A Prior-Guided Meta-Weight Network for Long-tailed Noisy Data in Item Categorization

Tianqi Wang[§]
University of Buffalo
Buffalo, NY, USA
twang47@buffalo.edu

Lei Chen
Rakuten Institute of Technology
Boston, MA, USA
lei.a.chen@rakuten.com

Jing Gao
Purdue University
West Lafayette, IN, USA
jinggao@purdue.edu

2023 IEEE International Conference on Big Data (BigData) | 979-8-3503-2445-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/BigData59044.2023.10386354

***Abstract*—Item categorization task aims to automatically infer the product category of an item based on its textual description. As a fundamental task in eCommerce domain, item categorization is widely adopted in many important applications such as product search, query understanding and product recommendation. However, this task faces several challenges. First, eCommerce data usually suffers from noisy facts since many key product values are self-reported by individual sellers and cannot be fully verified by experts. Second, eCommerce data usually follows the long-tail data distribution in which class distribution is highly imbalanced. To handle these challenges, some existing efforts simply combine approaches that are developed for noisy data and long-tail data separately. However, such a straightforward combination may not achieve satisfactory performance. In this paper, we propose a performance-driven Prior-Guided Meta-Weight Network (PGMWN) which handles the two challenges in a principled way. The proposed framework involves a meta re-weighting strategy to estimate the weights of samples mainly based on performance changes. Moreover, we leverage important data statistics to guide the meta re-weighting mechanism towards distribution-aware weights. A self-supervised representation learning component is utilized to further improve the framework's ability to address those two issues. To evaluate the effectiveness of the proposed PGMWN framework, comprehensive experiments are conducted on three public real-world datasets collected from Amazon. The proposed framework outperforms several state-of-the-art baselines in terms of various evaluation metrics. The experimental results show that the proposed model is able to handle the long tail data distribution and label noise issues and is effective in the item categorization task.**

***Index Terms*—item categorization, meta weight learning, long tail, noisy label**

I. Introduction

Item categorization is an important task in eCommerce. Its objective is to categorize the textual item description into a product category. This is a fundamental task which is widely adopted in many important applications including product search, query understanding and product recommendation in eCommerce. Despite the recent advances with pre-trained language models like BERT [9] and GPT [26] showing remarkable success in a variety of text processing tasks and researchers' efforts focusing on the item categorization task [4]–[6], [23], [31], the item categorization task is still challenging due to two reasons.

[§]The work was done when the first author was interning at Rakuten Institute of Technology.

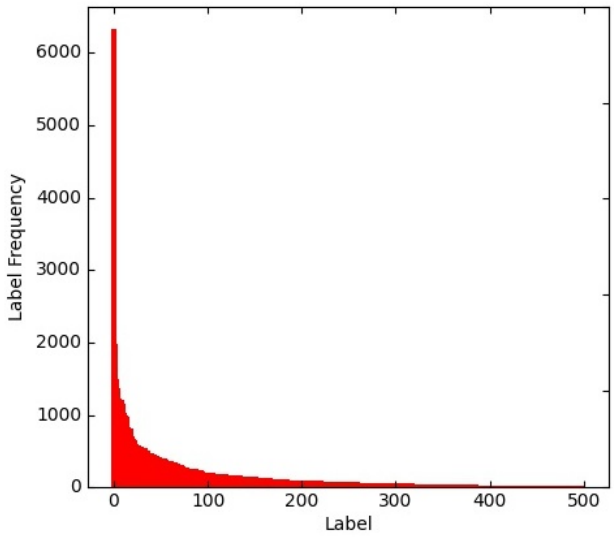

Fig. 1. The long-tail data distribution of the Amazon electronics dataset. The histogram is generated by counting the label frequencise where the x-axis denotes the number of labels and the y-axis represents the frequency of labels. For example, the (1, 6323) means there is a category with 6323 samples.

Noisy facts. The data collected from eCommerce platforms usually contains noisy facts. Because most of the information regarding products on eCommerce websites (e.g., Amazon, Ebay, and Walmart) is contributed by individual retailers, noisy facts about products are unavoidable. Meanwhile, it is prohibitively expensive to have those facts verified by experts considering the huge volumes of products. The existence of noisy facts will negatively impact the performance of product categorization. Therefore, how to handle noisy facts is one of the key challenges to develop item categorization models.

Long-tail distribution. As shown in Fig 1, the items on eCommerce platforms usually follow a long-tail distribution, where only few categories have a lot of items while most of the categories only include limited number of items. A classification model on such long-tailed data for item categorization may excel on a few head classes but perform poorly on a large set of tail classes. The unsatisfactory performance on tail classes could significantly degrade the shopping experience

for customers. How to handle this extremely imbalanced class distribution becomes another challenge.

To handle the aforementioned challenges, existing works [18], [34] investigate how to combine methods that handle noisy facts and methods that deal with long-tail distributions. However, a direct combination of methods developed separately for these two issues easily leads to conflicts in model design choices and thus may not achieve satisfactory performance. For example, when handling noisy data, the model design usually involves an assignment of lower weights to noisy data, while when dealing with long-tail data, a common strategy is to increase the weights of data points with a low frequency. Conflicts may occur when low-frequency data points are inferred to be noisy. Meanwhile, for data points belonging to tail classes, the judgement of their quality (whether they represent noisy facts or not) is difficult when there are not enough instances for achieving reliable quality estimation. In real eCommerce data, the two challenges exist simultaneously and are inseparable, and thus a new perspective that handles the two challenges simultaneously is needed.

Different from existing works, we propose a performance-driven framework, namely PGMWN, which deals with two challenges in a principled way. In the proposed PGMWN framework, the meta re-weighting method is introduced to estimate the weights of samples based on the performance changes. By this way, the framework tends to assign larger weights to data points that are more related to the categorization task, and thus improves the performance on the task of interest. Moreover, we rely on important data statistics to guide the data weight estimation mechanism so that weights are distribution aware. To further boost the performance, we propose to apply the self-supervised contrastive learning method, which learns informative representations without reliance on the label information. The proposed framework is evaluated on three datasets collected from Amazon. The experimental results show that the proposed PGMWN framework outperforms the state-of-the-art baselines. We also present case study analysis about the learned weights to explore insights behind the proposed framework.

The main contributions of this paper can be summarized as follows:

- We recognize the limitations of existing studies when they try to address the long-tail distribution and noisy label issues in item categorization task. Motivated by these limitations, we propose a novel performance-driven meta-re-weighting mechanism, namely PGMWN.
- The PGMWN framework leverages important statistical information to make re-weighting mechanism distribution-aware.
- Experimental results on three public real-world datasets collected from Amazon validate the effectiveness of the proposed PGMWN framework on item categorization task.

The rest of this paper is organized as follows: In the second section, we review related work. After that, the preliminary is introduced and then the notations, problem statement and technical details of the proposed framework are presented. Next, we demonstrate the experimental setup and results. Finally, we come to the conclusions.

II. Related Work

In this section, we summarize the existing work focusing on long-tailed classification and learning with noisy labels independently as well as some work addressing those two issues at the same time.

A. Methods addressing the long tail issue

There are many methods having been proposed to address the long tail (LT) issue, which can be categorized into the following categories: re-sampling, re-weighting and self-supervised learning methods. The re-sampling methods, such as SMOTE [3], samples the data to obtain a new dataset with balanced data distribution. Recently, a *two-stage* training strategy (exampled in [17], [37]), which decouples the learning of a feature encoder and the learning of a classifier and trains the classifier on a re-sampled balanced dataset, has become influential in computer vision and shows its superior performance on addressing the LT issue.

Another category methods try to re-weight the samples based on their label frequencies such as Focal loss [21], Class-balanced (CB) loss [8], Label-Distribution-Aware Margin loss (LDAM) [1] and so on. Those methods design functions of the label frequency to assign small weights to the samples in the head classes while large weights to the samples in the tail classes.

With the popularization of the two-stage methods in addressing the long-tail issue, Self-supervised methods [35] that do not rely on any label when training representations have been suggested to be effective for learning balanced representations in the representation learning stage.

B. Learning under label noise

Many efforts have also been devoted to the area of learning with noisy labels and the proposed methods can be classified into two groups. First, a group of methods utilize *sample selection* to identify clean samples from entire noisy dataset. For example, DivideMix [20] fits a Gaussian Mixture Model (GMM) on its training sample loss distribution to divide the training samples into a clean subset with labels and a noisy subset. Then, two subsets are jointly utilized in a semi-supervised learning way to avoid noisy labels' negative impact. Similarly, co-teaching [14] trains two networks at the same time to let them teach each other with selected clean samples. Next, the other group of methods adopt *loss correction* to shift model learning more on clean samples. For example, several noise-addressing loss weighting plans are manually designed [2], [36]. Recently, methods [27]–[29] have emerged to automatically learn the sample weighting plans to address the noisy label issue. In [11], contrastive learning is applied to better initialize models before using a set of robust learning methods such as [20], [29] for handling label noise. On the

item categorization task, [32] systematically compared several robust learning methods on real-world Amazon product data set.

C. Learning with long tail noisy data

Compared with a large number of previous investigations on addressing long-tail and noisy label issues separately, the investigations that jointly consider addressing those two issues are few. For example, on top of the two-stage LT-addressing method, which consists of a self-supervised pre-training stage and a classifier learning stage, [18] uses *superloss* [2] in the second stage with a goal of addressing the noisy label issue. Robust Long-Tailed Learning under Label Noise (RolT) [34] uses a semi-supervised approach to address the two issues at the same time. Compared to the method only considering the long-tailed issue, i.e., DivideMix [20], RoLT showed increased testing accuracy.

III. Background

In this section, we will introduce some background knowledge about self-supervised representation learning with SimCSE [10] and Meta-Weight-Net [29] for sample weight learning, which are utilized in the proposed framework to help alliveate the long-tail noisy label issue in the item categorization task.

A. Self-supervised Representation Learning with SimCSE

The self-supervised contrastive learning [7], [13] can help learn unbiased representations to help alleviate the long tail and noisy label issues [11], [16], [33]. In this paper, we apply the SimCSE [10] framework, which optimizes the InfoNCE loss [24] represented in Eq. (1) to maximize the agreement between the representations of the anchor sample and its augmentation to help learn unbiased representations. h, h^+ and H^- are the representations of the anchor sample s_a, a positive instance s^+ and the set of negative instances in the batch. $cos(\cdot)$ is the cosine similarity function. τ is the scaling hyper-parameter.

$$\mathcal{L}(h, h^+, H^-) = -\log \frac{\exp(cos(h, h^+)/\tau)}{\sum_{h^- \in H^-} \exp(cos(h, h^-)/\tau)} \quad (1)$$

The SimCSE obtains h and h^+ by feeding the inputs into the encoder twice using different random dropout masks [30] at the dropout rate 0.1 and all other samples in the same batch are considered as negative samples. The encoder is a fully-connected layer with tanh as activation function stacked on the BERT model.

B. Meta-Weight-Net for Sample Weight Learning

In this paper, we extend the Meta-Weight-Net framework in [29] with the prior guided weight learning regularization to re-weight the items to address the long-tail and noisy label issues at the same time. The Meta-Weight-Net framework can learn an explicit weight for the sample loss automatically. The weights are learned in a meta-learning manner. There are two main components in the Meta-Weight-Net [29]: the meta weight net to learn the weight of the samples with a deep neural network and the classifier to conduct the classification task. The parameters in the meta weight net and the classifier are updated alternatively with the other one fixed.

IV. Methodology

In this section, we first denote the terms and define the task. Followed by the notations and problem definition, the overview and the details of each component of the proposed framework are presented. Finally, we describe the loss function and parameter learning method of the proposed framework.

A. Notations and Problem Statement

Assume that there are $|\mathcal{C}|$ categories, and $\mathcal{C} = \{c_j\}_{j=1}^{|\mathcal{C}|}$, where c_j is the description of the j-th category. Let $\mathcal{I}$ denote the item set, and for each item $i \in \mathcal{I}$, its description is denoted as x_i, and its category is $l_i = c_j$. The correspond category index is denoted as a one-hot vector $y_i \in \{0, 1\}^{|\mathcal{C}|}$, where the element of the j-th position in the vector y_i is 1, while all other elements are 0.

Based on these notations, we can formally define the item categorization task as follows: Given the description x_i of an item i and the set of categories $\mathcal{C}$, item categorization needs to learn a model to predict which category the item belongs to (i.e., the category index $\mathbf{y}_i$ of the item i).

B. Overview of the PGMWN Framework

Fig. 2 shows the architecture overview of the PGMWN framework, which aims to address the long tail and noisy label issues simultaneously. PGMWN mainly consists of 4 modules, including the self-supervised representation learning network, the item weight learning network, the prior guided weight learning regularization network and the classifier.

We propose a novel meta weight learning framework based on the method in [29] with prior guided regularization for the item categorization task. The item weight learning network utilizes the meta re-weighting method introduced in [29] to assign the sample weights according to the changes of the performance, where larger weights tend to be assigned to samples that are more informative and thus help to improve the performance. Furthermore, we propose to incorporate the important data statistics to guide the sample weight learning to keep it aware of the sample characteristics via the prior guided item weight regularization network. To further improve the ability of the model addressing the long tail and noisy data issues, we utilize the aformentioned SimCSE [10] method to learn unbiased text representations. Finally, the classifier calculates the dot product between item and categories and categorizes the item into the category with the largest value. The details of the item weight learning network, the prior guided weight learning regularization network, the classifier and the way to learn the parameters will be introduced in the following part. Since we don't change the SimCSE [10] and briefly introduced it in the Background, we will not introduce the self-supervised representation learning component again in the following part.

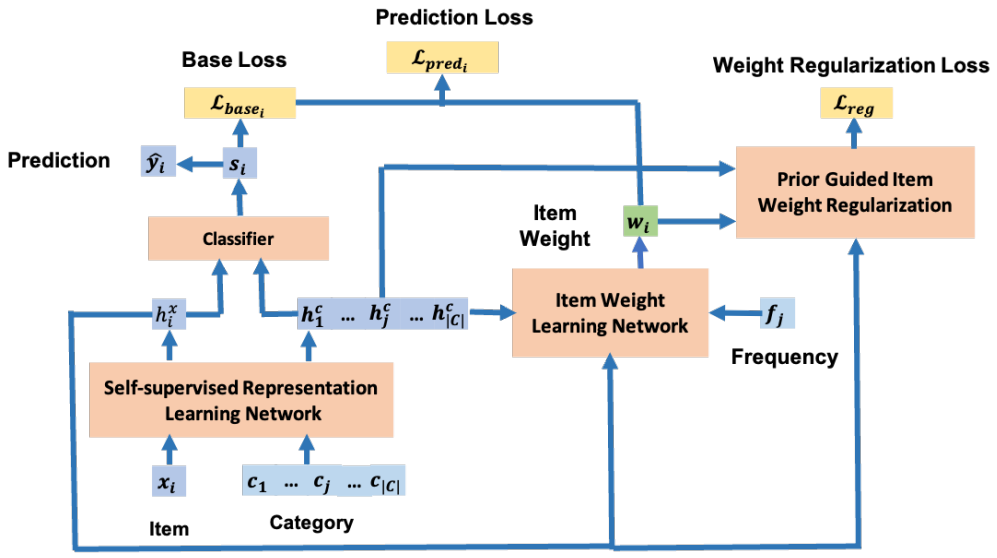

Fig. 2. The architecture overview of the proposed framework. The proposed framework consists of 4 modules, including the self-supervised representation learning module, the item weight learning module, the prior guided weight learning regularization network and the classifier. The training loss is the weighted base loss where the base loss is the classification loss such as cross entropy and the weight is derived from the item weight learning network with prior guided item weight regularization.

C. Item Weight Learning Network

Both the long-tail and noisy label issue will lead to insufficient high quality labeled data and consequently the lack of supervision. To fully exploit the labels, we propose to reweight the samples to assign different weights to different samples by the item weight learning network, where the weight of an item is related with its category frequency, the distance from the item to the corresponding category it belongs to, as well as the distance from its category to other categories. The weight learning network first learns the representations of the three factors and then obtains the weight for each item. The details of the item weight learning network are described as follows:

The weight learning network uses item embedding h_i^x and the category embedding h_j^c from the SimCSE self-representation learning network. First, the weight learning network obtains the category frequency representation of an item by taking its category representation and category frequency as inputs. For an item i belonging to category j, let f_j denote the frequency of the category c_j. Then the dense representation of the frequency of the category j can be obtained using a fully-connected layer represented as:

$$e_j^f = \text{ReLU}(\text{MLP}(h_j^c \oplus f_j)) \tag{2}$$

where $e_j^f \in \mathbb{R}^{d_e}$ is the dense representation of the category frequency of the category j, and d_j is the dimension of the dense representation. $\text{ReLU}(\cdot)$ represents the Rectified Linear Unit activation function where $\text{ReLU}(x) = max(x, 0)$. $\oplus$ denotes the vector concatenation operation.

Next, we consider the influence of the distance between the item and its corresponding category. For an item i and its corresponding category j, the distance between them can be represented as:

$$dist_i = -\cos(h_i^x, h_j^c) \tag{3}$$

Since $dist_i$ is related with both the item representation h_i^x and the category embedding h_j^c, to learn the dense representation of the distance, we consider both the item and category embeddings. The dense representation that is aware of the item, its corresponding category as well as the distance between them can be derived as:

$$e_i^x = \text{ReLU}(\text{MLP}(h_j^c \oplus h_i^x \oplus dist_i)) \tag{4}$$

where e_i^x is the dense representation aware of the item, category and their distance.

Then the weight learning network models the similarity of the category j that item i belongs to. To model the similarity of the category j, we consider the distances between the representation of category j and all other categories in the category set $\mathcal{C}$. Let $dist_j$ denote the nearest neighbor distance of the category which is the minimum distance between the category j and all other categories in the category set $\mathcal{C}$. Then $dist_j$ can be defined as:

$$dist_j = \min(-\cos(h_j^c, h_{j'}^c)), \quad \forall c_{j'} \in \mathcal{C} - \{c_j\} \tag{5}$$

where $h_{j'}^c$ denotes the embeddings of the category c_j and $c_{j'}$ separately.

Similar to modeling the category frequency, the nearest neighbor distance of the category can be represented by a dense vector, which can be derived as:

$$e_j^c = \text{ReLU}(\text{MLP}(h_j^c \oplus dist_j)) \tag{6}$$

where $e_j^c \in \mathbb{R}^{d_e}$ represents the dense representation of the confusion level of the category j of the item i. d_e is the dimension of the dense vector e_j^c.

With the three dense representations: e_j^f, e_j^c as well as e_i^x, the weight for the item i can be learned with a two-layer fully connected neural network with the aforementioned three representations as the inputs. Let $w_i \in \mathbb{R}$ denote the learned weight for the item i. Then w_i can be represented as:

$$w_i = \sigma(\text{MLP}(\text{ReLU}(\text{MLP}(e_j^f \oplus e_j^c \oplus e_i^x)))) \tag{7}$$

where $\sigma(\cdot)$ denote the Sigmoid activation function mapping the output to the range from 0 to 1.

D. Prior Guided Weight Learning Regularization

To learn a better weight w_i for the item i, in addition to using the signal from the back propagation from the supervision obtaining via labels as in existing work such as [29], we also consider to guide the weight learning via using prior knowledge of the items and categories as illustrated in Fig. 3. The details are shown as follows:

The order of category frequency: In the proposed framework, we follow the idea that the items belonging to the categories with lower frequencies should be assigned larger weights by the weight learning network. To satisfy this constraint, we first recover the frequency by using the learned weight and the category of the item as inputs which can be represented as:

$$\hat{e}_j^f = \sigma(\text{MLP}(w_i \oplus h_j^c)) \tag{8}$$

According to our analysis, the larger weight should be assigned to the item in the less frequent category. For two items i_1 and i_2 belonging to category j_1 and j_2 ($j_1 \neq j_2$) respectively, given $w_{i_2} > w_{i_1}$, then the probability that $f_1 > f_2$ can be represented as:

$$p(f_{j_1} > f_{j_2}) = \sigma(\hat{e_{j_2}^f} - \hat{e_{j_1}^f}) \tag{9}$$

where f_{j_1} and f_{j_2} are the frequency of categories j_1 and j_2, and e_{j_1} and e_{j_2} are the embeddings of the obtained via Eq. (8). w_{i_1} and w_{i_2} are the learned weights by the item weight learning network of the item i_1 and i_2.

The order of distance between the item and its corresponding category: Next, we consider to use the distance between the item and its corresponding category to regularize the item weight learning. To correctly classify items and obtain a model with better generalization, the classifier needs to capture the common characteristics of the category. For the items that are far from the category, they cannot be representative of the characteristics of the category and may be the noise influencing the performance of the model. Thus, we want the weight learning network to assign smaller weights to those items that are far from the category embeddings compared with the items that are closer to the category embeddings.

To recover the distance representation between the item and its corresponding category, a one-layer fully connected layer is applied, which can be represented as:

$$\hat{e}_i^x = \text{ReLU}(\text{MLP}(w_i \oplus h_i^x \oplus h_j^c)) \tag{10}$$

In such a way, the probability that the item i_2 is closer to the category j than item i_1 when i_1 and i_2 both belong to the same category j can be represented as:

$$p(dist_{i_1}^x > dist_{i_2}^x) = \sigma(\hat{e_{i_2}^x} - \hat{e_{i_1}^x}) \tag{11}$$

where $dist_{i_1}^x$ and $dist_{i_2}^x$ are the distances from the items i_1 and i_2 in the same category to the category embeddings they belong to. $\hat{e_{i_1}^x}$ and $\hat{e_{i_2}^x}$ are the embeddings obtained by Eq. (10). $p(dist_{i_1}^x > dist_{i_2}^x) \in \mathbb{R}$ denotes the probability that the item i_1 are further from the categories than item i_2 according to their assigned weights by the weight learning network.

The order of category distance: The category distance also affects the sample weights. The items belonging to the category with a closer nearest neighbor category may be more informative since their neighbors can borrow information from them. Hence, we want the weight learning network to assign larger weights to the items belonging to the categories that has a smaller nearest neighbor distance. To achieve this goal, we use the derived distance between the item category and its closest category to regularize the item weight learning.

Similar as the frequency constraint, we recover the distance by:

$$\hat{e}_i^d = \text{ReLU}(\text{MLP}(w_i \oplus h_j^c)) \tag{12}$$

Then the probability that the items belonging to the category that is more likely to be confused with other categories are assigned with larger weights by the item weight learning network can be represented as:

$$p(dist_{j_1} > dist_{j_2}) = \sigma(\hat{e}_{i_2}^d - \hat{e}_{i_1}^d) \tag{13}$$

where i_1 and i_2 are two items belonging to category j_1 and $j_2 (j_1 \neq j_2)$. $dist_{j_1}$ and $dist_{j_2}$ are the nearest neighbour distance of category j_1 and j_2. $\hat{e}_{j_1}^d$ and $\hat{e}_{j_2}^d$ are the embeddings obtained by Eq. (12). $p(dist_{j_1} > dist_{j_2}) \in \mathbb{R}^1$ denotes the probability that the category j_2 has a closer nearest neighbor category than j_1 according to their assigned weights by the weight learning network.

To regularize the weight learning network, for each constraint, a cross-entropy loss is applied for the valid pairs and the total regularization term is the average of the three losses. The loss for the three constraints can be represented as:

$$\begin{aligned}
\mathcal{L}_f &= \frac{1}{n_1} \sum \text{CE}(p(f_{j_1} > f_{j_2}), \mathbb{1}(f_{j_1} > f_{j_2})) \\
\mathcal{L}_d &= \frac{1}{n_2} \sum \text{CE}(p(dist_{i_1}^x > dist_{i_2}^x), \mathbb{1}(dist_{i_1}^x > dist_{i_2}^x)) \\
\mathcal{L}_c &= \frac{1}{n_3} \sum \text{CE}(p(dist_{j_1} > dist_{j_2}), \mathbb{1}(dist_{j_1} > dist_{j_2}))
\end{aligned} \tag{14}$$

where $\mathcal{L}_f$, $\mathcal{L}_c$ and $\mathcal{L}_d$ denote the losses of the three constraints, and n_1, n_2 and n_3 are the number of valid item pairs of the three constraints in a batch. $CE(\cdot, \cdot)$ represent the cross-entropy loss function, where the first argument is the predicted probability and the second argument is the label. $\mathbb{1}(\cdot)$ is the indentation function where if the condition is true the value of the function will be 1; otherwise, the value of the function will be 0.

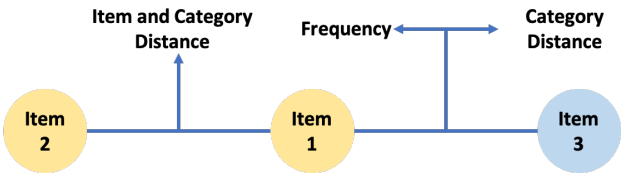

Fig. 3. Illustration of the item weight regularization. The figure shows order constraint applied cases where the colors of the items denote their categories. For the item 1 and 2 in the same category, ideally their weights should satisfy the constraint of item and category distance order. Considering item 1 and 3 belonging to different categories, their weights should meet the constraint of the category frequency order and category distance order constraints.

The total weight learning regularization loss $\mathcal{L}_{reg}$ is the average of the three constraint losses, which is:

$$\mathcal{L}_{reg} = \frac{1}{3}(\mathcal{L}_f + \mathcal{L}_d + \mathcal{L}_c) \tag{15}$$

E. Classifier

The classifier is used to classify the item with textual description to a category. Since the category representations which characterize the categories can be obtained using their textual descriptions by the encoder, they can be used to derive the classification matrix. We first use a two-layer fully-connected network to transform the representations of the item and all the categories to the latent space, which can be represented as:

$$\begin{aligned} h_i &= \mathrm{ReLU}(\mathrm{MLP}(\mathrm{ReLU}(\mathrm{MLP}(h_i^x)))), \\ h_j &= \mathrm{ReLU}(\mathrm{MLP}(\mathrm{ReLU}(\mathrm{MLP}(h_j^c)))), \quad \forall c_j \in \mathcal{C} \end{aligned} \tag{16}$$

where h_i and $h_j \in \mathbb{R}^{d_c}$ are the latent representations of the item i and category j.

Then the classifier determines the category of an item by assigning the item to the category with the largest dot product result, which is represented as:

$$\begin{aligned} s_i^j &= h_i \times h_j^T \\ \hat{y_i'} &= \mathrm{argmax}_j(s_i^j, \forall c_j \in \mathcal{C}) \\ \hat{y_i} &= \mathrm{onehot}(|\mathcal{C}|, \hat{y_i'}) \end{aligned} \tag{17}$$

where $\times$ denotes the vector multiplication operation. $s_i^j \in \mathbb{R}$ represents the score that item i belongs to category j. $y_i' \in \mathbb{R}$ is the index of the predicted category and $argmax$ denotes the function returning the index of the maximum value. $\hat{y_i} \in \mathbb{R}^{|\mathcal{C}|}$ represents the predicted label of the item i. $onehot(\cdot,\cdot)$ is the function generating one-hot embedding where the first argument ($|\mathcal{C}|$) representing the dimension of the vector and the second argument ($\hat{y_i'}$) set the index of 1 in the vector.

F. Prediction Loss

With the learned item weights, the prediction loss can be defined as the average of the base loss for the items:

$$\mathcal{L}_{pred} = \frac{1}{N}\sum w_i \mathcal{L}_{base_i} \tag{18}$$

where the $\mathcal{L}_{pred}$ is the prediction loss and $\mathcal{L}_{basei}$ represents the base loss for the item i and N is the number of samples in the batch. In our framework we consider to use the cross-entropy loss (PGMWN$_C$) and LDAM loss [1] (PGMWN$_L$), which has been proven to be among the most effective methods on imbalanced datasets for many tasks.

G. Parameter Learning

The training of the framework consists of two stages. The first stage is to train the self-supervised representation learning network and use the obtained model parameters as the initialization of the second stage. The second stage is to optimize the representation learning network, item weight learning network as well as the classifier jointly.

Following the online strategy to update the parameters in a single optimization loop in [29]. Let $\mathbf{W}$ denote all the model parameters in the representation learning network and the classifier and Θ denote all the model parameters in the item weight learning network.

Algorithm 1 PGMWN framework learning algorithm

Input : Item set $\mathcal{I}$, category set $\mathcal{C}$, batch size n, m, learning rate α, β, max iterations T
Output: Network parameters $\mathbf{W}^T, \Theta^T$

function PGMWN FRAMEWORK LEARNING
Stage 1:
Train the representation learning network with $\mathcal{I}$ and $\mathcal{C}$ by optimizeing Eq. (1).
Stage 2:
Initialize the representation network using the obtained parameters in Stage 1 and randomly initialize the classifier parameters $\mathbf{W}_{(0)}$
Randomly initialize item weight learning network parameters $\Theta_{(0)}$.
for $t = 0$ to $T-1$ **do**
  $\{x, y\} = MiniBatchSampler1(\mathcal{I}, N)$
  Formulate the representation network and classifier learning function $\hat{\mathbf{W}}^{(t)}(\Theta)$ by Eq. (19).
  $\{x', y'\} = MiniBatchSampler2(\mathcal{I}, M)$
  Update $\Theta^{(t+1)}$ by Eq. (20).
  Update $\mathbf{W}^{(t+1)}$ by Eq. (21)
end for
return $\mathbf{W}^T, \Theta^T$

The online strategy consists of the following three steps:

Representation network and classifier learning manner formulation. In this step, SGD is used to optimize the prediction loss in Eq. (18). The corresponding parameters $\mathbf{W}$ can be updated according to

$$\hat{\mathbf{W}}^{(t)}(\Theta) = \mathbf{W}^{(t)} - \frac{\alpha}{N}\sum_{n=1}^{N} V(I_i(\mathbf{W}^{(t)}); \Theta)\nabla_{\mathbf{W}}\mathcal{L}_{base_n}(\mathbf{W})|_{\mathbf{W}^{(t)}} \tag{19}$$

where α is the learning rate. $V(\cdot;\cdot)$ is the item weight learning network function with the first argument as the inputs and the second argument as the model parameters. $\mathcal{L}_{base_n}(\mathbf{W})$

represents the loss function respect to the variable of model parameters $\mathbf{W}$.

Item weight learning network parameter update. With the representation and classifier network updating formulation, the parameters Θ in the item weight learning network can be updated by

$$\Theta^{(t+1)} = \Theta^{(t)} - \beta \frac{1}{M} \sum_{m=1}^{M} \nabla_{\Theta} \mathcal{L}_{base_m}(\hat{\mathbf{W}}^{(t)}(\Theta))|_{\Theta^{(t)}} - \beta \nabla_{\Theta} \mathcal{L}_{reg}(\hat{\mathbf{W}}^{(t)}(\Theta))|_{\Theta^{(t)}} \tag{20}$$

where β denotes the learning rate.

Representation network and classifier update. The third step is to update the model parameters in the representation learning network and classifier with the updated $\Theta^{(t+1)}$ by

$$\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} - \frac{\alpha}{N} \sum_{n=1}^{N} V(I_i(\mathbf{W}^{(t)}); \Theta^{(t+1)}) \nabla_{\mathbf{W}} \mathcal{L}_{base_n}(\mathbf{W})|_{\mathbf{W}^{(t)}} \tag{21}$$

The two-stage learning strategy is summarized in Algorithm 1.

V. Experiments

In this section, we conduct extensive experiments on three public datasets to evaluate the effectiveness of the proposed PGMWN framework. The details of the experiments as well as the results are presented. We first introduce the used datasets and then list state-of-the-art baselines. To increase the reproducibility of the proposed model, we provide the details of experimental setup and implementation. Finally, we present the result comparison and model insight analysis.

A. Datasets

We use the Amazon product dataset [15], [22] to conduct experiments, which is a publicly available and commonly-used dataset in the natural language processing (NLP) domain. In particular, following the setting of existing work [32], we extract three sub categories from the whole Amazon product dataset, including **Automotive**, **Beauty**, and **Electronics**. Each data sample in these three datasets consists of a textual title and a corresponding textual category label. The textual titles are model inputs, and the goal of this task is to predict their category labels. Table I shows the statistics of the three datasets. From the quarter-tiles of label frequency in Table. I, we can find the label frequency distributions are right tail skewed for all the three datasets and as we mentioned in Section I, the category distributions of the three datasets all have long tail characteristics, where most categories have few samples while only few categories contain a huge amount of samples.

B. Models for Comparison

Our work focuses on addressing the long-tail and noisy label challenges and uses a two-stage training strategy and re-weights the samples. For fairly validating the performance of the proposed model, all the models are based on the BERT model and uses the cls tokens as the representations of the texts. We choose the following models as our baselines: CE, LDAM [1], cRT [17], unsupervised SimCSE [10], instance data parameters (DP) [28], self-supervised learning with LDAM superloss (SS+SL+LDAM) [18], MWNet [29] with CE (MWMet$_C$) and LDAM (MWNet$_L$).

The details of those compared methods are described as follows:

1) CE: It directly uses BERT to learn representations of the textual descriptions of item and categories and feeds the obtained representations to a classifier consisting of 3-layer fully connected neural network. The cross entropy is used as the prediction loss.
2) LDAM: The model architecture is the same as the CE, while it uses the LDAM [1] loss .
3) cRT: The model architecture is the same as CE. cRT applies the two-stage training strategy cRT in [17], where the a classifier is trained in the second stage on a resampled balanced dataset using the representations obtained from the first stage.
4) SimCSE: The model applies two-stage training strategy, which uses self-supervised contrastive learning loss to train the BERT first and then initializing the BERT in CE with the obtained parameter in the first stage.
5) DP: It uses the same model architecture as CE and the cross entropy loss is applied. The data parameters are learned following [28] to re-weight logits to learn a robust classifier.
6) $SS + SL + LDAM$: It follows the framework in [18], where in the first stage the self-supervised training is applied and in the second stage the Superloss [2] with LDAM [1] loss as the base loss is applied. The Superloss is a robust loss designed to handle noisy label issues.
7) $MWNet_C$: It learns the sample weights following the similar method in [29]. Instead of using the loss as the input for the weight learning network in [29], the $MWNet_C$ uses the same inputs as the proposed PGMWN framework.
8) $MWNet_L$: It shares the same architecture as the $MWNet_C$, while uses LDAM loss instead of CE loss.

C. Experimental Setup

1) Implementation: All the models are implemented using PyTorch [25]. The Xavier normal initializer [12] is applied to randomly initialize parameters. The batch size is set to 32. d_h is set to 768 and d_e is set to 32. The SGD is applied to optimize the parameter set $\mathbf{W}$, and Adam [19] is applied to optimize the parameter set Θ. The initial learning rate α and β are set to $1e^{-5}$ with a linear decay. The number of maximum epochs is 40. To avoid the overfitting issue, we also use the dropout technique [30],and set the dropout rate to 0.1.

2) Data Partition: We use the same way as [32] to preprocess the three datasets. Based on the preprocessed datasets, we randomly split the whole datasets into three subsets, i.e., training, validation, and testing sets, with the ratio $0.70 : 0.15 : 0.15$. For the proposed model and baselines, we use the training set to learn model parameters, and the validation set is used to select the best model and perform early stop.

TABLE I
STATISTICS OF DATASETS

	Labels	Samples	Title Length	Label Frequency Quarter-tile
Automotive	953	160,725	9.90 ± 5.51	[16, 30, 60, 134, 6,963]
Beauty	229	159,805	10.26 ± 5.61	[16, 62, 202, 744, 9,010]
Electronics	500	86,357	14.90 ± 9.56	[16, 34, 72, 170, 6,323]

TABLE II
MODEL PERFORMANCE ON LONG-TAILED NOISY ITEM CATEGORIZATION ON THE THREE DATASETS. THE BEST RESULTS ARE HIGHLIGHTED USING BOLD FONTS, AND THE SECOND BEST RESULTS ARE UNDERLINED.

Dataset	Automotive				Electronics				Beauty			
Metric	MiAcc	MaAcc	$WF1$	$MF1$	MiAcc	MaAcc	$WF1$	$MF1$	MiAcc	MaAcc	$WF1$	$MF1$
CE	78.23	64.73	78.03	63.95	68.27	54.09	67.68	52.94	71.68	56.67	71.44	56.64
LDAM	78.94	67.38	78.70	65.68	68.83	56.70	68.22	54.87	73.21	57.21	72.61	56.99
cRT	78.89	67.35	77.85	63.72	68.80	56.72	67.54	52.99	71.72	55.92	71.55	55.88
SimCSE	78.23	64.73	76.36	64.25	66.03	55.55	65.82	53.30	71.21	**60.10**	70.99	58.06
DP	78.12	65.30	78.35	64.30	68.25	54.08	67.50	52.90	71.56	57.30	71.29	56.94
SS+SL+LDAM	78.81	67.50	78.55	65.81	69.15	56.24	68.16	54.45	73.28	57.95	72.65	57.84
MWNet$_C$	78.45	65.84	78.17	64.65	68.20	54.04	67.51	53.05	71.82	57.58	71.57	57.42
MWNet$_L$	78.74	67.43	78.57	65.59	68.78	56.42	68.12	54.90	73.32	57.95	72.65	57.80
PGMWN$_C$	79.17	67.25	78.70	65.19	68.92	55.81	68.54	54.21	72.64	58.54	72.35	**58.62**
PGMWNL	**79.94**	**68.02**	**79.45**	**66.56**	**69.72**	**57.28**	**68.87**	**55.51**	**73.59**	58.62	**73.10**	58.54

3) Evaluation Metrics: As we discussed before, the category label distributions of all the three datasets are long-tailed. To evaluate the performance on such long-tailed data, existing work always uses the following four metrics, including Macro F1 (MF1), Weighted F1 (WF1), Macro Accuracy (MacAcc), and Micro Accuray (MicAcc). The MF1 and MacAcc are two metrics to average the F1 score and accuracy for each class and are frequently used in evaluating long-tailed data performance. WF1 and MicAcc computes the F1 score and accuracy for the whole dataset and represent the general performance on the dataset without considering the label frequency.

D. Experimental Results

Table II lists the experimental results on three datasets in terms of four evaluation metrics. We can observe that on the three datasets, the PGMWN methods achieve the best performance in terms of all the four evaluation metrics, except the MaAcc score on the Beauty dataset, which is the second best score. Especially for the approach using the LDAM loss, among 12 scores, 10 scores are ranked the first place and 2 ranked the second place. These results confidently validate the effectiveness of the proposed PGMWN in handling the long-tailed noisy data issue in item categorization task.

Although inferior to some methods using the LDAM loss, the proposed PGMWN$_C$ framework outperforms all the baselines using the cross entropy loss except for the MaAcc on Electronics and Beauty datasets. The improvement over other methods using cross entropy loss of the proposed PGMWN$_C$ framework can also demonstrate the advantage of the proposed PGMWN framework on addressing long-tailed noisy data in item categorization task.

Another finding is that LDAM loss is powerful to deal with not only the long-tailed data issue but also the noisy label issue in the item categorization task and can be effectively plugged into different frameworks to help improve their performance.

Compared with the MWNet models and the DP model, which also reweight the samples, the superior performance of the proposed framework shows the advantage of the specially designed mechanism for handling the long-tailed noisy data issue in item categorization. Although SS+SL+LDAM considers addressing both issues, it's even beaten by the LDAM model in some cases, which shows that simply combining the methods addressing those two issues will not necessarily improve the performance.

E. Ablation Study

To investigate the influence of different modules in the proposed PGMWN framework, we conduct ablation studies by removing the key components respectively. We compare the proposed frameworks with their variants to investigate the effectiveness of their different components. The variants we compared are: PGMWN$_C$-SS and PGMWN$_L$-SS which remove the self-supervised representation learning component, PGMWN$_C$-PG and PGMWN$_L$-PG removing the prior guided weight regularization network, as well as MWNet$_C$ and MWNet$_L$, which are the models generated by removing both the self-supervised representation learning network and the prior guided item weight regularization network from the proposed PGMWN network.

TABLE III
THE RESULTS OF THE ABLATION STUDY ON LONG-TAILED NOISY DATA ITEM CATEGORIZATION ON THE THREE DATASETS: AUTOMOTIVE, ELECTRONICS AND BEAUTY. THE BEST RESULTS ARE HIGHLIGHTED USING BOLD FONTS, AND THE SECOND BEST RESULTS ARE UNDERLINED.

Dataset	Automotive				Electronics				Beauty			
Metric	MicAcc	MacAcc	$WF1$	$MF1$	MicAcc	MacAcc	$WF1$	$MF1$	MicAcc	MacAcc	$WF1$	$MF1$
MWNet$_C$	78.45	65.84	78.17	64.65	68.20	54.04	67.51	53.05	71.82	57.58	71.57	57.42
MWNet$_L$	78.74	67.43	78.57	65.59	68.78	56.42	68.12	54.90	73.32	57.95	72.65	57.80
PGMWN$_C$-SS	78.70	65.72	78.38	64.67	68.70	54.35	67.90	53.84	72.00	57.82	71.92	57.81
PGMWN$_L$-SS	78.82	67.45	78.62	65.74	68.92	56.78	68.32	54.58	73.15	57.33	72.82	57.94
PGMWN$_C$-PG	78.62	66.25	78.24	64.72	68.54	54.62	67.84	53.45	71.98	58.02	71.84	57.92
PGMWN$_L$-PG	78.65	67.42	78.82	65.56	68.80	56.73	68.37	54.62	73.31	57.25	72.89	57.32
PGMWN$_C$	79.17	67.25	78.70	65.19	68.92	55.81	68.54	54.21	72.64	58.54	72.35	**58.62**
PGMWNL	**79.94**	**68.02**	**79.45**	**66.56**	**69.72**	**57.28**	**68.87**	**55.51**	**73.59**	58.62	**73.10**	58.54

TABLE IV
EXAMPLES OF ITEMS WITH THE LARGEST AND SMALLEST WEIGHTS

	ID	Item Description	Category
Top	1	Built NY Charger Notebook Accessory Organizer Bag - Leaf Green	Bags, Cases & Sleeves
	2	Pyle PLVWH1 In-Car Infrared Dual-Channel Wireless Stereo Headphones	Car Headphones
	3	XO Vision Universal IR in Car Entertainment Wireless Foldable Headphones, Orange	Car Headphones
Bottom	4	Konica Minolta NP400 Li-ion Battery for Dimage A1, A2, 5D & 7D Digital Cameras	Camera Batteries
	5	Battery for Canon PowerShot SD980 IS Digital Camera [Camera]	Camera Batteries
	6	Pearstone Duo Battery Charger for Canon BP-808/809/819/827	Camera

The results are presented in Table III. We can observe that the proposed frameworks PGMWN$_C$ and PGMWN$_L$ outperform all their variations by either removing the prior guided weight regularization network (PGMWN$_{C(L)}$-PG) or self-supervised representation network (PGMWN$_{C(L)}$-SS). MWNet$_C$ and MWNet$_L$ are the reduced models of the PGMWN$_C$ and PGMWN$_L$, where both the self-supervised representation learning and prior guided weight regularization modules are removed. We can also find as more modules are removed, the performance becomes worse in general. The better performance of the PGMWN demonstrates the advantage of incorporating the prior guided the item weight regularization and self-supervised representation learning to address the long-tailed noisy data issues in the item categorization task.

F. Case Study

To analyze the learned item weights, a case study is conducted. We show and analyze representative samples from the sample set with the 20 largest and smallest weights. The representative samples are shown in Table IV. The items 1 to 3 are among the items with the largest weights. They are in "Bags, Cases & Sleeves" and "Car Headphones", respectively, which are among the least frequent categories and there are similar categories easily to be confused with those category. There is a category named "Cases" similar as the "Bags, Cases & Sleeves". For the "Car Headphones" category, there is also a category named "Headphones". Those two factors may lead to the samples to be assigned larger weights.

For the items with the smallest weights, two of them belong to the "Camera Batteries", which is among the most frequent categories. Thus, it should be not difficult for the classifier to capture the characteristics and classify them correctly. For the item 6, it seems to be a noisy sample assigned a wrong category. The distance between the item description is far away from the category description so that the weight learning network assigns a small weight to it. In such a way, it can boost the performance of the classifier.

VI. CONCLUSION

The long tail data distribution and label noise naturally occur in item categorization task. These two data bias related challenges prevent the model to achieve satisfactory performance due to insufficient high quality labeled data for supervision. To address the challenges, we propose the PGMWN framework based on [27]–[29] with two major innovations, including (a) using an extra self-supervised representation learning module to improve text representations and (b) utilizing prior guided regularization. The experimental results prove that the proposed PGMWN framework can deal with the long tail and noisy data issues in item categorization task and the case study shows that the proposed PGMWN framework can learn meaningful item weights. In particular, ablation studies show that the proposed innovations consistently provide more accurate instance weigh prediction compared to the method in [29].

For future work, the possible directions include: (1) application to other large-sized text classification tasks with a large

number of classes and labeling noise, (2) applications to other tasks in e-commerce domain such as attribute extraction, and (3) further improving the item weight learning network.

VII. ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for their valuable comments and helpful suggestions, and would like to thank Fenglong Ma and Yaqing Wang for their insightful comments. The first two authors also would like to thank the graduate internship program of Rakuten Institute of Technology. This work is supported in part by the US National Science Foundation under grant NSF IIS-1747614 and NSF IIS-2141037. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] K. Cao, C. Wei, A. Gaidon, N. Arechiga, and T. Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *arXiv preprint arXiv:1906.07413*, 2019.

[2] T. Castells, P. Weinzaepfel, and J. Revaud. Superloss: A generic loss for robust curriculum learning. *Advances in Neural Information Processing Systems*, 33:4308–4319, 2020.

[3] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[4] L. Chen, H. Chou, Y. Xia, and H. Miyake. Multimodal item categorization fully based on transformer. In S. Malmasi, S. Kallumadi, N. Ueffing, O. Rokhlenko, E. Agichtein, and I. Guy, editors, *Proceedings of the 4th Workshop on e-Commerce and NLP*, pages 111–115, Online, Aug. 2021. Association for Computational Linguistics.

[5] L. Chen and H. W. Chou. Utilizing cross-modal contrastive learning to improve item categorization BERT model. In S. Malmasi, O. Rokhlenko, N. Ueffing, I. Guy, E. Agichtein, and S. Kallumadi, editors, *Proceedings of the Fifth Workshop on e-Commerce and NLP (ECNLP 5)*, pages 217–223, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[6] L. Chen and H. Miyake. Label-guided learning for item categorization in e-commerce. In Y.-b. Kim, Y. Li, and O. Rambow, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers*, pages 296–303, Online, June 2021. Association for Computational Linguistics.

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

[8] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9268–9277, 2019.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[10] T. Gao, X. Yao, and D. Chen. SimCSE: Simple contrastive learning of sentence embeddings. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

[11] A. Ghosh and A. Lan. Contrastive learning improves model robustness under label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2703–2708, 2021.

[12] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *thirteenth international conference on artificial intelligence and statistics*, 2010.

[13] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[14] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31, 2018.

[15] R. He and J. McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pages 507–517, 2016.

[16] B. Kang, Y. Li, S. Xie, Z. Yuan, and J. Feng. Exploring balanced feature spaces for representation learning. In *International Conference on Learning Representations*, 2021.

[17] B. Kang, S. Xie, M. Rohrbach, Z. Yan, A. Gordo, J. Feng, and Y. Kalantidis. Decoupling representation and classifier for long-tailed recognition. *arXiv preprint arXiv:1910.09217*, 2019.

[18] S. Karthik, J. Revaud, and B. Chidlovskii. Learning from long-tailed data with noisy labels. *arXiv preprint arXiv:2108.11096*, 2021.

[19] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[20] J. Li, R. Socher, and S. C. Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. *arXiv preprint arXiv:2002.07394*, 2020.

[21] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[22] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52, 2015.

[23] H. Nguyen and D. Khatwani. Robust product classification with instance-dependent noise. *arXiv preprint arXiv:2209.06946*, 2022.

[24] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[26] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. 2019.

[27] M. Ren, W. Zeng, B. Yang, and R. Urtasun. Learning to reweight examples for robust deep learning. In *International conference on machine learning*, pages 4334–4343. PMLR, 2018.

[28] S. Saxena, O. Tuzel, and D. DeCoste. Data parameters: A new family of parameters for learning a differentiable curriculum. 2019.

[29] J. Shu, Q. Xie, L. Yi, Q. Zhao, S. Zhou, Z. Xu, and D. Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. *Advances in neural information processing systems*, 32, 2019.

[30] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[31] T. M. Tashu, S. Fattouh, P. Kiss, and T. Horváth. Multimodal e-commerce product classification using hierarchical fusion. In *2022 IEEE 2nd Conference on Information Technology and Data Science (CITDS)*, pages 279–284, 2022.

[32] K. Tayal, R. Ghosh, and V. Kumar. Model-agnostic methods for text classification with inherent noise. In *Proceedings of the 28th International Conference on Computational Linguistics: Industry Track*, pages 202–213, 2020.

[33] T. Wang, L. Chen, X. Zhu, Y. Lee, and J. Gao. Weighted contrastive learning with false negative control to help long-tailed product classification. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 574–580, 2023.

[34] T. Wei, J.-X. Shi, W.-W. Tu, and Y.-F. Li. Robust long-tailed learning under label noise. *arXiv preprint arXiv:2108.11569*, 2021.

[35] Y. Yang and Z. Xu. Rethinking the value of labels for improving class-imbalanced learning. *arXiv preprint arXiv:2006.07529*, 2020.

[36] Z. Zhang and M. Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.

[37] B. Zhou, Q. Cui, X.-S. Wei, and Z.-M. Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9719–9728, 2020.