PRODUCT IMAGE REPRESENTATION LEARNING ON LARGE SCALE NOISY DATASETS

*Aniket Joshi** *Nilotpal Das** *Promod Yenigalla*

{anikjosh, nilodas, promy}@amazon.com

ABSTRACT

Learning product similarity using distance metric learning from real world catalog needs to take care of large number of product categories and noisy labels. On one hand, large number of product categories makes online hard mining (OHM) less effective as hard triplets become sparse and thus difficult to find. On the other hand, the validity of the hard-triplets themselves is less certain in the case of noisy labelled training data. In this paper, we address the problem of large-scale product representation learning in the presence of noisy training data. To address these challenges, we propose a novel co-teaching based label correction scheme for distance metric learning, that is motivated by the inconsistencies of variations relationships in the product catalog. To validate our approach, we conducted experiments on 20 different product categories, where we achieve up to 4% improvement in PR-AUC compared to the SOTA baseline and conclude by discussing the durable learnings we gained from these experiments and directions for future research.

1. INTRODUCTION

In this paper, we address the problem of detecting variation family defects using product images in a large scale setting and noisy catalog data. In e-commerce catalog, items from the same base product are grouped together in a single detail page (DP) known as variation family. Each variation family in the catalog is characteried by an entity called parent-item, and is identified with an unique id called parent-id. Accordingly, the items within a family (a.k.a child-items) are identified by their child-item-ids (we use the terms item and child-item interchangeably). The setup lets customers explore different variations of the same product, such as color, style, size, etc., and select the one that suits their liking or needs. The two major issues associated with variations are 1) Overleveraged Variation Family (OLV), and 2) Broken Variations (BV) - check Fig 1. A variation family is overleveraged when unrelated items are grouped together in a single family. It creates customer confusion and can lead to wrong buying decisions. In BV, items belonging to a same base product are split across multiple families. We develop an image similarity model for product matching which can be used to detect these above mentioned defects.

The task of learning image similarity is generally formulated as distance metric learning problem, where images are projected into a vector space such that images of the same product are closer and different products are projected further apart. Designing such a system for a large scale product catalog is challenging due the lack of clean labelled data for training. Particularly in our case where the training labels is derived from parent-id, primarily, there exists two source of label noise - a) inter-class label noise (or OLV), where different products share the same class labels, b) intra-class label noise (or BV), where instances of same products have different class labels. To conquer the above mentioned challenges, we propose a framework with the following novel contributions, 1) propose co-teaching [1] based *learning with noisy labels* (LNL) for distance metric learning and 2) graph based interactions for identifying and correcting the noisy labels.

The remaining of the paper is organized as follows: in the next section, we present some of the related works. In section 3, we explain our approach in detail followed by some experimental results in section 4. We conclude with a summary and an outlook on future work.

2. RELATED WORK

There is a lot of literature on learning with noisy labels for classification tasks which primarily focuses on detecting noisy samples and gradually excluding them from training ([2], [3], [4], [5], [6]). Also the idea of using only samples with small losses combined with co-teaching two networks for classification is widely used ([7], [1]). However, there is very little literature on the problem of noisy labels in metric learning. Ibrahimi et al. [8] use a student-teacher-based training setup to identify noisy batch interactions and eliminate them from retrieval loss calculation. Liu et al. [9] identify noisy data from a batch by averaging the similarity between the current image features and the corresponding image class center. Castells et al.[10] used curriculum learning by down weighting the samples with large loss values using Super loss. The above methods are tested on the datasets with a limited number of classes, having large number of samples per class. Here, we are dealing with a large scale dataset consisting of thousands of classes (each family being a separate class (Sec 3)) with a very limited number of samples per class.

*equal contribution.

3. APPROACH

Formally, given an image $\boldsymbol{i}$, let a, p and u denote the item-id, parent-id and its unique product-id respectively. In an ideal scenario, the values of p and u should have exact one-to-one mapping for all the images in the catalog, since a variation family is expected to group together all the variants of an unique product. Our goal is to learn a representation $f()$ such that, $d(f(\boldsymbol{i_1}), f(\boldsymbol{i_2})) < d(f(\boldsymbol{i_1}), f(\boldsymbol{i_3}))$ where, $u_1 = u_2 \neq u_3$ for some distance measure $d()$ and imagess, $\boldsymbol{i_1}$, $\boldsymbol{i_2}$, and $\boldsymbol{i_3}$. Now, optimizing the model with respect to u is difficult since it is generally unknown and figuring it out manually is costly. So instead, we take parent-id, p as our ground truth label and train the model in a weakly supervised fashion. Notice that under this scenario, we introduce noise in our training set labels due to the presence of errors in variation relationships in the catalog. More specifically, given two images $\boldsymbol{i_1}$ and $\boldsymbol{i_2}$, we can define, 1) BV as: $u_1 = u_2$ & $p_1 \neq p_2$ and 2) OLV as: $u_1 \neq u_2$ & $p_1 = p_2$.

First of all, in order to tackle noisy labels, we adopt the co-teaching based approach inspired by this paper [1]. The idea of co-teaching is to use two different models, and let them teach each other. Intuitively, when two deep networks are trained separately, they get affected differently by different noisy samples, thus making the whole training procedure more robust to label noise. Prior approaches to LNL deals with classification tasks, which infers the noisy samples from sample based loss. An important distinction between classification and metric learning methods in general is that metric learning methods use interaction losses, i.e. loss functions defined on a pair or tuple of samples, compared to sample-based losses for classification. To identify and correct noisy samples for our task, we propose a graph based label-correction technique, which relies on interaction between the items.

In the following subsections, we describe the various components of our framework for effectively learning product image representations which is robust to label noise in training data.

3.1. Noise Correction (Co-Teaching)

We start with a mini-batch of items, $B = \{I, L\}$ of size n. Each batch contains a set of images, $I : \{i_1, i_2, \ldots, i_n\}$ and their labels, $L : \{p_1, p_2, \ldots, p_n\}$, which are their corresponding parent-ids. Additionally, let $A : \{a_1, a_2, \ldots, a_n\}$ be the corresponding item-ids of the images. Similar to [1], our method relies on two networks, M_1 and M_2 (fig. 1) which produces image embeddings, E_1 and E_2 respectively for the given mini-batch. Let $D_1^{n \times n}$ and $D_2^{n \times n}$ be the pairwise distances corresponding to embeddings E_1 and E_2 respectively. In each mini-batch, each network filters and refines the labels for its peer network for further training. Since labels are the index to the parent-id of the corresponding items, we correct the labels by refining the family relationships by either merging (in case of BV) or splitting (in case of OLV) the families in the mini-batch. For this, we create a graph of n nodes, where each node corresponds to a sample in a mini-batch, with edges determined based on the pairwise dissimilarity (Sec 3.1.2). The corrected labels L_1 (L_2) from model M_1 (M_2) are then used to train models M_2 (M_1). All of these steps are described in detail in the next sub-sections.

Loss Functions: Minimizing triplet loss is an effective technique for learning representations. Triplet loss works by minimizing the distance between the anchor ($\boldsymbol{i_a}$) and positive ($\boldsymbol{i_p}$), and maximizing the distance between anchor and negative ($\boldsymbol{i_n}$) such that it satisfies the margin constraint α.

$$Loss_{triplet} = [d(\boldsymbol{i_a}, \boldsymbol{i_p})^2 - d(\boldsymbol{i_a}, \boldsymbol{i_n})^2 + \alpha] \tag{1}$$

We use triplet losses ($Loss_1$ & $Loss_2$) to train both the models (M_1 and M_2). One set of triplets is generated using E_1 and L_2, to compute $Loss_1$. Similarly, the other set generated by E_2 and L_1 computes $Loss_2$. We also include a consistency loss ($Loss_c$) i.e. given by the L2 distance between the 2 embeddings E_1 and E_2, which forces the two models to embed the images consistently.

$$L_{final} = w_1 Loss_1 + w_2 Loss_2 + w_c Loss_c \tag{2}$$

where w_1, w_2, and w_c are the hyperparameters which control the importance of the different losses.

3.1.1. Model Architecture

Our proposed architecture has two parallel image feature extractors (fig. 1). For the model M_1, we use ResNet-101 which returns a feature embedding of size 2048. For the model M_2, we use a DeepRank[11] based architecture which has three networks in parallel - ResNet-101 and two more shallow branches to extract fine grained features from the image. The output features from these three branches are concatenated to make a final feature embedding of size 2560. A linear fully connected layer followed by a tanh non linearity is added on top of both the model outputs to make the embedding size from both the models consistent.

3.1.2. Batch Correction

Given a mini-batch containing noisy-labels, we use a graph based algorithm to correct the labels in the batch.

Graph Interactions: We define an undirected graph **G(N, E)**, where nodes are the n items in the batch and edges are constructed between the items of the same family, based on the original ground truth as shown in eq 5.

$$E(n_1, n_2) = \begin{cases} 1, & \text{if } p_{n_1} = p_{n_2} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

Now for merging BVs, we will add edges in the graph G that satisfy,

$$E(n_1, n_2) = 1, \quad \text{if } D(n_1, n_2) \leq T_{bv} \tag{6}$$

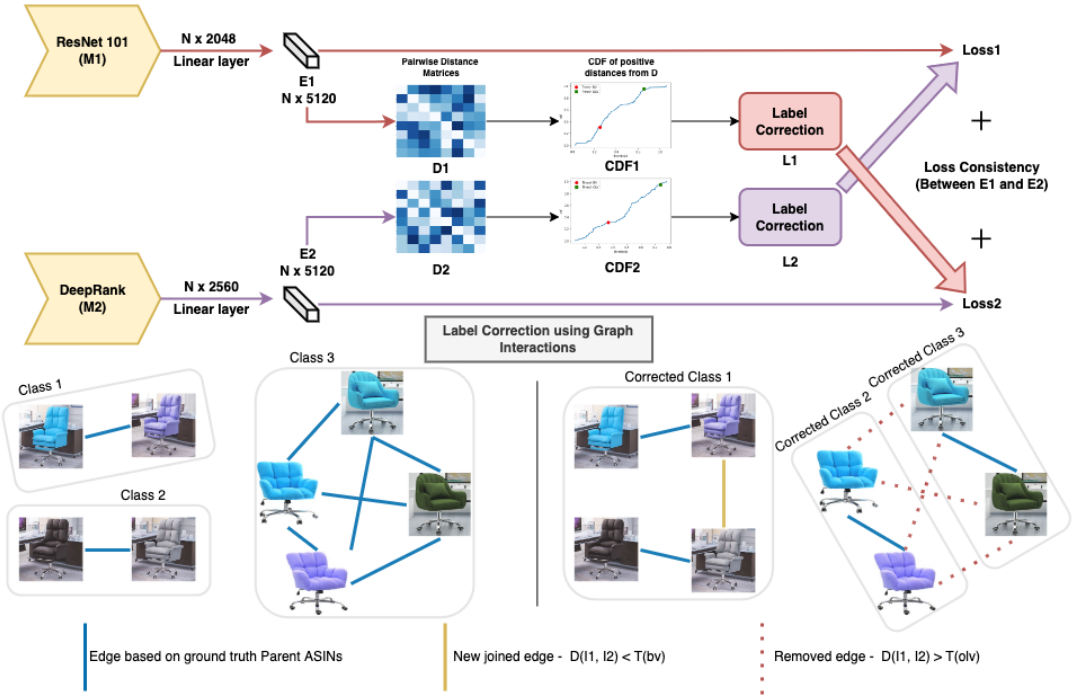

Fig. 1: Model architecture diagram and label correction example. In the example, class 1 and class 2 are BVs and merged into Corrected class 1 after label correction. Class 3 is a case of OLV family and splitted into Corrected class 2 & 3 after correction.

, giving us the graph G_{bv}. Now for splitting OLVs in the batch, we remove edges from the graph G_{bv} that satisfy,

$$E(n_1, n_2) = 0, \quad \text{if } D(n_1, n_2) \geq T_{olv} \tag{7}$$

, giving us the final graph G_{final}. The corrected labels, L for all the images in the mini-batch are then assigned according to the connected components in the graph G, i.e. nodes which are connected are assigned the same labels. The intuition behind using connected components is the transitivity of ground truth unique item-ids, i.e. if, $u_i = u_j$ and $u_j = u_k$, this implies $u_i = u_k$. Using this approach, we get two sets of corrected labels, L_1 and L_2 for the two networks.

In the above equation, the thresholds T_{olv} and T_{bv} are determined from the distribution of pairwise distances. Consider the cumulative distribution, $cdf(d)$ of pairwise distances of the images from the same family, i.e. having the same labels, $\{D(i,j) \mid p_i = p_j)\}$. Using this distribution, we select 2 thresholds: T_{bv} - threshold for broken variations, where $cdf(T_{bv}) = 0.3$, and T_{olv} - threshold for over-leveraged variations, where $cdf(T_{olv} = 0.95)$. Lastly, as our batch and embeddings are updated at every iteration, we take the exponential moving averaged (EMA) values for these thresholds (eqn. 4)

$$\begin{aligned} T_{bv} &= \lambda_{ema} T_{bv} + (1 - \lambda_{ema}) T_{bv_{-1}} \\ T_{olv} &= \lambda_{ema} T_{olv} + (1 - \lambda_{ema}) T_{olv_{-1}} \end{aligned} \tag{4}$$

where $T_{bv_{-1}}$ and $T_{olv_{-1}}$ are the exponential moving averages till the previous batch, and λ_{ema} is the weight to be given to the threshold in the current batch.

3.2. Batch Sampling & Inference

In this work, anchor and positive are sampled from the same parent ($p_a = p_p$), and negatives are sampled from separate parents ($p_a \neq p_n$). Randomly created triplets easily satisfy the loss constraints, and thus are not very useful for learning. Hence several hard mining strategies have been proposed in the literature [12]. Also, searching hard triplets in large datasets is computationally expensive, hence in practice, these are sampled from within a batch [13]. To ensure faster convergence with smaller batch in a dataset that consists of large number of product categories, we leverage product details such as item-brand and item-subcategories to group together items of similar types to create a batch.

Inference: We observe that as the consistency loss converges to a small value towards the end of the training, keeping both the models for inference is redundant as both the models generate nearly the same features. Hence, during inference we use only M_1 (ResNet 101) which reduces the latency. The distance between a pair of items, a_1 and a_2, is given by the L2 distance between the embeddings generated from model M_1. Our test data is labelled family wise, i.e. at a family level we have the markings if a particular family is OLV or non-OLV. During inference (fig. 2), we get the pairwise distances between all the items in the family and form

Table 1: Product Categories used for experimentation

Product Categories	Meal Holder, Drinking Cup, Pillow, Bedding Set, Artificial Plant, Tablecloth, Bed Linen Set, Hanging Ornament, Caddy, Towel, Blanket, Curtain, Shelf, Bed, Rug, Chair, Table, Desk, Pet Bed Mat, Hardware Handle

a pairwise distance matrix D, similar to what we did during training. Next, we construct a graph G with an edge between those items where pairwise distance in D is less then a predefined threshold (T_{test}). We classify the family as OLV if it has more then 1 connected components, otherwise we classify it as non-OLV.

4. EXPERIMENTS

4.1. Dataset Description & Evaluation

The dataset used in this work is obtained from the Amazon catalog and consists of 20 product categories (1). We use 5000 random families from the product detail page for creating the training data. Each sample in the test data (7000 families) is a family that are manually labelled as either OLV or a non-OLV. During testing, we measure how well our model is able to classify (binary classification) a family as OLV or non-OLV. We use three different metrics for evaluation, PR-AUC, Precision@Recall=50 and Precision@Recall=75.

4.2. Implementation Details

We ran the experiments in PyTorch. The training time for our baseline model is approximately 22 hours on a single Nvidia V100 GPU with a batch size of 48. The dimensions of the input image were kept as 224x224x3. During co-teaching, we train the model for 1000 steps without any batch cleaning so that the model learns the embedding space first and we do not merge or split the wrong families at the starting of the training procedure. We start our training with the easier triplets and gradually toughen them every 10000 steps. The loss functions weights w_1, w_2 and w_c were kept as 1, 1 and 0.2. λ_{ema} was kept as 0.3.

4.3. Results

We compare our method with ResNet-101, Co-teaching ((ResNet-101 + ResNet-101) & (DeepRank + ResNet-101))), and a recent state-of-the-art, T-SINT (Ibrahimi et al. [8]) (Table 2). We observe an improvement of about 2% in precision by applying the label correction strategy proposed in Sec 3.1.2 on top of ResNet model. Similar trend was observed for the DeepRank model as well.

In the co-teaching baseline (row 4 and 5), we train the 2 respective models in parallel without the label correction strategy. During the inference, we only take ResNet-101 features. This is to show the improvement using only knowledge distillation and also by using 2 different backbone networks instead of using the same. In T-SINT (row 3), where the noisy samples are discarded and not corrected, we adopt the same training strategy as described in the paper. Our proposed method with label correction (last row of Table 2) outperforms all the baselines with a significant improvement of about 4% (ResNet and T-SINT) & 2.5% (co-teaching) in precision. We use two models during the training phase, as described in Sec 3.1.1, but only use ResNet-101 during inference (Sec 3.2).

Table 2: Comparison with baseline

Model Type	P@50R	P@75R	PR-AUC
ResNet	87.23	84.41	84.73
ResNet + Label Correction	87.99	86.74	85.98
T-SINT [8]	87.62	84.78	85.35
Co-teaching (ResNet-101 + ResNet-101)	86.57	85.04	84.55
Co-teaching (ResNet-101 + Deeprank)	87.63	85.92	85.17
Proposed (Label Correction)	**91.03**	**88.37**	**88.46**

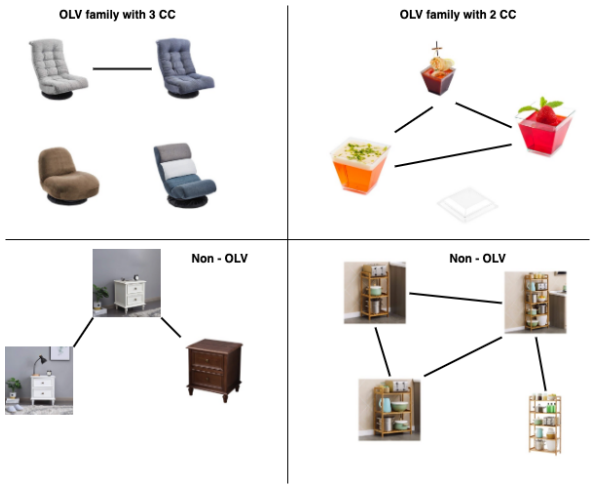

Fig. 2: Inference examples. CC stands for connected components in the graph.

4.4. Conclusion

In this paper, we propose a method for learning product representation using metric learning at scale. We consider the problems associated with the real-world catalog, such as its scale and the accompanying discrepancies in the weakly supervised labels. By proposing an online label correction scheme using a co-teaching distillation process and graph algorithm, we achieved an improvement of about 4% in PR-AUC compared to the baseline. Even though we have confined our experiments to product catalog images, our proposed pipeline is generic and can be useful in a variety of use-cases such as Face recognition and Person re-identification.

5. REFERENCES

[1] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," *Advances in neural information processing systems*, vol. 31, 2018. 1, 2

[2] Haozhi Zhang Chenfan Zhuang Dengke Dong Matthew R. Scott Sheng Guo, Weilin Huang and Dinglong Huang, "Curriculumnet: Weakly supervised learning from large-scale web images," in *European Conference on Computer Vision (ECCV)*, September. 1

[3] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu, "Making deep neural networks robust to label noise: A loss correction approach," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1944–1952. 1

[4] Aritra Ghosh, Himanshu Kumar, and PS Sastry, "Robust loss functions under label noise for deep neural networks," in *Proceedings of the AAAI conference on artificial intelligence*, 2017, vol. 31. 1

[5] Yisen Wang, Weiyang Liu, Xingjun Ma, James Bailey, Hongyuan Zha, Le Song, and Shu-Tao Xia, "Iterative learning with open-set noisy labels," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8688–8696. 1

[6] Görkem Algan and Ilkay Ulusoy, "Meta soft label generation for noisy labels," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 7142–7148. 1

[7] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama, "How does disagreement help generalization against label corruption?," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7164–7173. 1

[8] Sarah Ibrahimi, Arnaud Sors, Rafael Sampaio de Rezende, and Stéphane Clinchant, "Learning with label noise for image retrieval by selecting interactions," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, January 2022, pp. 2181–2190. 1, 4

[9] Chang Liu, Han Yu, Boyang Li, Zhiqi Shen, Zhanning Gao, Peiran Ren, Xuansong Xie, Lizhen Cui, and Chunyan Miao, "Noise-resistant deep metric learning with ranking-based instance selection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6811–6820. 1

[10] Philippe Weinzaepfel, Jérome Revaud, and Thibault Castells, "Superloss: A generic loss for robust curriculum learning," Apr. 14 2022, US Patent App. 17/383,860. 1

[11] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu, "Learning fine-grained image similarity with deep ranking," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1386–1393. 2

[12] Hong Xuan, Abby Stylianou, and Robert Pless, "Improved embeddings with easy positive triplet mining," 2020. 3

[13] Florian Schroff, Dmitry Kalenichenko, and James Philbin, "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. 3