe et al. [5], uncertainty is not a good measure as a model when given an image with small perturbations can result the image category wrong with high certainty.

2.2.3 Active Learning Methods for Deep Neural Networks

Many Active Learning methods are proposed recently which aim to work well for Deep neural Networks. These algorithms try to overcome the limitations faced by conventional Active learning methods when used with deep neural networks.

- Inspired from Query By Committee [23], Ducoffe & Precioso (2015) [4] used model dropouts to create model ensembles at test time. The partial CNNs created using dropout form a Committee of classifiers to decide the uncertainty of samples.
- Wang et al. (2016) [26] query most uncertain samples along with least certain samples and ask for their labels. According to the authors most certain samples help in improving the feature learning phase and least certain samples help in improving task learning phase of neural network.
- In Deep Bayesian Active Learning (BALD), Gal et al (2017) [6] used Bayesian CNNs and dropouts at time for sample acquisition.
- In Core Set approach by Sener et al. (2017) [21], samples are selected in batches instead of single sample as in traditional active learning. In this approach a

subset of samples is chosen (core-set) such that chosen points approximate the distribution of remaining points.

- In Deep Fool Algorithm proposed by Duffofe et al. (2018) [5] , most uncertain samples are those which have the shortest distance to boundary. Shortest distance to boundary is approximated as shortest distance to its adversarial example.

- Tao He et al. (2019) [9] suggest to compute uncertainty by not only output of last layer but also use outputs from previous layers also. Uncertainty from last layer indicates uncertainty in classification task. While uncertainty from previous layers indicates uncertainty in feature learning task. Final uncertainty is calculated by taking the weighted average of the uncertainties.

- Discriminative Active Learning proposed by Gissin el al. (2019) [7], selects samples such that labelled and unlabelled sets are indistinguishable.

- Adversarial Sampling for Active Learning proposed by Mayer et al. (2020) [18], is a GAN based method which synthesis samples and then selects similar synthetic samples from pool and add them to labelled set.

- A2-Link proposed by Suri et al. (2020) [25], uses adversarial noise to fine tune a model which is then used to select samples intelligently from target domain to labelled set.

Paper	Author	Dataset	Year
QBDC: query by dropout committee for training deep supervised architecture [4]	Ducoffe, Melanie and Precioso, Frederic	MNIST	2015
Cost-effective active learning for deep image classification [26]	Wang, Keze and Zhang, Dongyu and Li, Ya and Zhang, Ruimao and Lin, Liang	STL-10	2016
Deep bayesian active learning with image data [6]	Gal, Yarin and Islam, Riashat and Ghahramani, Zoubin	MNIST	2017
Active learning for convolutional neural networks: A core-set approach [21]	Sener, Ozan and Savarese, Silvio	CIFAR-10, CIFAR-100, SVHN, Caltech-256	2017
Adversarial active learning for deep networks: a margin based approach [5]	Ducoffe, Melanie and Precioso, Frederic	MNIST, Shoe-Bag, Quick-Draw	2018
Towards Better Uncertainty Sampling: Active Learning with Multiple Views for Deep Convolutional Neural Network [9]	He, Tao and Jin, Xiaoming and Ding, Guiguang and Yi, Lan and Yan, Chenggang	CIFAR-10, Fashoin-MNIST, SVHN	2019
Discriminative active learning [7]	Gissin, Daniel and Shalev-Shwartz, Shai	MNIST	2019
Adversarial sampling for active learning [18]	Mayer, Christoph and Timofte, Radu	MNIST, CIFAR-10, SVHN, Celeb-A	2020
A2-LINK: Recognizing Disguised Faces via Active Learning and Adversarial Noise based Inter-Domain Knowledge [25]	Suri, Anshuman and Vatsa, Mayank and Singh, Richa	Disguised Faces in the Wild [24]	2020

Table 2.1: Summary of Recent Active Learning Algorithms for Deep Neural Networks

2.3 Background on Loss Functions

In this research, we propose an Active Learning method which uses two models M1 and M2. M1 is a classification model which uses cross entropy loss and M2 is a embedding model based on triplet loss [11], [20]. So before moving into the proposed algorithm, a brief background of the loss functions used is discussed.

2.3.1 Cross Entropy Loss

Cross entropy loss is used to train a multi-class classification model. If y_i denotes label of class i and $y_i^{'}$ denotes softmax score of class i given by the model, Cross entropy loss L_{ce} for K class classification model is defined as-

$$L_{ce} = \sum_{i=1}^{K} y_i \times \log y_i^{'}$$

2.3.2 Triplet Loss

The goal of embedding learning algorithm is to learn a function $f_\theta : \mathbb{R}^M \longrightarrow \mathbb{R}^N$ where M is dimension of training data manifold and N is dimension of embedding space. Consider a training set where z_i^k represents i^{th} sample of k^{th} class. While training an embedding model using triplet loss, we have triplets of the form - $\{ z_a^C, z_b^C and z_c^{C'} \}$, where z_a and z_b are of same class C and z_c is of any other class except C. The aim is to minimize distance between anchor point z_a and positive point (i.e. point of same class) z_b and maximize distance between anchor point z_a and negative point (i.e. point of any other class) z_c (Refer Figure 2.5). The distance metric used is Euclidean distance. Euclidean distance D(x,y) between two points x and y is calculated as-

$$D(x, y) = \|f_\theta(x) - f_\theta(y)\|_2^2$$

For each sample X in training set we form a triplet consisting of X, a positive point P and a negative point N, Triplet loss L_{TL} proposed by Schroff et al. (2015) [20] is defined as -

$$L_{TL} = \max(D(X,P) - D(X,N) + \alpha, 0)$$
where α is margin parameter

Margin parameter α helps to separate positive and negative classes in the embedding space.

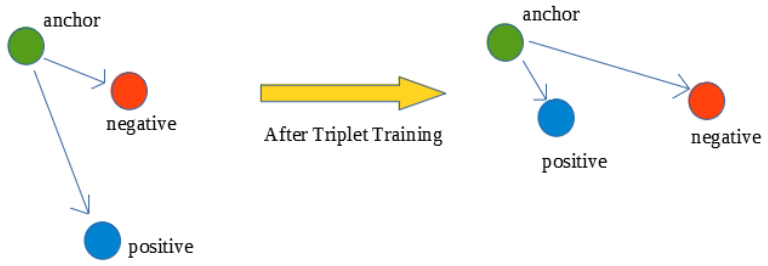


Figure 2.5: After Successful Training of Triplet Network, Distance Between Anchor and Positive is Minimized and Distance Between Anchor and Negative is Maximized [20]

In this research work we use Hard Mining Triplet loss [10]. In Hard mining triplet loss, to train embedding model we select a triplet which is hard. A hard triplet consists of 3 points Anchor point z_a^C, positive Point z_b^C and negative point $z_c^{C'}$ such that positive point is farthest from anchor point and negative point is closest to anchor point in the embedding space. In other words, for an anchor point, we choose a point of same class which is farthest to anchor point and we choose a negative point as a point of another class which is closest to the anchor point in the embedding space.

Chapter 3

Proposed Active Learning Method

In Semi Supervised Learning setting, we have initial labelled set L, a pool of unlabelled set P. Our goal in this research is to use the Unlabelled set P to improve performance of classification model.

In Conventional Active learning methods, we have a classification model M1 whose purpose is to learn a function $g_\theta : \mathbb{R}^M \longrightarrow \mathbb{R}^K$, where M is dimensionality of input manifold and K is no of classes. Model M1 is trained using initial labelled set. After initial training, samples from unlabelled pool P are selected based on confidence of model M1 on unlabelled samples.

In proposed novel **Triplet AL** method, along with classification model M1, a Triplet model M2 is also used. Purpose of model M2 is to learn a function $f_\phi : \mathbb{R}^M \longrightarrow \mathbb{R}^N$ using triplet loss, where M is the dimensionality of input data manifold and N is the size of embedding space. We start by training models M1 and M2 using initial labelled data. Then using confidence of model M2 on unlabelled samples, we select samples from unlabelled set P. To measure confidence of model M2 on a sample we propose a metric **Top-Two-Margin**. We pseudo label selected samples using model M2 and add them to labelled set. We then retrain models M1 and M2 using new labelled set.

Figure 3.1 shows side by side comparison of conventional active learning methods and proposed Triplet AL algorithm.

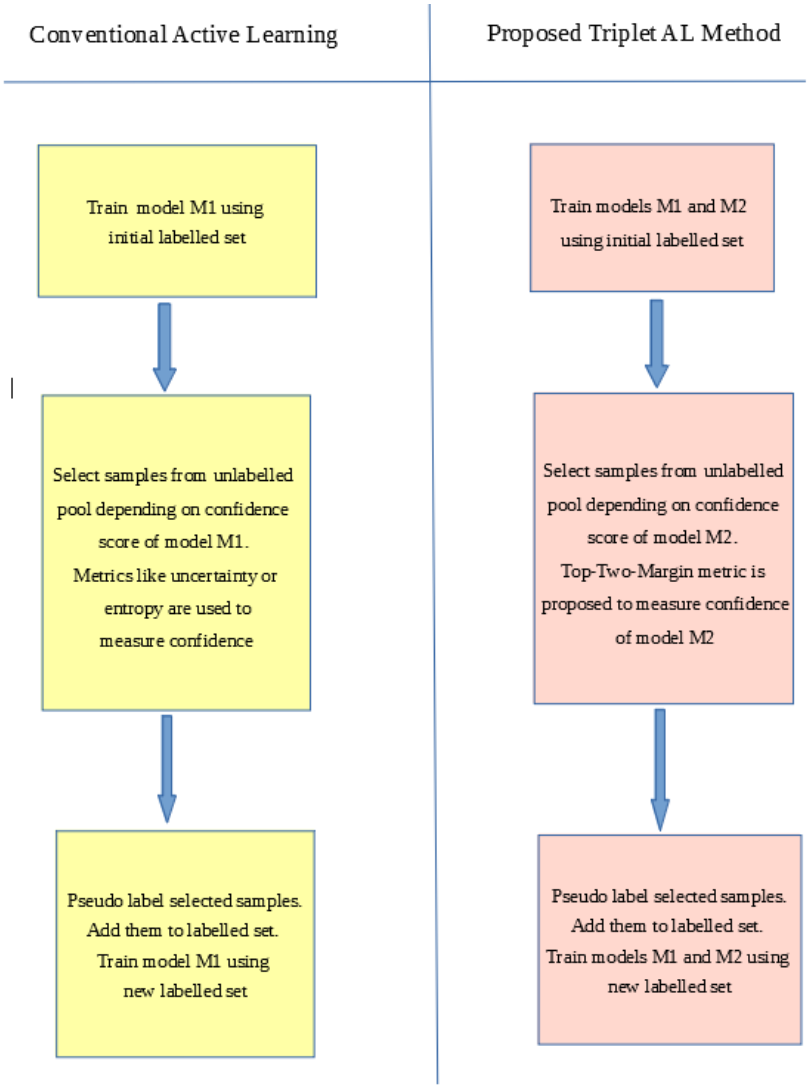


Figure 3.1: Comparison of Traditional Active Learning Algorithm and Proposed Triplet AL Algorithm

Overall algorithm (Refer Algorithm 4) to use unlabelled samples U for increasing performance of classification model M1 consists of the steps as shown in Figure 3.2.

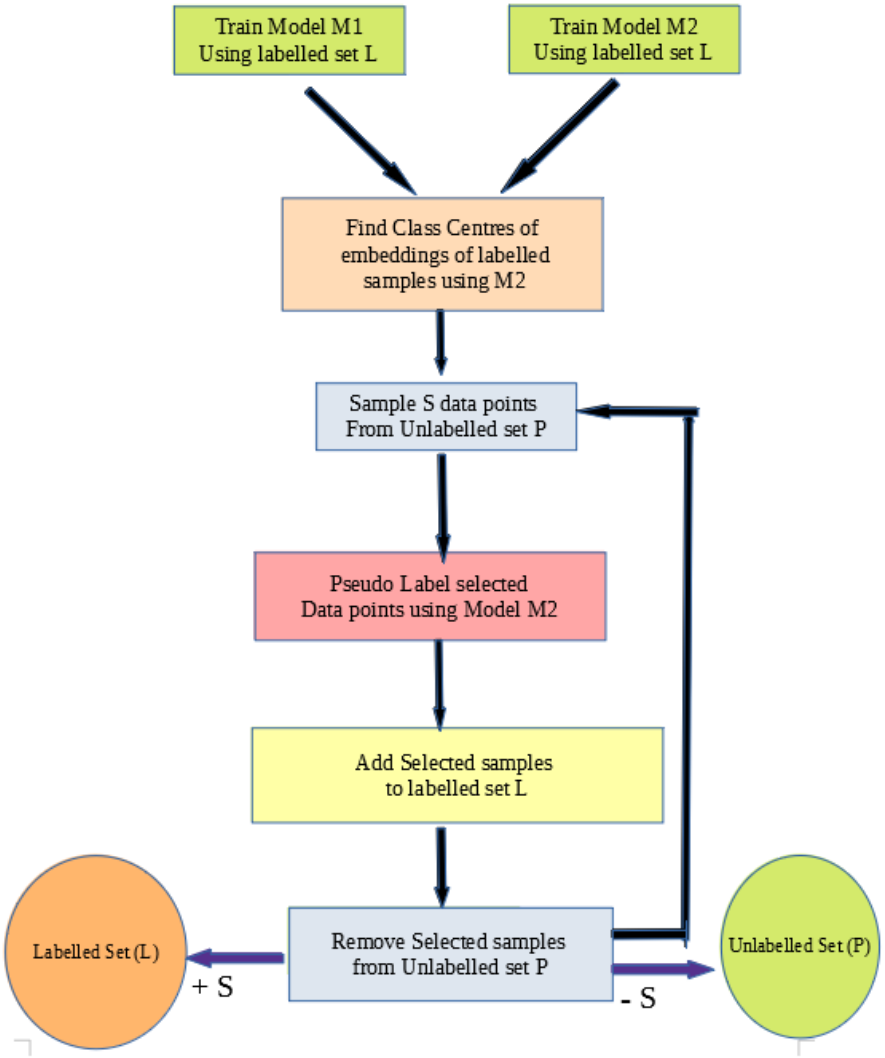


Figure 3.2: Flow Diagram of Overall Algorithm

3.1 Training Model M1 and M2 Using Initial Labelled Set

We use initial labelled set L to train models M1 and M2. To train model M1, we use Cross entropy loss L_{ce} while to train model M2, we use Triplet loss L_{tl}. Refer section 2.3 for details about loss functions.

Algorithm 1: Training M1

Input: (g_θ) Model M1, (L) labelled set
Parameters : (K) No of classes, (ep1) Epochs for training M1, (s1) Batch size for training model M1
Output: Trained model M1

for $i \leftarrow 1$ ***to*** $ep1$ **do**
  for *every batch X of size s1* **do**
    Forward pass through g_θ and find K class softmax output $y^{'} = g_\theta(\mathrm{X})$
    y = class labels of batch X
    Calculate Loss L_{ce}**:**
    $L_{ce} = \sum_{i=1}^{K} y_i \times \log y_i^{'}$
    Calculate Gradient:
    $\Delta W = \nabla_\theta 1/s1 \times \sum_{s1} L_{ce}$
    Update weights of model g_θ using ΔW
return g_θ

Algorithm 1 trains the classification model M1. After training model M1, for a given sample model M1 can output the K-class prediction probabilities.

Algorithm 2 trains embedding model M2. After training model M2, for a given sample model M2 can find out it's corresponding location in embedding space. Since we use Euclidean distance while using Triplet loss, we can find distances between two points in embedding space using Euclidean distance.

Algorithm 2: Training M2

Input: (f_ϕ) Model M2, (L) labelled set
Parameters : (K) No of classes, (ep2) Epochs for training M2, (s2) Batch size for training model M2
Output: Trained model M2

for $i \leftarrow 1$ ***to*** $ep1$ **do**
  Make Triplets:
  for *each Triplet* $z_a^C, z_b^C, z_c^{C'}$ **do**
    Calculate $f_\phi(z_a^C)$ **,** $f_\phi(z_b^C)$ **and** $f_\phi(z_c^{C'})$
    Calculate Loss L_{tl}**:**
    $L_{tl} = \max(\|f_\phi(z_a^C) - f_\phi(z_b^C)\|_2^2 - \|f_\phi(z_a^C) - f_\phi(\mathrm{z}_c^{C'})\ \|_2^2 + \alpha, 0)$
    Calculate Gradient
    $\Delta W = \nabla_\phi L_{tl}$
    Update weights of model f_ϕ using ΔW
return f_ϕ

3.2 Finding Centres of All Classes

After training model M2, we can find embedding of a training point $embedding(x) = f_\phi(x)$. To find center of a class in Algorithm 3, we find embeddings of all samples belonging to that class and take mean of the embeddings. For a classification problem involving K classes, we thus have k centres $c_1, c_2, c_3, ..., c_k$

Algorithm 3: Method to Find Class Centres

Input: (f_ϕ) Model M2 , (L) Initial labelled set
Parameters : (K) No of classes
Output: K Class centres

Initialize Centres to be an array of size K
for *each Class c* **do**
  $Sum_{Emb} = [0, 0, \ldots, 0]$
  for *each labelled example x in Class c* **do**
    embedding = $f_\phi(x)$
    $Sum_{Emb} = Sum_{Emb}$ + embedding
  $Centres[c] = Sum_{Emb}$/(No of samples in class c)
return $Centres[c]$

3.3 Defining the Oracle

In Active learning, Oracle is responsible for labelling of unlabelled samples before adding them to labelled set. There are two cases-

1. Case 1: Perfect Oracle present - In this case Oracle can perfectly label classes of unlabelled samples
2. Case 2: Perfect Oracle not present - In this case we don't have an Oracle to label the unlabelled data points

Since here we don't have perfect Oracle, we will use Model M2 to pseudo labelling the unlabelled samples before adding them to labelled set.

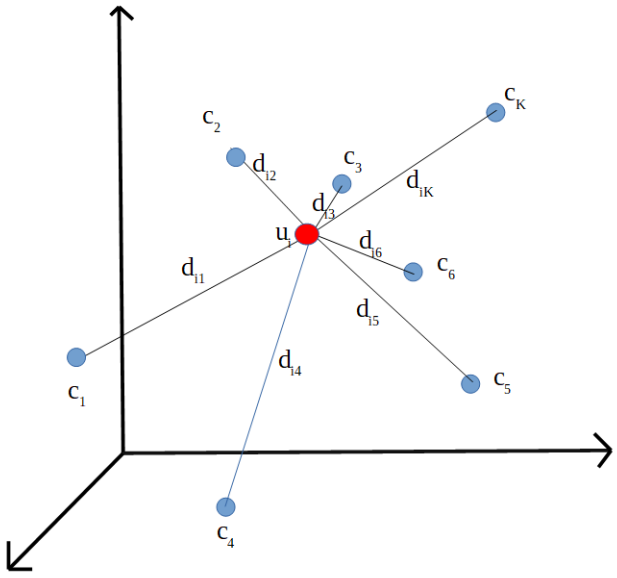


Figure 3.3: Distance of Unlabelled Data Point u_i to K Centers. (Note : Embedding Space of Size N is Shown as Embedding Space of Size 3 for Representation)

3.4 Selecting Samples Using Proposed Triplet AL Algorithm

Our goal is to sample S samples from unlabelled set P using Triplet AL Algorithm

- For each unlabelled sample p_i, convert it to its embedding e_i using model M2. Find its distance to all the K centers $[d_{i1}, d_{i2}, d_{i3}, ...d_{ik}]$ in the embedding space as shown in Figure 3.3. (Lines 8-10 of Algorithm 4)
- To find distances, we use Euclidean distance. Normalize distances using min-max normalization, so that distances are in range [0,1]. (Lines 11-12 of Algorithm 4)
- Sort distances in ascending order $[D_{i1}, D_{i2}, D_{i3}, ..., D_{iK}]$. Distance D_{i1} is the distance of unlabelled sample p_i from it's nearest class centre and distance D_{i2} is the distance of unlabelled sample p_i from it's second nearest class centre. (Line 13 of Algorithm 4)

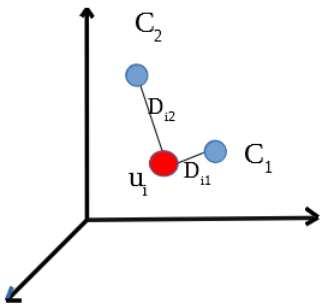


Figure 3.4: Top-Two-Margin of a point u_i is defined as difference between distance to it's nearest class centre C_1 and distance to it's second nearest class centre C_2 i.e. D_{i2} - D_{i1} (Note: Embedding Space of Size N is Shown as Embedding Space of Size 3 for Representation)

- We propose a term **Top-Two-Margin** as a difference between distance from nearest centre and distance from second nearest centre (refer Figure 3.4). Thus Top-Two-Margin (Line 14 of Algorithm 4) of an unlabelled sample p_i is calculated as-

$$Top - Two - Margin_i = D_{i2} - D_{i1}$$

Algorithm 4: Proposed Active Learning Algorithm - **Triplet AL**

Input: (g_θ) Model M1, (f_ϕ) Model M2 , (L) Initial labelled set, (P) Initial unlabelled set

Parameters : (I) Active Learning Iterations, (S) No of samples to select in each iteration, (α) Margin for triplet loss, (K) No of classes, (ep1) Epochs for training M1, (ep2) Epochs for training M2, (s1) Batch size for training model M1, (s2) Batch size for training model M2

Output: Trained model M1, M2 after I iterations of active learning

Initialize: Initialize models g_θ , f_ϕ
Training g_θ on Initial Labelled set using Algorithm 1
Training f_ϕ on Initial Labelled set using Algorithm 2
Centres = **Find Centres of all classes** using Algorithm 3
for $j \leftarrow 1$ ***to*** I **do**
    Initialize Margin Array M
    for *each unlabelled sample* $u_i \in P$ **do**
        $E_i = f_\phi(u_i)$
        for *each centre* $c \in$ *Centres* **do**
            $\mathrm{d}_{ic} = \|E_i - c\|_2^2$
        $\min_d$ =min(d_i) ; $\max_d$ =max(d_i)
        $d_i = d_i - min_d/(max_d - min_d)$
        Sort d_i to obtain $D_{i,1}, D_{i,2}, \ldots, D_{i,K}$
        $Top - Two - Margin_i = D_{i2} - D_{i1}$; $M[i] = Top - Two - Margin_i$
    indices = argsort(M)
    indices = indices[::-1]
    selected-indices = indices[:S]
    S = P[selected-indices]
    Pseudo Labelling
    for *selected sample* $s \in S$ **do**
        E_s =$\mathrm{f}_\phi(s)$
        for *each centre* $c \in$ *Centres* **do**
            $d_{sc} = \|E_s - c\|_2^2$
        $label = argsort(d_s)[0]$
        $Label[s] = label$
    Modify Labelled and Unlabelled set
    $L = L \cup \{S\}$
    $P = P \setminus \{S\}$
    Re-Train M1 and M2
    Training g_θ on New Labelled set using Algorithm 1
    Training f_ϕ on New Labelled set using Algorithm 2
return C

- Top-Two-Margin indicates certainty of Model M2 on class of unlabelled sample p_i. If Top-Two-Margin is large, then model M2 is sure that the sample belongs to the class of its nearest class centre. While if Top-Two-Margin is small, then model M2 is very less certain that it belongs to class of its nearest class centre.

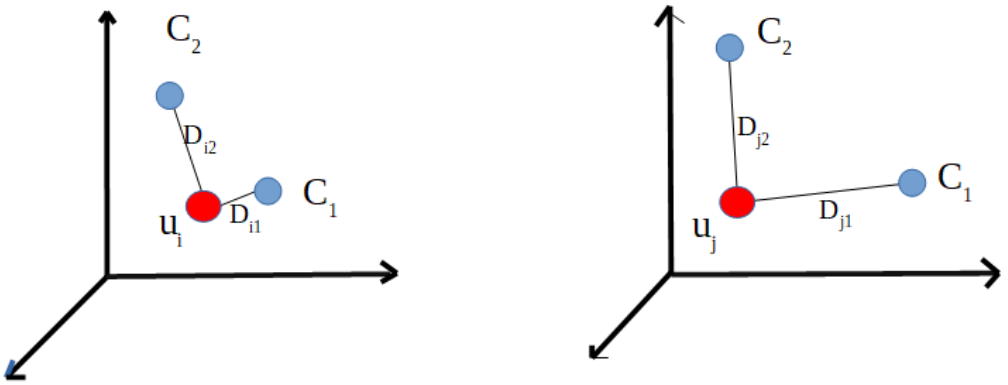


Figure 3.5: Two Points u_i and u_j Along With Their Closest Class Centres (Note : Embedding Space of Size N is Shown as Embedding Space of Size 3 for Representation)

- In Figure 3.5, Unlabelled sample u_i and u_j are shown along with their nearest two class centres C1 and C2 (C1 is nearest and C2 is second nearest). In left Figure there is large difference between distance of u_i from its nearest class centres C1 and C2 i.e. large Top-Two-Margin. So Model M2 has more confidence that the label or class of u_i is C1. On the other hand, right Figure there is very small difference between difference of u_j from from its nearest class centres C1 and C2 i.e. small Top-Two-Margin. So Model M2 has very less confidence that the label or class of u_i is C1.

- Sort unlabelled samples based on Top-Two-Margin in decreasing order. (Lines 15-16 of Algorithm 4)

- In each iteration, we select S samples whose Top-Two-Margin is largest i.e. model M2 is most confident about their class because we use M2 as Oracle. (Lines 17-18 of Algorithm 4)

3.5 Pseudo Labelling

We selected S $s_1, s_2, s_3, ...s_S$ data points using Active learning. Now our goal is to pseudo label the samples as we are not provided with the original labels for these samples)

- For pseudo labelling a sample s_i, we convert it to its embedding using M2. (Line 21 of Algorithm 4)
- Find distance of s_i to all class centres. (Lines 22-23 of Algorithm 4)
- We then assign a class C to the sample s_i whose centre is closest to the sample. (Lines 24-25 of Algorithm 4)

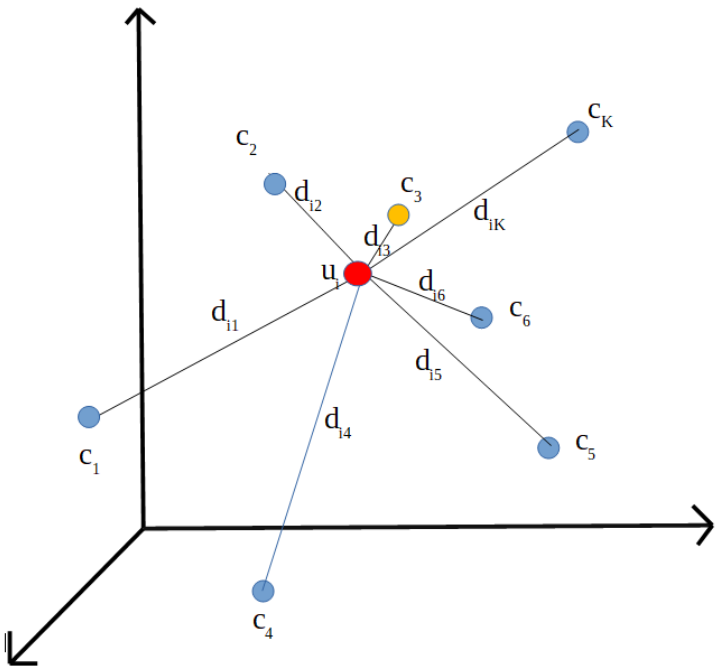


Figure 3.6: Pseudo Labelling of Sample u_i. C3 is Closest to u_i in Embedding Space, so u_i is Labelled as C3 (Note : Embedding Space of Size N is Shown as Embedding Space of Size 3 for Representation)

3.6 Re-Training

We selected S new samples from unlabelled set P and pseudo labelled them.

- Adding new points to labelled set L
  The pseudo labelled point are then added to the labelled set. So now the size of labelled set becomes $|L|$ + S. (Line 27 of Algorithm 4)
- Removing new points from unlabelled set U
  The points which are chosen and pseudo labelled have to be removed from unlabelled set, so that they are not chosen again. So now the size of unlabelled set becomes $|U|$ - S. (Line 28 of Algorithm 4)
- Re-Train Model M1 and M2
  Now we have new labelled set which includes new pseudo labelled points. We will now Re-Train both the models M1 and M2 using new labelled set. (Lines 30-31 of Algorithm 4)

Chapter 4

Experiments and Results

In this research, we present a new method for Active Learning. For testing the performance of proposed Active learning approach we experiment on Different datasets and different models.

We used two datasets STL-10 and CIFAR-10 to test proposed algorithm. We also test across different models for classification task. We used LeNet-5, ResNet150 and DenseNet-121 Architectures for classification task.

4.1 Dataset

4.1.1 STL-10 Data Set

STL-10 Dataset [2] consists of 10 classes - airplane, bird, car, cat, deer, dog, horse, monkey, ship and truck. The images are of size 96X96 with 3 colour channels. Along with labelled samples, STL-10 has high number of unlabelled samples. So it is suitable for the task in which we require to actively select samples from unlabelled pool. Some sample images from STL-10 dataset are shown in Figure 4.1. The dataset split is described in the Table 4.1

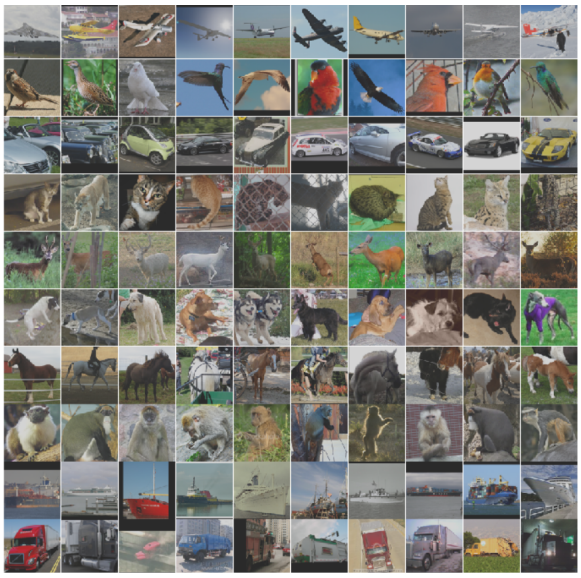


Figure 4.1: STL-10 Dataset

Split	Number of samples
Labelled	5000
Unlabelled	100000
Test	8000

Table 4.1: Dataset Split in STL-10

4.1.2 CIFAR-10 Data Set

CIFAR-10 Dataset [14] consists of 10 classes - airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck. The images are of size 32X32 with 3 colour channels. Some sample images from CIFAR-10 dataset are shown in Figure 4.2. The dataset split is described in the Table 4.2

Split	Number of samples
Train	50000
Test	10000

Table 4.2: Dataset Split in CIFAR-10

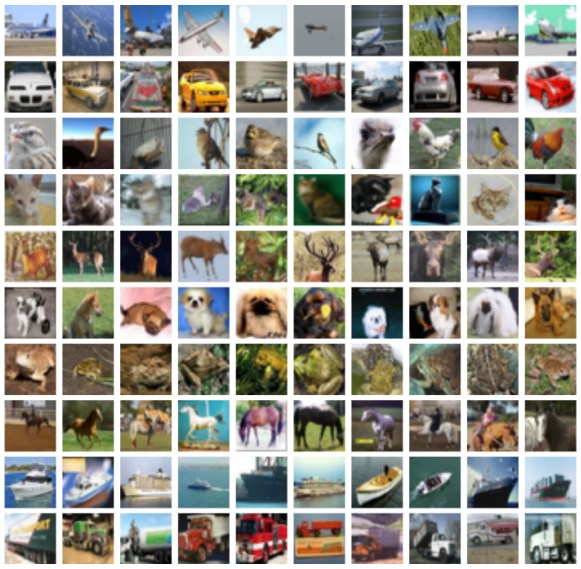


Figure 4.2: CIFAR-10 Dataset

4.2 Model Architecture

4.2.1 Architecture Used for Model M1

We used different models for classification task to see performance of proposed algorithm across different models.

1. LeNet-5
   LeNet-5 consists of two pairs of convolutional and average pooling layers followed by two fully connected layers and a softmax output layer [15].

2. ResNet-152
   ResNet-152 is a very deep neural network consisting of 152 layers. It consists of skip connections between residual blocks which help in mitigating vanishing gradient problem [8] .

3. DenseNet-121
   DenseNet-121 consists of 121 layers in which each layer gets input from all preceding layers and each layer gives its feature maps to subsequent layers. DenseNet architecture has many advantages like strengthening of feature propagation and feature res-use and mitigating vanishing gradient problem [12].

4.2.2 Architecture Used for Model M2

We used ResNet-50 [8] as Model M2 in all experiments as we want to test performance of model M1 with model M2 same in all cases. Model M2 is responsible for choosing samples and pseudo labelling samples in proposed Algorithm.

4.3 Experimental Protocol

4.3.1 Experimental Protocol for STL-10 Dataset

For STL-10 dataset, we are provided with a separate labelled set, an unlabelled set and a test set. We will use the same split as given to make Labelled set L and Unlabelled Set P. We will use 10% of the given labelled set as a Validation set for hyperparameter tuning and Early stopping. We test the performance of the models against unseen Test Set.

Set	**Number of samples**
Initial Labelled set for Training	4500
Validation Set	500
Unlabelled Set P	100000
Test Set	8000
Samples Added in each iteration	5000

Table 4.3: Set Split for STL-10

4.3.2 Experimental Protocol for CIFAR-10 Dataset

In CIFAR-10, we don't have a separate unlabelled set. So to test proposed algorithm, we will separate out samples from the labelled set and assume that their labels are unknown. We split given training set into labelled set of size 5000 and unlabelled set of size 45000. We use 500 labelled sample as validation set.

Set	Number of samples
Initial Labelled set for Training	4500
Validation Set	500
Unlabelled Set P	45000
Test Set	10000
Samples Added in each iteration	5000

Table 4.4: Set Split for CIFAR-10

4.4 Implementation Details

4.4.1 Re-Training Procedure

In proposed algorithm, we first use the initial labelled set to train models M1 and M2. We select S samples from Unlabelled set P and pseudo label them. Now for re-training there can be three alternatives-

1. Fine Tune Using New Samples Only
   Using the new samples, we fine tune the models M1 and M2 which were initially trained on labelled dataset. By fine-tuning we mean that we keep the old weights of models M1 and M2 which were learnt during training on initial labelled dataset and fine-tune weights of models by training on newly selected S samples.

2. Fine Tune Using New Labelled Set
   Use new labelled set formed from old current labelled set and newly selected samples to fine-tune using entire new data set.

3. Retraining From Scratch
   Initialize models M1 and M2 i.e. forget the weights which were learnt during training with initial labelled dataset. Train models M1 and M2 from scratch using newly formed labelled dataset comprising of initial labelled dataset and new added pseudo labelled .

- In first method where we fine-tune using new samples only, it is possibility that model learns more about new samples and forget about old samples

- In second method where we fine-tune using entire new labelled set, there is possibility that model learns more about old samples and less about new samples. This is because model has seen old samples twice, first when it was trained on old samples and second when it was trained on old combined with new samples.
- So the best way to train is to initialize model again and train on entire new labelled (i.e. old and new samples) set again.

4.4.2 Learning Rate Decay

During training, we reduce Learning rate by a factor of 10 if even after next 5 epochs validation loss is not decreasing.

4.4.3 Early Stopping

We use early stopping to stop model training to avoid overfitting of model. We monitor validation loss on validation dataset. During training, we save the model when we get a validation loss which is less than all the validation losses we get on previous epochs. We then wait for a number of epochs and see if we can achieve a lower validation loss than current best validation loss. If we get a lower validation loss, we save the model. If we don't get a lower validation loss even after waiting for some epochs, we stop training. After training is complete, we load the best model where we get least validation loss.

4.4.4 Tracking Best Model During Active Learning Iterations

We perform active learning iterations. In each active learning iteration, we select samples from unlabelled set, pseudo label them, add them to labelled set and retrain models. We check performance on test set and find out gain in accuracy. As we add new pseudo samples to labelled set. Model accuracy may decrease or increase. So During active learning iterations, we track best model which resulted in best accuracy compared to initial accuracy using initial labelled data.

4.4.5 GPU and Framework

We used following versions of GPU, python and frameworks to implement the algrotihm.

Tool	Version
GPU	NVIDIA 1080Ti
GPU RAM	12 GB
Python	3.6.10
PyTorch	1.1.0
Torchvision	0.3.0

Table 4.5: Tools Required

4.5 Experiment

Models M1 and M2 are trained using initial labelled dataset. After that we used active learning to select S new samples from unlabelled pool which are then pseudo labelled using model M2. Along with proposed method we used many other active learning methods to compare proposed algorithm.

- Random Selection
  Select S new samples randomly from unlabelled pool of samples
- Least Certainty Selection [16]
  Select S samples on which model M1 has low confidence. Low confidence means that score of highest score class/label is less.
- Most Certainty Selection
  It is reverse of least certainty where we sample S samples on which model M1 has high confidence. High confidence means that score of highest score class/label is high.
- Minimum Margin Based Selection [19]
  Select those samples whose difference between top two class scores given by

model M1 is less. Less difference means that model M1 is not sure about the class of M1.

- Maximum Margin Sased Selection
  This method is reverse of minimum margin based selection. We select those samples whose difference between top two class scores given by model M1 is high.

- Deep Bayesian Active Learning (BALD), Gal et al (2017) [6].
  This uses Bayesian CNNs and dropouts at time for sample acquisition.

- Deep Fool Algorithm proposed by Duffofe et al. (2018) [5].
  Most uncertain samples are those which have the shortest distance to boundary. Shortest distance to boundary is approximated as shortest distance to its adversarial example.

- Triplet AL (Proposed Method)
  We select those samples on which embedding model M2 is highly confident i.e. samples for which Top-Two-Margin is largest.

- Triplet AL Reverse (Proposed Method)
  It is reverse of Triplet AL Method where we select samples on which model M2 is least confident. In this method we select those samples on which model M2 has a middle level of confidence. The reason behind not to select high confidence sample is because the model M1 might be confident of the samples selected by Triplet AL method that it does not learn much even after adding new samples.

4.6 Results

We have tested proposed method along with other active learning method. In this section, we show results we obtained for all the architectures used for model M1. For each model architecture used for model M1, we show 2 plots-

- Accuracy Vs No of Samples Added
  (Left plot in Figures - 4.3, 4.4, 4.5, 4.6, 4.7 and 4.8)
  In this plot, we show how accuracy of model M1 changes as we add more and

more samples from unlabelled set and then retrain model M1. The graph starts from 0 on the x-axis which means that no new sample added i.e. training on initial labelled set.

These plots help to analyze the change in accuracy of classification model as new pseudo labelled samples are added. The accuracy of classification model may increase or decrease as new samples are added. If accuracy is increased this means that we have selected good samples and if accuracy is decreased this means we have select bad samples from unlabelled set.

- Chosen Accuracy Vs No of Samples added
  (Right plot in Figures - 4.3, 4.4, 4.5, 4.6, 4.7 and 4.8)
  In this plot, we show how much confident model M1 is on selected samples.

  This helps to get an insight of what type of new samples are being added to labelled set. If chosen accuracy is high, it implies that model M1 is already confident on selected samples. If chosen accuracy is low, it implies that model M1 is not confident about the selected samples.

4.6.1 Results on STL-10 Dataset

We have used entire 5000 samples samples given as per original STL-10 split as initial labelled set and used 100000 unlabelled samples as pool to select samples. We then tested accuracy of model M1 using 8000 test samples.

Hyper-Parameter Tuning

To train models M1 and M2, we performed hyper-parameter tuning (Table 4.6 and Table 4.7). We used Adam optimizer in all experiments.

Parameter	M1 - LeNet	M1 - ResNet152	M1 - DenseNet121
Learning Rate	0.0003	0.000003	0.0003
Adam $\beta 1$	0.9	0.9	0.9
Adam $\beta 1$	0.999	0.999	0.999
Adam Weight Decay	1e-5	1e-5	1e-5
Batch size	64	64	64

Table 4.6: STL-10 Model M1 Hyper-Parameter Tuning

Parameter	M2 Model
Learning Rate	0.000003
Adam $\beta 1$	0.9
Adam $\beta 1$	0.999
Adam Weight Decay	1e-5
Batch size	128
Triplet Margin (m)	0.2
Size of embedding space (N)	128

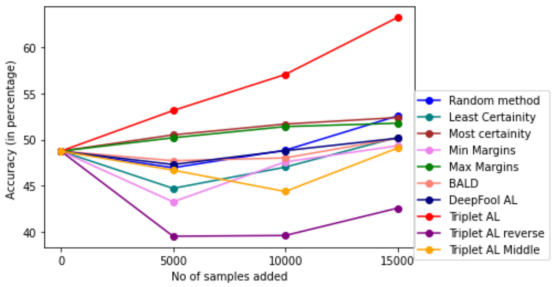
Table 4.7: STL-10 Model M2 Hyper-Parameter Tuning

Results

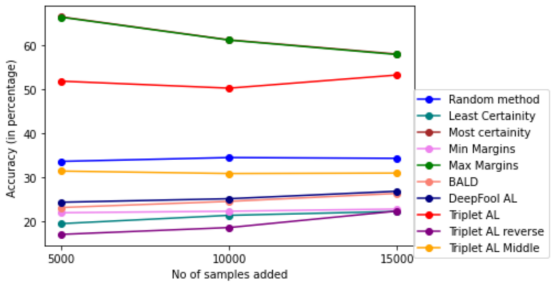
For STL-10 dataset, we achieved best results using proposed Triplet AL algorithm when compared to other Active learning methods used. We performed 3 iterations of active learning for each method. We queried 5000 new samples from STL-10 unlabelled dataset in each iteration. Accuracy vs number of samples added graphs in Figures 4.3 a), 4.4 a) and 4.5 a) show that we get better performance of classification model M1 when we queried new samples using proposed Triplet AL method.

- Model M1 - LeNet
- Model M1 - ResNet152
- Model M1 - DenseNet121

Table 4.8 shows maximum accuracy which can be achieved by adding 15000 samples from unlabelled pool to the labelled pool for STL-10 dataset. Initial accuracy

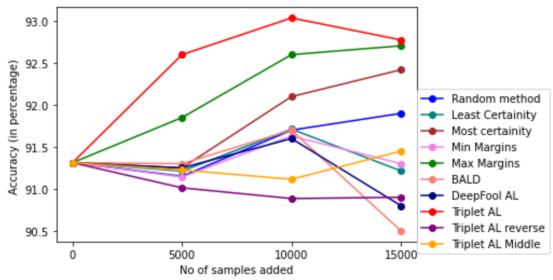


a) Accuracy of Model M1 vs No of Samples Added

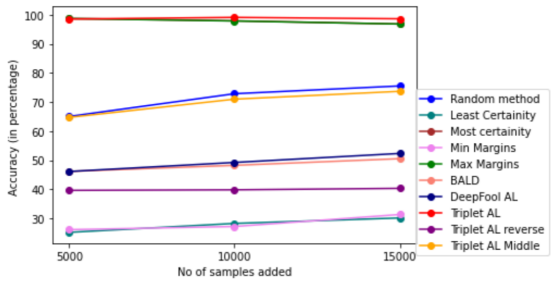


b) Accuracy of Model M1 on Chosen Samples

Figure 4.3: Result on STL-10 Dataset and Using LeNet Model as M1

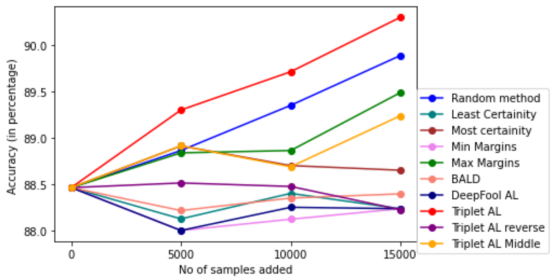


a) Accuracy of Model M1 vs No of Samples Added

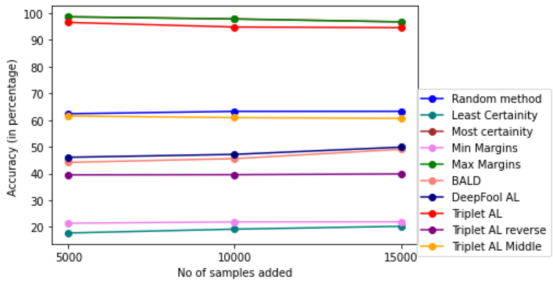


b) Accuracy of Model M1 on Chosen Samples

Figure 4.4: Result on STL-10 Dataset and Using ResNet152 Model as M1



a) Accuracy of Model M1 vs No of Samples Added



b) Accuracy of Model M1 on Chosen Samples

Figure 4.5: Result on STL-10 Dataset and Using DenseNet121 Model as M1

achieved using initial training data is also provided in the table for all architectures used.

Table 4.8 shows that using proposed Triplet AL algorithm, we have successfully increased performance of classification model. **We achieved increase of 14.48%, 1.72% and 1.84% in model M1 accuracy for models LeNet, ResNet152 and DenseNet121**. These gains in accuracy by proposed Triplet AL method exceeds gains achieved by other active learning methods used.

	M1 - LeNetl	**M1 - ResNet152**	**M1 - DenseNet121**
Initial Accuracy	48.75	91.31	88.46
Active Learning Method			
Random	52.56	91.9	89.88
Least Certainty	50.23	91.71	88.46
Most Certainty	52.38	92.42	88.91
Minimum Margin	49.325	91.63	88.46
Maximum Margin	51.775	92.70	89.48
BALD (2017)	50.15	91.70	88.46
Deep Fool AL(2018)	50.75	91.50	88.46
Triplet AL (Proposed)	**63.23**	**93.03**	**90.3**
Triplet AL Reverse (Proposed)	48.75	91.31	88.50
Triplet AL Middle (Proposed)	49.075	92.45	89.23

Table 4.8: Maximum Accuracy Obtained after Adding 15000 Samples from Unlabelled Set for STL-10 Dataset

4.6.2 Results on CIFAR-10 Dataset

Here we do not have any unlabelled data. So we partitioned given training data of size 50000 samples into labelled set of size 5000 and unlabelled set of size 45000. We then tested accuracy of model M1 using 10000 test samples.

Hyper-Parameter Tuning

To train models M1 and M2, we performed hyper-parameter tuning (Table 4.9 and Table 4.10). We used Adam optimizer in all experiments.

Parameter	**M1 - LeNet**	**M1 - ResNet152**	**M1 - DenseNet121**
Learning Rate	0.003	0.0003	0.0003
Adam $\beta 1$	0.9	0.9	0.9
Adam $\beta 1$	0.999	0.999	0.999
Adam Weight Decay	1e-5	1e-5	1e-5
Batch size	128	128	128

Table 4.9: CIFAR-10 Model M1 Hyper-Parameter Tuning

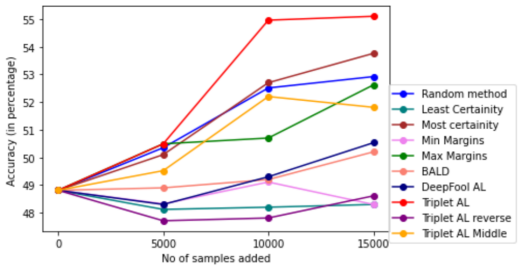
Parameter	**M2 Model**
Learning Rate	0.00003
Adam $\beta 1$	0.9
Adam $\beta 1$	0.999
Adam Weight Decay	1e-5
Batch size	128
Triplet Margin (m)	0.2
Size of embedding space (N)	128

Table 4.10: CIFAR-10 Model M2 Hyper-Parameter Tuning

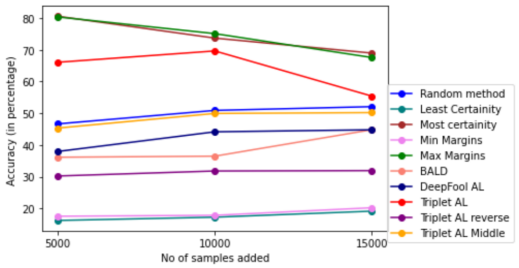
Results

For CIFAR-10 dataset, we achieved best results using proposed Triplet AL algorithm when compared to other Active learning methods used. We performed 3 iterations of active learning for each method. We queried 5000 new samples from CIFAR-10 unlabelled dataset in each iteration. Accuracy vs number of samples added graphs in Figures 4.6 a), 4.7 a) and 4.8 a) show that we get better performance of classification model M1 when we queried new samples using proposed Triplet AL method.

- Model M1 - LeNet



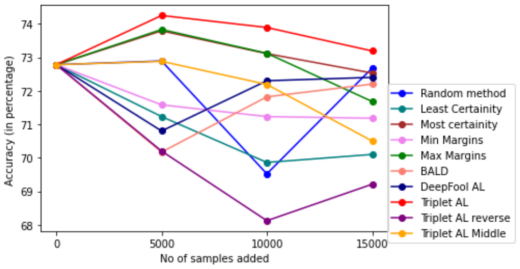
a) Accuracy of Model M1 vs No of Samples Added



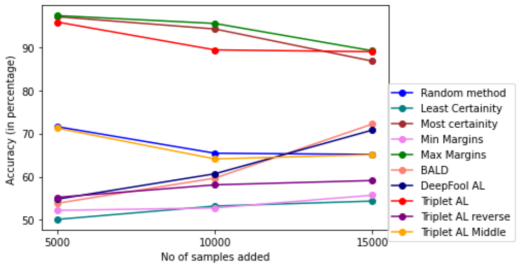
b) Accuracy of Model M1 on Chosen Samples

Figure 4.6: Result on CIFAR-10 Dataset and Using LeNet Model as M1

- Model M1 - ResNet152



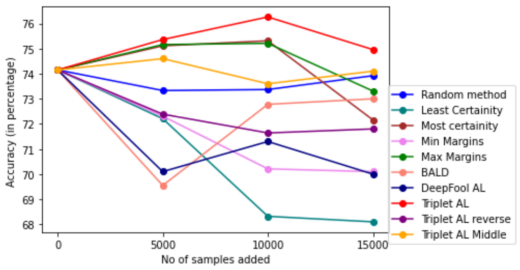
a) Accuracy of Model M1 vs No of Samples Added



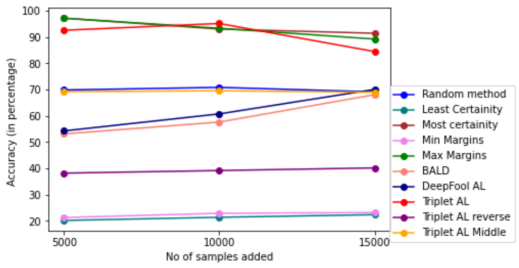
b) Accuracy of Model M1 on Chosen Samples

Figure 4.7: Result on CIFAR-10 Dataset and Using ResNet152 Model as M1

- Model M1 - DenseNet



a) Accuracy of Model M1 vs No of Samples Added



b) Accuracy of Model M1 on Chosen Samples

Figure 4.8: Result on CIFAR-10 Dataset and Using DenseNet Model as M1

Table 4.11 shows maximum accuracy which can be achieved by adding 15000 samples from unlabelled pool to the labelled pool for CIFAR-10 dataset. Initial accuracy achieved using initial training data is also provided in the table for all architectures used.

Table 4.11 shows that using proposed Triplet AL algorithm, we have successfully increased performance of classification model. **We achieved increase of 6.29%, 1.47% and 2.11% in model M1 accuracy for models LeNet, ResNet152 and DenseNet121**. These gains in accuracy by proposed Triplet AL method exceeds gains achieved by other active learning methods used.

	M1 - LeNet	**M1 - ResNet152**	**M1 - DenseNet**
Initial Accuracy	48.81	72.78	74.15
Active Learning Method			
Random	52.92	72.89	74.15
Least Certainty	48.81	72.78	74.15
Most Certainity	53.76	73.79	75.31
Minimum Margin	48.81	72.78	74.15
Maximum Margin	52.62	73.83	75.21
BALD (2017)	50.2	72.78	74.15
Deep Fool AL(2018)	50.53	72.78	74.15
Triplet AL (Proposed)	**55.1**	**74.25**	**76.26**
Triplet AL Reverse (Proposed)	48.81	72.78	74.15
Triplet AL Middle (Proposed)	52.20	73.19	74.60

Table 4.11: Maximum Accuracy Obtained after Adding 15000 Samples from Unlabelled Set for CIFAR-10 Dataset

4.7 Analyzing Performances of Active Learning Algorithms

In this research, we have used many Active learning algorithms. Chosen accuracy vs no of samples added graphs for STL-10 dataset in Figures 4.3 b), 4.4 b) and 4.5 b) and Chosen accuracy vs no of samples added graphs for CIFAR-10 dataset in Figures 4.6 b), 4.7 b) and 4.8 b) show that these algorithms are of 4 types-

1. Algorithms Choosing Most Confident Samples
    - Triplet AL (Proposed)
    - Most certainty
    - Maximum Margin
2. Algorithms Choosing Least Confident Samples
    - Least certainty
    - Minimum Margin
    - Triplet AL Reverse (Proposed)
    - DeepFool Al
    - BALD
3. Algorithms Choosing Samples of Mid Level Confidence Samples
    - Triplet Middle (Proposed)
4. Random Algorithm

To have a deeper look on what types of samples are by different algorithms, we show samples extracted by different active learning algorithms along with their pseudo labels. We show these results on Data set STL-10 and following protocol as illustrated in 4.8.1.

4.7.1 Analysis of Algorithms Choosing Least Confident Samples

Figure 4.9 shows some images selected by Least certainty algorithm in first iteration. Since the samples chosen are those on which model is least certain, these are difficult samples which model M1 does not know yet. So it might be possible that these are difficult samples for model M2 also which leads to wrong pseudo labelling of selected samples.

In Figure 4.9, first three images are of horse, monkey and cat but these are incorrectly labelled as dog, bird and dog respectively. Such new samples which are

Figure 4.9: Wrong Pseudo Labelling

wrongly pseudo labelled can lead to improper training of model M1. This is the reason we observe that algorithms which selects samples on which current model is least confident, leads to decrease in accuracy.

Active learning methods BALD and DeepFool AL also come under this category where samples on which current model has low confidence are selected. So these methods does not work well for the task which requires to increase accuracy using unlabelled samples.

4.7.2 Analysis of Algorithms Choosing High Confident Samples

Figure 4.10: Correct Pseudo Labelling

The samples selected by Algorithms like Triplet AL and Maximum certainty which selects samples on which current model is highly confident are easy samples. Most of these easy samples are pseudo labelled to their correct class by model M2.

Figure 4.10 shows some selected samples and their pseudo labels. All of them are

correctly pseudo labelled. Adding these new samples along with their pseudo labels to the labelled dataset helps to train classification model M1 better as we have now more number of samples along with their correct class labels. So the new samples added helps to increase accuracy of the classification model.

4.7.3 Analysis of Algorithms Choosing Mid Level Confident Samples

We also proposed Triplet middle method which selects samples with mid level of confidence. We implemented this algorithm to select samples on which we neither have too high confidence nor too low confidence. Motivation behind this method is to not give too easy samples for training as model may not learn much using easy samples. So we give samples with mid level of confidence to model M1 to train. Tough this method seems to help classification model M1 to train better, wrong pseudo labelling of many of mid level confidence samples leads to improper training of classification model M1

4.7.4 Analysis and Comparison of Triplet AL and Other High Confident Sampling Methods

Refer tables 4.8, 4.11 and 4.15, proposed Triplet AL method performs much better than other higher confidence sampling active learning methods.

The conventional high confidence selection methods only uses final k-class classification scores as a measure to decide confidence of model.

- According to Tao He et al. [9], Deep learning networks are made of feature learning and task learning phase. So using only task learning phase i.e. final k-class classification scores as uncertainty measure is not good practice.
- According to Duffofe et al. [5], uncertainty based on final k-class classification score is not good measure as these scores can change drastically if some small perturbations are added.

Our proposed method, Triplet AL measures confidence of a sample using it's position in the embedding space. We trained the embedding model M2 using Triplet loss.

We proposed a measure Top-Two-Margin, which is difference between distances of sample from it's closest two class centers in embedding space. If Top-Two-Margin is high, there is large difference in distances of sample from its nearest two class centers. In this case Model M2 is confident that class of sample is it's nearest class center. However if Top-Two-Margin is low, there is less difference in distances of sample from its nearest two class centers, then Model M2 is not confident about class of sample.

Thus proposed Triplet AL method presents a better method to measure confidence of a sample as confidence measure is based on distance measure and not just final K-class prediction scores.

4.8 Experiment Utilizing Entire Unlabelled Pool

In previous experiments, we added 15,000 samples from unlabelled set of size 100000 to labelled set in STL-10 experiment and added 15,000 samples from unlabelled set of size 45000 to labelled set in CIFAR-10 experiment. In this experiment we will consider entire unlabelled pool and see that how much increase in accuracy we can achieve by utilizing entire unlabelled pool.

4.8.1 Dataset and Protocol for Experiment Utilizing Entire Unlabelled Pool

We use STL-10 for this complete experiment but protocol is different than previous experiment on STL-10. Out of 5000 training samples of STL-10 provided as per original split, we assume that only 1000 samples are labelled and remaining 4000 samples are unlabelled. Out of 1000 labelled samples, we keep 100 samples as validation set used for hyper parameter tuning and early stopping. We use test set of size 8000 samples provided by original split to test the performance of model.

Set	Number of samples
Initial Labelled set for Training	900
Validation Set	100
Unlabelled Set P	4000
Test Set	8000
Number of samples added in each iteration	500

Table 4.12: Set Split for STL-10 for Experiment Utilizing Entire Unlabelled Pool

4.8.2 Model Architecture

We will use a different architecture not used in earlier experiments to test robustness of proposed algorithm. We will use Wide ResNet-50-2 [27] as model for M1. In Wide ResNet-50-2, 50 means depth of model and 2 means width factor.

4.8.3 Active Learning Methods

In this case we will compare performance of proposed Triplet AL method against recent Active learning methods and most confident methods only. We omit comparing with least confidence methods which were not good enough for this task.

- Random
- Most Certainty
- Maximum Margin
- BALD Active learning (2017) [6]
- Deep fool Active Learning (2018) [5]
- Triplet AL (Proposed)

4.8.4 Hyper-Parameter Tuning

To train models M1 and M2, we performed hyper-parameter tuning (Table 4.13 and Table 4.14). We used Adam optimizer in all experiments.

Parameter	M1 - Wide ResNet-50-2
Learning Rate	0.00003
Adam $\beta 1$	0.9
Adam $\beta 1$	0.999
Adam Weight Decay	1e-5
Batch size	64

Table 4.13: Utilizing Entire Unlabelled Set , STL-10 - Model M1 Hyper-Parameter Tuning

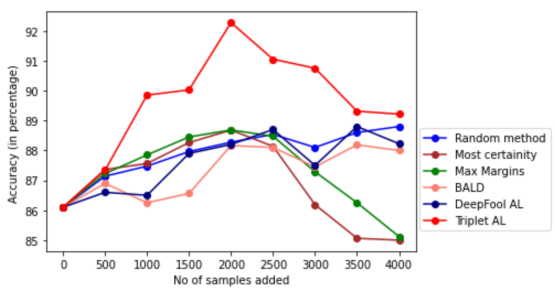
Parameter	M2 Model
Learning Rate	0.000003
Adam $\beta 1$	0.9
Adam $\beta 1$	0.999
Adam Weight Decay	1e-5
Batch size	128
Triplet Margin (m)	0.2
Size of embedding space (N)	128

Table 4.14: Utilizing Entire Unlabelled Set , STL-10 - Model M2 Hyper-Parameter Tuning

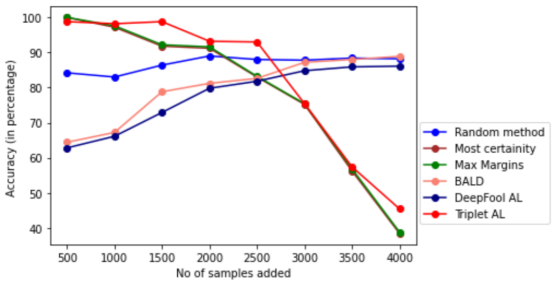
4.8.5 Results

We started with an initial labelled set of size 1000. We then queried 500 samples in each iteration, pseudo labelled them, added them to labelled set and retrained the models. With initial sample set of size 1000, model M1 gave an accuracy score of 86.1% on test set.

We obtained the results as shown in Figure 4.11. As shown in table 4.15, proposed Triplet AL method outperformed all other active learning methods used and gave better increase in performance as we added more samples from unlabelled set. Our proposed algorithm reached a maximum accuracy of 92.275% on test set during active learning iterations. **Our proposed Triplet AL method resulted in a gain of 6.175% in model accuracy which is much higher when compared to other approaches**.



a) Accuracy of Model M1 vs No of Samples Added



b) Accuracy of Model M1 on Chosen Samples

Figure 4.11: Result of Experiment Utilizing Entire Unlabelled Pool

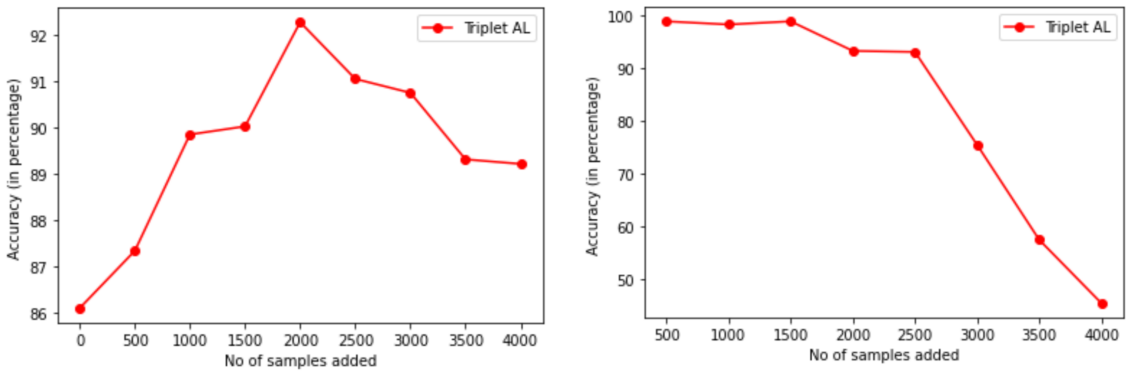
Method	Initial Accuracy	Maximum Accuracy	Gain in Accuracy
Random	86.1	88.8	2.7
Most Certainty	86.1	88.68	2.58
Maximum Margin	86.1	88.685	2.585
BALD (2017)	86.1	88.18	2.08
DFAL (2018)	86.1	88.8	2.7
Triplet AL (Proposed)	**86.1**	**92.275**	**6.175**

Table 4.15: Maximum Accuracy Obtained and Gain in Accuracy After Utilizing Entire Unlabelled Dataset

4.8.6 Analysis of Result Obtained Using Triplet AL method

We analyse the results obtained by proposed Triplet AL method where we performed experiment on complete unlabelled dataset. In proposed Triplet AL method we select those samples on which embedding model M2 is highly confident. We choose highly confident samples because we don't have their correct labels and we have to pseudo label them. If we would have selected least confident samples then their pseudo labelling would be incorrect and model M1 may not get any gain in model performance even after adding those samples.

In figure 4.12, we only show results for proposed Triplet AL algorithm. In each iteration we add 500 pseudo labelled samples to labelled dataset. Plot of chosen accuracy vs No. of samples added in Figure 4.12 b) shows that in initial iterations proposed Triplet AL algorithm selects highly confident samples or easy samples which leaves samples with low confidence in remaining unlabelled set.



a) Accuracy of Model M1 vs No of Samples Added

b) Accuracy of Model M1 on Chosen Samples

Figure 4.12: Result of Triplet AL Method in Experiment Utilizing Entire Unlabelled Pool

In early iterations since easy samples are added, the pseudo labelling process results in correct labelling of most of the samples which results in better training of classification model M1 as we have now more number of samples with their correct samples. Till 4^{th} (i.e. till 2000 samples), model M1 have an accuracy of above 90% pseudo labelled samples chosen. Adding these highly confident data samples results in increase of model accuracy as we have more number of data samples with correct labels.

In later iterations, since most of the samples with high confidence are chosen earlier, we are left with low confidence samples. These samples are hard samples and pseudo labelling them results in wrong labelling of many of the selected samples. Adding many samples which are wrongly labelled can decrease model performance. Thus we see a decrease of model M1 test accuracy after 5^{th} iteration.

We get best performance of model M1 at 4^{th} iteration with test accuracy of 92.275% which is 6.175% more than initial accuracy of 86.1%. In this experiment we assumed a set of given labelled samples as unlabelled. So we have their correct labels available with us. By using correct labels of all 5000 samples we get accuracy of Model M1 as 93.725% on test set. On the other hand we achieved an accuracy of 92.275% using just 1000 labelled samples. This shows that proposed Triplet AL algorithm which uses only 1000 labelled samples is powerful enough that it can reach close to accuracy when 5000 labelled samples are used.

Chapter 5

Conclusions And Future Work

In this research we proposed a new Active Learning approach **Triplet AL** which uses Triplet network. To check dataset and architectural independence of proposed algorithm, we have tested proposed algorithm using two datasets (STL-10 and CIFAR-10) and have used various model architectures for classification model like Lenet, ResNet152, DenseNet-121 and Wide ResNet. Using unlabelled set proposed Triplet AL algorithm is able to increase the performance of classification model on test set. We compared proposed algorithm with some active learning methods like Most certainty, Maximum margin, Random sampling, DeepFool Al [5], BALD [6] etc. Our algorithm outperformed other methods we used for comparison. We have also done a detailed analysis of why proposed algorithm performs better.

In future, we plan to use a Multi Task Learning model instead of separate classification and triplet models. We also plan to test the proposed algorithm on real world applications like face recognition in video surveillance system.

Bibliography

[1] Baum, E. B., and Lang, K. Query Learning Can Work Poorly When a Human Oracle is Used. In *International Joint Conference on Neural Networks* (1992), vol. 8, p. 8.

[2] Coates, A., Ng, A., and Lee, H. An Analysis of Single-Layer Networks in Unsupervised Feature Learning. In *Proceedings of The Fourteenth International Conference on Artificial Intelligence and Statistics* (2011), pp. 215–223.

[3] Cohn, D., Atlas, L., and Ladner, R. Improving Generalization with Active Learning. *Machine Learning 15*, 2 (1994), 201–221.

[4] Ducoffe, M., and Precioso, F. QBDC: Query by Dropout Committee for Training Deep Supervised Architecture. *arXiv preprint arXiv:1511.06412* (2015).

[5] Ducoffe, M., and Precioso, F. Adversarial Active Learning for Deep Networks: A Margin Based Approach. *arXiv preprint arXiv:1802.09841* (2018).

[6] Gal, Y., Islam, R., and Ghahramani, Z. Deep Bayesian Active Learning with Image Data. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70* (2017), JMLR. org, pp. 1183–1192.

[7] Gissin, D., and Shalev-Shwartz, S. Discriminative Active Learning. *arXiv preprint arXiv:1907.06347* (2019).

[8] He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 770–778.

[9] He, T., Jin, X., Ding, G., Yi, L., and Yan, C. Towards Better Uncertainty Sampling: Active Learning with Multiple Views for Deep Convolutional Neural Network. In *IEEE International Conference on Multimedia and Expo (ICME)* (2019), IEEE, pp. 1360–1365.

[10] Hermans, A., Beyer, L., and Leibe, B. In Defense of the Triplet Loss for Person Re-Identification. *arXiv preprint arXiv:1703.07737* (2017).

[11] Hoffer, E., and Ailon, N. Deep Metric Learning Using Triplet Network. In *International Workshop on Similarity-Based Pattern Recognition* (2015), Springer, pp. 84–92.

[12] Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely Connected Convolutional Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 4700–4708.

[13] King, R. D., Whelan, K. E., Jones, F. M., Reiser, P. G., Bryant, C. H., Muggleton, S. H., Kell, D. B., and Oliver, S. G. Functional Genomic Hypothesis Generation and Experimentation by a Robot Scientist. *Nature 427*, 6971 (2004), 247–252.

[14] Krizhevsky, A., and Hinton, G. Learning multiple layers of features from tiny images. Tech Report, 2009.

[15] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE 86*, 11 (1998), 2278–2324.

[16] Lewis, D. D., and Catlett, J. Heterogeneous Uncertainty Sampling For Supervised Learning. In *Machine Learning Proceedings 1994*. Elsevier, 1994, pp. 148–156.

[17] Lewis, D. D., and Gale, W. A. A Sequential Algorithm for Training Text Classifiers. In *SIGIR'94* (1994), Springer, pp. 3–12.

[18] Mayer, C., and Timofte, R. Adversarial Sampling for Active Learning. In *The IEEE Winter Conference on Applications of Computer Vision* (2020), pp. 3071–3079.

[19] Scheffer, T., Decomain, C., and Wrobel, S. Active Hidden Markov Models for Information Extraction. In *International Symposium on Intelligent Data Analysis* (2001), Springer, pp. 309–318.

[20] Schroff, F., Kalenichenko, D., and Philbin, J. Facenet: A Unified Embedding for Face Recognition and Clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 815–823.

[21] Sener, O., and Savarese, S. Active Learning for Convolutional Neural Networks: A Core-Set Approach. *arXiv preprint arXiv:1708.00489* (2017).

[22] Settles, B. Active learning literature survey. Tech. rep., University of Wisconsin-Madison Department of Computer Sciences, 2009.

[23] Seung, H. S., Opper, M., and Sompolinsky, H. Query By Committee. In *Proceedings of The Fifth Annual Workshop on Computational Learning Theory* (1992), pp. 287–294.

[24] Singh, M., Chawla, M., Singh, R., and Vatsa, M. Disguised Faces in the Wild 2019. In *IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)* (2019).

[25] Suri, A., Vatsa, M., and Singh, R. A2-LINK: Recognizing Disguised Faces via Active Learning and Adversarial Noise based Inter-Domain Knowledge. *IEEE Transactions on Biometrics, Behavior, and Identity Science* (2020). doi: `https://doi.org/10.1109/tbiom.2020.2998912`.

[26] Wang, K., Zhang, D., Li, Y., Zhang, R., and Lin, L. Cost-Effective Active Learning for Deep Image Classification. *IEEE Transactions on Circuits and Systems for Video Technology 27*, 12 (2016), 2591–2600.

[27] Zagoruyko, S., and Komodakis, N. Wide Residual Networks. *arXiv preprint arXiv:1605.07146* (2016).