Tencent ML-Images: A Large-Scale Multi-Label Image Database for Visual Representation Learning

BAOYUAN WU[1], **WEIDONG CHEN**[1], **YANBO FAN**[1], **YONG ZHANG**[1], **JINLONG HOU**[1],
JIE LIU[1], **AND TONG ZHANG**[2]

[1]Tencent AI Lab, Shenzhen 518057, China
[2]Department of Computer Science and Mathematics, Hong Kong University of Science and Technology, Hong Kong

Corresponding author: Yanbo Fan (fanyanbo0124@gmail.com)

ABSTRACT In existing visual representation learning tasks, deep convolutional neural networks (CNNs) are often trained on images annotated with single tag, such as ImageNet. However, single tag annotation cannot describe all important contents of one image, and some useful visual information may be wasted during training. In this work, we propose to train CNNs from images annotated with multiple tags, to enhance the quality of visual representation of the trained CNN model. To this end, we build a large-scale multi-label image database with 18M images and 11K categories, dubbed *Tencent ML-Images*. We efficiently train the ResNet-101 model with multi-label outputs on Tencent ML-Images, taking 90 hours for 60 epochs, based on a large-scale distributed deep learning framework, *i.e.*, TFplus. The good quality of the visual representation of the Tencent ML-Images checkpoint is verified through three transfer learning tasks, including single-label image classification on ImageNet and Caltech-256, object detection on PASCAL VOC 2007, and semantic segmentation on PASCAL VOC 2012. The Tencent ML-Images database, the checkpoints of ResNet-101, and all the training codes have been released at *https://github.com/Tencent/tencent-ml-images*. It is expected to promote other vision tasks in the research and industry community.

INDEX TERMS Visual representation learning, multi-label, image database.

I. INTRODUCTION

This work presents the large-scale visual representation learning on a newly built multi-label image database, dubbed *Tencent ML-Images*. We start from the discussions of the following two questions.

- Why we need large-scale image database? Deep learning had been in a long trough, until 2012 when AlexNet [1] shows surprising results on the single-label image classification task of ILSVRC2012 challenge [2]. The excellent potential of deep neural networks is released through the large-scale image database ImageNet [2]. Besides, the cost of acquiring training data for many visual tasks, such as object detection and semantic segmentation is very high. Due to the insufficient training data, they usually need certain pre-trained model with

good visual presentation on other large-scale database (*e.g.*, single-label image classification model trained on ImageNet) as initialization.

- Why we need multi-label image database? As there are multiple objects in most natural images, single-label annotation may miss some useful information and mislead the training of CNNs. For example, two visually similar images that include both *cow* and *grass* may be annotated as *cow* and *grass* separately. The reasonable approach is "telling" the CNN model that these two images contain both *cow* and *grass*.

The above discussions explain why we need large-scale multi-label image database for visual representation learning with deep neural networks. However, annotating one image with multiple tags is much more time-consuming than annotating one image with single tag, and it is difficult to control the annotation quality. To the best of our knowledge, the largest public multi-label image database is Open

The associate editor coordinating the review of this manuscript and approving it for publication was Michele Nappi.

Images [3], [4], which includes about 9 million images. Recently, Sun et al. [5] fine-tuned a ResNet-101 model [6] that pre-trained on JFT-300M (a multi-label image database with 300 million images), leading to 79.2% top-1 accuracy on the validation set of ImageNet. In contrast, the ResNet-101 model trained on ImageNet from scratch only gives 77.5% top-1 accuracy. However, the training on JFT-300M takes 2 months for 4 epochs, as the training size of JFT-300M is 200 times more than that of ImageNet. Besides, JFT-300M and its checkpoint are not publicly available.

In this work, we build a new large-scale multi-label image database, dubbed *Tencent ML-Images*. Instead of collecting new images from Google search or Flickr as did in other databases, we collect images from existing image databases, *i.e.*, Open Images [3] and ImageNet [2]. Specifically, we merge their class vocabularies into one unified vocabulary. We further remove rare and redundant categories as well as the corresponding images, and obtain about 18 million images with 11,166 categories. We then build semantic hierarchy of the unified vocabulary, according to semantic information extracted from WordNet [7]. We also derive the class co-occurrence among categories, which are then used together with the constructed semantic hierarchy to augment the annotations, based on the original annotations from Open Images and ImageNet.

To verify the quality of the built Tencent ML-Images, we conduct large-scale visual representation learning with the popular deep neural network ResNet-101. There are two main difficulties in the large-scale representation learning using the multi-label image database, including the severe class imbalance and the long training process. To alleviate the side-effect of class imbalance, we design a novel loss function that simultaneously considers weighted cross entropy, adaptive loss weight along the training process and down-sampling of negative training images in each mini-batch. To accelerate the training process, we utilize the large-scale distributed deep learning framework, *i.e.*, TFplus, with Message Passing Interface (MPI) and NVIDIA Collective Communications Library (NCCL) [8]. Consequently, the whole training process takes 90 hours of 60 epochs, using 128 GPUs. Furthermore, to verify the quality of visual representation of the ResNet-101 model pre-trained on Tencent ML-Images, we conduct transfer learning on three other vision tasks, including single-label image classification, object detection and semantic segmentation. We compare with the transfer learning using the checkpoints pre-trained on JFT-300M and ImageNet, respectively. The better transfer learning results using the checkpoint pre-trained on Tencent ML-Images demonstrate the good quality of Tencent ML-Images and the trained checkpoints.

The main contributions of this work are four-fold.

- We build a multi-label image database with 18M images and 11K categories, dubbed *Tencent ML-Images*, which is the largest publicly available multi-label image database until now.

- We efficiently train the ResNet-101 model on Tencent ML-Images, utilizing a large-scale distributed deep learning framework. Besides, we design a novel loss function to alleviate the side-effect of the severe class-imbalance in large-scale multi-label database.

- We demonstrate that the good quality of Tencent ML-Images and its pre-trained checkpoint through the transfer learning on three different vision tasks.

- We release the Tencent ML-Images database, the trained ResNet-101 checkpoints, as well as the complete codes of data preperation, pre-training and fine-tuning, at the GitHub address *https://github.com/Tencent/tencent-ml-images*. It is expected to promote other vision tasks for the research and industry community.

The organization of this manuscript is as follows. Related work is reviewed in section II. The built multi-label image database is introduced in section III, including the image source, class vocabulary, semantic hierarchy, tag augmentation and statistic informations. The visual representation learning on Tencent ML-Images is presented in section IV. Transfer learning to single-label image classification, object detection, and semantic segmentation are presented in section V, followed by the conclusion in section VI.

II. RELATED WORK

In this section, we review the image databases that are used for visual representation learning. They can be generally partitioned into two categories. One category is the single-label image database, where each image is annotated with only one tag. The other category is the multi-label image database, where each image is annotated with multiple tags.

Widely used single-label image databases include CIFAR-10 [9], Caltech-256 [10], MNIST [11], ImageNet [2], WebVision [12], SUN [13] and Places [14], etc. Before the deep learning era, the scales of most image databases are not very large. CIFAR-10 [9] includes 60K small-sized natural images with 10 categories. Caltech-256 [10] includes 30,607 images with 256 object categories. MNIST [11] includes 70K images of handwritten digits, from "0" to "9". In the deep learning era (from 2012), ImageNet [2] is the most popular database. Its first version that was used for ILSVRC 2012 includes 1.28M images and 1000 object categories. And now it has been expanded to 14M images. Many deep learning models (*e.g.*, AlexNet [1], VGG [15] and ResNet [6]) are trained and evaluated on ImageNet to demonstrate their performance, and the checkpoints pre-trained on ImageNet are widely used to help other vision tasks, such as image annotation, object detection, etc. WebVision [12] includes 2.4M images, with the same 1,000 object categories in ImageNet. The main difference between WebVision and ImageNet is that the annotations of WebVision are noisy, while the annotations of ImageNet are accurate. However, the authors of WebVision have experimentally demonstrated that the AlexNet trained on sufficient images with noisy labels has comparable or even better performance in visual

representation than those trained on ImageNet. In addition to above image databases of object categories, there are two popular databases of scene categories, including SUN and Places. SUN [13] includes 108,754 images, with 397 scene semantic categories. Places [14] includes 10M images, with 434 scene semantic categories. However, scene categories describe higher level information than object categories. The visual representation of the deep model trained on scene databases may not be suitable for other vision tasks like object recognition or detection. However, as mentioned in section I, the main contents of one image cannot be well described by a single label. Visual representation learning on single-label images will waste useful information of training images, and may bring in confusion to deep models, as two visually similar images could be annotated with two different categories.

There are also many multi-label image databases. Before the deep learning era, most multi-label image databases are used to evaluate multi-label models or image annotation methods. Some widely used databases include Corel 5k [16] (including 4,999 images with 260 object categories), ESP Game [17] (including 20,770 images with 268 categories), IAPRTC-12 [18] (including 19,627 images with 291 categories), NUSWIDE [19] (including 270K images with 81 categories), MS COCO [20] (including 330K images with 80 categories), and PASCAL VOC 2007 [21] (including 9,963 images with 2.47 averaged annotated tags per image). However, they are rarely used to train deep models for visual representation learning. Their scales are not big enough to train good parameters of popular deep models, such as VGG or ResNet. Besides, their small-scale category vocabularies are not diverse enough to train models with good generalization to other vision tasks. In contrast, there are also large-scale multi-label image databases. For example, Open Images-v1 [3] includes 9M image with 6K categories. JFT-300M is an ''internal dataset'' in Google, including 300M images and 18,291 categories, as well as 1.25 averaged annotated tags per image. Sun et al. [5] trained the ResNet-101 model on JFT-300M, and transferred the trained checkpoint to other vision tasks, including single-label image classification on ImageNet, object detection on MS-COCO and PASCAL VOC 2007, semantic segmentation on PASCAL VOC 2012, and human pose estimation on MS-COCO. Specifically, the checkpoint of ResNet-101 pre-trained on JFT-300M is fine-tuned on ImageNet, leading to 79.2% top-1 accuracy on the validation set of ImageNet. In contrast, the ResNet-101 model trained on ImageNet from scratch gives 77.5% top-1 accuracy. This improvement demonstrates that JFT-300M is helpful for learning more generalized visual representation. However, it is notable that the scale of JFT-300M is more than 200 times of ImageNet. Training ResNet-101 on 300M images with 18,291 categories is very costly. As reported in [5], their training process takes 2 months for 4 epochs, using ''asynchronous gradient descent training on 50 NVIDIA K80 GPUs and 17 parameter servers''. Moreover, JFT-300M and its checkpoints have not

been published. They cannot be utilized by the research community to help other vision tasks. In contrast, the built Tencent ML-Images is publicly available, and our training based on distributed training framework is much more efficient.

III. THE TENCENT ML-IMAGES DATABASE
A. IMAGE SOURCE AND CLASS VOCABULARY
The images and class vocabulary of Tencent ML-Images are collected from ImageNet [2] and Open Images [3]. In the following we introduce the construction of training set, validation set and class vocabulary, respectively.

Firstly, we extract image URLs from ImageNet-11k.[1] It is a subset of the whole database of ImageNet, collected by MXNet. It originally includes 11,797,630 training images, covering 11,221 categories. However, 1,989 categories out of 11,221 categories are very abstract in visual domain, such as *event* and *summer*. We think the images annotated with such abstract categories is helpless for visual representation learning. Thus, we remove these abstract categories and their corresponding images, with 10,322,935 images of 9,232 categories left. Moreover, according to the semantic relationship among categories, we add other 800 finer-grained categories from the whole vocabulary of ImageNet. For example, if *dog* is included in the above 9,232 categories, we also add *Husky* into the vocabulary of Tencent ML-Images, as well as the corresponding images from ImageNet. Consequently, we obtain 10,756,941 images, covering 10,032 categories from ImageNet. We randomly select 50,000 images as validation set, while ensuring that the number of selected images of each category is no larger than 5. On the other hand, the Open Images-v1 contains about 9M images and 6K categories. We filter all images of Open Images using a per-category criteria. If one category occurs in less than 650 images, then we remove this category. We also remove some abstract categories in visual domain as did above. Besides, as some categories from Open Images are similar to or synonyms of the above 10,032 categories, we merge these redundant categories into unique ones. If all tags of one image are removed, then this image is also abandoned. Consequently, 6,902,811 training images and 38,739 validation images are remained, covering 1,134 unique categories. Finally, we merge the selected images and categories from ImageNet and Open Images to construct the Tencent ML-Images database, which includes 17,609,752 training and 88,739 validation images, covering 11,166 categories.

B. TAG AUGMENTATION OF IMAGES
Note that each image from ImageNet-11K is originally annotated by a single tag. As analyzed above, there usually exists multiple contents in one nature image. Single tag annotation may not cover the whole content in each image, and thus misses some helpful information. Meanwhile, as the size of class vocabulary is very large, there may also exist missing

---

[1]Downloaded from http://data.mxnet.io/models/imagenet-11k/

tags for multi-label annotations in Open Images. Due to the large scale of images and class vocabulary, it is challenging and time-consuming to manually augment the tags for each image. We thus propose to augment the tags of these images by utilizing the semantic hierarchy and the co-occurrence among categories as follows.

We firstly map the categories of Tencent ML-Images to the WordIDs in WordNet. According to the WordIDs, we construct the semantic hierarchy among these 11,166 categories. It include 4 independent trees, of which the root nodes are *thing, matter, physical object* and *atmospheric phenomenon*, respectively. The length of the longest semantic path from root to leaf nodes is 16, and the average length is 7.47. The constructed semantic hierarchy captures the semantic relations among different categories and is used for tag augmentation. Specifically, according to the semantic hierarchy, all ancestor categories of the original tag are also annotated as positive tags of the same image. For example, if one image is originally annotated as "dog", we also label is as "animal". Secondly, we compute the co-occurrence matrix CO between categories from ImageNet-11k and categories from Open Images. Specifically, we train a ResNet-101 model with 1,134 outputs, based on Open Images. Using this trained model, we predict the labels among these 1,134 categories for the images from ImageNet-11k. If the posterior probability with respect to one category of one image is larger than 0.95, then we set this category as the positive tag of this image. Then, we compute the co-occurrence matrix as follows: for category i from ImageNet-11k and category j from Open Images, we denote the number of positive images of category i in ImageNet-11k as n_i, among which the number of images also annotated as category j is denoted as $n_{i,j}$, then $CO(i, j) = n_{i,j}/n_i \in [0, 1]$. If $CO(i, j) > 0.5$ and there is no semantic relationship between category i and j (*i.e.*, there is no path from i to j or reverse in the semantic hierarchy), then we determine that category i and j is a strongly co-occurrent pair of categories. Then, we augment the tags of images from ImageNet-11k as follows: if one image is originally annotated as i, then we also label it as category j. For example, if an image is originally annotated as "sea snake", then we also label it as "sea".

C. DATA STATISTICS
1) DISTRIBUTION OF ANNOTATIONS
The number of images per category is shown in Fig. 1. Specifically, the maximum number of images per category is 13,217,523, corresponding to the category "object, physical object"; the minimum number is 0; the average number is 13,843. The distributions of different categories are extremely imbalanced. Some categories are frequent, while many others are very rare. It is referred to as the imbalance among categories [22]. There are 10,505 trainable categories, of which the numbers of images are larger than 100. The statistics of the numbers of annotated tags of all training images are shown in Fig. 2. Specifically, the numbers of annotated tags
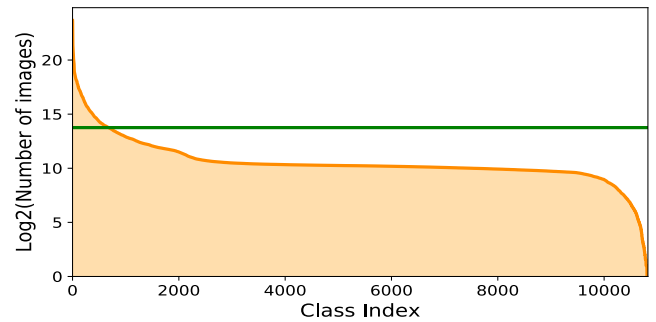

FIGURE 1. Number of images (log$_2$) per category in Tencent ML-Images. The green line indicates the average number of images of all categories.
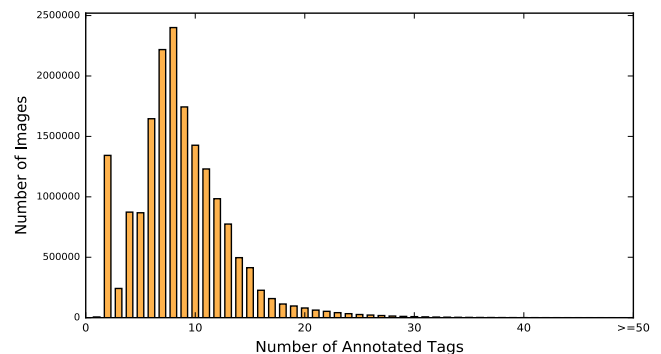

FIGURE 2. The statistics of the numbers of annotated tags of all training images in Tencent ML-Images.

of all training images range from 1 to 91, and the average number is 8.72. Considering the size of the label vocabulary (*i.e.*, 11K), the number of annotated tags per image is very small. In other words, the number of positive tags of each image is much smaller than the number of negative tags. It is referred to as the imbalance between positive and negative tags per image [22]. Above two types of imbalance bring in difficulty to model training. They will be considered during the training process of our model, as shown in section IV.

2) NOISY AND MISSING TAGS
Noisy tag indicates the incorrectly annotated tag. And missing tag [23]–[27] means that one class occurs in the image, but it is not annotated. As demonstrated in [3], the annotated tags for most images in Open Images are generated by machine, while the annotations of only a few fraction of images are verified by humans. The noisy annotations are unavoidable and they are also included in Tencent ML-Images. Most missing tags occur in images from ImageNet-11K, as they are originally annotated by single tag. As demonstrated in section III-B, we augment the tags of these single-label annotations by the category co-occurrence and semantic hierarchy. Compared to automatically generating tags by machine as did in Open Images, our augmentation is rather conservative. Our concern is that it is difficult to control the noise proportion of the machine-generated annotations, and we believe that the negative influence of noisy tags is larger than that of missing

ones. Some works (*e.g.*, [5], [28]) have demonstrated that learning from massive noisy labeled images is still able to show good visual representation. But they have not studied the trade-off between noisy and missing annotations, as the accurate proportions of both types of annotations are costly to calculate on large-scale databases. In this work, we choose the setting of more missing but less noisy annotations.

IV. VISUAL REPRESENTATION LEARNING ON TENCENT ML-IMAGES

A. TRAINING RESNET-101 WITH MULTI-LABEL OUTPUTS ON TENCENT ML-IMAGES

1) MODEL AND LOSS FUNCTION

For visual representation learning on Tencent ML-Images databse, we implement the popular ResNet-101 model. As our task is multi-label classification, the outputs of ResNet-101 are the activations of m independent Sigmoid functions, with m being the size of the class vocabulary. To alleviate the imbalance problems described in section III-C, we propose a novel weighted cross entropy loss function. For clarity, in the following, we present the loss function with respect to one training image $\mathbf{x}_i$:

$$\mathcal{L}_{\mathbf{W}}(\mathbf{x}_i, \mathbf{y}_i) = \frac{1}{m} \sum_{j}^{m} r_t^{y_{ij}} \left[-\eta y_{ij} \log(p_{ij}) - (1 - y_{ij}) \log(1 - p_{ij}) \right],$$

(1)

where $p_{ij} = f_{\mathbf{W}}(\mathbf{x}_i, j) \in [0, 1]$ denotes the posterior probability with respect to category j, with $\mathbf{W}$ being the trainable parameters. $\mathbf{y}_i = [y_{i1}, \ldots, y_{ij}, \ldots, y_{im}] \in \{0, 1\}^m$ indicates the ground-truth label vector of image $\mathbf{x}_i$.

- Due to the highly imbalance of Tencent ML-Images, for many categories, the number of positive images is far less than that of negative ones. The cost parameter $\eta > 1$ is thus introduced to set a larger cost on positive labels than negative labels, to alleviate the imbalance between positive and negative images in each category, which is a common strategy in imbalance learning [29]. In our experiments, η is set to 12.
- $r_t^{y_{ij}}$ denotes an adaptive weight during the training process. It is formulated as follows:

$$r_t^{y_{ij}} = \begin{cases} \max\{0.01, \log_{10}(\frac{10}{0.01 + t})\} \in [0.01, 1), & \text{if } y_{ij} = 1; \\ \max\{0.01, \log_{10}(\frac{10}{8 + t})\} \in [0.01, 0.1), & \text{if } y_{ij} = 0. \end{cases}$$

(2)

For category j, if all training images in one mini-batch are negative, then we record the status as 0; if at least one training image in this mini-batch is positive, then we record the status as 1. Consequently, we record a status vector like $(\ldots, 0, 1, 1, 1, 0, 0, 1, 0, \ldots)$. Then, t is defined as follows: if the status of the current mini-batch is different with that of the previous mini-batch, *i.e.*, 01 or 10, then $t = 1$; if the current status is same with the previous status, then $t = t + 1$.
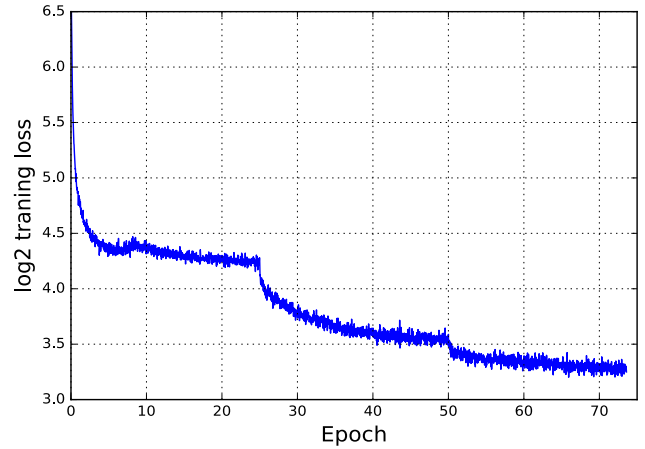

FIGURE 3. The curve of the $\log_2$ training loss of pre-training the ResNet-101 model on Tencent ML-Images.

With $r_t^{y_{ij}}$, if the parameters corresponding to category j are positively or negatively updated in sequential mini-batches, the weight of the corresponding loss is decayed. It helps to alleviate the imbalance between frequent and rare categories. Besides, as the positive sequential mini-batches is less frequent than the negative sequential mini-batches, we set $r_t^1 > r_t^0$ to alleviate the imbalance between positive and negative labels.

2) IMAGE PRE-PROCESSING

For image pre-processing, data augmentation and normalization are widely adopted in image classification to improve the generalization [1], [6], [30]. Following [30], our image pre-processing consists of the following six sequential steps.

1) Random crop a bounding box from input image, for which the box area is within [0.05, 1.0] of the whole image area, and the aspect ratio between its width and height is within $[\frac{3}{4}, \frac{4}{3}]$.
2) Resize the cropped box to 224×224.
3) Random flip the cropped box horizontally with probability of 0.5.
4) Random rotate the cropped box with probability of 0.25, and the rotation degree is evenly sampled from $[-45, 45]$.
5) Random shift the color with probability of 0.5.
6) Linearly rescale pixel value to $[-1, 1]$.

Besides, we use a relatively small value (*i.e.*, 0.05) for the lower bound of box area ratio in step-1 to include more small patches. We also add a random rotation (*i.e.*, step-4) for data augmentation. The range of rotation degree is set to [-45,45] experimentally.

3) TRAINING ALGORITHM AND HYPER-PARAMETERS

We adopt stochastic gradient descent (SGD) with momentum and back-propagation [31] to train the ResNet-101 model. There are 17,609,752 training images. To speed up the training process, we follow the "large minibatch SGD"
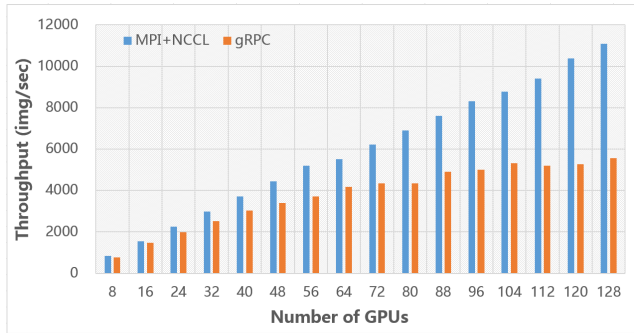
FIGURE 4. Throughout of distributed training based on MPI+NCCL and gRPC.

FIGURE 5. Scaling efficiency of each GPU in distributed training based on MPI+NCCL and gRPC.

and ''linear scaling rule'' proposed by [32]. Specifically, our batch-size is set to 4,096, and each epoch includes 4,300 steps. According to the ''linear scaling rule'', when the batch-size is multiplied by k, the learning rate should be multiplied by k. In our implementation, we adopt a reference learning rate of 0.01 for batch-size 512, and the learning rate is 0.08 for our large batch-size 4,096. Moreover, as the ''linear scaling rule'' may not hold in early epochs when the network is changing rapidly, warm-up strategy is proposed by [32]. The key idea of warm-up is to use less aggressive learning rates at the start of training. We adopt the gradual warm-up strategy in [32] that gradually ramps up the learning from a small (which is 0.01 in our implementation) to a desired value (which is 0.08 in our implementation). Our warm-up phase contains 8 epochs, and the learning rate increasing factor during warm-up is 1.297. After the warm-up phase, the learning rate reaches 0.08, and we go back to original learning rate decay schedule. That is, the learning rate decays with factor 0.1 in every 25 epochs. The maximal epoch number is 60. The momentum is 0.9. For the updating of the parameters of BatchNorm, the decay factor of moving average is 0.9, and the constant ϵ is set to 0.001 to avoid the 0 value of the variance. The weight decay is 0.0001.

In each batch, for many categories, due to the highly imbalance between positive and negative images, most training images are negative. For category i, if there are no positive images in the current batch, the model parameters corresponding to category i will only be updated with a probability of 0.1; if there are positive images for category i, we will down-sample negative images with the number of 5 times of positive images, then the model parameters are updated according to positive and down-sampled negative images. Besides, we set the cost parameter η to 12, *i.e.*, a higher cost for positive labels than negative ones. It helps to alleviate the negative influence of the imbalance between positive and negative images. The curve of the $\log_2$ training loss is shown in Fig. 3.

4) ACCELERATION BY DISTRIBUTED TRAINING

The training of Resnet-101 model with 11K categories and 18M images requires lots of computation. It will take a few dozen days if using one single GPU. In this work,
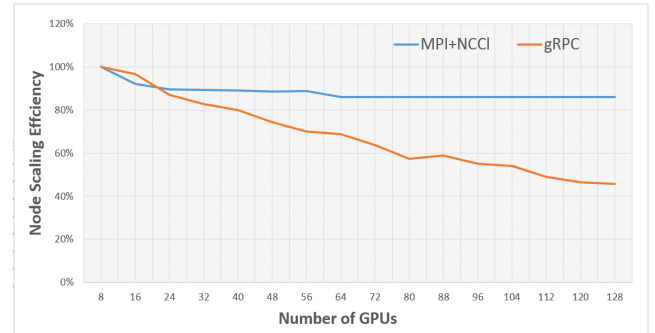
all training experiments are conducted on a large-scale distributed deep learning framework, *i.e.*, TFplus, which is built upon Tensorflow with several communication optimized techniques. We replace the original gRPC implementation with Message Passing Interface (MPI) and NVIDIA Collective Communications Library (NCCL) [8]. NCCL provides a highly optimized version of routines, such as all-gather, all-reduce, broadcast, reduce, reduce-scatter, and the integrated bandwidth-optimal ring all-reduce algorithm [33], to achieve high bandwidth over PCIe on NVIDIA GPU. In order to scale from one GPU to multiple nodes and multiple GPUs, we implement several APIs for communication: 1) a broadcast operation to synchronize parameters among all GPUs at the initialization stage or the recovery from the checkpoint; 2) a distributed optimizer wrapper for synchronization update of parameters; 3) some operations for data partition and barrier, etc. Since both MPI and NCCL support the remote direct memory access (RDMA), we run all distributed training jobs over a 40-GbE RDMA-capable networking. We achieve about $2\times$ speed up compared with the original gRPC based distributed implementation on a cluster of 16 nodes and each node with 8 NVIDIA M40 GPUs, as shown in Fig. 4. Specifically, when training with 128 GPUs, the throughout (*i.e.*, the number of processed images per second) of MPI+NCCL is up to 11077, while the throughout of gRPC is 5551. Besides, the distributed training jobs based on MPI+NCCL achieve 86% scaling efficiency from 8 to 128 GPUs, while those based on gRPC are only 46%, as shown in Fig. 5. The whole training process on Tencent ML-Images takes 90 hours for 60 epochs with 128 GPUs, *i.e.*, 1.5 hours per epoch.

B. EVALUATIONS AND RESULTS
To evaluate the performance of the trained ResNet-101 model with multi-label outputs, we adopt the widely used instance-level metrics in multi-label learning, including instance-level precision, recall and F1 score [34]. As the output for each category is the posterior probability, we need to transform the continuous predictions to binary predictions to calculate above metrics. Specifically, for image i, we determine the categories corresponding to top-k largest posterior probabilities as positive labels (*i.e.*, 1), while all other categories are negative labels (*i.e.*, 0). We obtain a binary prediction vector

TABLE 1. Results (%) of the ResNet-101 with multi-label outputs, evaluated on the validation set of Tencent ML-Images.

Evaluation metric → Top-k prediction ↓	Precision	Recall	F1
Top-5 prediction	35.7	18.2	23.3
Top-10 prediction	29.0	29.5	28.1

$\hat{\mathbf{y}}_i^k \in \{0, 1\}^m$. Then, the evaluation metrics are calculated as follows [34]:

$$P_k = \frac{1}{n} \sum_i^n P_{i,k} = \frac{1}{n} \sum_i^n \frac{\mathbf{y}_i * \hat{\mathbf{y}}_i^k}{k}, \qquad (3)$$

$$R_k = \frac{1}{n} \sum_i^n R_{i,k} = \frac{1}{n} \sum_i^n \frac{\mathbf{y}_i * \hat{\mathbf{y}}_i^k}{\mathbf{1}^\top * \mathbf{y}_i}, \qquad (4)$$

$$F1_k = \frac{1}{n} \sum_i^n \frac{2P_{i,k} * R_{i,k}}{P_{i,k} + R_{i,k}}. \qquad (5)$$

We present results of top-5 and top-10 predictions in Table 1. The evaluation values are not very high. As demonstrated in section III-A, the size of the validation set is only about $\frac{1}{200}$ of the size of the training set. And there should be many missing labels for validation images. Thus, the evaluation scores on this small validation set are not reliable enough to measure the visual representation performance of the model trained on Tencent ML-Images. Instead, its performance could be evaluated through transfer learning to some other visual tasks, as follows.

V. TRANSFER LEARNING
A. TRANSFER LEARNING TO SINGLE-LABEL IMAGE CLASSIFICATION ON IMAGENET
To verify the quality of the visual representation of the ResNet-101 model pre-trained on Tencent ML-Images, we conduct transfer learning to image classification on the benchmark single-label image database, *i.e.*, ImageNet. Specifically, we utilize the ResNet-101 model pre-trained on Tencent ML-Images as the initial checkpoint, and replace the output layer with 1,000 output nodes, as well as the loss with standard softmax loss. Then, we fine-tune the model on ImageNet. Experimental results are given below.

1) FINE-TUNING APPROACHES
a: LEARNING RATE
Note that there are significant differences between Tencent ML-Images and ImageNet. First, the distributions of visual features and class vocabulary are different. Second, images in Tencent ML-Images are annotated with multiple tags, while images in ImageNet are annotated with single label. Last, the annotations of Tencent ML-Images are noisy, while the annotations of ImageNet are clean. Considering these significant differences, one cannot expect that the ResNet-101 model pre-trained on Tencent ML-Images show good classification performance on ImageNet, if without fine-tuning of the parameters. The standard fine-tuning approach adopts one consistent learning rate of all layers.

TABLE 2. Hyper-parameters of different checkpoints in transfer learning to ImageNet. "ckpt" denotes our ResNet-101 checkpoint. ckpt-1 means the checkpoint trained on ImageNet from scratch; ckpt-2 indicates the checkpoint pre-trained on Tencent ML-Images and fine-tuned on ImageNet with layer-wise consistent learning rates; ckpt-3/4/5 represent three checkpoints pre-trained on Tencent ML-Images and fine-tuned on ImageNet with layer-wise adaptive learning rates, and they are different at the image size in fine-tuning. "LR" denotes learning rate. ResNet-101 consists of 4 stages of residual blocks, and top2-stages indicates the two stages close to the output layer, while bottom2-stages represents those close to the input layer. Note that if there is only one value in a row, then it means that the checkpoints of different columns are trained with the same value of the corresponding hyper-parameter.

Hyper-parameters	ckpt-1	ckpt-2	ckpt-3/4/5
Batch size		2048	
Maximum epoch		120	
LR-top2-stages	0.8	0.008	0.8
LR-bottom2-stages	0.8	0.008	0
LR-decay-factor		0.1	
LR-decay-step		18750 (30 epochs)	
Weight-decay		0.0001	
warmup-steps		2500 (the first 4 epochs)	
LR-warmup	0.1	0.001	0.1
LR-warmup-decay-factor		1.681	
LR-warmup-decay-step		625 (one epoch)	
BatchNorm-decay		0.9	
BatchNorm-eps		0.001	

It is referred to as fine-tuning with layer-wise consistent learning rates. However, as verified in later experiments, the ResNet-101 model with the above fine-tuning approach even shows worse performance than the ImageNet baseline. It reveals that the useful information contained in the Tencent ML-Images checkpoint has not been well utilized, due to the aforementioned three differences. To tackle this issue, we adopt the fine-tuning approach with layer-wise adaptive learning rates. Specifically, we set larger learning rates on top layers, while smaller learning rates on bottom layers. The rationale behind this setting is that the parameters of top-layers are more dependent on training images and labels, while the parameters in bottom layers represent low-level visual features. To alleviate the negative influence of the significant differences between Tencent ML-Images and ImageNet, the top-layers' parameters should be changed to be further from the initialized parameters from the checkpoint, compared to the bottom layers' parameters.

b: IMAGE SIZE
As demonstrated in [35], the image size in training and testing has a significant influence to the results. Further, YOLO9000 [36] proposes to adjust the image size during training, which is proved to be helpful for the detection performance. Besides, we note that the image size of pre-training, fine-tuning and evaluation used in [5] is 299×299, while the image size in our pre-training is 224×224. To conduct a fair comparison with the JFT-300M, and also to explore the influence of image size to image classification, we design three fine-tuning settings with different image sizes, including: 1) the image size is kept at 224×224; 2) the image size in early epochs is set as 224×224, while 299×299 in late epochs; 3) the image size is kept at 299×299.

2) COMPARISONS AND HYPER-PARAMETERS

To verify the quality of visual representation of the Tencent ML-Images checkpoint, we compare five ResNet-101 checkpoints with different fine-tuning approaches. The hyper-parameters of training these checkpoints are summarized in Table 2. Besides, we also present the reported results of others' implementations, including: the MSRA checkpoint of training on ImageNet from scratch; the Google's checkpoint of training on ImageNet from scratch [5]; the checkpoint of pre-training on JFT-300M and fine-tuning on ImageNet [5]. Please refer to Table 3.

3) RESULTS

All compared results are shown in Table 3. **(1)** In terms of the baseline (*i.e.*, train on ImageNet from scratch), our ckpt-1 is higher than both MSRA ckpt [6] and Google ckpt-1 [5]. Our implementation and MSRA implementation are same at the model architecture and the size of the input image (*i.e.*, 224 × 224). The main difference is the pre-processing of the input image (see section IV-A.2) and hyper-parameters, which should be the main reasons of different performance of this two baselines. In contrast, the details of the model architecture and the image pre-processing are not demonstrated in [5], and the image size is 299 × 299. But our ckpt-1 still performs better than Google ckpt-1 on the evaluation with the size of 299 × 299. It demonstrates the good quality of our implementation of the baseline. **(2)** Moreover, the comparison between our baseline checkpoint (*i.e.*, our ckpt-1 in Table 3) and our fine-tuning checkpoints (*i.e.*, our ckpt-2/3/4/5 in Table 3) demonstrate two points. The accuracy of our ckpt-2 is much lower than that of our ckpt-1. As analyzed in section V-A.1, the significant difference between Tencent ML-Images and ImageNet, as well as the label noises in Tencent ML-Images, could bring in negative influence to the model performance. In contrast, our ckpt-3 with the layer-wise adaptive fine-tuning learning rate shows the improvements of 1.0% at top-1 accuracy and 0.6% at top-5 accuracy under validation size of 224 × 224. This demonstrates that the fine-tuning with layer-wise adaptive learning rate can not only utilize the good visual representation encoded in bottom layers of the Tencent ML-Images checkpoint, but also alleviate the significant difference between these two databases. It reveals that the Tencent ML-Images checkpoint includes good visual representation, but it should be carefully fine-tuned to help other vision tasks. We also evaluate the checkpoints of different epochs using top-1 accuracy on the validation set of ImageNet, as shown in Fig. 6. **(3)** The results of Google ckpt-2 are 79.2% top-1 accuracy and 94.7% top-5 accuracy. The improvements over Google ckpt-1 are 1.7% at top-1 accuracy and 0.8% at top-5 accuracy. It demonstrates the good quality of the initial checkpoint pre-trained on JFT-300M. In contrast, all of our ckpt-3/4/5 show higher top-1 accuracies than Google ckpt-2 when the image size of validation set is set to 299 × 299. And, our ckpt-4 with adaptive image sizes in fine-tuning achieves the highest 80.73%
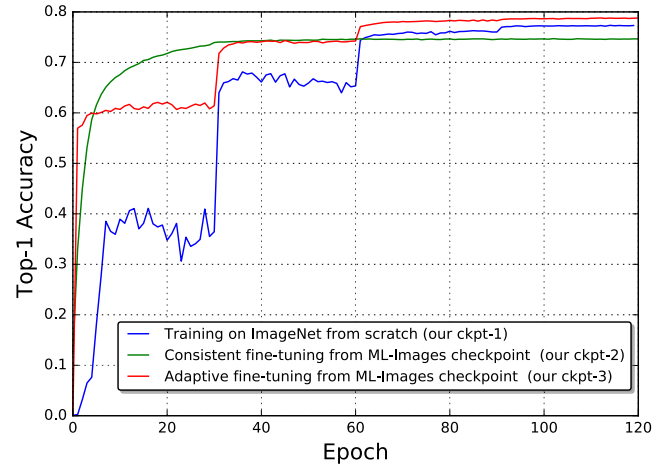

FIGURE 6. The curves of top-1 accuracy of different checkpoints on the validation set of ImageNet.

top-1 and 95.5% top-5 accuracy, and the improvements over our ckpt-1 are 1.73% at top-1 and 1% at top-5 accuracy. Our checkpoints exceed Google ckpt-2 on both the accuracy and the accuracy improvement over baseline. Considering that the size of JFT-300M is about 17 times of Tencent ML-Images, these verify the high quality of Tencent ML-Images and our training and fine-tuning.

B. TRANSFER LEARNING TO CALTECH-256

We also conduct transfer learning to another small-scale single-label image database, *i.e.*, Caltech-256 [10], which includes 30,607 images with 256 object categories. We utilize a pre-trained checkpoint of the ResNet-101 model to extract features for each image of Caltech-256. Specifically, we adopt the output of the global average pooling in ResNet-101 as the feature vector (2,048 dimensions). Then we train a multi-category SVM classifier to predict the labels for each image. We compare with three checkpoints, including: training on ImageNet from scratch; training on Tencent ML-Images from scratch; pre-training on Tencent ML-Images, and fine-tuning on ImageNet with layer-wise adaptive learning rate. The results of different checkpoints are shown in Table 4. The result of the ImageNet checkpoint is higher than that of the Tencent ML-Images checkpoint. It again demonstrates that the significant difference of distribution between Tencent ML-Images and single-label image databases. The checkpoint of adaptive fine-tuning from Tencent ML-Images gives 86.5% accuracy, which is higher than 86% accuracy of the ImageNet checkpoint. It also verifies that the good generalization of visual representation of the Tencent ML-Images could be well explored through adaptive fine-tuning.

C. TRANSFER LEARNING TO OBJECT DETECTION

We conduct transfer learning to object detection on the benchmark PASCAL VOC database [21], [37], including 20 categories. To fairly compare with [5], we also adopt the "trainval" images from both PASCAL VOC 2007 and

TABLE 3. Results (%) of single-label image classification on the validation set of ImageNet, using single model and single crop inference. In the column of "Train and fine-tune settings", "224" indicates that the size of training image is 224 × 224, while "299" indicating 299 × 299; "224 to 299" means that the size of training image in early epochs is 224 × 224, and that in later epochs is 299 × 299. For Our ckpt-3/4/5, we adopt the layer-wise adaptive learning rate in fine-tuning.

ResNet-101 checkpoints	Training and fine-tuning settings	Validation size 224		Validation size 299	
		top-1 acc	top-5 acc	top-1 acc	top-5 acc
MSRA ckpt [6]	Train on ImageNet, 224	76.4	92.9	–	–
Google ckpt-1 [5]	Train on ImageNet, 299	–	–	77.5	93.9
Our ckpt-1	Train on ImageNet, 224	77.8	93.9	79.0	94.5
Google ckpt-2 [5]	Train on JFT-300M, fine-tune on ImageNet, 299	–	–	79.2	94.7
Our ckpt-2	Pre-train on Tencent ML-Images, fine-tune on ImageNet with layer-wise consistent learning rate	74.7	92.6	–	–
Our ckpt-3	Pre-train on Tencent ML-Images, fine-tune on ImageNet	**78.8**	**94.5**	79.5	94.9
Our ckpt-4	Pre-train on Tencent ML-Images, fine-tune on ImageNet 224 to 299	78.3	94.2	**80.73**	**95.5**
Our ckpt-5	Pre-train on Tencent ML-Images, fine-tune on ImageNet 299	75.8	92.7	79.6	94.6

TABLE 4. Classification accuracy (%) of transfer learning to Caltech-256. Checkpoints 1 to 3 respectively indicate: training on ImageNet from scratch; training on Tencent ML-Images from scratch; pre-training on Tencent ML-Images, and fine-tuning on ImageNet with a layer-wise adaptive learning rate.

Checkpoints	Checkpoint 1	Checkpoint 2	Checkpoint 3
Accuracy	86.0	83.6	86.5

TABLE 5. Results of object detection on the testing set of PASCAL VOC 2007. Note that "VOC" in "fine-tune on VOC" indicates the combined training set of PASCAL VOC 2007 and 2012.

Checkpoints	Pre-training and fine-tuning settings	mAP@0.5
Google ckpt-1	Pre-train on ImageNet, fine-tune on VOC	76.3
Our ckpt-1	Pre-train on ImageNet, fine-tune on VOC	80.1
Google ckpt-2	Pre-train on JFT-300M, fine-tune on VOC	81.4
Google ckpt-3	Pre-train on JFT-300M+ImageNet, fine-tune on VOC	81.3
Our ckpt-2	Pre-train on Tencent ML-Images, fine-tune on ImageNet, and then fine-tune on VOC	**81.5**

2012 as the training set, including 16,551 training images. All models are evaluated on the testing set of PASCAL VOC 2007, including 4,952 images. We use mean average precision at 50% IOU threshold (mAP@.5) for performance evaluation.

1) COMPARISONS

We compare with the transfer learning did in [5], including their baseline checkpoint pre-trained on ImageNet, and their checkpoints pre-trained on JFT-300M and JFT-300M+ImageNet. Our first checkpoint (i.e., our ckpt-1) is also pre-trained on ImageNet from scratch, then fine-tuned on VOC. Our second checkpoint (i.e., our ckpt-2) is firstly pre-trained on Tencent ML-Images, then fine-tuned on ImageNet (using the same setting with "Our ckpt-3" in Table 3), and further fine-tuned on VOC.

2) IMPLEMENTATION DETAILS

Our implementation is based on the TensorFlow implementation of the Faster RCNN framework [38], [39]. Specifically, we use stochastic gradient descent with the momentum of 0.9 for training; the initial learning rate is set to 8×10^{-4} and decays by 0.1 at every 80k steps; the batch size is set to 256; the model is trained for 180k steps; the weight decay is set to 10^{-3}. The input image is resized such that the short side is fixed to 600-pixels, while maintaining the aspect ratio.

3) RESULTS

The results are summarized in Table 5. In comparison of the baseline checkpoints, our ckpt-1 achieves 80.1%, while Google ckpt-1 gives 76.3%. With the same databases and models, it demonstrates that our implementation quality is much better than Google's implementation. Our ckpt-2 shows the improvement of 1.4% than our ckpt-1. This verifies

the good visual representation of the checkpoint pre-trained on Tencent ML-Images. In contrast, Google ckpt-2 and ckpt-3 show 81.4% and 81.3%, respectively, which are slightly lower than our ckpt-2. It demonstrates that the checkpoint pre-trained on Tencent ML-Images and fine-tuned on ImageNet has similar generalization with those pre-trained on JFT-300M and JFT-300M+ImageNet. Considering that JFT-300M is about 17 times larger than our Tencent ML-Images, we could claim that Tencent ML-Images is a high-quality database. We also tried the checkpoint that is pre-trained on Tencent ML-Images, then fine-tuned on VOC. But the performance is not as good as other checkpoints, thus we didn't report its results here. We think the reason is that the big gap of the data distributions and tasks, between Tencent ML-Images and VOC.

D. TRANSFER LEARNING TO SEMANTIC SEGMENTATION

We conduct transfer learning to semantic segmentation on the benchmark PASCAL VOC 2012 database [37], including 20 foreground categories and 1 background category. To fairly compare with [5], we also adopt the augmented training set of PASCAL VOC 2012 as the training set of fine-tuning, which includes 10,582 training images. All models are evaluated on the validation set of PASCAL VOC 2012, including 1,149 images, using the mean itersection-over-union (mIOU) metric.

1) COMPARISONS

We adopt the same setting of checkpoints as did in the above object detection experiments (see section V-C). Thus, here we didn't repeat it to keep clarity.

TABLE 6. Results of semantic segmentation on the validation set of PASCAL VOC 2012. Note that "VOC" in "fine-tune on VOC" indicates the augmented training set of PASCAL VOC 2012.

Checkpoints	Pre-training and fine-tuning settings	mIOU
Google ckpt-1	Pre-train on ImageNet, fine-tune on VOC	73.6
Our ckpt-1	Pre-train on ImageNet, fine-tune on VOC	74.0
Google ckpt-2	Pre-train on JFT-300M, fine-tune on VOC	75.3
Google ckpt-3	Pre-train on JFT-300M+ImageNet, fine-tune on VOC	**76.5**
Our ckpt-2	Pre-train on Tencent ML-Images, fine-tune on ImageNet, and then fine-tune on VOC	76.3

2) IMPLEMENTATION DETAILS

Our implementation is based on the semantic segmentation architecture of DeepLab [40]. For fair comparison, our implementation also adopts DeepLab-ASPP-L structure that has four branches after the Conv5 block of ResNet-101 model. All ASPP branchs use 3×3 kernels but with different atrous rates (*i.e.*, {6, 12, 18, 24}). During training, we use the "poly" learning rate policy (with power $= 0.9$) and the initial learning rate is set to 3×10^{-3}. The weight decay is set to 5×10^{-4}. The model is trained for 50k steps using the stocastic gradient descent with momentum of 0.9. The batch size is set to 6, and the input image is resized to 513×513.

3) RESULTS

The results are summarized in Table 6. We could obtain the similar observations with the transfer learning to object detection (see section V-C and Table 5). 1) Our implementation of the baseline checkpoint is better than that of Google, *i.e.*, our ckpt-1 74.0% vs. Google ckpt-1 73.6%. 2) Our ckpt-2 shows the improvement up to 2.3% over our ckpt-1, which verifies the good visual representation of the checkpoint pre-trained on Tencent ML-Images. 3) Google ckpt-2 shows 75.3% and Google ckpt-3 gives 76.5%. It demonstrates that the checkpoint pre-trained on Tencent ML-Images and fine-tuned on ImageNet has better generalization than that pre-trained on JFT-300M, while is similar with the checkpoint pre-trained on JFT-300M+ImageNet.

VI. CONCLUSION

In this work, we built a large-scale multi-label image database, dubbed *Tencent ML-Images*, including about 18M images and 11K categories. It is the largest-scale public multi-label image database until now. We presented the lage-scale visual presentation learning of deep convolutional neural networks on Tencent ML-Images, employing a distributed training framework with MPI and NCCL. A novel loss function was carefully designed to alleviate the side-effect of the severe class imbalance in the large-scale multi-label database. Extensive experiments of transfer learning to other visual tasks, including single-label image classification, object detection and semantic segmentation, verified that Tencent ML-Images is of very high quality and the pre-trained checkpoint has very good visual representation. We hope that this work could promote other visual tasks in the research and industry community. The Tencent

ML-Images database, the complete code of data preparation, pre-training and fine-tuning, and the pre-trained and fine-tuned checkpoints of the ResNet-101 model have been released at *https://github.com/Tencent/tencent-ml-images*.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. CVPR*, 2009, pp. 248–255.

[3] I. Krasin, T. Duerig, N. Alldrin, V. Ferrari, S. Abu-El-Haija, A. Kuznetsova, H. Rom, J. Uijlings, S. Popov, A. Veit, S. Belongie, V. Gomes, A. Gupta, C. Sun, G. Chechik, D. Cai, Z. Feng, D. Narayanan, and K. Murphy. (2017). *OpenImages: A Public Dataset for Large-Scale Multi-Label and Multi-Class Image Classification*. [Online]. Available: https://github.com/openimages

[4] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, V. Ferrari, and T. Duerig, "The open images dataset V4: Unified image classification, object detection, and visual relationship detection at scale," 2018, *arXiv:1811.00982*. [Online]. Available: https://arxiv.org/abs/1811.00982

[5] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," in *Proc. ICCV*, 2017, pp. 843–852.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016, pp. 770–778.

[7] C. Fellbaum, *WordNet*. Hoboken, NJ, USA: Wiley, 1998.

[8] NVIDIA. (2017). *NVIDIA Collective Communications Library (NCCL)*. https://developer.nvidia.com/nccl

[9] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009, vol. 1, no. 4.

[10] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Inst. Technol., Pasadena, CA, USA, Tech. Rep. 7694, 2007.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[12] W. Li, L. Wang, W. Li, E. Agustsson, and L. Van Gool, "Web-Vision database: Visual learning and understanding from Web data," 2017, *arXiv:1708.02862*. [Online]. Available: https://arxiv.org/abs/1708.02862

[13] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "SUN database: Large-scale scene recognition from abbey to zoo," in *Proc. CVPR*, 2010, pp. 3485–3492.

[14] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1452–1464, Jun. 2018.

[15] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," 2014, *arXiv:1405.3531*. [Online]. Available: https://arxiv.org/abs/1405.3531

[16] P. Duygulu, K. Barnard, J. F. G. de Freitas, and D. A. Forsyth, "Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary," in *Proc. ECCV*, 2002, pp. 97–112.

[17] L. von Ahn and L. Dabbish, "Labeling images with a computer game," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, 2004, pp. 319–326.

[18] M. Grubinger, P. Clough, H. Müller, and T. Deselaers, "The IAPR TC-12 benchmark: A new evaluation resource for visual information systems," in *Proc. Int. Workshop OntoImage*, 2006, pp. 13–23.

[19] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, "NUS-WIDE: A real-world Web image database from National University of Singapore," in *Proc. CIVR*, 2009, pp. 368–375.

[20] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. ECCV*, 2014, pp. 740–755.

[21] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Sep. 2009.

[22] B. Wu, S. Lyu, and B. Ghanem, "Constrained submodular minimization for missing labels and class imbalance in multi-label learning," in *Proc. AAAI*, 2016, pp. 2229–2236.

[23] B. Wu, Z. Liu, S. Wang, B.-G. Hu, and Q. Ji, "Multi-label learning with missing labels," in *Proc. ICPR*, 2014, pp. 1964–1968.

[24] B. Wu, S. Lyu, and B. Ghanem, "ML-MG: Multi-label learning with missing labels using a mixed graph," in *Proc. ICCV*, 2015, pp. 4157–4165.

[25] B. Wu, S. Lyu, B.-G. Hu, and Q. Ji, "Multi-label learning with missing labels for image annotation and facial action unit recognition," *Pattern Recognit.*, vol. 48, no. 7, pp. 2279–2289, Jul. 2015.

[26] Y. Li, B. Wu, B. Ghanem, Y. Zhao, H. Yao, and Q. Ji, "Facial action unit recognition under incomplete data based on multi-label learning with missing labels," *Pattern Recognit.*, vol. 60, pp. 890–900, Dec. 2016.

[27] B. Wu, F. Jia, W. Liu, B. Ghanem, and S. Lyu, "Multi-label learning with missing labels using mixed dependency graphs," *Int. J. Comput. Vis.*, vol. 126, no. 8, pp. 875–896, 2018.

[28] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. Belongie, "Learning from noisy large-scale datasets with minimal supervision," in *Proc. CVPR*, 2017, pp. 839–847.

[29] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.

[30] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. CVPR*, 2015, pp. 1–9.

[31] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Cognit. Model.*, vol. 5, no. 3, p. 1, 1988.

[32] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, large minibatch SGD: Training imagenet in 1 hour," 2017, *arXiv:1706.02677*. [Online]. Available: https://arxiv.org/abs/1706.02677

[33] P. Patarasuk and X. Yuan, "Bandwidth optimal all-reduce algorithms for clusters of workstations," *J. Parallel Distrib. Comput.*, vol. 69, no. 2, pp. 117–124, Feb. 2009.

[34] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1819–1837, Aug. 2014.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. ECCV*, 2016, pp. 630–645.

[36] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. CVPR*, 2017, pp. 7263–7271.

[37] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes challenge: A retrospective," *Int. J. Comput. Vis.*, vol. 111, no. 1, pp. 98–136, Jan. 2014.

[38] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. NIPS*, 2015, pp. 91–99.

[39] X. Chen and A. Gupta, "Spatial memory for context reasoning in object detection," in *Proc. ICCV*, 2017, pp. 4086–4096.

[40] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2017.

WEIDONG CHEN received the B.S. degree in computer science and technology from the Harbin Institute of Technology, Harbin, China, in 2013, and the M.S. degree from the Harbin Institute of Technology, Shenzhen, China, in 2016. He is currently a Senior Researcher with the Tencent AI Lab. His research interests are computer vision and machine learning.

YANBO FAN received the B.S. degree in computer science and technology from Hunan University, in 2013, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences (CASIA), Beijing, China, in 2018. He is currently a Senior Researcher with the Tencent AI Lab. His research interests are computer vision and machine learning.

YONG ZHANG received the Ph.D. degree in pattern recognition and intelligent systems from the Institute of Automation, Chinese Academy of Sciences, in 2018. From 2015 to 2017, he was a Visiting Student with the Rensselaer Polytechnic Institute. He is currently with the Tencent AI Lab. His research interests include computer vision, machine learning, and probabilistic graphical models.

JINLONG HOU received the B.S. degree from the Huazhong University of Science and Technology, in 2007, and the M.S. degree from Shanghai Jiao Tong University, in 2010. He has been a Senior Researcher with the Tencent AI Lab, since 2018. His research interests include machine learning and computer vision.

JIE LIU received the B.S. degree from Hunan University, in 2016, and the M.S. degree from Shanghai Jiao Tong University, in 2019. Before graduation, he was a Research Intern with the Tencent AI Lab. His research interests include computer version and recommender systems.

TONG ZHANG received the B.A. degree in mathematics and computer science from Cornell University and the Ph.D. degree in computer science from Stanford University. He is currently a Machine Learning Researcher. Previously, he was a Professor with Rutgers University, and worked at IBM, Yahoo, Baidu, and Tencent. His research interests include machine learning algorithms and theory, statistical methods for big data and their applications. His research has been supported by many grants from funding agencies, such as NSF and NIH. He is a Fellow of ASA and IMS. He has been in the editorial boards of leading machine learning journals and program committees of top machine learning conferences.

BAOYUAN WU received the Ph.D. degree from the National Laboratory of Pattern Recognition, Chinese Academy of Sciences (CASIA), in 2014. He held a postdoctoral position at the IVUL Lab, KAUST, from August 2014 to November 2016. He has been a Principal Researcher with the Tencent AI Lab, since November 2016. His work has been published in TPAMI, IJCV, CVPR, ICCV, and ECCV. His research interests are machine learning, computer vision, and optimization.

• • •