This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the version available on IEEE Xplore.

Online Collaborative Learning for Open-Vocabulary Visual Classifiers

Hanwang Zhang[†], Xindi Shang[†], Wenzhuo Yang[†], Huan Xu[†], Huanbo Luan[‡], Tat-Seng Chua[†]
[†]National University of Singapore, [‡]Tsinghua University
{hanwangzhang,xindi1992,luanhuanbo}@gmail.com; {wenzhuo,mpexuh,dcscts}@nus.edu.sg

Abstract

We focus on learning open-vocabulary visual classifiers, which scale up to a large portion of natural language vocabulary (e.g., over tens of thousands of classes). In particular, the training data are large-scale weakly labeled Web images since it is difficult to acquire sufficient well-labeled data at this category scale. In this paper, we propose a novel online learning paradigm towards this challenging task. Different from traditional N-way independent classifiers that generally fail to handle the extremely sparse and inter-related labels, our classifiers learn from continuous label embeddings discovered by collaboratively decomposing the sparse image-label matrix. Leveraging on the structure of the proposed collaborative learning formulation, we develop an efficient online algorithm that can jointly learn the label embeddings and visual classifiers. The algorithm can learn over 30,000 classes of 1,000 training images within 1 second on a standard GPU. Extensively experimental results on four benchmarks demonstrate the effectiveness of our method.

1. Introduction

In recent years, we have witnessed the impressive progress of visual classifiers that help to move a large variety of visual applications from academic prototypes into industrial products [27, 41]. However, when we communicate with vision systems using natural language, those classifiers with a predefined vocabulary (*e.g.*, ImageNet [35]) generally fail due to vocabulary discrepancy and scarcity. For example, "dolphin" is usually used instead of "grampus griseus" [33] and adjective classifiers like "romantic" or "exciting" are usually scarce as compared to nouns [28, 19]. Recently, much attention has been paid to scale up visual classifiers to *open-vocabulary*, which covers the full range of vocabulary in natural language [18, 16, 46].

One major limitation of the scale-up is the difficulty in acquiring sufficient well-labeled datasets with many classes, *e.g.*, even the full ImageNet only contains images

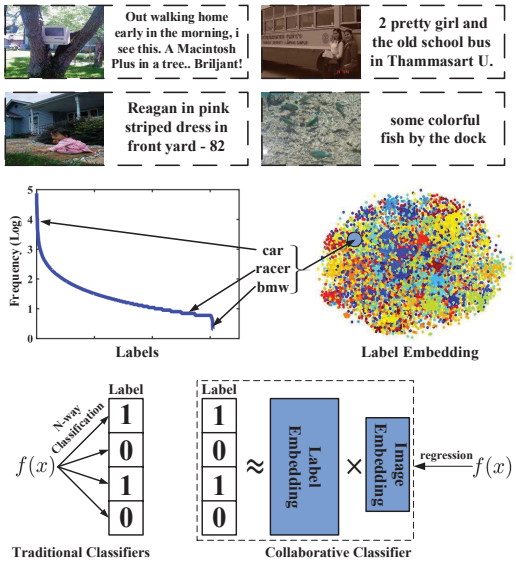


Figure 1. **Top**: Web images are usually noisy and incomplete. These examples are from SBU 1M Flickr image dataset [34], which is used as the training data. **Middle**: The power-law distribution of labels suggests that the images are sparsely labeled. We transform such extreme sparsity to continuous label space that preserve semantic relations. Different colors suggest various semantic clusters by K-means. **Bottom**: As compared to traditional classifiers, ours are learned from the latent embeddings.

of around 20K nouns. Fortunately, a promising method to acquire more labels has been recently studied on learning concepts from millions of noisy Web images. This method utilizes user-generated textual descriptions as labels and can naturally accommodate an open vocabulary [34, 29, 49]. However, learning open-vocabulary classifiers from a large weakly-labeled dataset is not a trivial extension to the conventional N-way classification (*e.g.*, one-vs-all SVM or softmax that treats the N classes independently), due to the following three challenges:

Extreme Sparse Labels. Web images are usually weakly annotated, *i.e.*, an image is usually described by only a small number of words as compared to the whole vocabulary. This results in extreme sparse labels, where the miss-

ing entries have no clear "positive" or "negative" supervised samples that confuse the resultant classifiers. As shown in Figure 1 (top), missing labels such as "green" and "grass" should also be considered as positive.

Complex Semantic Relations. Traditional N-way classifiers assume that the N labels are mutually exclusive [1]. However, when N is large, this assumption will no longer hold. Although some efforts are made by hand-crafting semantic relations among the labels [10, 11, 44, 22], it is increasingly impractical as more realistic open-vocabulary labels are considered. On one hand, different labels are usually used to describe the same image, *e.g.*, "BMW" is a "car", which may or may not be a "racer"; treating them independently will violate semantic relations. On the other hand, the vocabulary of Web image labels follow a power-law distribution (cf. Figure 1(middle))—only a few labels correspond to many training data while the large number of the rest labels correspond to little data. If the semantic relations among labels are ignored, we cannot transfer the knowledge of learning frequent classes to help learning rare classes [37].

Inexhaustible Web Data. Nearly 1.83 million images are uploaded to Flickr everyday[1]! It is necessary to keep classifiers up-to-date since: 1) receiving more data will improve the performance of the classifiers; and 2) the semantics of classifiers may evolve when feeding additional examples [9]. Obviously, it is impractical to retrain our visual classifiers each time when new samples arrive. Therefore, we require the open-vocabulary classifiers to be trained in an online fashion, which will help us to realize a practical never-ending visual learner [5].

In this paper, we present a novel classification paradigm towards tackling the above challenges. As shown in Figure 1(bottom), our key idea is to learn latent representations for labels and images, and then cast the classifier learning from label *discrimination* to image embedding *regression*. We call this paradigm *Collaborative Learning* since we exploits the joint collaboration among images, labels and visual features during classifier learning. In fact, this technique can be considered as a visual extension to the well-known Collaborative Filtering that is used to learn latent representations for the sparsely linked users and items in recommendation systems [26]. Collaborative learning can effectively transform the extreme sparse labels into compact latent space, where the semantic relations are also preserved in terms of similarities in the space (cf. Figure 1(b)). By doing this, we can effectively handle the patterns of missing labels and the complex label relations.

To address the dynamic nature of the ever-evolving Web images, we develop a computationally efficient online algorithm to solve the proposed collaborative learning problem. As illustrated in the colored components in Figure 2, given a new image and its label, our model is updated by two online steps: 1) update the image and label embeddings given new images and labels; and 2) update the classifiers given the updated embeddings. In sharp contrast to the widely-used SGD, our method has two advantages: 1) it does not require the tuning of learning rate that has a significant effect on the performance of SGD, and 2) it can "memorize" the history data and thus only one pass of training data is sufficient. We apply the proposed method to learn a set of 30K-vocabulary classifiers from a publicly available 1M Flickr image dataset with weak labels [34]. Our method achieves fast online training of 1K images within a second on a GPU (or 10s on a single CPU). Cross-dataset experiments on several multi-labeled benchmarks demonstrate that the learned open-vocabulary classifiers outperform several state-of-the-art learning methods. Figure 3 illustrates qualitative images ranked by several sample classifiers in our vocabulary. In summary, our contributions are as follows:

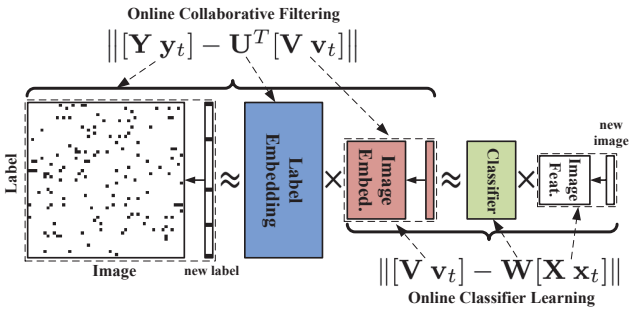


Figure 2. The proposed online collaborative learning paradigm. Colored components are alternatively updated.

- We propose a novel open-vocabulary classification paradigm named collaborative learning, which tackles the three challenges of: extreme label sparsity, complex label relations and inexhaustible Web data.
- We develop a fast online algorithm for collaborative learning which requires no learning rate and retraining.
- Promising results on cross-datasets demonstrate the high potential of our 30,456-vocabulary classifiers trained from 1M Flickr Images.

2. Related Work

Our idea of transforming multi-label classification to regression is inspired by recent studies on learning visual models from semantic embeddings [16, 2, 38] that are fundamentally different from those studies on webly-supervised learning [12, 4]. As compared to hand-crafted semantic relations, *e.g.*, semantic hierarchy [10, 37] and relation graph [11], mapping discrete label space to continuous semantic space offers a more flexible scalability. Frome *et al.* [16] applied Word2Vec model [32] to obtain

[1]https://www.flickr.com/photos/franckmichel/6855169886

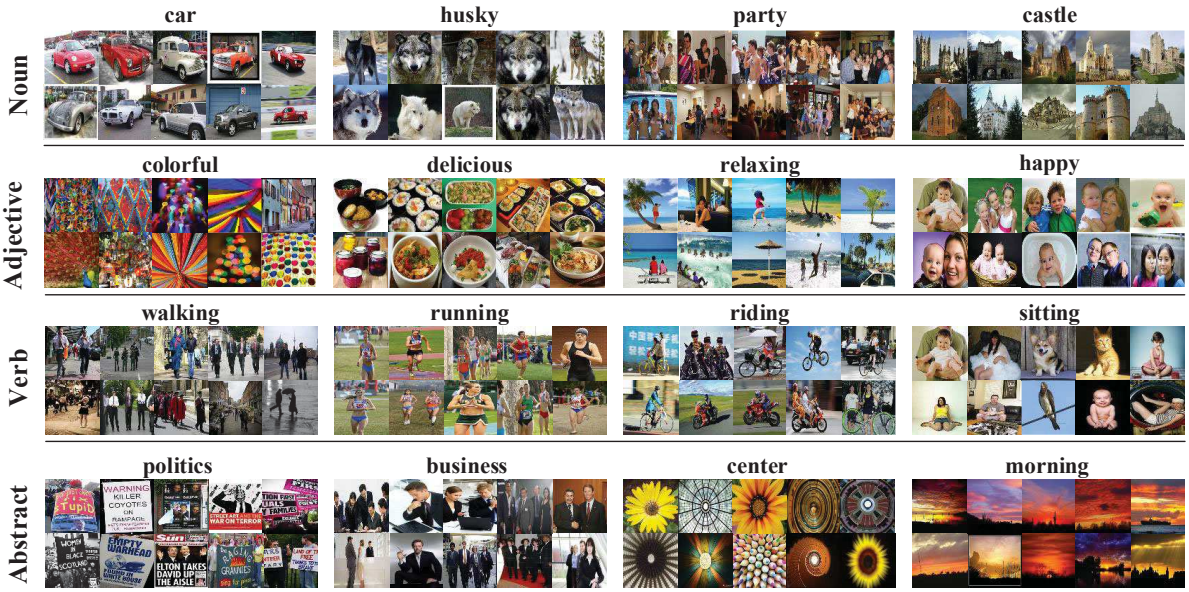


Figure 3. Top 10 images classified by our open-vocabulary classifiers, including various types of semantics such as nouns, adjectives, verbs and abstract concepts. Note that the images are from NUSWIDE [7] which are different from the training set. We can see that our classifiers well generalize the visual patterns, *e.g.*, there is no "husky" in NUSWIDE but we can return the most similar "wolf". This is mainly due to the effectiveness of learning from semantic embeddings. Best viewed in color and zoom in.

the embeddings of the 1,000 ImageNet classes by learning from a large Wiki corpus and then fitted a visual CNN model to the 1,000 embeddings. Therefore, by similarity calculation between class embeddings, the visual model can be generalized to unseen classes in the corpus. Similar models with different regression function can be found in [38]. Compared to our work, their semantic embeddings were pretrained by an external textual corpus while ours are jointly learned by collaborating images and visual features. This joint learning approach can also be found in recent visual-semantic embedding work [45, 24]. However, they require training data with well-annotated image-sentence pairs, while our method can deal with weakly labeled data.

Technically speaking, the work mentioned above and ours are closely related to Label Space Dimension Reduction (LSDR) [21, 42]—a new paradigm in multi-label classification that can be traced back to the very classic CCA [20]. LSDR maps the label-space into a new low dimensional space, trains the classifier in the constructed space and then projects the predictions back to the original labels. In experiments, we compare our method with two recent works [46, 48] since they can also be applied in large-scale settings. Compared to their optimizaiton procedure, our method needs no sensitive learning rates or subproblem iterations.

Collaborative filtering via matrix factorization has been successfully applied in many recommender systems [26], which inspires our collaborative learning formulation. Our modeling of sparse label-image annotations is analogous to their sparse user-item relations. This idea has also been applied in the latest work on visual learning [17, 14], where the CNN model is fine-tuned with image embeddings learned by collaborative filtering. On the other hand, we develop an efficient online algorithm for the proposed collaborative learning with the help of matrix factorization. Our algorithm can be considered as a visual extension for the recent online dictionary learning methods [31, 15].

3. Formulation

For a typical multi-label classification problem with n training samples $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^d$ is the image feature vector and $\mathbf{y}_i \in \{0,1\}^m$ is the sparse label vector with vocabulary size m, we denote the nonzero entries of $\mathbf{y}_i$ as the corresponding labels that are "present" or "on" whereas the zeros are "absent" or "off". When m is large, traditional N-way classifiers $\{f_i\}_{i=1}^m$, where f_i: $\mathbb{R}^d \rightarrow \{0,1\}$, will be problematic due to the complex semantic relations among the sparsely annotated labels.

3.1. Label Space Dimension Reduction

Recall that the proposed collaborative learning is closely related to Label Space Dimension Reduction (LSDR) [42, 6]—a generic framework that scales up to large label size. LSDR transforms the original linear classification model $\mathbf{y}_i \approx \widetilde{\mathbf{W}}\mathbf{x}_i$ into $\mathbf{v}_i \approx \mathbf{W}\mathbf{x}_i$, where $\widetilde{\mathbf{W}} \in \mathbb{R}^{m\times d}$ and $\mathbf{W} \in \mathbb{R}^{r\times d}$ are the parameters of linear models, $\mathbf{v}_i \in \mathbb{R}^r$ ($r \ll m$) lies the reduced label space (cf. Figure 1(bottom)). Formally, the objective of LSDR can be formulated as a coupled linear regression problem [48]:

$$\min_{\mathbf{U},\mathbf{W}} \left\|\mathbf{Y} - \mathbf{U}^T\mathbf{W}\mathbf{X}\right\|_F^2 + \beta\left(\|\mathbf{U}\|_F^2 + \|\mathbf{W}\|_F^2\right), \qquad (1)$$

where the columns of $\mathbf{Y} \in \mathbb{R}^{m\times n}$ and $\mathbf{X} \in \mathbb{R}^{d\times n}$ represent $\{\mathbf{y}_i\}$ and $\{\mathbf{x}_i\}$, respectively; $\mathbf{U} \in \mathbb{R}^{r\times m}$ is a reduced classifier model. LSDR essentially seeks a low-rank decomposition for $\widetilde{\mathbf{W}}$, *i.e.*, $\widetilde{\mathbf{W}} = \mathbf{U}^T\mathbf{W}$, and the model of the i-th classifier can be written as $\widetilde{\mathbf{w}}_i = \mathbf{u}_i^T\mathbf{W}$, *i.e.*,

$$f_i(\mathbf{x}) = \mathbf{u}_i^T\mathbf{W}\mathbf{x}. \tag{2}$$

The assumption of LSDR is that $\widetilde{\mathbf{W}} = \{\widetilde{\mathbf{w}}_i\}$ are interrelated. For example, $\widetilde{\mathbf{w}}_{dog}$ should be more similar to $\widetilde{\mathbf{w}}_{puppy}$ than $\widetilde{\mathbf{w}}_{car}$. Due to the coupled linear $\mathbf{U}^T\mathbf{W}\mathbf{X}$, solving $\mathbf{W}$ and $\mathbf{U}$ requires large matrix inverse[2], which is usually solved by iterative conjugate gradient descent [48]. Therefore, they are impractical for designing online solutions.

3.2. Collaborative Learning

Instead of hand-crafting the semantic relations between classifier models $\widetilde{\mathbf{W}}$, we directly explore the relations from the extreme sparse labels $\mathbf{Y}$ by leveraging collaborative filtering [26], which is especially powerful in learning latent representations that preserve semantic relations for sparse user-item matrix like $\mathbf{Y}$. Recall that $\mathbf{V} = \{\mathbf{v}_i\}$ is the image embedding in the reduced label space, by forcing $\mathbf{V} = \mathbf{W}\mathbf{X}$, Eq. (1) can be reformulated as:

$$\min_{\mathbf{U},\mathbf{V},\mathbf{W}} \left\|[\mathbf{Y} - \mathbf{U}^T\mathbf{V}\right\|_F^2 + \beta\left(\|\mathbf{U}\|_F^2 + \|\mathbf{W}\|_F^2\right), s.t.\ \mathbf{V} = \mathbf{W}\mathbf{X}. \tag{3}$$

By relaxing the equality constraint, our collaborative learning formulation can be obtained:

$$\min_{\mathbf{U},\mathbf{V},\mathbf{W}} \left\|\mathbf{Y} - \mathbf{U}^T\mathbf{V}\right\|_F^2 + \alpha\|\mathbf{V} - \mathbf{W}\mathbf{X}\|_F^2 + \beta\left(\|\mathbf{U}\|_F^2 + \|\mathbf{V}\|_F^2 + \|\mathbf{W}\|_F^2\right). \tag{4}$$

To ensure the constraint in Eq. (3) is satisfied, we can enforce $\|\mathbf{W}\mathbf{X} - \mathbf{V}\|_F^2 = 0$ by imposing a very large α. Optimizing Eq. (3) involves a joint collaboration among the label embedding $\mathbf{U}$, image embedding $\mathbf{V}$, and the visual features $\mathbf{X}$. This joint learning is different from other works that use two-stage learning, including separated semantic embedding and visual-semantic mapping [38, 14, 16].

Note that the original collaborative filtering loss is only defined for the observed entries, *i.e.*, $\|(\mathbf{Y} - \mathbf{U}^T\mathbf{V}) \odot \mathbf{I}\|_F^2$, where $\mathbf{I}$ indicates whether Y_{ij} is taken into account or not. However, our method cannot apply this partially collaborative filtering formulation due to the following three reasons. First, $\mathbf{Y}$ is extremely sparse, *e.g.*, for the 1M Flickr dataset used in this paper, over 99.7% of the enries are missing. Thus, only modeling the nonzero entries will cause severely overfitting. Second, most of the missing labels should be considered as "negative" although there do exist missing "positive" labels. In fact, some results show that treating "missing" as "zero" is much more effective than considering "missing" as "undefined", especially for preserving the latent semantic relations [39, 8, 40]. Finally, the mask $\mathbf{I}$ will cause difficulties in deriving matrix-form solutions and hence cannot be easily speed-up by parallel computing via GPU. Figure 4 illustrates some results of the label embedding qualities using different matrix decompositions.

[2]To see this, denoting $\mathbf{w} = \text{vec}(\mathbf{W})$, $\widetilde{\mathbf{X}}_i = [\mathbf{u}_1 \otimes \mathbf{x}_i, ..., \mathbf{u}_m \otimes \mathbf{x}_i]$ (vec is the column-wise vectorization of a matrix and $\otimes$ is matrix outer product), the subproblem can be reformulated into $\min_{\mathbf{W}} \sum_i \|\mathbf{y}_i - \widetilde{\mathbf{X}}_i^T\text{vec}(\mathbf{W})\|_2^2 + \lambda_2\|\text{vec}(\mathbf{W})\|_2^2$, which requires impractical inversion of the size $md \times md$.

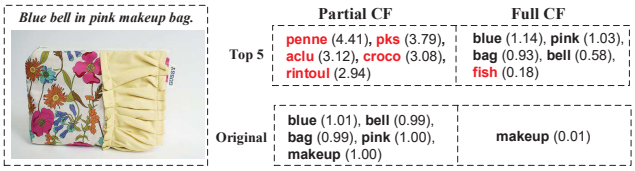


Figure 4. **Top5**: top 5 predicted labels ranked by inner product values (in brackets) between the image embedding and label embeddings, obtained by Partial CF and Full CF. Words in red are wrong labels. **Original**: the product value between the embeddings of the image and the original labels. We can see that due to severe overfitting, the observed labels receive nearly perfect (close to 1) predictions; while unobserved but wrong labels receive high product values.

4. Online Algorithm

Eq. (4) can be easily solved via alternating minimization that involves solving several quadratic programming problems. However, in dealing with large-scale and ever-evolving training data, such batch methods become impractical due to the limitation of storage and computational power.

The batch objective function in Eq. (4) with a batch of samples can be rewritten as:

$$J_n(\mathbf{U},\mathbf{W}) = \frac{1}{n}\sum_{i=1}^{n}\ell(\mathbf{U},\mathbf{W};\mathbf{x}_i,\mathbf{y}_i) + \frac{\beta}{n}\left(\|\mathbf{U}\|_F^2 + \|\mathbf{W}\|_F^2\right), \tag{5}$$

where $\ell(\mathbf{U},\mathbf{W};\mathbf{x}_i,\mathbf{y}_i) = \min_{\mathbf{v}} \|\mathbf{y}_i - \mathbf{U}^T\mathbf{v}\|_F^2 + \alpha\|\mathbf{W}\mathbf{x}_i - \mathbf{v}\|_F^2 + \beta\|\mathbf{v}\|_2^2$. Note that the regularization for $\mathbf{U}$ and $\mathbf{W}$ is imposed on all the training samples, so we have $\frac{\beta}{n}$. This reasonable since when $n \to \infty, \frac{\beta}{n} \to 0$, *i.e.*, the regularization is no longer necessary due to sufficient training data. In the online optimization setting, we usually focus on minimizing the expected loss $\mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(\mathbf{U},\mathbf{W});\mathbf{x},\mathbf{y}]$ where the expectation is taken w.r.t. samples $(\mathbf{x},\mathbf{y})$, instead of directly minimizing the empirical loss. The reason is that online algorithms such as SGD can lead to a lower expected loss than a perfect batch minimization [3].

4.1. Solution

Inspired by recent online matrix factorization algorithms [31, 15], we develop an online stochastic optimization algorithm to minimize the empirical loss in Eq. (5), which can process *one training sample at a time*. In particular, at time t, we estimate $\ell(\mathbf{U},\mathbf{W};\mathbf{x},\mathbf{y})$ by using $\mathbf{v}_t$, which is

Algorithm 1: Online Collaborative Learning

Input : $\{\mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^n$: training image features,
$\{\mathbf{y}_i \in \{0,1\}^m\}_{i=1}^n$: training labels,
r: embedding dimension,
α and β: trade-off parameters

Output: $\mathbf{W} \in \mathbb{R}^{r\times d}$: image embedding model, $\mathbf{U} \in \mathbb{R}^{r\times m}$: label embeddings

Initialization: randomly initialize $\mathbf{U}_0$ and $\mathbf{W}_0$, set $\mathbf{A}_0 = \mathbf{0}_{r\times d}$, $\mathbf{C}_0 = \mathbf{0}_{r\times m}$, $\mathbf{B}_0^{-1} = \frac{\alpha}{\beta}\mathbf{I}_{d\times d}$, $\mathbf{D}_0^{-1} = \frac{1}{\beta}\mathbf{I}_{r\times r}$

for *t=1* **to** *n* **do**

  Reveal the sample $(\mathbf{x}_t, \mathbf{y}_t)$

  Update
  $\mathbf{v}_t \leftarrow \left(\mathbf{U}_{t-1}\mathbf{U}_{t-1}^T + (\alpha+\beta)\mathbf{I}\right)^{-1}(\alpha\mathbf{W}_{t-1}\mathbf{x}_t + \mathbf{U}_{t-1}\mathbf{y}_t)$;

  Update $\mathbf{W}_t \leftarrow \mathbf{A}_t\mathbf{B}_t^{-1}$, where $\mathbf{A}_t = \mathbf{A}_{t-1} + \mathbf{v}_t\mathbf{x}_t^T$,
  $\mathbf{B}_t^{-1} \leftarrow \mathbf{B}_{t-1}^{-1} - \mathbf{B}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^T\mathbf{B}_{t-1}^{-1}/(1+\mathbf{x}_t^T\mathbf{B}_{t-1}^{-1}\mathbf{x}_t)$;

  Update $\mathbf{U}_t \leftarrow \mathbf{D}_t^{-1}\mathbf{C}_t$, where $\mathbf{C}_t = \mathbf{C}_{t-1} + \mathbf{v}_t\mathbf{y}_t^T$,
  $\mathbf{D}_t^{-1} \leftarrow \mathbf{D}_{t-1}^{-1} - \mathbf{D}_{t-1}^{-1}\mathbf{v}_t\mathbf{v}_t^T\mathbf{D}_{t-1}^{-1}/\left(1+\mathbf{v}_t^T\mathbf{D}_{t-1}^{-1}\mathbf{v}_t\right)$;

end

return $\mathbf{U} = \mathbf{U}_t, \mathbf{W} = \mathbf{W}_t$

solved by using last updated $\mathbf{U}_{t-1}$ and $\mathbf{W}_{t-1}$. Then, we use the collected $\{\mathbf{v}_i\}_{i=1}^t$ to compute $\mathbf{U}_t$ and $\mathbf{W}_t$. The detailed procedure is described as follows.

Update $\mathbf{v}_t$ with Fixed $\mathbf{W}_{t-1}$ and $\mathbf{U}_{t-1}$: We estimate $\ell(\mathbf{U}, \mathbf{W}; \mathbf{x}, \mathbf{y})$ in Eq. (5) by using $\mathbf{v}_t$ obtained by:

$$\mathbf{v}_t = \arg\min_{\mathbf{v}} \left\|\mathbf{y}_t - \mathbf{U}_{t-1}^T\mathbf{v}\right\|_F^2 + \alpha\left\|\mathbf{W}_{t-1}\mathbf{x}_t - \mathbf{v}\right\|_F^2 + \beta\|\mathbf{v}\|_2^2, \quad (6)$$

which has a closed-form update rule:

$$\mathbf{v}_t \leftarrow \left(\mathbf{U}_{t-1}\mathbf{U}_{t-1}^T + (\alpha+\beta)\mathbf{I}\right)^{-1}(\alpha\mathbf{W}_{t-1}\mathbf{x}_t + \mathbf{U}_{t-1}\mathbf{y}_t). \quad (7)$$

Note that $\mathbf{U}_{t-1}\mathbf{U}_{t-1}^T$ is only of size $r\times r$, thus computing the inverse is fast.

Update $\mathbf{U}_t$ and $\mathbf{W}_t$ with fixed $\mathbf{v}_t$: The loss function for updating $\mathbf{U}_t$ and $\mathbf{W}_t$ is:

$$G_t(\mathbf{U}, \mathbf{W}) = \frac{\beta}{t}\left(\|\mathbf{U}\|_F^2 + \|\mathbf{W}\|_F^2\right) + \frac{1}{t}\sum_{i=1}^{t}\left(\left\|\mathbf{y}_t - \mathbf{U}^T\mathbf{v}_t\right\|_F^2 + \alpha\left\|\mathbf{W}\mathbf{x}_t - \mathbf{v}_t\right\|_F^2 + \beta\|\mathbf{v}_t\|_2^2\right). \quad (8)$$

Minimizing $G_t(\mathbf{U}, \mathbf{W})$ can be solved by optimizing $\mathbf{W}$ and $\mathbf{U}$ alternatively. With $\mathbf{U}$ fixed, $G_t(\mathbf{U}, \mathbf{W})$ can be easily calculated by solving equation $\nabla_{\mathbf{W}} G_t = \mathbf{0}$, which leads to the following update for $\mathbf{W}_t$:

$$\mathbf{W}_t \leftarrow \mathbf{V}_t\mathbf{X}_t^T\left(\mathbf{X}_t\mathbf{X}_t^T + \beta/\alpha\mathbf{I}\right)^{-1}, \quad (9)$$

where $\mathbf{V}_t = [\mathbf{V}_{t-1}, \mathbf{v}_t]$ and $\mathbf{X}_t = [\mathbf{X}_{t-1}, \mathbf{x}_t]$. Since the size of $\mathbf{X}_t\mathbf{X}_t^T$ is $d\times d$ (*e.g.*, d is typically several thousands for modern visual features), computing the inverse of $\mathbf{X}_t\mathbf{X}_t^T$ is impractical for online algorithm due to $\mathcal{O}(d^3)$ computational cost. Fortunately, note that $\mathbf{X}_t\mathbf{X}_t^T = \mathbf{X}_{t-1}\mathbf{X}_{t-1}^T + \mathbf{x}_t\mathbf{x}_t^T$, which is a low-rank modification for the original matrix, we can apply Sherman-Morrison-Woodbury formula to simplify the inverse. Define $\mathbf{B}_0 = \frac{\beta}{\alpha}\mathbf{I}$ and $\mathbf{B}_t = \mathbf{B}_{t-1} + \mathbf{x}_{t-1}\mathbf{x}_{t-1}^T$, we have:

$$\mathbf{B}_t^{-1} = \left(\mathbf{B}_{t-1} + \mathbf{x}_t\mathbf{x}_t^T\right)^{-1} = \mathbf{B}_{t-1}^{-1} - \frac{\mathbf{B}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^T\mathbf{B}_{t-1}^{-1}}{1+\mathbf{x}_t^T\mathbf{B}_{t-1}^{-1}\mathbf{x}_t}. \quad (10)$$

which only requires matrix-vector multiplication. Let $\mathbf{A}_t = \mathbf{A}_{t-1} + \mathbf{v}_t\mathbf{x}_t^T$ and $\mathbf{A}_0 = \mathbf{0}$, then the update rule of $\mathbf{W}_t$ in Eq. (9) can be rewritten as:

$$\mathbf{W}_t \leftarrow \mathbf{A}_t\mathbf{B}_t^{-1}. \quad (11)$$

Similarly, the updating rule of $\mathbf{U}_t$ can be reformulated as:

$$\mathbf{U}_t \leftarrow \mathbf{D}_t^{-1}\mathbf{C}_t, \quad (12)$$

where $\mathbf{C}_t = \mathbf{C}_{t-1} + \mathbf{v}_t\mathbf{y}_t^T$, $\mathbf{D}_0 = \beta\mathbf{I}$, and

$$\mathbf{D}_t^{-1} = \left(\mathbf{D}_{t-1} + \mathbf{v}_t\mathbf{v}_t^T\right)^{-1} = \mathbf{D}_{t-1}^{-1} - \frac{\mathbf{D}_{t-1}^{-1}\mathbf{v}_t\mathbf{v}_t^T\mathbf{D}_{t-1}^{-1}}{1+\mathbf{v}_t^T\mathbf{D}_{t-1}^{-1}\mathbf{v}_t}. \quad (13)$$

There are two significant advantages of our method over SGD: 1) as shown in the update rules discussed above, our method requires no learning rate; 2) our method explicitly records the historical information in $\mathbf{A}_t$, $\mathbf{B}_t$, $\mathbf{C}_t$ and $\mathbf{D}_t$. Thus, our method is expected to achieve better performance via only one pass of the data, while SGD usually needs several passes which are impossible for online cases.

4.2. Algorithmic Analysis

Our online algorithm is summarized in Algorithm 1. Theorem 1 provides the theoretical guarantee for the convergence of our algorithm, which states that the update series $\{(\mathbf{W}_t, \mathbf{U}_t)\}$ obtained by Algorithm 1 will converge to a local minimum of the optimization problem with cost function J_t shown in Eq. (5).

Theorem 1 (Convergence of Algorithm 1) *Assume $(\mathbf{x}_i, \mathbf{y}_i)$ is bounded; the solution $\mathbf{W} \in \mathbb{R}^{r\times d}$ and $\mathbf{U} \in \mathbb{R}^{r\times m}$ obtained by Algorithm 1 is full rank. Then, we have the following properties with probability one: (a) G_t converges; (b) $J_t - G_t$ converges to 0; (c) J_t converges; and (d) $(\mathbf{W}_t, \mathbf{U}_t)$ converges to a stationary point.*

Since the image visual features and the labeling vectors are always bounded, and we empirically observe that $(\mathbf{W}_t, \mathbf{U}_t)$ is always full rank, the assumptions in Theorem 1 hold. The proof can be done similarly to [31, 15] with additional efforts on showing $G_t(\mathbf{W}, U)$ in Eq. (8) is strictly convex and the update rules in Eq. (7), (11) and (12) satisfy the optimality conditions of their corresponding objective functions. Note that Theorem 1 also guarantees the convergence of the mini-batch extension of Algorithm 1. Suppose the batch size is b, we slightly abuse the notation $\mathbf{X}_t$ as the t-th mini-batch $\{\mathbf{x}_i\}_{i=t}^{t+b}$, and $\mathbf{V}_t$ and $\mathbf{Y}_t$ are defined similarly. Then, the extended version of Algorithm 1 can be derived by replacing $\mathbf{x}_t$, $\mathbf{y}_t$ and $\mathbf{v}_t$ to $\mathbf{X}_t$, $\mathbf{Y}_t$ and $\mathbf{V}_t$, respectively. Moreover, we replace $\mathbf{A}_t$ and $\mathbf{B}_t$ in Step 5 in Algorithm 1:

$$\begin{cases}\mathbf{A}_t \leftarrow \mathbf{A}_{t-1} + \frac{1}{b}\mathbf{V}_t\mathbf{X}_t^T, \\ \mathbf{B}_t \leftarrow \mathbf{B}_{t-1}^{-1} - \frac{1}{b}\mathbf{B}_{t-1}^{-1}\mathbf{X}_t(\mathbf{I} + \frac{1}{b}\mathbf{X}_t^T\mathbf{B}_{t-1}^{-1}\mathbf{X}_t)^{-1}\mathbf{X}_t^T\mathbf{B}_{t-1}^{-1},\end{cases} \quad (14)$$

and replace $\mathbf{C}_t$ and $\mathbf{D}_t$ in Step 6 in Algorithm 1 by averaging the observations in a minibatch:

$$\begin{cases} \mathbf{C}_t \leftarrow \mathbf{C}_{t-1} + \frac{1}{b}\mathbf{V}_t\mathbf{Y}_t^T, \\ \mathbf{D}_t \leftarrow \mathbf{D}_{t-1}^{-1} - \frac{1}{b}\mathbf{D}_{t-1}^{-1}\mathbf{V}_t(\mathbf{I} + \frac{1}{b}\mathbf{V}_t^T\mathbf{D}_{t-1}^{-1}\mathbf{V}_t)^{-1}\mathbf{V}_t^T\mathbf{D}_{t-1}^{-1}. \end{cases} \quad (15)$$

Note that the matrix inverse can be computed efficiently since the corresponding size is only $b \times b$.

We now provide the complexity analysis for the proposed algorithm. In the training stage, despite the output model $\mathbf{W}$ and $\mathbf{U}$, the memory consumption to store $\mathbf{A}_t$, $\mathbf{B}_t$, $\mathbf{C}_t$ and $\mathbf{D}_t$ are $\mathcal{O}(rd)$, $\mathcal{O}(d^2)$, $\mathcal{O}(rm)$ and $\mathcal{O}(r^2)$, respectively. At each training time t, the computational complexity for Step 4, 5 and 6 are as follows: Step 4 requires $\mathcal{O}(r^3 + r^2m)$, including inverse and matrix multiplication; Step 5 requires $\mathcal{O}(rd^2 + rbd + bd^2 + b^2d + b^3)$ including $\mathcal{O}(rd^2)$ for updating $\mathbf{W}_t$, $\mathcal{O}(rbd)$ for updating $\mathbf{A}_t$ and $\mathcal{O}(bd^2 + b^2d + b^3)$ for $\mathbf{B}_t$, where b is the minibatch size and $b = 1$ in Algorithm 1; and Step 6 requires $\mathcal{O}(mr^2 + rbm + b^2r + br^2 + b^3)$. We can see the memory and computational compelxity for our algorithm is relatively low—except for several small-size inverses (*e.g.*, $r \times r$ and $b \times b$ matrices), our update rules only require matrix multiplications, which can be easily speed-up by using parallel computing such as GPU.

5. Experiments

5.1. Datasets

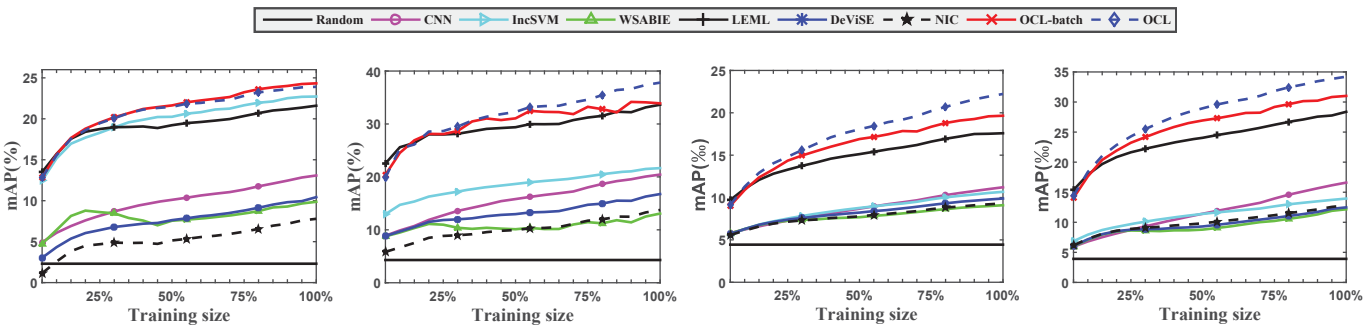
We used **SBU** captioned photo dataset [34] as our large-scale weakly-labeled training data. It contains 1M images with user-generated descriptive text, which covers noisy and wide variety of semantics including objects, attributes, actions, stuff and scenes. The stopwords and words with frequency less than 5 are removed from the text. This gives rises to a vocabulary of size 3,0456. Since we are interested in examining whether the open-vocabulary classifiers learned with one dataset can generalize to others, we conducted **cross-dataset** evaluations, *i.e.*, we test the classifiers on datasets different from SBU. We used the official test split of four multi-labeled visual benchmarks: 1) **NUSWIDE** [7], containing 107,859 test images across 81 concepts; 2) **CCV** [23], containing 4,658 test videos across 20 concepts. Note that some labels of CCV are more complex than those of NUSWIDE (*e.g.*, "MusicPerformance" vs. "car"); 3) **Flickr30K** [47], containing 31,783 Flickr images focusing on events involving people and animals. Each image is associated with five high-quality sentences independently written by five native English speakers from Mechanical Turk. Since this dataset has no train/test split, we use the whole dataset as test data; 4) **COCO** [30], containing 40K official validation images with 3-5 high-quality sentences. We consider Flickr30K and COCO as multi-labeled datasets, where the labels correspond to the words in the associated sentences. After removing the words with frequency less than 5, it resulted in 4,015 and 3,638 labels, which reside in our 30,456-word open-vocabulary, respectively for Flickr30K and COCO. For images, we used DeCAF 4,096-d DCNN features [13]. For videos, we sampled 1 frame image with a step size of 5 in each video and used the mean DeCAF feature as the final video feature. All the features are normalized by ℓ_2-norm.

5.2. Compared Methods and Details

Recall that the two key model parameters for open-vocabulary classifiers as described in Eq. (2) are: the visual-to-semantic mapping $\mathbf{W}$ and the label embedding $\mathbf{U}$. We compared the proposed Online Collaborative Learning (**OCL**) against 6 state-of-the-art large-scale classification methods, which have different definitions of $\mathbf{W}$ and $\mathbf{U}$. 1) **CNN**: a standard 3,0456-way AlexNet [27] with softmax classifiers. Such deep architecture has been widely used in training classifiers recently. Since we used DeCAF visual features, other methods can be considered as CNNs of various classification layer but with fixed lower-level networks. With the last fully-connected layer as features, $\mathbf{W}$ can be defined as $\mathbf{I}$ and $\mathbf{U}$ is the softmax model; 2) **IncSVM**: Incremental SVM [43]. It applies a warm start strategy to efficiently update the previously trained SVM. We used the one-vs-all strategy to train 30,456 independent classifiers. Similar to CNN which has no label space reduction, it defines $\mathbf{W} = \mathbf{I}$ and $\mathbf{U}$ as the SVM model; 3) **WSABIE**: an online large-scale vocabulary image annotation method [46]. It adopts a max-margin metric learning method to learn an image mapping $\mathbf{W}$ and and label embedding $\mathbf{U}$; 4) **DeViSE**: a Deep Visual-Semantic Embedding Model [16]. Different from WSABIE, it adopts an external method—Word2Vec [32] to obtain label embedding $\mathbf{U}$, which are fixed during metric learning. In this paper, we used the SBU captions as the textual corpus; 5) **NIC**: Google Neural Image Caption generator [45]. Based on [25], we used the visual-semantic mapping as $\mathbf{W}$ and word embeddings as $\mathbf{U}$; 6) **LEML**: Low-rank Empirical risk minimization for Multi-Label Learning [48], where $\mathbf{U}$ and $\mathbf{W}$ are described in Eq. (1).

The experiments showed that our method is insensitive to the trade-off parameters α and β. We empirically set $\alpha = 1$ and $\beta = 10^{-4}$, which can achieve good performance. Except for WSABIE and DViSE which have no released source code, we used the implementations of the methods suggested by the authors. For WSABIE, our implementation can roughly reproduce the reported results on NUSWIDE. Since DeViSE has no released textual corpus, we used the sentences of SBU instead to obtain the label embeddings by gensim[3], and then strictly sticked to the suggested hyperparameter settings. Except for CNN and IncSVM, the dimensionality r is a crucial parameter. We

[3]https://radimrehurek.com/gensim/about.html



(a) NUSWIDE (b) CCV (c) Flickr30K (d) COCO

Figure 5. Performance (mAP% or mAP‰) on four datasets using various training size. All the results are obtained in an online setting, *i.e.*, when new data arrive, we do not retrain the model. Therefore, the performance at 100% is obtained by only one pass of the data.

tested $r \in [100, 300, 500, 1000, 1500]$ and found that 500 was best for NIC and 1,000 or 1,500 were best for the rest (1,500 slightly better). Therefore, for efficiency, we chose $r = 500$ for NIC and $r = 1000$ for OCL, WASABIE, DeViSE and LEML. All the methods were run 5 times with different minibatch orders and averaged the results.

The experiments showed that the minibatch extension as described in Eq. (14) and Eq. (15) improves the resultant accuracy and the overall training speed and different batch size $b \in [64, 128, 256, 512]$ provided similar results. We used $b = 256$ as the default batch size for all the methods. Table 1 lists the time for training one minibatch of size 256 on a standard desktop 6-core 3.0GHz CPU with 64GB memory [4]. Our method is computationally efficient due to the fact that each iteration only requires matrix-vector multiplication and inversion of small-size matrices. Our method only costs ∼0.3s for a minibatch when running on a Titan Z GPU with 5K cuda cores and 12GB memory.

Table 1. CPU time of various methods in processing a minibatch of size 256 with 30,456 labels. For example, $\sim 10\times$ denotes the CPU time is about 10 times as OCL.

CNN	IncSVM	WSABIE	DeViSE	NIC	LEML	OCL
$\sim 70\times$	$\sim 10\times$	$\sim 20\times$	$\sim 20\times$	$\sim 60\times$	$\sim 10\times$	∼2s

5.3. Results

We first evaluated the performance of our online learned open-vocabulary classifiers. After collecting the visual-semantic mapping $\mathbf{W}$ and the label embedding $\mathbf{U}$ of various methods, we computed a score between label and image as the final classification score. Except for OCL, the other methods run for several epochs (*e.g.*, scans of training data) until convergence. Table 2 lists the performance of all the methods. We can observe that:

1) Except on NUSWIDE, our method considerably outperforms the others. One possible reason is that the 81 classes in NUSWIDE, *e.g.*, "car" and "dog" are relatively simple, *e.g.*, with large inter- and small intra-class visual discrepancy. Therefore, N-way classifiers such as CNN can still perform well. When more complex classes appear such as "MusicPerformance" in CCV and many more classes in Flickr30K and COCO, our method becomes superior.

2) Visual-semantic embedding methods like WSABIE, DeViSE and NIC generally fail on all the datasets. This shows that they are ineffective on weakly supervised data.

3) LEML and our method consistently perform well on all the datasets since they effectively take advantages of explicitly learning latent representations from the extreme sparse label-image matrix.

4) The online version of OCL even outperforms the batch version (OCL-batch). This finding is consistent with [3], which states that a perfect minimization of the empirical loss is not a necessary guarantee to minimize the expected loss.

Table 2. Performance (mAP%) of various methods on the four benchmarks. Numbers in the bracket are the number of classes.

Dataset/Method	CNN	IncSVM	WSABIE	DeViSE	NIC	LEML	OCL-batch	OCL
NUSWIDE (81)	**24.3**	20.4	9.87	9.51	7.79	23.6	22.7	23.9
CCV (20)	36.2	39.1	13.1	12.1	13.8	37.8	33.9	**40.5**
Flickr30K (4,015)	1.91	1.85	0.61	0.73	0.93	2.28	1.97	**2.48**
COCO (3,638)	2.36	2.31	1.22	1.11	1.28	3.40	3.11	**3.52**

We further evaluated the effectiveness of online learning of our method. Figure 5 shows the performance of various methods by only one pass of the training data (SBU). Besides similar observations as above, we can see that the performance of CNN and IncSVM considerably drop by using only one epoch. We believe that online learning is important for training Web images due to its large-scale and ever-evolving nature. The effectiveness of the proposed online learning suggests a practical way for learning open-vocabulary classifiers. Moreover, we can see that there is still a large potential when our method is fed with more training data. That is to say, by continuously learning from the "free" Web images with weak labels, we can obtain

[4]This comparison is coarse since the implementations are different, *e.g.*, C++ for CNN, IncSVM and LEML, MATLAB for OCL, WASABIE and DeViSE, Python for NIC. However, the underlying linear algebra calculation of them benefits from Intel hardware acceleration

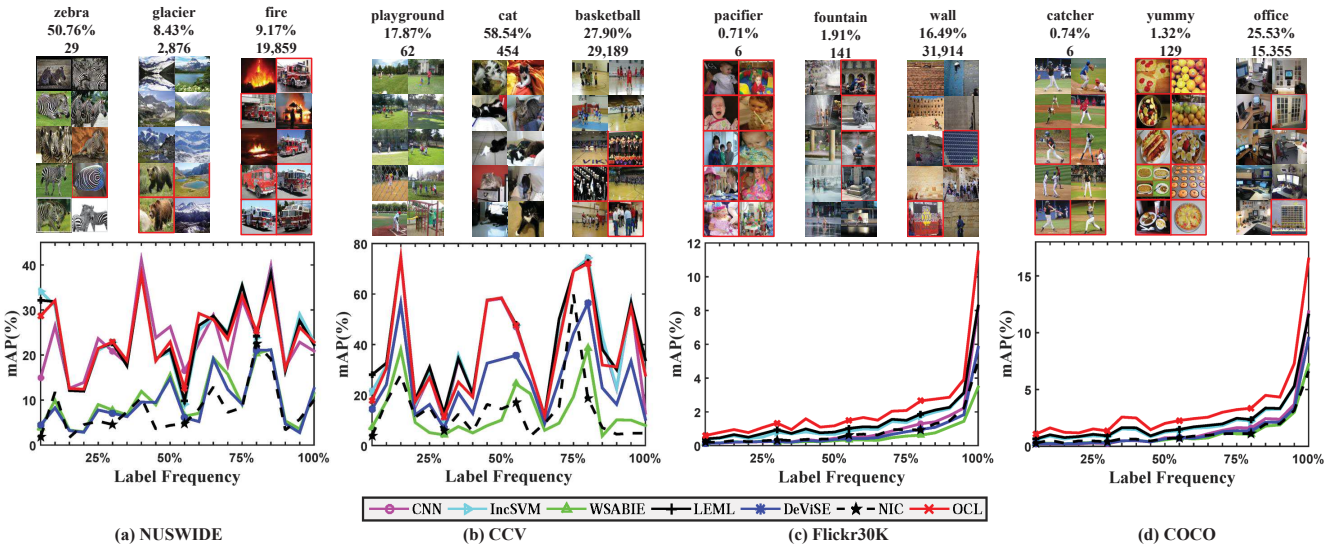


Figure 6. Performance (mAP%) of various classifiers vs. the frequency of their semantic labels in training data. For example, performance at 10% is the mAP% of all the classifiers whose labels fall in the lowest 5% to 10% freqency. We also show the top 10 results of our classifiers (OCL) at different frequency levels, including label names, AP%, and label freqency. Red lines indicate wrongly classified images according to the groundtruth. Best viewed in color and zoom in.

more and more accurate classifiers at a large scale.

Now we investigate how well the classifiers perform in more detail. Figure 6 illustrates the the peformance of various classifiers at different label frequency levels. We can see that as compared to conventional datasets of uniformly distributed data over small label size (*e.g.*, NUSWIDE and CCV), the performance on large-scale labeled datasets (*e.g.*, Flickr30K and COCO) clearly reflects the power-law distribution of real-world label distribution (cf. Figure 1 (middle)). Therefore, knowledge transfer from "many" classes to "few" classes is crucial in learning open-vocabulary classifiers [37]. We can see that our method consistently offers more accurate classifiers at almost all the frequency levels. This demonstrate the effectiveness of our method that learns from semantic embeddings and transfers semantic knowledge. Next, we would like to gain more insights about the failure cases:

Over generalization. Learning from semantic embeddings is not always effective. As shown in Figure 6, the "glacier" classifier is confused with "bear" and "lake" on NUSWIDE. The reason is that our model successfully learns that "glacier" is semantically related to "polar bear", "water" and "mountain". This is meaningful but it confuses the visual patterns of "glacier". When the visual cues are subtle, this confusion becomes more severe. For example, the "catcher" classifier on COCO is confused with "pitcher" and "batter", which are all very close to "baseball". Nevertheless, we believe that such confusion is more favorable than the semantically unrelated confusions, *e.g.*, when there is only 6 training samples of "pacifier", it can still returns "baby" images as reasonable wrong results; moreover, as shown in Figure 3, there is no "husky" in NUSWIDE but we can return the most similar "wolf". This demonstrates a great potential in zero-shot learning.

Word ambiguiation. We can see that "fire" classifier is confused with "fire truck", even there are 19,859 training images. The reason is that we did not adopt any word disambiguation. However, we believe such failure cases can be eliminated once we use advanced word sense disambiguation and part-of-speech techniques [36] in the future work.

Subjective labels. Human labeling is usually partially given in Flickr30K and COCO. For example, even though the top 10 images of our "yummy" classifier are all delicious food, none of them is labeled by human as "yummy".

6. Conclusions

This paper presents a novel large-scale classifier learning paradigm, called Online Collaborative Learning, which continuously learns many visual classifiers in an online fashion. The key difference between our method and conventional ones is that we explicitly learn from the discovered semantic embeddings, and hence we can effectively tackle the extreme sparse and complex semantic relations in weakly-supervised Web-scale images. We trained 3,0456 classifiers on 1M Flickr images and test them on four visual benchmarks. Promising results demonstrate the effectiveness of the proposed approach.

Acknowledgements

NExT research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its IRC@SG Funding Initiative.

References

[1] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Good practice in large-scale learning for image classification. *TPAMI*, 2014. 2

[2] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for image classification. *TPAMI*, 2015. 2

[3] O. Bousquet and L. Bottou. The tradeoffs of large scale learning. In *NIPS*, 2008. 4, 7

[4] X. Chen and A. Gupta. Webly supervised learning of convolutional networks. In *ICCV*, 2015. 2

[5] X. Chen, A. Shrivastava, and A. Gupta. Neil: Extracting visual knowledge from web data. In *ICCV*, 2013. 2

[6] Y.-N. Chen and H.-T. Lin. Feature-aware label space dimension reduction for multi-label classification. In *NIPS*, 2012. 3

[7] 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 *CIVR*, 2009. 3, 6

[8] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Recommender systems*, 2010. 4

[9] J. R. Curran, T. Murphy, and B. Scholz. Minimising semantic drift with mutual exclusion bootstrapping. In *PACL*, 2007. 2

[10] J. Deng, A. C. Berg, K. Li, and L. Fei-Fei. What does classifying more than 10,000 image categories tell us? In *ECCV*. 2010. 2

[11] J. Deng, N. Ding, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, and H. Adam. Large-scale object classification using label relation graphs. In *ECCV*. 2014. 2

[12] S. K. Divvala, A. Farhadi, and C. Guestrin. Learning everything about anything: Webly-supervised visual concept learning. In *CVPR*, 2014. 2

[13] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. *arXiv preprint arXiv:1310.1531*, 2013. 6

[14] C. Fang, H. Jin, J. Yang, and Z. Lin. Collaborative feature learning from social media. In *CVPR*, 2015. 3, 4

[15] J. Feng, H. Xu, and S. Yan. Online robust pca via stochastic optimization. In *NIPS*, 2013. 3, 4, 5

[16] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, T. Mikolov, et al. Devise: A deep visual-semantic embedding model. In *NIPS*, 2013. 1, 2, 4, 6

[17] X. Geng, H. Zhang, J. Bian, and T.-S. Chua. Learning image and user features for recommendation in social networks. In *ICCV*, 2015. 3

[18] S. Guadarrama, E. Rodner, K. Saenko, N. Zhang, R. Farrell, J. Donahue, and T. Darrell. Open-vocabulary object retrieval. In *Robotics Science and Systems (RSS)*, 2014. 1

[19] A. Gupta and L. S. Davis. Beyond nouns: Exploiting prepositions and comparative adjectives for learning visual classifiers. In *ECCV*. 2008. 1

[20] H. Hotelling. Relations between two sets of variates. *Biometrika*, 1936. 3

[21] D. Hsu, S. Kakade, J. Langford, and T. Zhang. Multi-label prediction via compressed sensing. In *NIPS*, 2009. 3

[22] M. Jain, J. C. van Gemert, and C. G. Snoek. What do 15,000 object categories tell us about classifying and localizing actions? In *CVPR*, 2015. 2

[23] Y.-G. Jiang, G. Ye, S.-F. Chang, D. Ellis, and A. C. Loui. Consumer video understanding: A benchmark database and an evaluation of human and machine performance. In *ICMR*, 2011. 6

[24] A. Karpathy, A. Joulin, and F. F. F. Li. Deep fragment embeddings for bidirectional image sentence mapping. In *NIPS*, 2014. 3

[25] R. Kiros, R. Salakhutdinov, and R. S. Zemel. Unifying visual-semantic embeddings with multimodal neural language models. In *NIPS*, 2014. 6

[26] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 2009. 2, 3, 4

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1, 6

[28] A. Lazaridou, G. Dinu, A. Liska, and M. Baroni. From visual attributes to adjectives through decompositional distributional semantics. *TACL*, 2015. 1

[29] Q. Li, J. Wu, and Z. Tu. Harvesting mid-level visual concepts from large-scale internet images. In *CVPR*, 2013. 1

[30] 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 *ECCV*. 2014. 6

[31] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *JMLR*, 2010. 3, 4, 5

[32] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013. 2, 6

[33] V. Ordonez, J. Deng, Y. Choi, A. C. Berg, and T. Berg. From large scale image categorization to entry-level categories. In *ICCV*, 2013. 1

[34] V. Ordonez, G. Kulkarni, and T. L. Berg. Im2text: Describing images using 1 million captioned photographs. In *NIPS*, 2011. 1, 2, 6

[35] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, pages 1–42, 2014. 1

[36] K. Saenko and T. Darrell. Unsupervised learning of visual sense models for polysemous words. In *NIPS*, 2009. 8

[37] R. Salakhutdinov, A. Torralba, and J. Tenenbaum. Learning to share visual appearance for multiclass object detection. In *CVPR*, 2011. 2, 8

[38] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng. Zero-shot learning through cross-modal transfer. In *NIPS*, 2013. 2, 3, 4

[39] N. Srebro, T. Jaakkola, et al. Weighted low-rank approximations. In *ICML*, 2003. 4

[40] H. Steck. Training and testing of recommender systems on data missing not at random. In *KDD*, 2010. 4

[41] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014. 1

[42] F. Tai and H.-T. Lin. Multilabel classification with principal label space transformation. *Neural Computation*, 2012. 3

[43] C.-H. Tsai, C.-Y. Lin, and C.-J. Lin. Incremental and decremental training for linear classification. In *KDD*, 2014. 6

[44] D. Tsai, Y. Jing, Y. Liu, H. Rowley, S. Ioffe, J. M. Rehg, et al. Large-scale image annotation using visual synset. In *ICCV*, 2011. 2

[45] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. *arXiv preprint arXiv:1411.4555*, 2014. 3, 6

[46] J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*, 2011. 1, 3, 6

[47] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *TACL*, 2014. 6

[48] H.-F. Yu, P. Jain, P. Kar, and I. S. Dhillon. Large-scale multi-label learning with missing labels. In *ICML*, 2013. 3, 4, 6

[49] H. Zhang, Y. Yang, H. Luan, S. Yang, and T.-S. Chua. Start from scratch: Towards automatically identifying, modeling, and naming visual attributes. In *MM*, 2014. 1