# EECS 498/598-012 Deep Learning
# Project Progress Report

Hansheng Zou  Kevin He  Mike Wang  Yuanhang Luo  Yumou Wei

{hanshenz, kevhe, boyuan, royluo, yumouwei}@umich.edu

## 1 Problem Statement

Metric Learning is concerned with constructing a high-dimensional manifold where the distance measured by some metrics between two points directly reflects their similarity. For example, on a well constructed such manifold, an image of a cat, represented by a point (vector), should have a smaller distance to another image of a cat than to an image of a dog. Mathematically, an effective Metric Learning algorithm seeks for a parameterised function $f(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\theta})$ such that

$$D(\boldsymbol{x}, \boldsymbol{x}^+) = f(\boldsymbol{x}, \boldsymbol{x}^+; \boldsymbol{\theta}) < D(\boldsymbol{x}, \boldsymbol{x}^-) = f(\boldsymbol{x}, \boldsymbol{x}^-; \boldsymbol{\theta}) \tag{1}$$

where $\boldsymbol{x}, \boldsymbol{x}^+$ are samples of the same class and $\boldsymbol{x}, \boldsymbol{x}^-$ are samples from different classes.

*Deep* Metric Learning, like many other phrases starting with "deep", refers to the use of deep neural networks to represent the parameterised function $f(\boldsymbol{x}_i, \boldsymbol{x}_j; \boldsymbol{\theta})$. A very popular choice of loss function to guide the neural network to learn useful representations is the Triplet Loss [Hoffer and Ailon, 2015]:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{x}^+, \boldsymbol{x}^-; \boldsymbol{\theta}) = \max\{0, D(\boldsymbol{x}, \boldsymbol{x}^+) - D(\boldsymbol{x}, \boldsymbol{x}^-) + m\} \tag{2}$$

where $m$ represents a margin by which the negative example $\boldsymbol{x}^-$ should be farther away from the anchor $\boldsymbol{x}$ than the positive example $\boldsymbol{x}^+$.

One of the important issues that needs to be addressed is the selection of the negative example $\boldsymbol{x}^-$, given that the current training of neural networks highly rely on gradient-based optimisations. If the negative example is too "easy", that is, when the margin requirement is already almost satisfied, the loss shown in Equation 2 would be too small to guide the neural work to update its parameters. On the other hand, if the network is presented with too "hard" a negative example, it may be trapped at a bad local minimum and hence learning stagnates, especially at the very beginning of training. The problem of selecting negative examples, also known as Hard Negatives Mining, is very closely related to the idea of curriculum learning [Bengio et al., 2009]. Ideally, the neural network should be presented with negative examples of increasing difficulty as the training progresses. In this paper, borrowing the ideas from Adversarial Training and generative models, we propose a novel method that *generates* appropriately difficult negative examples based on the network's current performance.

## 2 Significance

Common neural network architectures for Deep Metric Learning, such as the Siamese Network with Contrastive Loss [Chopra et al., 2005] or Triplet Loss [Hoffer and Ailon, 2015], are known to be very hard to train. We expect that selecting negative examples appropriately not only allows the neural network to converge to a (potentially good) local minimum faster, but also improves its generalisation capabilities, as claimed in [Bengio et al., 2009]. Built upon previous work, our proposed method generates feature embeddings for both positive and negative examples in an adversarial manner. When the network is performing poorly, especially at the beginning, it receives easier negative examples to guide it through; however, when the performance is quite good, it is challenged by hard negative examples so that it can learn more robust features. The "performance" is quantified by the Triplet Loss, which is used to guide the adversarial generator. In this way the curriculum is adaptively adjusted as the learning progresses.

A good metric has enormous practical applications. First, any classification problems can be solved effectively using a good metric coupled with a distance-based classifier, such as the K-Nearest Neighbour classifier. In fact, a good metric can handle more than just the ordinary classification problems: it can deal with the case where there is a large number of classes, some of which are even *unknown during training*, but each class has only a few examples. This is often the case for speaker identification or face verification, where the potential users could be in millions but training data only contains a few thousand persons, each with a few examples. Another closely related application would be one-shot learning, where a good metric is often proved to be effective [Koch, 2015]. A good metric can also be used for ranking, with applications such as image retrieval and recommendation systems.

## 3 Related Work

**Metric Learning:** In recent years, many metric learning methods have been proposed, which aim to learn a good distance metric to reduce the distance between positive pairs and enlarge the distance between negative pairs as much as possible. The simplest work in metric learning learns a Mahalanobis distance to measure the similarities among images, which has been successfully applied to many real-world problems [Globerson and Roweis, 2006] [Weinberger and Saul, 2009] [Guillaumin et al., 2009]. Some methods only learn a linear transformation to map samples into new feature space [Davis et al., 2007] [Schultz and Joachims, 2004] [Globerson and Roweis, 2006] [Shalev-Shwartz et al., 2004]. For example, [Globerson and Roweis, 2006] presented an approach called Maximally Collapsing Metric Learning algorithm, which relies on the simple geometric intuition that points in the same classes should be near to one another while points from different classes should be far away from one another after mapping. This method aims to learn a quadratic Gaussian metric for classification. [Schultz and Joachims, 2004] proposed an online algorithm to learn a pseudo-metric, which is a positive semi-definite matrix, to predict the similarity of instances.

Since linear metric learning may not be good enough to deal with complex nonlinear relations, the kernel trick was proposed to address this limitation in many work [Chen et al., 2016] [Weinberger and Tesauro, 2007] [Lu et al., 2013]. For example, [Feng et al., 2013] presented a robust kernel metric learning (RKML) algorithm that leverages on regression techniques to improve the efficiency for image annotation. However, one limitation always remains: it is almost impossible to obtain a explicit kernel expression.

**Deep Metric Learning:** Using CNNs with contrastive [Oh Song et al., 2016] or triplet [Weinberger and Saul, 2009] embeddings, Deep Metric Learning has achieved a better performance in computer vision tasks, as compared to other traditional Metric Learning algorithms. It has been successfully applied in computer vision, semantic hashing, and style matching. For example, [Cui et al., 2016] proposed a triplet-based Deep Metric Learning approach in order to handle all three challenges in Fine-grained Visual Categorization (FGVC). [Liao et al., 2015] presented an efficient feature representation called Local Maximal Occurrence (LOMO), and a subspace and metric learning method called Cross-view Quadratic Discriminant Analysis (XQDA), which was shown to be effective and efficient through person re-identification experiments.

**Hard Negative Mining:** In many cases, hard negative mining is becoming a common technique for deep metric networks training. Hard negative mining can be seen as creating hard negative examples out of the training data and retraining the classifier to make it more robust against negative instances. For example, [Yuan et al., 2017] proposed the Hard-Aware Deeply Cascaded Embedding to solve the under-fitting and over-fitting problem. [Yu et al., 2018] shown the Loss Rank Mining (LRM) could be applied to all state-of-the-art real-time detectors. The Online Hard Example Mining (OHEM) algorithm presented by [Shrivastava et al., 2016] could also be applied to train state-of-the-art detection models based on deep CNNs.

Different from many existing Hard Negative Mining approaches, building upon the work by [Duan et al., 2018], we aim to utilize both positive and negative examples in the training set to generate synthetic hard negative examples to guide the learning of the neural network.

# 4 Proposed method/approach

In this section, we first present what the baseline model in our project is, and then the improvements we add upon the baseline model. Lastly, we will discuss the novelty of our model compared with the baseline model.

## 4.1 Original Deep Adversarial Metric Learning

In the ordinary deep metric learning approaches, there are lots of negative samples that are not hard enough. These negative samples are far from the anchor points, and thus they are wasted, producing very small gradients. Moreover, most hard negative samples are not very diversified.

To solve these drawbacks in the negative samples, Deep Adversarial Metric Learning (DAML) proposed an additional generator network generating synthetic negative examples that are supposed to be harder than the normal negative examples. This generator is fed by a triplet input of $\{\boldsymbol{x}_i, \boldsymbol{x}_i^+, \boldsymbol{x}_i^-\}$ (anchor, positive, and negative point, respectively), and then a harder negative point $\boldsymbol{g}_i^-$ is generated by minimizing the generator loss $J_{gen}$, which aims to make the synthetic negative point $\boldsymbol{g}_i^-$ closer to the anchor.

The overall network architecture of the original DAML model is as follows. Anchor, positive, and negative points are fed into convolutional neural networks to get features extracted. These CNNs are GoogLeNets adjusted for this DAML model, and they share the same parameters and architectures. After that, the extracted features for the triplet input will be fed into a generator to generate synthetic negative samples, supposedly hard negative samples. Then, the synthetic negative samples, alongside with the extracted features of positive and anchor samples, will be fed

into fully connected layers, after which we calculate the final similarity score. The fully connected layers share the same parameters and architectures. There are various embeddings used in the loss objective function: contrastive, triplet, lifted, and N-pair.

To train this model, firstly the model is trained without the hard negative generator. After that, the hard negative generator is trained simultaneously with the deep metric learning model to jointly optimize both networks. In the testing phase, we test the samples and measure their similarity without the generator network.

### 4.2 Deep Adversarial Metric Learning with Synthetic Positive

Building upon the idea of hard negative generator, we proposed a new synthetic positive sample generator. This synthetic positive generator receives the extracted features of triplet input $\{\boldsymbol{x}_i, \boldsymbol{x}_i^+, \boldsymbol{x}_i^-\}$, and tries to generate a synthetic positive sample that helps in generating a synthetic negative sample in the hard negative generator.

To achieve this, the positive generator aims to generate a positive sample $\boldsymbol{g}_i^+$ as close to the anchor $\boldsymbol{x}_i$ and the original positive point $\boldsymbol{x}_i^+$ as possible. This positive generator aims to minimize the loss function $J_{gen}^+$, defined as:

$$\begin{aligned} J_{gen}^+ &= J_{hard}^+ + \lambda_1 J_{reg}^+ \\ &= \sum_{i=1}^{N} (\|\boldsymbol{g}_i^+ - \boldsymbol{x}_i\|_2^2 + \lambda_1 \|\boldsymbol{g}_i^+ - \boldsymbol{x}_i^+\|_2^2) \end{aligned} \tag{3}$$

N is the number of input samples, and $\lambda_1$ is a term introduced to control the weight. $J_{hard}^+$ helps to make generated positive points closer to anchor, and $J_{reg}^+$ helps to regularize this loss function and prevent the generated positive from deriving too far from the original positive point.

After this positive generator network, the synthetic positive sample is sent to a hard negative generator together with negative and anchor samples. The hard negative generator takes into $\{x_i, g_i^+, x_i^-\}$, and tries to generate a hard negative point $g_i^-$ by minimizing the loss function $J_{gen}^-$, defined as:

$$\begin{aligned} J_{gen}^- &= J_{hard}^- + \lambda_1 J_{reg}^- + \lambda_2 J_{adv} \\ &= \sum_{i=1}^{N} (\|\boldsymbol{g}_i^- - \boldsymbol{x}_i\|_2^2 + \lambda_1 \|\boldsymbol{g}_i^- - \boldsymbol{x}_i^-\|_2^2 + \lambda_2 max\{0, D(\boldsymbol{g}_i^-, \boldsymbol{x}_i)^2 - D(\boldsymbol{g}_i^+, \boldsymbol{x}_i)^2 + \alpha\}) \end{aligned} \tag{4}$$

$\alpha$ is a margin between $(\boldsymbol{g}_i^-, \boldsymbol{x}_i)$ and $(\boldsymbol{g}_i^+, \boldsymbol{x}_i)$. Similar to the positive generator, $\lambda_1$ and $\lambda_2$ are parameters introduced to control the weights. $J_{hard}^+$ helps to push generated negative points closer to anchor, $J_{reg}^+$ prevents the generated negative from being too far from the negative point, and $J_{adv}$ encourages the difference between the similarity of $(\boldsymbol{g}_i^-, \boldsymbol{x}_i)$ and the similarity of $(\boldsymbol{g}_i^+, \boldsymbol{x}_i)$ are smaller than the margin $\alpha$.

The triplet of {anchor point $\boldsymbol{x}_i$, positive point $\boldsymbol{x}_i^+$, and generated hard negative point $\boldsymbol{g}_i^-$} is then sent to fully connected layers and then a similarity is calculated with the various embeddings objective function: contrastive, triplet, lifted, and N-pair. The whole architecture is in Figure 1.

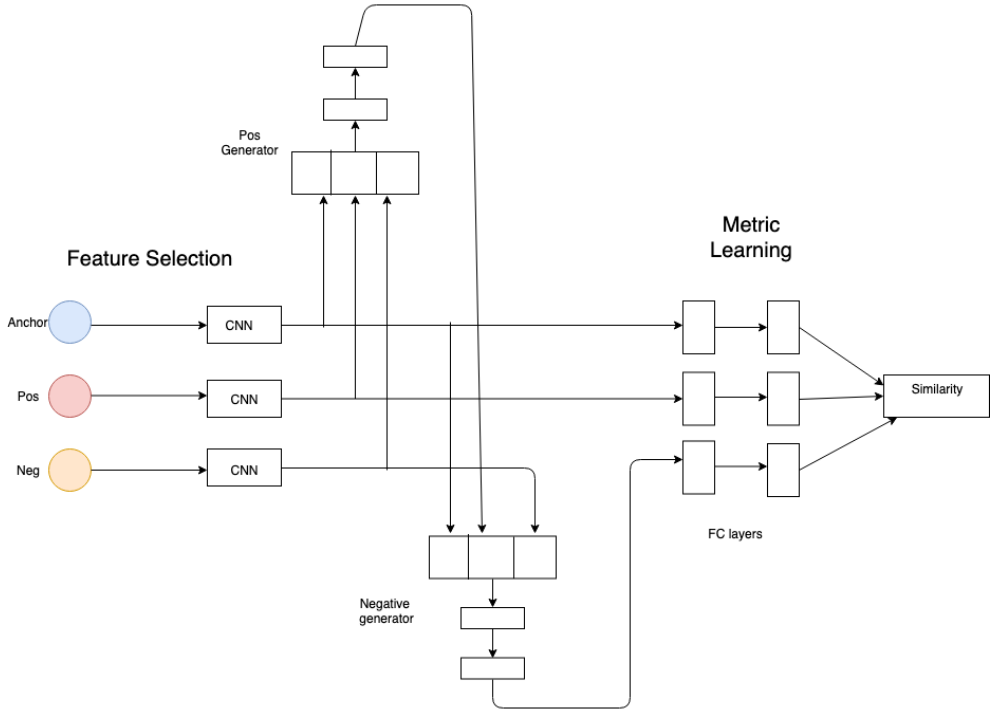


Figure 1: Deep Adversarial Metric Learning with Synthetic Positive

### 4.3 Novelty

The main idea of the original DAML algorithm aims to learn the metric better by generating hard negative samples. On top of the original idea of extracting features from the triplet input and feeding into a generator to generate synthetic negative samples, we want to add more complexity into the neural net to generate more appropriate negative samples. This is done by adding a synthetic positive generator in between. In this way, not only is the synthetic negative sample harder, but also more appropriate under the help of the synthetic positive generator. The metric this neural network learns given the triplets will be better.

## 5 Datasets and Evaluation

### 5.1 Datasets

We plan to conduct the experiments using the same datasets in the original paper of DAML [Y.Duan, 2018].Those datasets are widely used in the topics of deep metric learning [H.O.Song et al., 2017] [H.O.Song et al., 2016]

1. The CUB-200-2011 dataset. It includes 11,788 images of 200 bird species. In the DAML paper, they use the first 100 speices with 5,864 images for training and rest for testing [Y.Duan, 2018]. We will perform the same operation.

2. The Cars196 dataset. It includes 16,185 images of 196 car models. In the DAML paper, they used the first 98 models with 8,054 images for training and remaining for testing [Y.Duan, 2018]. We will perform the same operation.

3. The Stanford Online Products dataset [H.O.Song et al., 2016]. It includes 120,053 images of 22,634 products from eBay.com. In the DAML paper, they used the first 11,328 products with 59,551 images for training and remaining for testing [Y.Duan, 2018]. We will perform the same operation.

### 5.2 Evaluation Method

To evaluate the performance of our new model, we will use the same evaluation metrics used in [Y.Duan, 2018] and compare the performance.

1. Normalized mutual information(NMI). The input of NMI is a set of clusters $\Omega = \{\omega_1, ...\omega_K\}$ and ground truth class $C = \{c_1, ...c_K\}$. $\omega_i$ represents samples that belong to the $i$th cluster and $c_j$ represents a set of samples with label $j$. NMI is defined as the ratio of mutual information and mean entropy of clusters and the ground truth $NMI(\Omega, C) = \frac{2I(\omega;C)}{H(\Omega)+H(C)}$.

2. $F_1$, calculated as $F_1 = \frac{2PR}{P+R}$ (P is precision and R is recall).

3. The percentage of test samples which have at least one example from the same category in R nearest neighbors. [Y.Duan, 2018]

### 5.3 Baseline

In the original paper for DAML, they use constrastive embedding, triplet embedding, lifted structure [H.O.Song et al., 2016], N-Pair Loss, Angular loss and DDML [J.Hu et al., 2017] as the baseline. The baseline we will use in our project includes both the baseline methods and results of DAML paper.

| Method | NMI | $F_1$ | R@1 | R@2 | R@4 | R@8 |
|---|---|---|---|---|---|---|
| DDML | 41.7 | 10.9 | 32.7 | 43.9 | 56.5 | 68.8 |
| Triplet+N-pair | 54.3 | 19.6 | 46.3 | 59.9 | 71.4 | 81.3 |
| Angular | 62.4 | 31.8 | 71.3 | 80.7 | 87.0 | 91.8 |
| Contrastive | 42.3 | 10.5 | 27.6 | 38.3 | 51.0 | 63.9 |
| DAML (cont) | **42.6** | **11.4** | **37.2** | **49.6** | **61.8** | **73.3** |
| Triplet | 52.9 | 17.9 | 45.1 | 57.4 | 69.7 | 79.2 |
| DAML (tri) | **56.5** | **22.9** | **60.6** | **72.5** | **82.5** | **89.9** |
| Lifted | 57.8 | 25.1 | 59.9 | 70.4 | 79.6 | 87.0 |
| DAML (lifted) | **63.1** | **31.9** | **72.5** | **82.1** | **88.5** | **92.9** |
| N-pair | 62.7 | 31.8 | 68.9 | 78.9 | 85.8 | 90.9 |
| DAML (N-pair) | **66.0** | **36.4** | **75.1** | **83.8** | **89.7** | **93.5** |

Table 1: Baseline methods on dataset Car196

# 6 References


[Bengio et al., 2009] Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 41–48, New York, NY, USA. ACM.

[Chen et al., 2016] Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., and Abbeel, P. (2016). Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems*, pages 2172–2180.

[Chopra et al., 2005] Chopra, S., Hadsell, R., and LeCun, Y. (2005). Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1.

[Cui et al., 2016] Cui, Y., Zhou, F., Lin, Y., and Belongie, S. (2016). Fine-grained categorization and dataset bootstrapping using deep metric learning with humans in the loop. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1153–1162.

[Davis et al., 2007] Davis, J. V., Kulis, B., Jain, P., Sra, S., and Dhillon, I. S. (2007). Information-theoretic metric learning. In *Proceedings of the 24th international conference on Machine learning*, pages 209–216. ACM.

[Duan et al., 2018] Duan, Y., Zheng, W., Lin, X., Lu, J., and Zhou, J. (2018). Deep adversarial metric learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2780–2789.

[Feng et al., 2013] Feng, Z., Jin, R., and Jain, A. (2013). Large-scale image annotation by efficient and robust kernel metric learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1609–1616.

[Globerson and Roweis, 2006] Globerson, A. and Roweis, S. T. (2006). Metric learning by collapsing classes. In *Advances in neural information processing systems*, pages 451–458.

[Guillaumin et al., 2009] Guillaumin, M., Verbeek, J., and Schmid, C. (2009). Is that you? metric learning approaches for face identification. In *2009 IEEE 12th International Conference on Computer Vision*, pages 498–505. IEEE.

[Hoffer and Ailon, 2015] Hoffer, E. and Ailon, N. (2015). Deep metric learning using triplet network. *Lecture Notes in Computer Science*, page 8492.

[H.O.Song et al., 2017] H.O.Song, S.Jegelka, V.Rathod, and K.Murphy (2017). Deep metric learning via facility location. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5382–5390.

[H.O.Song et al., 2016] H.O.Song, Y.Xiang, S.Jegelka, and S.Savarse (2016). Deep metric learning via lifted structuredfeature embedding. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*, pages 4004–4012.

[J.Hu et al., 2017] J.Hu, J.Lu, and Y.P.Tan (2017). Discriminative deep metric learning for face and kinship verification. In *IEEE Transactions on Image Processing*, volume 26, pages 4269–4282.

[Koch, 2015] Koch, G. R. (2015). Siamese neural networks for one-shot image recognition.

[Liao et al., 2015] Liao, S., Hu, Y., Zhu, X., and Li, S. Z. (2015). Person re-identification by local maximal occurrence representation and metric learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2197–2206.

[Lu et al., 2013] Lu, J., Wang, G., and Moulin, P. (2013). Image set classification using holistic multiple order statistics features and localized multi-kernel metric learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 329–336.

[Oh Song et al., 2016] Oh Song, H., Xiang, Y., Jegelka, S., and Savarese, S. (2016). Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4004–4012.

[Schultz and Joachims, 2004] Schultz, M. and Joachims, T. (2004). Learning a distance metric from relative comparisons. In *Advances in neural information processing systems*, pages 41–48.

[Shalev-Shwartz et al., 2004] Shalev-Shwartz, S., Singer, Y., and Ng, A. Y. (2004). Online and batch learning of pseudo-metrics. In *Proceedings of the twenty-first international conference on Machine learning*, page 94. ACM.

[Shrivastava et al., 2016] Shrivastava, A., Gupta, A., and Girshick, R. (2016). Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 761–769.

[Weinberger and Saul, 2009] Weinberger, K. Q. and Saul, L. K. (2009). Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244.

[Weinberger and Tesauro, 2007] Weinberger, K. Q. and Tesauro, G. (2007). Metric learning for kernel regression. In *Artificial Intelligence and Statistics*, pages 612–619.

[Y.Duan, 2018] Y.Duan, W. (2018). Deep adversarial metric learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2780–2789.

[Yu et al., 2018] Yu, H., Zhang, Z., Qin, Z., Wu, H., Li, D., Zhao, J., and Lu, X. (2018). Loss rank mining: A general hard example mining method for real-time detectors. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

[Yuan et al., 2017] Yuan, Y., Yang, K., and Zhang, C. (2017). Hard-aware deeply cascaded embedding. In *Proceedings of the IEEE international conference on computer vision*, pages 814–823.