Semi-Supervised Metric Learning: A Deep Resurrection

Ujjal Kr Dutta,[1,2] Mehrtash Harandi,[3] C Chandra Sekhar[2]

[1] Data Sciences, Myntra, India
[2]Dept. of Computer Science and Eng., Indian Institute of Technology Madras, India
[3]Dept. of Electrical and Computer Systems Eng., Monash University, Australia
ukd@cse.iitm.ac.in, ujjal.dutta@myntra.com, mehrtash.harandi@monash.edu, chandra@cse.iitm.ac.in

Abstract

Distance Metric Learning (DML) seeks to learn a discriminative embedding where similar examples are closer, and dissimilar examples are apart. In this paper, we address the problem of Semi-Supervised DML (SSDML) that tries to learn a metric using a few labeled examples, and abundantly available unlabeled examples. SSDML is important because it is infeasible to manually annotate all the examples present in a large dataset. Surprisingly, with the exception of a few classical approaches that learn a linear Mahalanobis metric, SSDML has not been studied in the recent years, and lacks approaches in the deep SSDML scenario. In this paper, we address this challenging problem, and revamp SSDML with respect to deep learning. In particular, we propose a stochastic, graph-based approach that first propagates the affinities between the pairs of examples from labeled data, to that of the unlabeled pairs. The propagated affinities are used to mine triplet based constraints for metric learning. We impose orthogonality constraint on the metric parameters, as it leads to a better performance by avoiding a model collapse.

Introduction and Motivation

Distance Metric Learning (DML) seeks to learn a discriminative embedding where similar examples are closer, and dissimilar examples are apart. The importance of DML is evidenced by the plethora of recent approaches introduced at the premier conferences in Computer Vision (CV) (Yu and Tao 2019; Wang et al. 2019; Yuan et al. 2019; Sun et al. 2020; Cakir et al. 2019; Movshovitz-Attias et al. 2017; Deng et al. 2019; Qian et al. 2019), as well as Artificial Intelligence (AI) (Dutta, Harandi, and Sekhar 2020; Gu and Ko 2020; Chen et al. 2020; Li et al. 2020; Gong, Yuan, and Bao 2020) in general. In contrast to *classification losses* that learn a class logit vector, *embedding losses* in DML capture the relationships among examples. This makes embedding losses more general in nature, because of their flexibility to provide supervisory signals in the form of pairs (Chopra, Hadsell, and LeCun 2005), triplets (Schroff, Kalenichenko, and Philbin 2015; Weinberger, Blitzer, and Saul 2006), tuples (Sohn 2016) etc.

Despite the presence of many exemplary, state-of-the-art DML approaches as mentioned above, they are *supervised* in nature. In order to provide constraints (pairs/ triplets/ tuples) for metric learning, they require manual annotations (*class labels*). For large scale datasets, it is infeasible to obtain class labels for all the examples. In practical scenarios where it is possible to have a few examples annotated, with an abundance of additional unlabeled data, *semi-supervised learning* (Oliver et al. 2018; Zhu 2005; Chapelle, Scholkopf, and Zien 2009; Stretcu et al. 2019; Miyato et al. 2018; Iscen et al. 2019) can be applied. Surprisingly, with the exception of a few classical approaches that learn a linear Mahalanobis metric (Liu et al. 2010; Hoi, Liu, and Chang 2010; Niu et al. 2014; Ying et al. 2017; Dutta and Sekhar 2018; Shen, Du, and Li 2016), the problem of Semi-Supervised DML (SSDML) has not been studied in the recent years, not to mention the lack of SSDML approaches in the deep DML scenario. In this paper, we address this challenging problem, and revamp SSDML with respect to deep learning. To the best of our knowledge, deep SSDML has been studied for the first time (this is despite the presence of deep semi-supervised approaches for classification losses).

As with any DML technique, our approach covers the following **two major aspects of DML**: 1. **Constraint Mining:** To appropriately mine constraints of examples (eg., pairs or triplets), and 2. **DML Loss:** An appropriate loss formulation to learn a metric using the mined constraints. In the recent years, a huge number of *supervised, state-of-the-art* DML techniques have been proposed: Tuplet Margin (Yu and Tao 2019), Multi-Similarity (Wang et al. 2019), SNR (Yuan et al. 2019), Circle Loss (Sun et al. 2020), FastAP (Cakir et al. 2019), ArcFace (Deng et al. 2019), Soft Triple (Qian et al. 2019) etc. All of these approaches have made contributions either in terms of constraint mining, or a novel loss.

However, recently (Musgrave, Belongie, and Lim 2020) and (Roth et al. 2020) observed that when ran under the exact same experimental protocol (network architecture, embedding size, optimizers etc), classical metric learning losses involving *pairs* (Chopra, Hadsell, and LeCun 2005) or *triplets* (Schroff, Kalenichenko, and Philbin 2015; Weinberger, Blitzer, and Saul 2006) perform at par (and at times better) with that of the *state-of-the-art* loss functions. For this reason, we make use of a classical *triplet based* approach (because of the fact that the *pairwise* loss still has the theoretical downside of applying the same distance threshold to all the pairs, irrespective of variances in the similarities

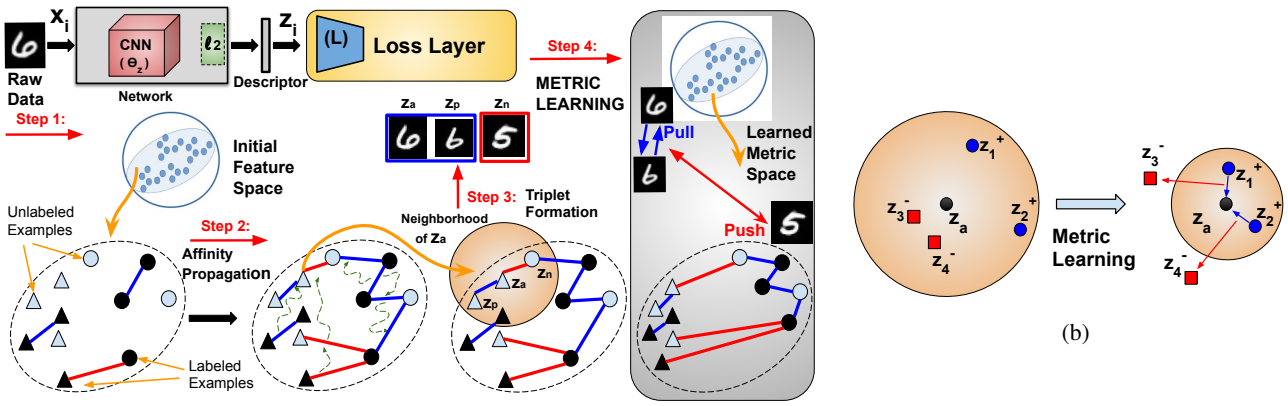


Figure 1: (Best viewed in color) (a) Illustration of our method. The raw image belongs to the MNIST dataset (LeCun et al. 1998), (b) Triplet mining around an anchor $\boldsymbol{z}_a$ (shown in black), within its k- neighborhood $\mathcal{N}_k(\boldsymbol{z}_a)$, k= 4. Points in blue ($\boldsymbol{z}_1^+, \boldsymbol{z}_2^+$) are more *semantically* similar (by virtue of *propagated affinities*) to the anchor, than the points in red ($\boldsymbol{z}_3^-, \boldsymbol{z}_4^-$). Hence, they should be pulled closer to the anchor in the learned space, compared to the red ones.

or dissimilarities). In particular, we make use of an angular variant of a triplet-based metric learning loss, due to certain well-established theoretical benefits (Wang et al. 2017) (eg., ease of tuning an angle as compared to a distance hyperparameter, capture of third-order information by the angular variant, scale and rotation invariance, and so on).

Following are the **major contributions** of the paper:

- Though end-to-end training is widely studied in supervised DML, the same is not the case for SSDML (a few classical SSDML approaches have attempted to learn a linear Mahalanobis metric, but not in an end-to-end fashion). **We revisit key linear SSDML baselines, and discuss their deep extensions**. Additionally, to address the theoretical limitations of prior SSDML methods, we also propose a **novel approach consisting of *well-studied components***: i) Affinity propagation (Liu et al. 2010; Dutta and Sekhar 2018) to propagate information from the limited labeled examples to the unlabeled data using a graph, and ii) Making use of an *adapted* angular variant of a triplet-based metric learning loss (Wang et al. 2017). It should be noted that we formulate a simple approach with well-known components, and the superiority and *novelty* of our approach lies by the ***design composition***, which we establish both empirically, as well as qualitatively.
- **Constraint mining** and scalability: It should be noted that having propagated the affinities, we propose a *novel triplet mining strategy* by considering the neighborhood of an example. Also, to make the approach scalable to large datasets, our approach is *stochastic* in nature (absent in earlier classical SSDML methods).
- **Addressing model collapse**: A *model collapse* could occur in DML which can project representations of close, but distinct examples to a single point (degenerate embeddings). To address this, we impose orthogonality on the metric and establish that it can lead to better embeddings. Arguably, some supervised works (Xie et al. 2018) have studied orthogonality while trying to alleviate class imbalance, achieving compact bases, etc. But our context is different. We explicitly show that it alleviates a collapse. This is particularly important in the semi-supervised case where we do not have much supervision, and a model may easily lead to degenerate embeddings.

An illustration of our method is shown in Figure 1a.

Extending Classical SSDML to Deep SSDML

We begin our paper with the introduction of a few representative SSDML methods, while discussing their main principles. Let $\mathcal{X} = \mathcal{X}_L \cup \mathcal{X}_U$ be a given dataset, consisting of a set of labeled examples $\mathcal{X}_L = \{\boldsymbol{z}_i \in \mathbb{R}^d\}_{i=1}^{N_L}$ with the associated label set $\{y_i\}_{i=1}^{N_L}, (y_i \in \{1, \cdots, C\}; C$ is the number of classes), and a set of unlabeled examples $\mathcal{X}_U = \{\boldsymbol{z}_j \in \mathbb{R}^d\}_{j=N_L+1}^{N}$. Existing SSDML approaches learn the parametric matrix $\boldsymbol{M} \in \mathbb{R}^{d\times d}, \boldsymbol{M} \succeq 0$ of the squared Mahalanobis distance metric $\delta^2_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j) = (\boldsymbol{z}_i - \boldsymbol{z}_j)^\top \boldsymbol{M}(\boldsymbol{z}_i - \boldsymbol{z}_j)$, for a pair of descriptors $\boldsymbol{z}_i, \boldsymbol{z}_j \in \mathbb{R}^d$ (classically obtained using hand-crafted features). The SSDML approaches can mainly be categorized under two major paradigms: i) entropy minimization (Grandvalet and Bengio 2005), and ii) graph-based. The SERAPH (Niu et al. 2014; Shen, Du, and Li 2016) approach is the only SSDML representative from the first class, and is expressed as (Niu et al. 2014):

$$\min_{\boldsymbol{M}} -[\sum_{\substack{(\boldsymbol{z}_i,\boldsymbol{z}_j):\boldsymbol{z}_i,\boldsymbol{z}_j\in\mathcal{X}_L \\ y_{ij}=-1, y_i\neq y_j \\ y_{ij}=1, y_i=y_j}} \log p_{ij}^{\boldsymbol{M}}(y_{ij}) + \mu \sum_{\substack{(\boldsymbol{z}_i,\boldsymbol{z}_j): \\ \boldsymbol{z}_i,\boldsymbol{z}_j\in\mathcal{X}_U}} \sum_{y\in\{-1,1\}} p_{ij}^{\boldsymbol{M}}(y) \log p_{ij}^{\boldsymbol{M}}(y)] + \lambda \mathrm{Tr}(\boldsymbol{M}). \quad (1)$$

The first term maximizes the entropy of a conditional probability distribution over the pairwise constraints obtained from labeled data, while the second term minimizes the entropy over the unlabeled data. Here, $p_{ij}^{\boldsymbol{M}}(y_{ij})$ (and correspondingly $p_{ij}^{\boldsymbol{M}}(y)$) denotes a probability parameterized

Algorithm 1 Stochastic extension of the SSDML baselines

1: Initialize $\theta_z^0, \boldsymbol{M}_0$.
2: **for** $t \leftarrow 1$ to T **do**
3: Fix θ_z^{t-1} and learn $\boldsymbol{M}_t$ using the given SSDML method.
4: Fix $\boldsymbol{M}_t$ and learn θ_z^t by backpropagation using SGD.
5: **end for**
6: **return** $\theta_z^T, \boldsymbol{M}_T$

by the distance metric $\delta^2_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j)$, as follows: $p_{ij}^{\boldsymbol{M}}(y_{ij}) = \frac{1}{1+\exp(y_{ij}(\delta^2_{\boldsymbol{M}}(\boldsymbol{z}_i,\boldsymbol{z}_j)-\eta))}$. The regularizer $\text{Tr}(\boldsymbol{M})$ ensures a *projection sparsity* (Niu et al. 2014). $\eta, \mu, \lambda > 0$ are hyperparameters in (1).

The other category of SSDML approaches, i.e., graph-based ones, has majority of approaches to its disposal. Here we discuss the most prominent ones. The LRML (Hoi, Liu, and Chang 2010) method is expressed as:

$$\min_{\boldsymbol{M}\succeq 0, \log|\boldsymbol{M}|\geq 0} \gamma_S \sum_{\substack{(\boldsymbol{z}_i,\boldsymbol{z}_j):\boldsymbol{z}_i,\boldsymbol{z}_j\in\mathcal{X}_L \\ y_i=y_j}} \delta^2_{\boldsymbol{M}}(\boldsymbol{z}_i,\boldsymbol{z}_j) - \gamma_D \sum_{\substack{(\boldsymbol{z}_i,\boldsymbol{z}_j):\boldsymbol{z}_i,\boldsymbol{z}_j\in\mathcal{X}_L \\ y_i\neq y_j}} \delta^2_{\boldsymbol{M}}(\boldsymbol{z}_i,\boldsymbol{z}_j) + \text{Tr}(\boldsymbol{M}\boldsymbol{X}\mathcal{L}\boldsymbol{X}^\top). \tag{2}$$

The first and second terms in (2) *minimize* and *maximize* the pairwise distances among similar and dissimilar pairs from labeled data, respectively. The third term is a *Laplacian regularizer* that preserves the pairwise distances among all the examples in $\mathcal{X}$ (and hence, unlabeled data as well). Here, columns of $\boldsymbol{X} \in \mathbb{R}^{d\times N}$ represent the descriptors of elements of $\mathcal{X}$, and $\mathcal{L}$ is the *graph Laplacian* matrix obtained using the affinity matrix of the underlying kNN graph used.

The recently proposed graph-based ISDML (Ying et al. 2017) approach is a state-of-the-art among SSDML methods. Similar to LRML, it seeks to optimize the pairwise distances between labeled examples. However, it also takes into account the density around an unlabeled example while computing the Laplacian regularizer. The APLLR and APIT methods have also been proposed under a common framework (Dutta and Sekhar 2018), that instead of performing naive optimization of the distances among labeled pairs, first computes a prior metric using the labeled data. The APLLR method computes the prior metric by optimizing a log-likelihood ratio, while the APIT method computes the prior metric using an information-theoretic approach. The final metric is learned using affinity propagation (Liu et al. 2010) on unlabeled data while staying close to the prior one.

Extending the above methods for end-to-end learning: Although trivial, the methods discussed above have not been studied in the context of end-to-end deep learning. It is noteworthy that as the losses in these methods are differentiable, we can backpropagate their gradients while using mini-batch based stochastic optimization. Due to the graph-based nature of LRML, ISDML, APLLR and APIT, one could first sample a random partition of the unlabeled examples, and construct a sub-graph along with the limited number of available labeled examples. Then, the mini-batches can be drawn from this partition alone for a number of epochs (over this partition). This process could be iterated over a number of partitions over the entire dataset. The same partition based strategy could be used for SERAPH as well, with the only exception that it does not require graph construction.

Formally, we can learn a deep embedding that induces a distance metric of the form $\delta^2_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j)$, such that the descriptors $\boldsymbol{z}_i, \boldsymbol{z}_j \in \mathbb{R}^d$ are obtained using a deep neural network $z : \mathcal{X} \rightarrow \mathbb{R}^d$ with parameters θ_z, while simultaneously learning $(\theta_z, \boldsymbol{M})$ in an end-to-end manner. This general stochastic approach is illustrated in Algorithm 1.

Limitations with existing SSDML methods: Having proposed stochastic extensions for the existing classical SSDML techniques LRML, SERAPH, ISDML, APLLR and APIT, we now discuss their limitations. A fundamental weakness of the LRML method is the use of a naive Laplacian regularizer as: $\text{Tr}(\boldsymbol{M}\boldsymbol{X}\mathcal{L}\boldsymbol{X}^\top) = \frac{1}{2}\sum_{i,j=1}^N \delta^2_{\boldsymbol{M}}(\boldsymbol{z}_i,\boldsymbol{z}_j)W_{ij}$. Here, W_{ij} refers to the affinity between the i^{th} and j^{th} examples, and is computed directly using the distances among the initial representations. Hence, the distance $\delta^2_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j)$ in the learned space could be affected by a poor initial W_{ij}. The ISDML method scales the affinity terms by considering the densities around each individual example. However, both these techniques are prone to be affected by a poor initial representation. Moreover, they fail to adapt their affinities by the additionally present labeled data information.

The APLLR and APIT techniques make use of an affinity propagation principle (Liu et al. 2010) to propagate the information from the labeled pairs to the unlabeled ones. This leads to a more informative affinity matrix. However, their dependency on a prior pre-computed metric sometimes have adverse effects. Especially, in scenarios where there is an extremely limited amount of labeled data, the prior parametric matrix is often close to singular, and leads to a poor metric. Furthermore, despite the affinity propagation, they do not *mine* informative constraints for metric learning (which is very crucial). This fails to fully leverage the benefits of affinity propagation. On the other hand, the SERAPH method is based on the *entropy minimization* principle. Although it is a good principle due to its capability of minimizing the class overlap, the SERAPH method takes a long time to converge, and often gets trapped in poor local minima.

To address these limitations with the existing SSDML methods, we propose a new method. However, we propose that instead of learning the metric wrt the matrix $\boldsymbol{M}$ ($O(d^2)$ parameters), we utilize the property that $\boldsymbol{M} \succeq 0$, and factorize it as: $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^\top$ s.t. $\boldsymbol{L} \in \mathbb{R}^{d\times l}, l \leq d$, and learn the metric wrt $\boldsymbol{L}$ with lesser parameters ($O(dl)$). The pseudocode of our method is same as Algorithm 1 (after replacing $\boldsymbol{M}$ with $\boldsymbol{L}$). We now discuss our proposed method.

Proposed Approach

Triplet Mining using Affinity Propagation

In the semi-supervised DML setting, the first task is to *mine* informative constraints, and then use an appropriate loss function. As the first stage, we propose a novel method for mining constraints using a graph-based technique while leveraging affinity propagation (Liu et al. 2010). Let $\mathcal{X}_U^{(p)}$ be

a randomly selected partition of unlabeled data. We construct a kNN graph using $\mathcal{X}_L \cup \mathcal{X}_U^{(p)}$, s.t. the nodes represent the examples, and edge weights denote the *affinities* (or similarities) among the examples. The initial affinity matrix $\boldsymbol{W}^0 \in \mathbb{R}^{(N_L+N_p)\times(N_L+N_p)}$ is defined as follows: i) $W^0_{ij} = +1$, if $i \neq j$ and $\exists y_i, y_j$, s.t. $y_i = y_j$, ii) $W^0_{ij} = -1$, if $i \neq j$ and $\exists y_i, y_j$, s.t. $y_i \neq y_j$, iii) $W^0_{ij} = +1$, if $i = j$, and iv) $W^0_{ij} = 0$, otherwise. Here, N_L and N_p are respective cardinalities of $\mathcal{X}_L$ and $\mathcal{X}_U^{(p)}$.

The neighborhood structure of the kNN graph can be indicated using a matrix $\boldsymbol{Q} \in \mathbb{R}^{(N_L+N_p)\times(N_L+N_p)}$ defined as: i) $Q_{ij} = 1/k$, if $\boldsymbol{z}_j \in \mathcal{N}_k(\boldsymbol{z}_i)$, and ii) $Q_{ij} = 0$, otherwise. Here, $\mathcal{N}_k(\boldsymbol{z}_i)$ is the set of k-nearest neighbor examples of $\boldsymbol{z}_i$. The flow of information from the non-zero entries of $\boldsymbol{W}^0$ (representing the labeled pairs) to the zero entries (representing the unlabeled pairs), can be performed by using a closed-form expression described by Liu *et al.* (Liu et al. 2010), as: $\boldsymbol{W}^* = (1-\gamma)(\mathbf{I}_{N_L+N_p} - \gamma\boldsymbol{Q})^{-1}\boldsymbol{W}^0$. This step is called *affinity propagation*. This step essentially performs a random-walk to propagate the affinities, where $0 < \gamma < 1$ is a weight hyper-parameter between the affinities obtained at the current step to that of the initial one. As $N_L + N_p \ll N$, we do not encounter any difficulty while scaling up to large datasets. To obtain a symmetric affinity matrix $\boldsymbol{W}$, we perform a final symmetrization step as follows: $W_{ij} = (W^*_{ij} + W^*_{ji})/2$.

The finally obtained representation of the symmetric affinity matrix $\boldsymbol{W}$ is used to mine triplets for metric learning. In doing so, we take into account the k-neighborhood $\mathcal{N}_k(\boldsymbol{z}_a)$ of an example $\boldsymbol{z}_a \in \mathcal{X}_L \cup \mathcal{X}_U^{(p)}$ that we consider as an anchor (Figure 1b). Let $\mathcal{N}_W(\boldsymbol{z}_a) = \{\boldsymbol{z}_1^+, \boldsymbol{z}_2^+, \cdots, \boldsymbol{z}_{k/2}^+, \boldsymbol{z}_{k/2+1}^-, \boldsymbol{z}_{k/2+2}^-, \cdots, \boldsymbol{z}_k^-\}$ be the k-neighboring examples of $\boldsymbol{z}_a$ sorted in descending order of their *propagated affinities* w.r.t. $\boldsymbol{z}_a$, i.e., $\boldsymbol{W}(\boldsymbol{z}_a, \boldsymbol{z}_1^+) > \cdots > \boldsymbol{W}(\boldsymbol{z}_a, \boldsymbol{z}_{k/2}^+) > \boldsymbol{W}(\boldsymbol{z}_a, \boldsymbol{z}_{k/2+1}^-) > \cdots > \boldsymbol{W}(\boldsymbol{z}_a, \boldsymbol{z}_k^-)$. Essentially, $\mathcal{N}_W(\boldsymbol{z}_a)$ is simply a sorted version of $\mathcal{N}_k(\boldsymbol{z}_a)$. As the obtained affinities are an indication of the semantic similarities among examples, we take it as a guidance to form triplets. Given an anchor $\boldsymbol{z}_a$, intuitively we can consider an example $\boldsymbol{z}_i^+$ with more affinity towards $\boldsymbol{z}_a$ as a *positive*, and another example $\boldsymbol{z}_j^-$ with lesser affinity towards $\boldsymbol{z}_a$ as a *negative*, and form a triplet $(\boldsymbol{z}_a, \boldsymbol{z}_i^+, \boldsymbol{z}_j^-)$. By considering first half of examples in the sorted neighborhood $\mathcal{N}_W(\boldsymbol{z}_a)$ as *positives*, and remaining half as *negatives*, we can form the following triplets: $(\boldsymbol{z}_a, \boldsymbol{z}_1^+, \boldsymbol{z}_{k/2+1}^-), (\boldsymbol{z}_a, \boldsymbol{z}_2^+, \boldsymbol{z}_{k/2+2}^-), \cdots, (\boldsymbol{z}_a, \boldsymbol{z}_{k/2}^+, \boldsymbol{z}_k^-)$. One may select the set of anchors from entire $\mathcal{X}_L \cup \mathcal{X}_U^{(p)}$, or by seeking the modes of the graph, without loss of generality.

Triplet-based Orthogonality-Promoting SSDML

Given $\mathcal{X}_L \cup \mathcal{X}_U^{(p)}$ and the corresponding $\boldsymbol{W}$, assume that we have obtained a triplet set $\mathcal{T}_p = \bigcup_b \mathcal{T}_p^{(b)}$. Here, $\mathcal{T}_p^{(b)}$ is a mini-batch of T_b triplets, and $b \in [1, \cdots, \left\lfloor \frac{|\mathcal{T}_p|}{T_b} \right\rfloor]$. Let, $\mathcal{T}_p^{(b)} = \{(\boldsymbol{z}_i, \boldsymbol{z}_i^+, \boldsymbol{z}_i^-)\}_{i=1}^{T_b}$. Then, we propose a smooth objective to learn the parameters $(\boldsymbol{L}, \theta_z)$ of our deep metric as follows:

$$\min_{\boldsymbol{L}, \theta_z} J_{metric} = \sum_{i=1}^{T_b} \log(1 + \exp(m_i)). \tag{3}$$

Here, $m_i = \delta^2_{\boldsymbol{L}}(\boldsymbol{z}_i, \boldsymbol{z}_i^+) - 4\tan^2\alpha\, \delta^2_{\boldsymbol{L}}(\boldsymbol{z}_i^-, \boldsymbol{z}_{i-avg})$, s.t., $\boldsymbol{z}_{i-avg} = (\boldsymbol{z}_i + \boldsymbol{z}_i^+)/2$, and $\delta^2_{\boldsymbol{L}}(\boldsymbol{z}_i, \boldsymbol{z}_j) = (\boldsymbol{z}_i - \boldsymbol{z}_j)^\top \boldsymbol{L}\boldsymbol{L}^\top (\boldsymbol{z}_i - \boldsymbol{z}_j)$. m_i tries to pull the anchor $\boldsymbol{z}_i$ and the positive $\boldsymbol{z}_i^+$ together, while moving away the negative $\boldsymbol{z}_i^-$ from the mean $\boldsymbol{z}_{i-avg}$, with respect to an angle $\alpha > 0$ at the negative $\boldsymbol{z}_i^-$ (Wang et al. 2017). Given the fact that J_{metric} in (3) is fully-differentiable, we can backpropagate the gradients to learn the parameters θ_z using SGD. This helps us in integrating our method within an end-to-end deep framework using Algorithm 1 (except, we learn wrt $\boldsymbol{L}$ instead of $\boldsymbol{M}$).

Additionally, we constrain $\boldsymbol{L} \in \mathbb{R}^{d\times l}, l \leq d$ to be a orthogonal matrix, i.e., $\boldsymbol{L}^\top \boldsymbol{L} = \mathbf{I}_l$. This is because in contrast to other regularizers that only constrain the values of the elements of a parametric matrix (like l_1 or l_2 regularizers), *orthogonality* omits redundant correlations among the different dimensions, thereby omits redundancy in the metric which could hurt the generalization ability due to overfitting (Xie et al. 2018). We further show that this avoids a *model collapse* (degenerate embeddings of distinct, close examples collapsing to a singularity). Theoretically, this is by virtue of the Johnson–Lindenstrauss Lemma (Dasgupta and Gupta 2003), which ensures the following near-isometry property of the embedding: $(1-\epsilon)\left\|\boldsymbol{z}_i - \boldsymbol{z}_j\right\|_2^2 \leq \frac{d}{l}\left\|\boldsymbol{L}^\top \boldsymbol{z}_i - \boldsymbol{L}^\top \boldsymbol{z}_j\right\|_2^2 \leq (1+\epsilon)\left\|\boldsymbol{z}_i - \boldsymbol{z}_j\right\|_2^2, 0 < \epsilon < 1$.

Given a mini-batch of triplets $\mathcal{T}_p^{(b)} = \{(\boldsymbol{z}_i, \boldsymbol{z}_i^+, \boldsymbol{z}_i^-)\}_{i=1}^{T_b}$, we provide an efficient matrix-based implementation to learn $\boldsymbol{L}$ (assuming a fixed θ_z), in the supplementary. Here, we present the computational time complexity of the major steps involved: **i) Cost Function:** Computing the cost requires four matrix multiplications resulting in complexity of O(ldT_b). Next, the transpose of the matrix products need to be computed, requiring O(lT_b). Finally, the sum of losses across all triplets can be computed using a matrix trace operation requiring O(T_b) complexity. **ii) Gradients:** The gradient with respect to $\boldsymbol{L}$ requires the following computations: transposes requiring O(dT_b), outer products requiring O(d^2T_b). The subsequent products require O(d^2l). Hence, the overall complexity is O($dT_b + d^2T_b + d^2l$).

Our proposed algorithm is linear in terms of the number of triplets in a mini-batch, i.e., T_b, which is usually low. The complexity of our algorithm is either linear or quadratic in terms of the original dimensionality d (which in practice is easily controllable within a neural network).

It should be noted that in contrast to existing SSDML approaches, our method also enjoys a lower theoretical computational complexity. For example, LRML has a complexity of O(d^3), while APLLR and APIT are of the order $O(d^3 + N^3)$.

Experiments

In this section we evaluate our proposed method in terms of its effectiveness in clustering and retrieval tasks on a number of benchmark datasets.

Dataset	MNIST					Fashion-MNIST					CIFAR-10				
Method	**NMI**	**R@1**	**R@2**	**R@4**	**R@8**	**NMI**	**R@1**	**R@2**	**R@4**	**R@8**	**NMI**	**R@1**	**R@2**	**R@4**	**R@8**
Initial	17.4	86.5	92.4	95.9	97.6	32.6	71.9	82.0	89.5	94.8	20.4	38.2	54.9	71.0	83.6
Deep-LRML	**48.3**	88.7	93.2	95.9	97.6	**53.2**	75.3	83.9	90.2	95.0	16.0	40.2	56.3	71.5	83.7
Deep-SERAPH	45.3	92.5	95.7	97.6	98.7	53.0	75.9	85.1	91.5	95.4	21.3	39.5	56.2	71.1	83.7
Deep-ISDML	44.1	92.1	95.7	97.5	98.6	51.3	74.4	84.2	90.7	95.1	20.0	36.4	52.4	68.5	82.1
Deep-APLLR	30.5	57.6	71.1	82.1	90.1	38.9	59.5	73.3	84.1	91.4	13.3	24.4	40.0	58.8	76.4
Deep-APIT	31.7	87.6	92.9	96.0	97.9	37.3	69.4	80.1	88.5	94.1	11.4	26.0	41.4	59.6	76.4
Ours	47.5	**93.9**	**96.6**	**98.2**	**98.9**	52.1	**77.6**	**86.0**	**91.8**	**95.6**	**25.3**	**41.4**	**57.3**	**72.6**	**84.9**

Table 1: Comparison against state-of-the-art SSDML approaches on MNIST, Fashion-MNIST and CIFAR-10.

Dataset	CUB-200					Cars-196				
Method	**NMI**	**R@1**	**R@2**	**R@4**	**R@8**	**NMI**	**R@1**	**R@2**	**R@4**	**R@8**
Initial	34.3	31.7	42.2	55.4	67.9	23.7	24.2	32.8	43.4	54.9
Deep-LRML	49.9	34.9	45.0	55.4	65.7	40.5	33.2	41.2	49.4	58.3
Deep-SERAPH	50.5	39.7	52.2	64.4	76.5	37.2	34.8	46.7	59.4	71.7
Deep-ISDML	50.6	43.7	55.7	67.6	78.4	25.8	30.6	40.8	52.2	63.2
Deep-APLLR	38.9	25.7	36.6	48.7	61.9	33.2	33.0	43.6	55.6	67.9
Deep-APIT	52.1	42.2	54.5	66.7	78.2	37.3	38.5	50.7	62.7	74.9
Ours	**54.0**	**44.8**	**56.9**	**69.1**	**79.9**	**40.7**	**45.7**	**58.1**	**69.5**	**80.4**

Table 2: Comparison against state-of-the-art SSDML approaches on fine-grained datasets.

Dataset	CUB-200					Cars-196				
Method	**NMI**	**R@1**	**R@2**	**R@4**	**R@8**	**NMI**	**R@1**	**R@2**	**R@4**	**R@8**
Exemplar	45.0	38.2	50.3	62.8	75.0	35.4	36.5	48.1	59.2	71.0
Rotation	49.1	42.5	55.8	68.6	79.4	32.7	33.3	44.6	56.4	68.5
NCE	45.1	39.2	51.4	63.7	75.8	35.6	37.5	48.7	59.8	71.5
DeepCluster	53.0	42.9	54.1	65.6	76.2	38.5	32.6	43.8	57.0	69.5
Synthetic	53.4	43.5	56.2	68.3	79.1	37.6	42.0	54.3	66.0	77.2
Ours	**54.0**	**44.8**	**56.9**	**69.1**	**79.9**	**40.7**	**45.7**	**58.1**	**69.5**	**80.4**

Table 3: Comparison of our method against state-of-the-art deep unsupervised methods on fine-grained datasets.

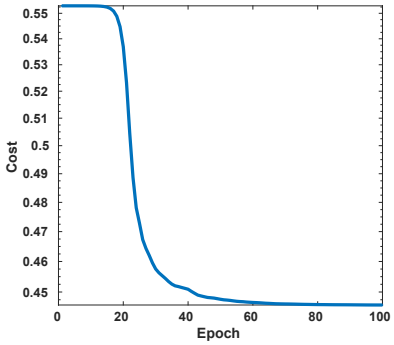
Datasets: Following recent literature, the benchmark datasets that have been used are as follows:

- **MNIST** (LeCun et al. 1998): It is a benchmark dataset that consists of 70000 gray-scale images of handwritten digits. Each image is of 28×28 pixels. There are 60000 training images and 10000 test images in the standard split.
- **Fashion-MNIST** (Xiao, Rasul, and Vollgraf 2017): It is a similar dataset as the MNIST, but consists of images from 10 categories of fashion products. There are 60000 training images and 10000 test images in the standard split.
- **CIFAR-10** (Krizhevsky, Hinton et al. 2009): This dataset consists of colour images of 32×32 pixels, containing animal or vehicle objects from 10 different categories. There are 50000 training images and 10000 test images.
- **CUB-200** (Welinder et al. 2010): This dataset consists of images of 200 species of birds with first 100 species for training (5864 examples) and remaining for testing (5924 examples).
- **Cars-196** (Krause et al. 2013): It consists of images of cars belonging to 196 models. The first 98 models containing 8054 images are used for training. The remaining 98 models containing 8131 images are used for testing.

The MNIST, Fashion-MNIST and CIFAR-10 datasets are widely used benchmarks with sufficiently large number of images for a comparative evaluation of different approaches. The CUB-200 and Cars-196 datasets are well-known for their use in *Fine-Grained Visual Categorization* (FGVC), and have huge intra-class variances and inter-class similarities.

Implementation details: We adapted the network architectures in the MatConvNet tool (Vedaldi and Lenc 2015). For MNIST and Fashion datasets, the network for MNIST has been adapted as: Conv1($5 \times 5, 20$) $\rightarrow$ max-pool $\rightarrow$ Conv2($5 \times 5, 50$) $\rightarrow$ max-pool $\rightarrow$ Conv3($4 \times 4, 500$) $\rightarrow$ ReLU $\rightarrow$ FC(500×128) $\rightarrow$ $l2$ $\rightarrow$ $\boldsymbol{L}(128 \times 64)$. For CIFAR-10, we used the following adapted network: Conv1($5 \times 5, 32$) $\rightarrow$ max-pool $\rightarrow$ ReLU $\rightarrow$ Conv2($5 \times 5, 32$) $\rightarrow$ ReLU $\rightarrow$ avg-pool $\rightarrow$ Conv3($5 \times 5, 64$) $\rightarrow$ ReLU $\rightarrow$ avg-pool $\rightarrow$ Conv4($4 \times 4, 64$) $\rightarrow$ ReLU $\rightarrow$ $l2$ $\rightarrow$ $\boldsymbol{L}(64 \times 32)$. For our method, we set $\gamma = 0.99$ in the affinity propagation step, $k = 10$ in the kNN graph, $\alpha = 40^\circ$ in (3), and initial learning rate 10^{-4}. For MNIST and Fashion datasets, we choose 100 labeled examples (10 per class), while for CIFAR-10, we choose 1000 labeled examples (100 per class). We sample a random subset of 9k unlabeled examples and use it along with the labeled data to mine triplets. For each random subset, we run our method for 10 epochs (with mini-batch size of 100 triplets). In total, we run for a maximum of 50 epochs and choose the best model from across all runs. For MNIST and Fashion, we train upon randomly initialized networks. For CIFAR-10, we observed a better performance by pretraining with labeled examples (by replacing the $\boldsymbol{L}$ *layer* with softmax) for 30 epochs, and then fine-tune using our loss for 50 epochs. For all datasets, the graph has been constructed using the $l2$-normalized representations obtained just before $\boldsymbol{L}$.

For the FGVC task, the GoogLeNet (Szegedy et al. 2015) architecture pretrained on ImageNet (Russakovsky et al. 2015), has been used as the backbone CNN, using MatConvNet (Vedaldi and Lenc 2015). We used the Regional Maximum Activation of Convolutions (R-MAC) (Tolias, Sicre, and Jégou 2016) right before the average pool layer, aggregated



α	NMI	R@1	R@2	R@4	R@8
35°	40.5	45.0	57.2	69.0	79.8
40°	40.3	45.5	57.8	69.1	80.0
45°	40.7	45.7	58.1	69.5	80.4
55°	40.4	45.7	57.9	69.4	80.1

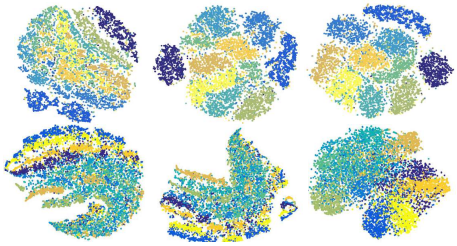


Figure 2: Left: Convergence behaviour of our method on the Cars-196 dataset (Krause et al. 2013). Middle: Ablation study showing sensitivity of our method towards α on Cars-196 (Krause et al. 2013) dataset. Right: tSNE embeddings for the test examples of MNIST (top row) and CIFAR-10 (bottom row). The left column represents the embeddings obtained right after random initialization. The embeddings obtained by our method: without orthogonality constraint on $\boldsymbol{L}$ (middle column) and with orthogonality constraint on $\boldsymbol{L}$ (rightmost column). Orthogonality leads to better embeddings (see Table 4).

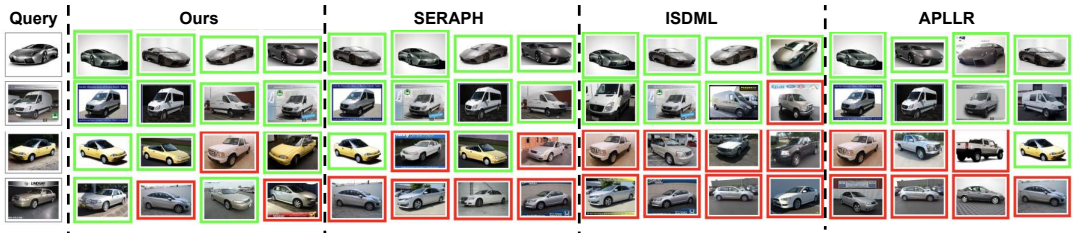


Figure 3: Qualitative comparison of retrieval performance against state-of-the-art SSDML approaches on the Cars-196 dataset. For a retrieved image, a red box denotes an incorrect retrieval (different class as the query), and a green box denotes a correct retrieval (same class as the query).

over three input scales (512, $512/\sqrt{2}$, 256). We choose five labeled examples per class. For our method, we set $\gamma = 0.99$, $k = 50$, $\alpha = 45°$ in (3), and embedding size of 128. We take the entire dataset without partition based sampling and run for a maximum of 200 epochs (mini-batch size of 100 triplets), and choose the best model. Specifically, using our triplet mining strategy, we mined 146600 triplets for the CUB dataset, and 201350 triplets for the Cars dataset. In all experiments, we fix a validation dataset by sampling 15% examples from each class of the training data. This validation dataset is used to tune the hyperparameters without taking any feedback from the test data. Note that to learn $\boldsymbol{L}$ with orthogonal constraints we made use of the Manopt (Boumal et al. 2014) tool with CGD ($max\_iter = 10$, with all other default settings).

Compared state-of-the-art baseline approaches: We compare our proposed SSDML method against the following baseline techniques:

- **Deep-LRML**: This is the stochastic extension of the classical LRML method (Hoi, Liu, and Chang 2010) discussed earlier, by making use of our proposed stochastic Algorithm 1. It follows the *min-max principle* for the labeled data (minimizing distances between similar pairs, while maximizing distances between dissimilar pairs). A Laplacian regularizer is used to capture information from the unlabeled data.
- **Deep-ISDML**: Stochastic extension of the ISDML (Ying et al. 2017) method. It is similar to LRML, but makes use of densities around an example to adapt the Laplacian regularizer.
- **Deep-SERAPH**: Stochastic extension of the SERAPH (Niu et al. 2014) method that makes use of the entropy minimization principle.
- **Deep-APLLR**: Stochastic extension of the APLLR (Dutta and Sekhar 2018) method. Makes use of a Log-Likelihood Ratio (LLR) based prior metric. The affinity propagation principle is used to propagate information from the labeled pairs to the unlabeled ones, and adapt the Laplacian (but no triplet mining like ours).
- **Deep-APIT**: Stochastic extension of the APIT (Dutta and Sekhar 2018) method. Makes use of an information-theoretic prior metric. The affinity propagation principle is used to adapt the Laplacian as in APLLR.
- **Exemplar** (TPAMI'16): This method (Dosovitskiy et al. 2016) attempts to learn the parameters associated with certain elementary transformation operations like translation, scaling, rotation, contrast, and colorization applied on random image patches.
- **Rotation Net** (ICLR'18): This method (Gidaris, Singh, and Komodakis 2018) aims to learn representations of data that can accurately capture the information present in an image despite any rotation of the subject.
- **NCE** (CVPR'18): This method (Wu et al. 2018) aims at bringing augmentations of an example together while moving away augmentations from different examples.
- **DeepCluster** (ECCV'18): This method (Caron et al. 2018) aims to jointly cluster metric representations while learn-

ing pseudo-labels for data in an end-to-end manner.

- **Synthetic** (AAAI'20): This method (Dutta, Harandi, and Sekhar 2020) learns a metric using synthetic constraints generated in an adversarial manner.
- **Triplet** (CVPR'15): This method (Schroff, Kalenichenko, and Philbin 2015) learns a metric using a standard triplet loss.
- **Angular** (ICCV'17): This method (Wang et al. 2017) learns a metric using an angular loss on a triplet of examples.

Observations: For comparing the approaches, we first learn a metric with the training data, and using it we obtain the test embeddings. The methods are compared based on their clustering (wrt NMI) and retrieval (wrt Recall@K, K=1,2,4,8) performances on the test embeddings. NMI is defined as the ratio of mutual information and the average entropy of clusters and entropy of actual ground truth class labels. The Recall@K metric gives us the percentage of test examples that have at least one K nearest neighbor from the same class. A higher value of all these metrics indicates a better performance for an approach.

The methods Deep-LRML, Deep-ISDML, Deep-SERAPH, Deep-APLLR and Deep-APIT are *semi-supervised* in nature. They have been chosen as direct counterparts for our proposed SSDML method. These baselines, including ours make use of both labeled (limited in number) and unlabeled examples to learn a metric. As seen in Tables 1-2, we outperform the baselines on all the datasets that vary in sizes and complexities. In Figure 3, we also show some qualitative retrieval comparisons of our method against the baselines on the Cars-196 dataset (successful retrieval is shown in green and failure is shown in red). Our method performs fairly better retrieval than the baselines.

The Exemplar, Rotation Net, NCE, DeepCluster and Synthetic techniques are *unsupervised* in nature, i.e., they learn a metric using only unlabeled data. As shown in Table 3, a better performance of our method in contrast to these approaches demonstrates the benefit of additionally available labeled examples while learning a metric.

Lastly, the Triplet and Angular methods are fully-supervised baselines that only learn from labeled examples. As shown in Table 5, a better performance of our method in contrast to these approaches demonstrates the benefit of leveraging additional unlabeled examples via affinity propagation and our mining technique. On the other hand, the fully-supervised baselines overfit to the training data because of the availability of only a limited number of labeled examples.

Ablation studies: We now perform a few ablation studies to understand the different components of our method. Table 4 reports the performance of our method for the two cases: i) **w/o orth**: $\boldsymbol{L}$ is not orthogonal, and ii) **w/ orth**: $\boldsymbol{L}$ is orthogonal. We observed that with orthogonality, the performance is better. To observe this qualitatively, we plot the tSNE embeddings of the test examples of MNIST and CIFAR-10 in Figure 2-(Right). We could observe that with orthogonality the embeddings are relatively better separated. Especially, in contrast to the simpler MNIST, orthogonality seems to have a more profound effect on the complex CIFAR-10 dataset. As

Dataset	Method	**NMI**	**R@1**	**R@2**	**R@4**	**R@8**
MNIST	w/o orth	42.9	91.5	95.3	97.3	98.6
	w/ orth	**47.5**	**93.9**	**96.6**	**98.2**	**98.9**
Fashion-MNIST	w/o orth	50.3	73.3	82.4	89.7	94.4
	w/ orth	**52.1**	**77.6**	**86.0**	**91.8**	**95.6**
CIFAR-10	w/o orth	19.0	37.4	54.2	69.7	83.4
	w/ orth	**25.3**	**41.4**	**57.3**	**72.6**	**84.9**

Table 4: Quantitative comparison of the performance of our method, without (w/o orth) and with orthogonality (w/ orth) on the metric parameters.

Method	**R@1**	**R@2**	**R@4**	**R@8**
Triplet	42.6	55.2	66.9	77.6
Angular	41.4	54.5	67.1	78.4
Ours	**44.8**	**56.9**	**69.1**	**79.9**

Table 5: Comparison of our method against supervised deep metric learning baselines on CUB.

conjectured, without orthogonality one may learn degenerate embeddings due to a model collapse. Additionally, for the Cars-196 dataset we show the convergence behaviour of our method for the first 100 epochs in Figure 2-(Left), and sensitivity of our method towards α in Figure 2-(Middle). The performance is fairly stable in the range $35^\circ - 55^\circ$.

Why our method addresses the limitations of the existing SSDML methods, and performs better ? 1. **LRML / ISDML vs Ours:** Both the LRML and ISDML methods define the affinity matrices directly using the distances among the initial representations. If the initial affinities are poor, the learned metric would be poor as well. On the other hand, our affinity matrix is adapted by the affinity propagation principle, while leveraging labeled data information as well. 2. **APLLR / APIT vs Ours:** Although APLLR and APIT make use of affinity propagation, they do not mine constraints using the enriched affinity matrix information, whereas we do. While their prior metric may be singular and hence poor, our method has no such dependency on a pre-computed metric. 3. **SERAPH vs Ours:** In contrast to SERAPH, our method has a stronger regularizer by virtue of orthogonality.

Conclusions

In this paper we revisit and revamp the important problem of Semi-Supervised Distance Metric Learning (SSDML), in the end-to-end deep learning paradigm. We discuss a simple stochastic approach to extend classical SSDML methods to the deep SSDML setting. Owing to the theoretical limitations of the existing SSDML techniques, we propose a method to address these limitations. While the components in our method have been studied earlier, their composition has not been performed before. We show that by using this design composition, the overall approach could outperform existing approaches by the collective use of some of the best practices. In short, following are our major contributions: 1. Extension of classical SSDML approaches for end-to-end stochastic deep learning, with the proposal of a new method, 2. A novel triplet mining strategy leveraging graph-based affinity propagation, and 3. Adaptation of an angular variant of a metric learning loss with orthogonality imposed on the metric parameters to avoid a model collapse.

Acknowledgements

The work was initiated when UKD was in IIT Madras. We would like to thank the anonymous reviewers for their efforts at reviewing our work and providing valuable comments to polish our paper.

References

Boumal, N.; Mishra, B.; Absil, P.-A.; and Sepulchre, R. 2014. Manopt, a Matlab toolbox for optimization on manifolds. *The Journal of Machine Learning Research* 15(1): 1455–1459.

Cakir, F.; He, K.; Xia, X.; Kulis, B.; and Sclaroff, S. 2019. Deep metric learning to rank. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1861–1870.

Caron, M.; Bojanowski, P.; Joulin, A.; and Douze, M. 2018. Deep clustering for unsupervised learning of visual features. In *Proc. of European Conference on Computer Vision (ECCV)*, 132–149.

Chapelle, O.; Scholkopf, B.; and Zien, A. 2009. Semi-supervised learning. *IEEE Transactions on Neural Networks* 20(3): 542–542.

Chen, Z.; Gong, M.; Xu, Y.; Wang, C.; Zhang, K.; and Du, B. 2020. Compressed Self-Attention for Deep Metric Learning. In *Proc. of Association for the Advancement of Artificial Intelligence (AAAI)*, 3561–3568.

Chopra, S.; Hadsell, R.; and LeCun, Y. 2005. Learning a similarity metric discriminatively, with application to face verification. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, 539–546. IEEE.

Dasgupta, S.; and Gupta, A. 2003. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures & Algorithms* 22(1): 60–65.

Deng, J.; Guo, J.; Xue, N.; and Zafeiriou, S. 2019. Arcface: Additive angular margin loss for deep face recognition. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4690–4699.

Dosovitskiy, A.; Fischer, P.; Springenberg, J.; Riedmiller, M.; and Brox, T. 2016. Discriminative Unsupervised Feature Learning with Exemplar Convolutional Neural Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 38(9): 1734–1747.

Dutta, U. K.; Harandi, M.; and Sekhar, C. C. 2020. Unsupervised Metric Learning with Synthetic Examples. In *Proc. of Association for the Advancement of Artificial Intelligence (AAAI)*.

Dutta, U. K.; and Sekhar, C. C. 2018. Affinity Propagation Based Closed-Form Semi-supervised Metric Learning Framework. In *Proc. of International Conference on Artificial Neural Networks (ICANN)*, 556–565. Springer.

Gidaris, S.; Singh, P.; and Komodakis, N. 2018. Unsupervised Representation Learning by Predicting Image Rotations. In *Proc. of International Conference on Learning Representations (ICLR)*.

Gong, X.; Yuan, D.; and Bao, W. 2020. Online Metric Learning for Multi-Label Classification. In *Proc. of Association for the Advancement of Artificial Intelligence (AAAI)*, 4012–4019.

Grandvalet, Y.; and Bengio, Y. 2005. Semi-supervised learning by entropy minimization. In *Proc. of Neural Information Processing Systems (NeurIPS)*, 529–536.

Gu, G.; and Ko, B. 2020. Symmetrical Synthesis for Deep Metric Learning. In *Proc. of Association for the Advancement of Artificial Intelligence (AAAI)*.

Hoi, S. C.; Liu, W.; and Chang, S.-F. 2010. Semi-supervised distance metric learning for collaborative image retrieval and clustering. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 6(3): 18.

Iscen, A.; Tolias, G.; Avrithis, Y.; and Chum, O. 2019. Label propagation for deep semi-supervised learning. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 5070–5079.

Krause, J.; Stark, M.; Deng, J.; and Fei-Fei, L. 2013. 3d object representations for fine-grained categorization. In *Proc. of IEEE International Conference on Computer Vision Workshops (ICCVW)*, 554–561.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images. Technical report, Citeseer.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11): 2278–2324.

Li, M.; Zhang, S.; Zhu, F.; Qian, W.; Zang, L.; Han, J.; and Hu, S. 2020. Symmetric Metric Learning with Adaptive Margin for Recommendation. In *Proc. of Association for the Advancement of Artificial Intelligence (AAAI)*, 4634–4641.

Liu, W.; Ma, S.; Tao, D.; Liu, J.; and Liu, P. 2010. Semi-supervised sparse metric learning using alternating linearization optimization. In *Proc. of ACM International Conference on Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD)*, 1139–1148.

Miyato, T.; Maeda, S.-i.; Ishii, S.; and Koyama, M. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* .

Movshovitz-Attias, Y.; Toshev, A.; Leung, T. K.; Ioffe, S.; and Singh, S. 2017. No fuss distance metric learning using proxies. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, 360–368.

Musgrave, K.; Belongie, S.; and Lim, S.-N. 2020. A metric learning reality check. In *Proc. of European Conference on Computer Vision (ECCV)*.

Niu, G.; Dai, B.; Yamada, M.; and Sugiyama, M. 2014. Information-theoretic semi-supervised metric learning via entropy regularization. *Neural computation* 26(8): 1717–1762.

Oliver, A.; Odena, A.; Raffel, C. A.; Cubuk, E. D.; and Goodfellow, I. 2018. Realistic evaluation of deep semi-supervised

learning algorithms. In *Proc. of Neural Information Processing Systems (NeurIPS)*, 3235–3246.

Qian, Q.; Shang, L.; Sun, B.; Hu, J.; Li, H.; and Jin, R. 2019. Softtriple loss: Deep metric learning without triplet sampling. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, 6450–6458.

Roth, K.; Milbich, T.; Sinha, S.; Gupta, P.; Ommer, B.; and Cohen, J. P. 2020. Revisiting training strategies and generalization performance in deep metric learning. In *Proc. of International Conference on Machine Learning (ICML)*.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision (IJCV)* 115(3): 211–252.

Schroff, F.; Kalenichenko, D.; and Philbin, J. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 815–823.

Shen, P.; Du, X.; and Li, C. 2016. Distributed semi-supervised metric learning. *IEEE Access* 4: 8558–8571.

Sohn, K. 2016. Improved deep metric learning with multi-class n-pair loss objective. In *Proc. of Neural Information Processing Systems (NeurIPS)*, 1857–1865.

Stretcu, O.; Viswanathan, K.; Movshovitz-Attias, D.; Platanios, E.; Ravi, S.; and Tomkins, A. 2019. Graph agreement models for semi-supervised learning. In *Proc. of Neural Information Processing Systems (NeurIPS)*, 8713–8723.

Sun, Y.; Cheng, C.; Zhang, Y.; Zhang, C.; Zheng, L.; Wang, Z.; and Wei, Y. 2020. Circle loss: A unified perspective of pair similarity optimization. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6398–6407.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1–9.

Tolias, G.; Sicre, R.; and Jégou, H. 2016. Particular object retrieval with integral max-pooling of CNN activations. In *Proc. of International Conference on Learning Representations (ICLR)*.

Vedaldi, A.; and Lenc, K. 2015. Matconvnet: Convolutional neural networks for matlab. In *Proceedings of the 23rd ACM international conference on Multimedia*, 689–692. ACM.

Wang, J.; Zhou, F.; Wen, S.; Liu, X.; and Lin, Y. 2017. Deep metric learning with angular loss. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*.

Wang, X.; Han, X.; Huang, W.; Dong, D.; and Scott, M. R. 2019. Multi-similarity loss with general pair weighting for deep metric learning. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 5022–5030.

Weinberger, K. Q.; Blitzer, J.; and Saul, L. 2006. Distance metric learning for large margin nearest neighbor classification. In *Proc. of Neural Information Processing Systems (NeurIPS)*, 1473–1480.

Welinder, P.; Branson, S.; Mita, T.; Wah, C.; Schroff, F.; Belongie, S.; and Perona, P. 2010. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology.

Wu, Z.; Xiong, Y.; Yu, S. X.; and Lin, D. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3733–3742.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747* .

Xie, P.; Wu, W.; Zhu, Y.; and Xing, E. P. 2018. Orthogonality-promoting distance metric learning: convex relaxation and theoretical analysis. In *Proc. of International Conference on Machine Learning (ICML)*.

Ying, S.; Wen, Z.; Shi, J.; Peng, Y.; Peng, J.; and Qiao, H. 2017. Manifold preserving: An intrinsic approach for semisupervised distance metric learning. *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)* 29(7): 2731–2742.

Yu, B.; and Tao, D. 2019. Deep metric learning with tuplet margin loss. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, 6490–6499.

Yuan, T.; Deng, W.; Tang, J.; Tang, Y.; and Chen, B. 2019. Signal-to-noise ratio: A robust distance metric for deep metric learning. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4815–4824.

Zhu, X. J. 2005. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.