AdaHGNN: Adaptive Hypergraph Neural Networks for Multi-Label Image Classification

Xiangping Wu[1,2], Qingcai Chen[1,2,3,*], Wei Li[1,2], Yulun Xiao[1,2], Baotian Hu[1,2]
[1] Harbin Institute of Technology, Shenzhen, China
[2] Shenzhen Chinese Calligraphy Digital Simulation Engineering Laboratory
[3] Peng Cheng Laboratory, Shenzhen, China
{wxpleduole, baotian.nlp}@gmail.com, qingcai.chen@hit.edu.cn*, weili_hitwh@163.com, xiaoyulun@stu.hit.edu.cn

ABSTRACT

Multi-label image classification is an important and challenging task in computer vision and multimedia fields. Most of the recent works only capture the pair-wise dependencies among multiple labels through statistical co-occurrence information, which cannot model the high-order semantic relations automatically. In this paper, we propose a high-order semantic learning model based on adaptive hypergraph neural networks (AdaHGNN) to boost multi-label classification performance. Firstly, an adaptive hypergraph is constructed by using label embeddings automatically. Secondly, image features are decoupled into feature vectors corresponding to each label, and hypergraph neural networks (HGNN) are employed to correlate these vectors and explore the high-order semantic interactions. In addition, multi-scale learning is used to reduce sensitivity to object size inconsistencies. Experiments are conducted on four benchmarks: MS-COCO, NUS-WIDE, Visual Genome, and Pascal VOC 2007, which cover large, medium, and small-scale categories. State-of-the-art performances are achieved on three of them. Results and analysis demonstrate that the proposed method has the ability to capture high-order semantic dependencies.

CCS CONCEPTS

• **Computing methodologies** → **Computer vision**; • **Mathematics of computing** → *Hypergraphs*.

KEYWORDS

Multi-label image classification; Hypergraph neural networks; High-order semantic learning; Adaptive hypergraph

ACM Reference Format:
Xiangping Wu[1,2], Qingcai Chen[1,2,3,*], Wei Li[1,2], Yulun Xiao[1,2], Baotian Hu[1,2]. 2020. AdaHGNN: Adaptive Hypergraph Neural Networks for Multi-Label Image Classification. In *Proceedings of the 28th ACM International Conference on Multimedia (MM '20), October 12–16, 2020, Seattle, WA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3394171.3414046

*Corresponding author.

MM '20, October 12–16, 2020, Seattle, WA, USA

ACM ISBN 978-1-4503-7988-5/20/10…$15.00
https://doi.org/10.1145/3394171.3414046

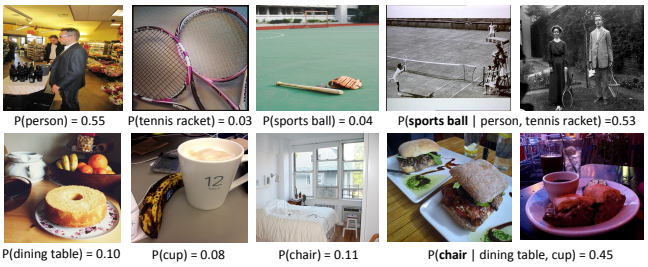

Figure 1: Illustration of high-order label dependencies on the MS-COCO dataset. When "person" and "tennis racket" appear at the same time, "sports ball" comes together with a high probability in an image. Similarly, "chair" co-occurs with "dining table + cup" is as high as 0.45.

1 INTRODUCTION

The multi-label image classification (MLIC) task has attracted important attention in computer vision. It can be widely applied to scene recognition [3, 31, 33], automatic image annotation [19, 36], and human attribute recognition [14, 51]. Unlike single-label image classification, the multi-label task is more challenging due to two main issues: associating multi-labels with image regions and correlation among multiple labels.

For the first issue, some works [41, 45] utilize object detection technologies to extract region proposals and to enhance the feature learning of object region. These works usually need extra bounding box annotations of objects in training, which severely limits the practical application. Some researchers [1, 13, 44, 50] introduce attention mechanisms [43]. These methods capture the associations between image regions and labels with image-level supervision. However, they do not take the label dependencies into account [5].

For the second issue, a popular method is to use recurrent neural networks (RNNs) or long short-term memory (LSTM) to model the label correlations [2, 26]. Though good performance has been achieved, it suffers from the problem that only sequential label relations are modeled [6]. Another popular method is to capture two label dependencies based on the probabilistic graph model [23], such as ChowLiu Tree [7], PLEM [24], etc. These methods utilize label co-occurrence pairs to construct a maximum spanning tree structure for MLIC tasks. However, probabilistic graph models have high computational complexity [38].

To overcome the aforementioned limitations, recent works introduce graph neural networks (GNN) to explicitly model label

correlations, such as ML-GCN [6], knowledge and statistics superimposing network (KSSNet) [38], etc. These methods leverage a graph to model the label correlations from the statistical label co-occurrence or human prior knowledge, which makes a significant improvement in MLIC performance. Even so, existing methods only capture relationships among pair-wise labels and cannot model high-order semantic dependencies [12]. And the hand-crafted correlation graph makes most of these models very inflexible [22]. Real-world objects usually have high-order correlations. For the first row of Fig. 1, we can see that "sports ball" comes together with "person + tennis racket" with a high probability. Similarly, "chair" co-occurs with "dining table + cup" is as high as 0.45 on the MS-COCO dataset. It is obvious that the high-order semantic relations own great potential for improving MLIC performance.

In this paper, we propose an adaptive hypergraph neural network (AdaHGNN) to learn high-order semantic relations. Instead of using statistical co-occurrence information, we utilize label embeddings to automatically construct the adaptive hypergraph. The main contributions of this paper include:

- Propose a model based on hypergraph neural networks for learning high-order semantic relations and guiding label-related feature learning.
- Propose a novel method for constructing an adaptive hypergraph, which is more flexible and effective than hand-crafted methods.
- Experiments are conducted on four benchmarks and state-of-the-art performances are achieved on three of them in multi-label image classification tasks.

2 RELATED WORKS

2.1 Multi-Label Image Classification (MLIC)

Some works use object detection techniques for MLIC tasks. Zhang *et al.* [49] use a regional proposal network like layer to localize the regions corresponding to labels. Yang *et al.* [45] extract object proposals and do single-label classification in each local region. Though these methods can enhance feature learning, they require extra object-level annotations [46]. To overcome this issue, some other works exploit attention mechanisms based on image-level annotations to obtain local information. For instance, Wang *et al.* [39] propose a recurrent memorized-attention module to locate attentional regions. Li *et al.* [21] propose a recurrent highlight network (RHN) to generate candidate glimpses and locate related regions for improving features learning.

Recently, several studies attempt to model label correlations by RNNs or LSTM. Specifically, Hua *et al.* [18] use a bidirectional LSTM-based network to capture the label correlations in both directions. Lyu *et al.* [28] use RNNs to encode the label dependencies sequentially. In addition, the probabilistic graph model is also used to model label dependencies. For instance, a cyclic directed graphical model [15] and a tree-structured graph [24] are proposed to capture label relevance.

2.2 Graph Neural Network (GNN)

With the great success of GNN on visual tasks [40, 42], GNN has been introduced to MLIC and achieved impressive progress. For instance, Chen *et al.* [6] propose a novel graph convolutional network (GCN) based model (ML-GCN) to learn the label relationships. Wang *et al.* [38] add lateral connections between GCN and CNN at different stages to enhance the information transmission of feature learning and label system. A semantic-specific graph representation learning (SSGRL) [5] framework is proposed to explore the interactions between semantics and regions. Despite the significant improvements have been achieved, these methods only capture the relations of two label, which can't model high-order semantic dependencies [12].

Some works [34, 47, 52] introduce the hypergraph structure to model high-order relations among data. These methods treat each sample as one vertex and iteratively optimize each variable by fixing others. Different from these works, we regard each label as one vertex and integrate the adaptive hypergraph into HGNN for end-to-end training. Recently, HGNN [12] is proposed to learn multi-modal and complex data by a hyperedge convolution operation. Unlike an edge in general graphs, which only connect two vertices, a hyperedge in hypergraphs connects two or more vertices. It satisfies the characteristics of high-order relationships in multi-labels. So inspired by [12], this paper proposes adaptive hypergraph neural networks to deal with MLIC tasks. Most of the prior works of GNN and HGNN use the statistical co-occurrence information of two labels to construct the graph manually. And A-GCN [22] uses two 1×1 convolutional layers and a dot product operation to learn the correlation matrix of pair-wise labels with fixed dimensions. Different from these works, we use label embeddings to directly initialize the adaptive hypergraph with an arbitrary number of hyperedges, which can model high-order semantic relations automatically. The adaptive hypergraph not only avoids the inflexibility of hand-crafted correlation graphs but also avoids statistical bias caused by imbalanced labels in the training set.

3 METHOD

In this section, we elaborate on the proposed AdaHGNN model for MLIC tasks, as illustrated in Fig. 2. The overall architecture mainly consists of 3 modules, i.e., the construction of adaptive hypergraph, HGNN module, and multi-scale learning. Among them, the adaptive hypergraph module is proposed to construct and learn label associations. The HGNN module is used to correlate the label-related features and explore semantic interactions. And multi-scale learning is utilized to improve the robustness to object size.

3.1 Construction of Adaptive Hypergraph

Following [12], a hypergraph is defined as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \boldsymbol{W}\}$, where $\mathcal{V}$ denotes a set of vertices, $\mathcal{E}$ denotes a set of hyperedges and $\boldsymbol{W}$ is a diagonal matrix of hyperedges weights, which can be initialized with an identity matrix for meaning equal weights for all hyperedges. A hypergraph incidence matrix is denoted as $\boldsymbol{H} \in \mathbb{R}^{n \times m}$, where n and m denote the number of vertices and hyperedges, respectively. Generally, the distance between two vertices is calculated to build a hypergraph or statistical information is used to build a graph. These methods depend on calculations or statistics information from the training set, which may suffer bias caused by imbalanced labels. To solve this issue, we propose a novel method to construct an adaptive hypergraph.

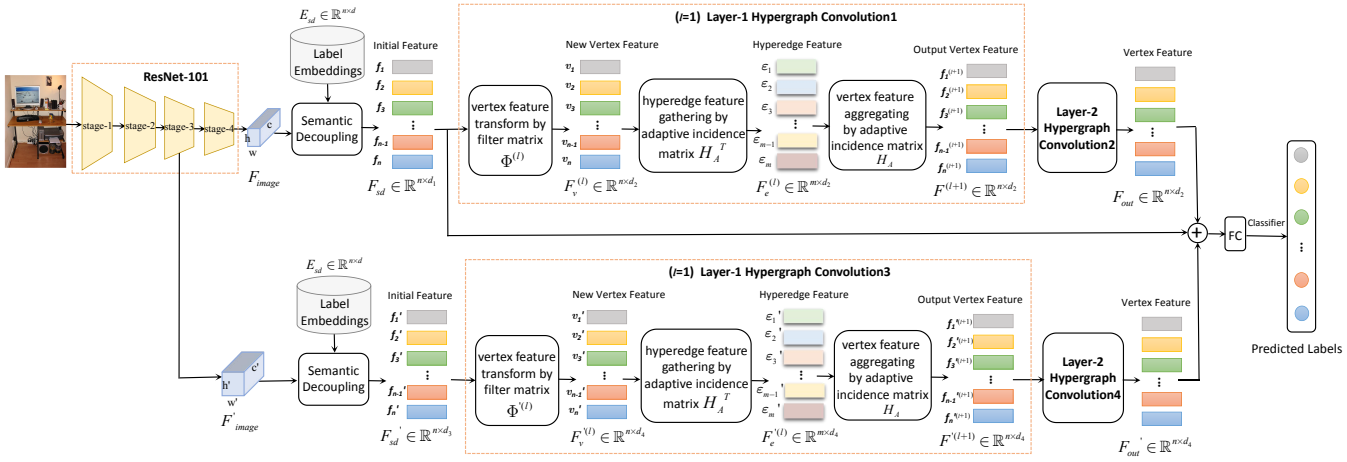

Figure 2: The overall architecture of the proposed AdaHGNN. Firstly, ResNet-101 [16] is employed to extract the image features. A semantic decoupling module [5] is used to decouple the features into label-related feature vectors by label embeddings. Secondly, two-layer hypergraph neural networks based on adaptive hypergraph are employed to correlate the label-related features and explore the high-order semantic interactions. Finally, merge image features of stage-3 and stage-4 to improve the robustness to object size. "⊕" denotes the concatenate operation and "FC" denotes the fully-connected layer.

Hand-Crafted Hypergraph. For comparison, we first introduce the construction of a hand-crafted hypergraph for the MLIC task. Assuming a classification task contains n labels and regards each label as one vertex, the process of hand-crafted hypergraph construction is as follows:

Step 1: construct a probability matrix $\boldsymbol{P} \in \mathbb{R}^{n \times n}$ from the label co-occurrence of the training dataset. The element of probability matrix $\boldsymbol{P}$ can be formulated as

$$P_{ij} = P(L_i|L_j) \tag{1}$$

where $P(L_i|L_j)$ denotes the conditional probability of appearance of label L_i when label L_j appears, $i, j = 1, 2, ..., n$.

Step 2: construct the hypergraph incidence matrix $\boldsymbol{H_S} \in \mathbb{R}^{n \times m}$ with vertex as row and hyperedge as column. Each hyperedge selects one vertex as the center and connects the K nearest neighbors by co-occurrence probability. Let $\boldsymbol{V_j}$ denotes the vertex set of the K nearest neighbors of the j^{th} vertex, then the element of hypergraph incidence matrix $\boldsymbol{H_S}$ can be formulated as

$$[H_S]_{ij} = \begin{cases} P_{ij}, & v_i \in \boldsymbol{V_j} \\ 0, & v_i \notin \boldsymbol{V_j} \end{cases} \tag{2}$$

where v_i denotes the i^{th} vertex, $i = 1, 2, ..., n$ and $j = 1, 2, ..., m$. Here $n = m$, that is, the number of vertices equals the number of hyperedges.

Step 3: add an identity matrix $\boldsymbol{I_n} \in \mathbb{R}^{n \times n}$ to obtain the final hypergraph incidence matrix:

$$\boldsymbol{H_S}' = \boldsymbol{H_S} + \boldsymbol{I_n} \tag{3}$$

Fig. 3 (b) shows the example of hand-crafted hypergraph construction. The hyperedge e_1 takes vertex v_1 as the center and connects other vertices with the co-occurrence probability as the distance. When $K = 2$, v_3 with the probability of 0.67 and v_5 with the probability of 0.5 are the top 2 nearest neighbors of v_1. So in the hypergraph incidence matrix $\boldsymbol{H_S}$, the element values of v_3 and v_5 corresponding to column e_1 are 0.67 and 0.5 respectively, while others are equal to 0. In this way, we finally get the hand-crafted hypergraph based on the statistical co-occurrence information. However, the number of the hyperedge suffers limitation and the hyperparameter K has an impact on the construction of hypergraph.

Adaptive Hypergraph. To address the above issue, this paper proposes an automatic learning method of the incidence matrix to construct the adaptive hypergraph. It is impossible that all label relationships can be obtained by statistical co-occurrence information. Therefore, we redefine the hyperedge of hypergraphs, which no longer centers on a vertex to connect its K nearest neighbor vertices. We define a hyperedge as a kind of abstract relation between vertices, rather than a specific prior relationship. Let $\boldsymbol{\mathcal{E}}$ be a set of hyperedge in a hypergraph, then hyperedges can be denoted as:

$$\boldsymbol{\mathcal{E}} = \{\boldsymbol{e_1}, \boldsymbol{e_2}, ..., \boldsymbol{e_m}\}, m > 0 \tag{4}$$

where m is the number of hyperedges. Each hyperedge indicates a potential relationship between two or more vertices, which can be learned automatically in the training phase. Then the adaptive hypergraph incidence matrix $\boldsymbol{H_A} \in \mathbb{R}^{n \times m}$ can be obtained by aggregating the learnable hyperedges. We can choose random values or label embeddings to initialize $\boldsymbol{H_A}$. To accelerate the convergence, this paper uses label embeddings as initialization. The label embeddings $\boldsymbol{E} \in \mathbb{R}^{n \times m}$ can be denoted as

$$\boldsymbol{E} = \{\boldsymbol{b_1}, ..., \boldsymbol{b_i}, ..., \boldsymbol{b_m}\}, m > 0 \tag{5}$$

where n is the number of labels and m denotes the dimensionality of the embedding. $\boldsymbol{b_i}$ denotes the i^{th} dimension vector of the label embedding. They can be obtained by the pre-trained word embedding, such as GloVe [30], BERT [10], etc. Intuitively, each dimension of the label embedding indicates an attribute or relation of labels, which is consistent with the characteristics of the hyperedge. So

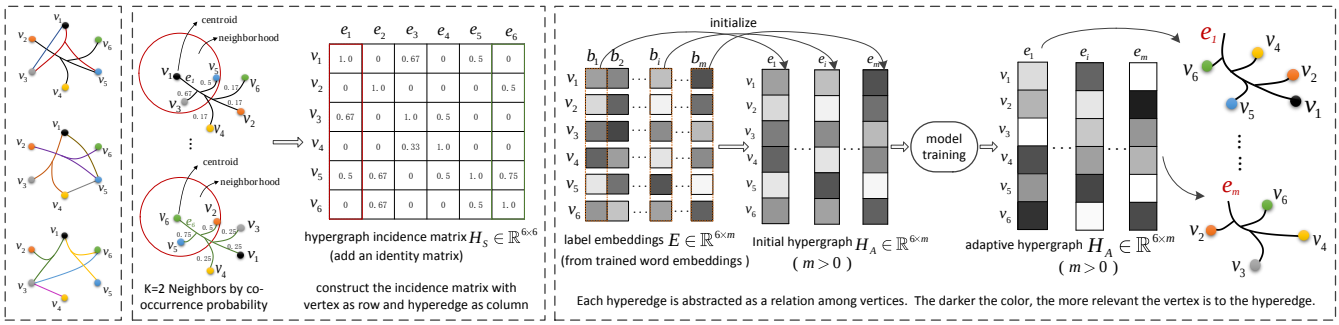

Figure 3: The comparison of hand-crafted and adaptive hypergraph construction processes. (a) The input data examples with 6 objects and 11 examples. Each colored dot denotes an object label. The same color of edges connect the labels co-occurred in the same image sample. (b) For each vertex, aggregate its $n-1$ neighbor vertices by co-occurrence probability to generate a hyperedge. Then select the K nearest neighbors to construct the hypergraph incidence matrix. The red circle region denotes the neighborhood of $K=2$. The greater the probability, the closer they are. (c) The adaptive hypergraph is initialized by label embedding and automatically learned during training.

it is meaningful to use label embedding to initialize the adaptive hypergraph incidence matrix $\boldsymbol{H_A}$. Fig. 3 (c) shows an example of the construction of an adaptive hypergraph. There are 6 vertices and m hyperedges. Label embedding is used to initialize the hypergraph incidence matrix. And the adaptive hypergraph can be learned after model training. The darker the color, the stronger the correlation. Take e_m as an example, we can draw its potential relationship with four vertices (v_2, v_6, v_3, v_4) according to the degree of correlation. The closer the distance, the stronger the association.

3.2 Feature-Semantics Interaction through HGNN

The overall framework of our approach is shown in Fig. 2. Given an input image I, ResNet-101 [16] is utilized to extract the image feature maps. The last average pooling layer has a size of 2×2 and a stride of 2. Then the output of stage-4 is $\boldsymbol{F}_{image} \in \mathbb{R}^{w\times h\times c}$, where w, h and c denote the width, height and the channel of the feature maps, respectively. To learn label-related features, a semantic decoupling module [5] is employed to decouple image features into the semantic-specific feature representation by the label embeddings. The module uses a low-rank bilinear pooling method and an attention function to calculate the attention coefficient. The label embeddings $\boldsymbol{E} \in \mathbb{R}^{n\times d}$ are obtained by the pre-trained GloVe [30] model. Then we can get the semantic-specific feature representation $\boldsymbol{F_{sd}} \in \mathbb{R}^{n\times d_1}$, $\boldsymbol{F_{sd}} = \{\boldsymbol{f_1}, \boldsymbol{f_2}, ..., \boldsymbol{f_n}\}$, where n denotes the number of labels, $\boldsymbol{f_i}$ denotes the feature vector related to label i, and d_1 denotes the dimensionality of feature vector $\boldsymbol{f_i}$.

To capture high-order semantic relations automatically, the two-layer hypergraph neural networks with adaptive hypergraphs are used to correlate the feature vectors and explore semantic dependencies. Let $\boldsymbol{\Phi}^{(l)}$ be the learnable filter matrix of hypergraph neural network at l layer, $\boldsymbol{F}^{(l)}$ be the vertex features of hypergraph at l layer, then a hypergraph convolutional layer [12] $HConv(\boldsymbol{F}, \boldsymbol{W}, \boldsymbol{\Phi})$ can be formulated as

$$\boldsymbol{F}^{(l+1)} = \sigma(\boldsymbol{D_v}^{-1/2}\boldsymbol{H_A}\boldsymbol{W}\boldsymbol{D_e}^{-1}\boldsymbol{H_A}^{T}\boldsymbol{D_v}^{-1/2}\boldsymbol{F}^{(l)}\boldsymbol{\Phi}^{(l)}) \tag{6}$$

where $\sigma(.)$ is the nonlinear activation function, $\boldsymbol{H_A}$ is the adaptive hypergraph incidence matrix, $\boldsymbol{D_e}$ and $\boldsymbol{D_v}$ denote the degrees of the edge and the degrees of vertex, respectively. They are used for normalization. For a vertex $v \in \mathcal{V}$ and a hyperedge $e \in \mathcal{E}$, their degrees can be calculated by

$$d(v) = \sum_{e\in\mathcal{E}} w(e)a(v,e), \quad d(e) = \sum_{v\in\mathcal{V}} a(v,e) \tag{7}$$

where $w(e)$ is an element on the diagonal of matrix $\boldsymbol{W}$ and indicates the weight of hyperedge e. $a(v,e)$ is the element of $\boldsymbol{H_A}$. According to Equation 6, the process of a hypergraph convolution layer is as follows. Firstly, the learnable filter matrix at l layer $\boldsymbol{\Phi}^{(l)}$ is used to transform the l layer vertex features $\boldsymbol{F}^{(l)}$ to the new vertex features. Note that the initial vertex features $\boldsymbol{F}^{(1)}$ is the output of the semantic decoupling module. Secondly, the new vertex features on the hyperedges are gathered to obtain the hyperedge features by the multiplication of $\boldsymbol{H_A}^{T}$. Finally, the related hyperedge features are associated to obtain the final vertex features $\boldsymbol{F}^{(l+1)}$, which is implemented by multiplying matrix $\boldsymbol{H_A}$. Through the vertex-hyperedge-vertex transform, HGNN can effectively capture semantic dependencies and explore the interaction between features and semantics.

3.3 Multi-Scale Learning

To be more robust to object size, we propose the multi-scale learning by using the image features of stage-3. Different from other multi-scale feature fusion methods, such as MS-CMA [46], which average the predicted probability score of classifier of multi-scale features, while we fuse multi-scale results at the output of hypergraph. Let $\boldsymbol{F'}_{image} \in \mathbb{R}^{w'\times h'\times c'}$ be the output of stage-3, and go through the semantic decoupling module and two-layer HGNN. Then the final vertex features $\boldsymbol{F'_{out}}$ of stage-3 can be obtained. Following [5], we also concatenate the output $\boldsymbol{F_{sd}}$ of the semantic decoupling module of stage-4 with the final vertex features. Then the final vertex features of multi-stage learning can be written as

$$\boldsymbol{F_{ms}} = \boldsymbol{F_{sd}} \oplus \boldsymbol{F_{out}} \oplus \boldsymbol{F'_{out}} \tag{8}$$

where $\oplus$ denotes the concatenate operation by channels. Finally, the vertices are classified after two fully connected layers.

3.4 Optimization

Let the training set be $\boldsymbol{X} = \{I_i, \boldsymbol{y_i}\}$, here $i = 1, 2, ..., T$, T is the number of images in training set. I_i and $\boldsymbol{y_i}$ are the i^{th} input image and ground truth, respectively. For the MLIC task with n objects, $\boldsymbol{y_i} = \{y_{i1}, ..., y_{ij}, ..., y_{in}\}$, y_{ij} is equal to 1 if the ground truth contains the label j and 0 otherwise. We employ the cross entropy as the loss function:

$$L_{entropy} = \sum_{i=1}^{T} \sum_{j=1}^{n} y_{ij} \log p_{ij} + (1 - y_{ij}) \log(1 - p_{ij}) \tag{9}$$

where p_{ij} is the probability of the classifier via a sigmoid function.

4 EXPERIMENTS

In this section, we conduct experiments on four benchmarks: MS-COCO [25], NUS-Wide [8], Pascal VOC 2007 [11], and Visual Genome 500 [20]. These benchmarks cover large, medium and small-scale categories. Experimental results demonstrate the effectiveness and generality of AdaHGNN.

4.1 Evaluation Metrics

To fairly compare with existing approaches, we report the mean average precision (***mAP***) over all labels on all datasets. ***mAP*** is a key metric used for MLIC tasks, which is more important than other metrics. Following [27] on the MS-COCO dataset, we further present the average per-label F1 (CF1) and the average overall F1 (OF1). The results of top-3 labels are also reported. Following [5] on Pascal VOC 2007 dataset, we further present the average precision (AP) of each label.

4.2 Datasets

MS-COCO. Microsoft COCO [25] is a widely used dataset for MLIC tasks. It contains 82,081 training images and 40,137 validation images for testing. There are 80 labels on the dataset and about 2.9 labels per image.

NUS-Wide. NUS-Wide [8] is a web dataset from Flickr, which contains 269,648 images and 5018 labels. Following the processing and split used in [46], we use 150,000 images for training and 59,347 for testing. There are 81 labels with 2.4 labels per image on average.

Visual Genome. Visual Genome [20] is a dataset with 80,138 labels. Because most labels contain very few images, we follow [5] to use a VG-500 subset. The VG-500 subset contains 108,249 images and 500 labels. Following the split of [5], there are 10,000 images for testing and the rest 98,249 images for training.

Pascal VOC 2007. Pascal VOC 2007 [11] is also a popular dataset for MLIC tasks. It contains 5,011 images for training and 4,952 images for testing. The Pascal VOC 2007 dataset only contains 20 labels, which is a small-scale dataset.

4.3 Implementation Details

Data Preprocessing. Data preprocessing is the same for all datasets. During training, we first resize the input image to 640×640 and randomly select a number from {640, 576, 512, 384, 320} as the height and width to randomly crop the resized image. Finally, the training image can be obtained by further resizing the cropped patches to 576 × 576. During testing, we simply resize the test image to 640 × 640 for evaluation.

Experimental settings. All experiments on AdaHGNN are optimized by ADAM algorithm with momentums of 0.999 and 0.9. We employ ResNet-101 pre-trained on ImageNet dataset [9] to extract image features and freeze the parameters of stage-1 and stage-2. The learning rate is initialized to 10^{-5}, and it is divided by 10 when the error stops dropping. 300-dim GloVe is used as word embeddings to obtain label representation. When a label contains multiple words, the average of word embeddings for all words in it is used as the label embeddings. The final vertex features $\boldsymbol{F_{ms}}$ of multi-scale learning are classified through 5,120-to-2,048 and 2,048-to-1 fully connected layers. The proposed AdaHGNN is trained in an end-to-end manner with a batch size of 4. Other parameters are as follows: $d = 300$, $d_1 = 2048$, $d_2 = 2048$, $d_3 = 1024$, $d_4 = 1024$.

4.4 Compared Methods

To evaluate the effectiveness of the proposed AdaHGNN, we compare it with the following state-of-the-art methods, which can be grouped into two main categories:

(1) Classical deep learning methods. **CNN-RNN** [35] and **CNN-SREL-RNN** [26] use CNN to encode the image and RNN to decode it into sequences. **RNN-Attention** [39] and **Order-Free RNN** [4] are based on a LSTM framework with the attention module. **RHN-GRRE** [21] employs RHN to consider related regions and a gated recurrent relation extractor (GRRE) to capture the label correlation. **ResNet-SRN** [50] develops a spatial regularization network to explore semantic and spatial relations of labels. **Multi-Evidence** [13] proposes a weakly supervised curriculum learning method for MLIC tasks. **CMA** and **MS-CMA** [46] utilize cross-modality attention module to generate semantic attention maps. **KD-WSD** [27], **Distillation** [17], **FitsNet** [32], and **Attention transfer** [48] focus on knowledge distillation. **FeV+LV** [45] extracts object proposals to obtain local information. **RCP** [37] proposes an object-proposal-free framework based on random crop pooling.

(2) GNN based methods. **ML-GCN** [6] employs GCN to capture label dependencies by a re-weighted correlation matrix. **A-GCN** [22] uses an adaptive label graph module to a learn adjacent matrix with fixed dimension. **Attention-GCN** [29] employs attention mechanism to associate labels and regions and uses GCN to learn label dependencies. **KSSNet** [38] adds knowledge prior graph and lateral connections between CNN and GCN. **SSGRL** [5] uses a semantic decoupling module to guide the feature learning and a semantic interaction module to capture correlations.

4.5 Experimental Results

MS-COCO. We compare the proposed AdaHGNN with state-of-the-art methods and report the main results in Table 1. It can be observed that AdaHGNN achieves the best performances at all evaluation metrics. Compared with CNN-RNN [35], CNN-SREL-RNN [26], Order-Free RNN [4], RNN-Attention [39], RHN-GRRE [21] and ResNet-SRN [50], which rely on RNN framework, AdaHGNN outperforms them by significant margins. In comparison with weakly-supervised learning (KD-WSD [27] and Multi-Evidence [13]), better

Table 1: Performance comparisons (%) between state-of-the-art methods and AdaHGNN on MS-COCO dataset. Upper part presents the results of classical deep learning methods and lower part presents GNN based methods.

Method		All		Top-3	
	mAP	CF1	OF1	CF1	OF1
CNN-RNN [35]	61.2	-	-	60.4	67.8
CNN-SREL-RNN [26]	-	63.4	72.5	-	-
Order-Free RNN [4]	-	-	-	62.1	67.7
RNN-Attention [39]	-	-	-	67.4	72.0
RHN-GRRE [21]	66.7	-	-	65.2	71.8
KD-WSD [27]	74.6	69.2	74.0	66.8	72.7
ResNet-SRN [50]	77.1	71.2	75.8	67.4	72.9
ResNet101-ACfs [14]	77.5	72.2	76.3	68.0	73.1
Multi-Evidence [13]	-	74.9	78.4	70.6	74.7
CMA [46]	82.8	77.5	80.9	73.8	77.0
MS-CMA [46]	83.8	78.4	81.0	74.9	77.1
ML-GCN [6]	83.0	78.0	80.3	74.6	76.7
A-GCN [22]	83.1	78.0	80.3	74.6	76.6
Attention-GCN [29]	83.3	78.0	80.7	74.4	77.0
KSSNet [38]	83.7	77.2	81.5	-	-
SSGRL [5]	83.8	76.8	79.7	72.7	76.2
AdaHGNN (Ours)	**85.0**	**79.9**	**81.8**	**75.5**	**77.6**

performances are also obtained on AdaHGNN. Moreover, AdaHGNN exceeds the previous state-of-the-art result (MS-CMA: 83.8%) by 1.2% on mAP. From the lower part of Table 1, comparing to other GNN-based methods, e.g., ML-GCN [6], A-GCN [22], Attention-GCN [29], KSSNet [38] and SSGRL [5], AdaHGNN also achieves a notable performance improvement on all metrics, which demonstrates the effectiveness of our model for multi-label image classification. Specifically, it is 1.2%, 3.1%, 2.1%, 2.8% and 1.4% higher than the previous best GNN-based method SSGRL [5] on mAP, CF1 (all), OF1 (all), CF1 (Top-3), and OF1 (Top-3), respectively.

NUS-WIDE. The comparison results on the NUS-WIDE dataset are also shown in Table 2. Comparing to distillation-based methods, e.g., Distillation [17], FitsNet [32], Attention transfer [48] and KD-WSD [27], AdaHGNN improves by at least 2.2%. AdaHGNN also outperforms the latest model MS-CMA [46] by achieving 0.9% gain. On the NUS-WIDE dataset, AdaHGNN ranks second, which is 1.2% lower than RHN-GRRE [21]. However, on the MS-COCO dataset, the proposed AdaHGNN (85.0%) outperforms RHN-GRRE [21] (66.7%) by a large margin of 18.3% on mAP. The possible reason for such performance difference is that, on the NUS-WIDE dataset, the labels may have the hyponymy relation, e.g., "cat, dog, cow" are all belonged to the concept "animal". For images only containing "cat", some ground truths are "cat" and some are "cat, animal", which greatly affect the performance of the proposed method since it pays more attention to the modeling of the semantic relationships among multiple labels.

VG-500. To evaluate the performance of large-scale categories classification, we conduct experiments on the VG-500 dataset, as depicted in Table 2. The proposed AdaHGNN performs much better than existing baseline methods (ResNet-101 and ResNet-SRN [50])

Table 2: Performance of mAP (%) between state-of-the-art methods and AdaHGNN on NUS-WIDE and VG-500 dataset.

Method	NUS-WIDE	VG-500
CNN-RNN [35]	56.1	-
Distillation [17]	57.2	-
FitsNet [32]	57.4	-
Attention transfer [48]	57.6	-
KD-WSD [27]	60.1	-
RHN-GRRE [21]	**63.5**	-
CMA [46]	60.8	-
MS-CMA [46]	61.4	-
ResNet-101 [16]	-	30.9
ResNet-SRN [50]	-	33.5
ML-GCN [6]	-	33.8
SSGRL [5]	-	<u>36.6</u>
AdaHGNN (Ours)	<u>62.3</u>	**38.2**

by achieving 4.7% mAP improvement. Comparing to GNN-based models, e.g., ML-GCN [6] and SSGRL [5], AdaHGNN achieves the mAP of 38.2%, which exceeds the previous state-of-the-art by 1.6%. The performance gain on the VG-500 dataset indicates that our AdaHGNN model is suitable for recognizing large-scale categories.

Pascal VOC 2007. Table 3 shows the comparisons of average precision (AP) of each label and mean average precision (mAP) with state-of-the-art methods on the Pascal VOC 2007 dataset. Following SSGRL [5], we also pre-train the AdaHGNN model on the COCO dataset. It can be observed that of the 20 categories, we have 17 with the best on AP. Comparing to SSGRL [5], AdaHGNN gets the absolute performance gain of 0.20% on mAP, i.e., error relative dropping of 4.0%. The results show that AdaHGNN is also effective for small data with small-scale labels.

4.6 Ablation Studies

To demonstrate the effectiveness and influence of each component of the AdaHGNN, we perform a series of ablation experiments on the MS-COCO dataset. We evaluate the effect of the HGNN module, the adaptive hypergraph module, the multi-scale learning module, and the different initialization methods of adaptive Hypergraph.

4.6.1 Effect of the HGNN Module. To explore the effect of the HGNN module, we replace the HGNN module in the AdaHGNN model by the GNN module. The GNN module is following SSGRL [5] but without gated recurrent update mechanism. The graph construction is based on the statistical label information from training data. As shown in Table 4, the mAP drops from 85.0% to 84.2% when GNN is used instead of HGNN. The result demonstrates that HGNN is superior to GNN in exploring label associations and interactions. The possible reason is that GNN only captures the label pair relationship, while HGNN can discover high-order semantic relations of multi-labels.

4.6.2 Effect of Adaptive Hypergraph. To evaluate the effectiveness of the proposed adaptive hypergraph, we compare different construction methods of hypergraph, as depicted in Table 5. The handcrafted hypergraph is constructed as described in section 3.1 and

Table 3: Comparisons of AP and mAP (%) with state-of-the-art methods on the Pascal VOC 2007 dataset. Upper part presents the results of classical deep learning methods and lower part presents GNN based methods.

Method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motor	person	plant	sheep	sofa	train	tv	**mAP**
CNN-RNN [35]	96.7	83.1	94.2	92.8	61.2	82.1	89.1	94.2	64.2	83.6	70.0	92.4	91.7	84.2	93.7	59.8	93.2	75.3	99.7	78.6	84.0
ResNet-101 [16]	99.5	97.7	97.8	96.4	65.7	91.8	96.1	97.6	74.2	80.9	85.0	98.4	96.5	95.9	98.4	70.1	88.3	80.2	98.9	89.2	89.9
FeV+LV [45]	97.9	97.0	96.6	94.6	73.6	93.9	96.5	95.5	73.7	90.3	82.8	95.4	97.7	95.9	98.6	98.6	77.6	88.7	98.3	89.0	90.6
RNN-Attention [39]	98.6	97.4	96.3	96.2	75.2	92.4	96.5	97.1	76.5	92.0	87.7	96.8	97.5	93.8	98.5	81.6	93.7	82.8	98.6	89.3	91.9
RCP [37]	99.3	97.6	98.0	96.4	79.3	93.8	96.6	97.1	78.0	88.7	87.1	97.1	96.3	95.4	99.1	82.1	93.6	82.2	98.4	92.8	92.5
ML-GCN (Binary) [6]	99.6	98.3	97.9	97.6	78.2	92.3	97.4	97.4	79.2	94.4	86.5	97.4	97.9	97.1	98.7	84.6	95.3	83.0	98.6	90.4	93.1
ML-GCN (Re-weighted) [6]	99.5	98.5	98.6	98.1	80.8	94.6	97.2	98.2	**82.3**	95.7	86.4	98.2	98.4	96.7	99.0	84.7	96.7	84.3	98.9	93.7	94.0
SSGRL [5]	99.7	98.4	98.0	97.6	85.7	96.2	98.2	98.8	82.0	98.1	**89.7**	98.8	98.7	97.0	99.0	86.9	98.1	**85.8**	99.0	93.7	95.0
AdaHGNN (Ours)	**99.8**	**98.9**	**98.9**	**98.1**	**88.1**	**97.3**	**98.3**	**98.8**	81.7	**98.2**	87.2	**99.1**	**99.3**	**97.7**	**99.1**	**87.8**	**98.4**	82.5	**99.7**	**94.5**	**95.2**

Table 4: Performance comparisons (%) between HGNN and GNN in the AdaHGNN model on the MS-COCO dataset.

Method		All		Top-3	
	mAP	CF1	OF1	CF1	OF1
GNN	84.2	78.8	81.4	74.8	77.3
HGNN	**85.0**	**79.9**	**81.8**	**75.5**	**77.6**

Table 5: Comparisons (%) with several construction methods of hypergraph on the MS-COCO dataset.

Method		All		Top-3	
	mAP	CF1	OF1	CF1	OF1
Hand-crafted hypergraph	84.8	79.3	81.6	75.2	77.3
Static hypergraph	84.8	79.8	81.6	75.1	77.2
Adaptive hypergraph	**85.0**	**79.9**	**81.8**	**75.5**	**77.6**

Table 6: Performance of different scales of AdaHGNN on the MS-COCO dataset.

Method		All		Top-3	
	mAP	CF1	OF1	CF1	OF1
Stage-4	84.5	79.1	81.2	75.2	77.2
Stage-3,4	**85.0**	**79.9**	**81.8**	**75.5**	**77.6**
Stage-2,3,4	84.9	79.6	81.8	75.3	77.5

K is set to 80. The static hypergraph is initialized with label embedding, the parameters of the hypergraph incidence matrix are frozen and not learnable in the training process. From Table 5, it is obvious to see that the proposed adaptive hypergraph performs better than other construction methods at all evaluation metrics, which indicates the effectiveness of adaptive hypergraph.

4.6.3 Effect of Multi-Scale Learning. The comparison result of different scales is shown in Table 6. When the features of the stage are used, the network parameters of the stage are no frozen and but learnable. From Table 6 we can see that "Stage-3,4" achieves the best performance, which obtains 0.5% improvement comparing to single-scale learning. When using three stages of fusion, the possible reason of performance dropping is that when more stage features are used, the more network parameters are, which result in difficulty to train and possible over-fitting. We also use the general metric FPS (frames per second) to measure the inference time. On the machine with 1 GPU of Nvidia V100, the average inference time of stage-3,4 and stage-4 are 34 FPS and 39 FPS, respectively. It shows that the multi-scale learning achieves a 0.5% improvement with the acceptable cost of inference time.

Table 7: Performance of different initialization methods of adaptive Hypergraph on the MS-COCO dataset.

Method		All		Top-3	
	mAP	CF1	OF1	CF1	OF1
GloVe-300 (AdaHGNN)	85.0	79.9	81.8	75.5	77.6
Random-300	84.9	79.5	81.6	75.1	77.3
BERT-base-768	85.1	79.8	81.9	75.5	77.6
Random-768	85.0	79.7	81.8	75.5	77.6
BERT-large-1024	85.1	79.6	81.9	75.6	77.6
Random-1024	84.9	79.6	81.6	75.2	77.3

4.6.4 Effect of different initialization methods of adaptive Hypergraph. The results of different initialization methods are depicted in Table 7. Three pre-trained word embeddings including 300-dim GloVe [30], 768-dim BERT-base [10], and 1024-dim BERT-large [10] are used to obtain label embedding. Random vectors of the same dimension are also used for comparison. The effectiveness of random initialization proves that adaptive hypergraph can be learned in the training process. The label embedding method is slightly better than the random initialization method. The possible reason is that the label embedding contains external prior knowledge. In addition, the number of hyperedges does not have a great impact on performance. The possible reason is that the relationship is relatively simple on the MS-COCO dataset. Because there are only 80 labels, each image is about 2.9 labels.

4.7 Visualization and Analysis

4.7.1 Component Analysis. Fig. 4 visualizes the results of replacing different components on AdaHGNN. Columns 3, 4, and 5 show the results of using GNN instead of HGNN, hand-crafted hypergraphs instead of adaptive hypergraphs, and single-scale learning (stage-4) instead of multi-scale learning (stage-3,4) on AdaHGNN, respectively. From the last row, we can see that the highlighted regions of the semantic feature maps on GNN, hand-crafted hypergraphs,

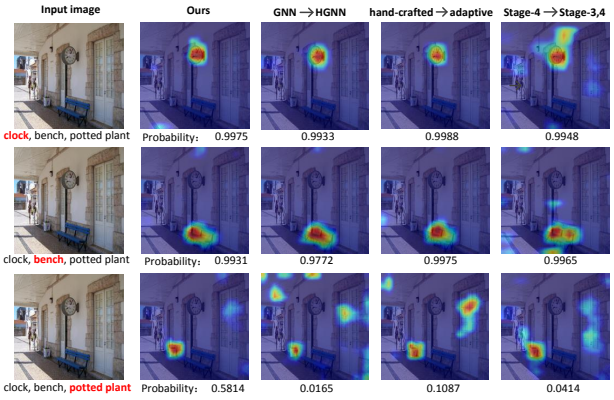

Figure 4: The visualization of replacing different components of AdaHGNN on MS-COCO dataset. In each row, the red label indicates the label corresponding to the semantic feature maps of that row. "module A-> module B" indicates that on AdaHGNN, replace the module B with module A. The number below the image indicates the prediction probability of the corresponding label.

Input Image	Ground Truth	AdaHGNN (ours)	ML-GCN [6]	SSGRL [5]
	person tennis racket sports ball chair	person tennis racket sports ball chair	person tennis racket skateboard	person tennis racket skateboard chair
	person tennis racket sports ball	person tennis racket sports ball	person tennis racket tie	person tennis racket tie
	dining table cup chair bowl sandwich	dining table cup chair bowl sandwich	dining table cup bowl cake knife	dining table cup bowl cake sandwich
	dining table cup chair sandwich knife	dining table cup chair sandwich knife	dining table cup sandwich fork knife person	dining table cup sandwich fork knife cell phone

Figure 5: Comparison of capability for high-order semantic association capturing on MS-COCO dataset. Column 1, 2 are input images and ground truth. Column 3-5 are the predicted labels of different methods. The red-colored text indicates classification errors. The green-colored text indicates the labels predicted according to the higher-order relation.

and stage-4 are not very accurate for the label "potted plant". And the confidences of the corresponding label are relatively low. It demonstrates the effectiveness of each component of AdaHGNN.

4.7.2 Comparison Analysis. Fig. 6 shows the predicted labels and the corresponding semantic feature maps of AdaHGNN, ML-GCN [6], and SSGRL [5]. The predicted labels with confidence greater than 0.5 are positive. They are sorted by confidence from high to low. It can be observed that the predicted labels and corresponding highlighted regions of AdaHGNN are more consistent and accurate than those of ML-GCN [6] and SSGRL [5]. Moreover, the semantic-aware areas of AdaHGNN are more concentrated and not scattered. It shows that AdaHGNN is superior to the existing methods in learning label-related features and exploring label and region interaction.

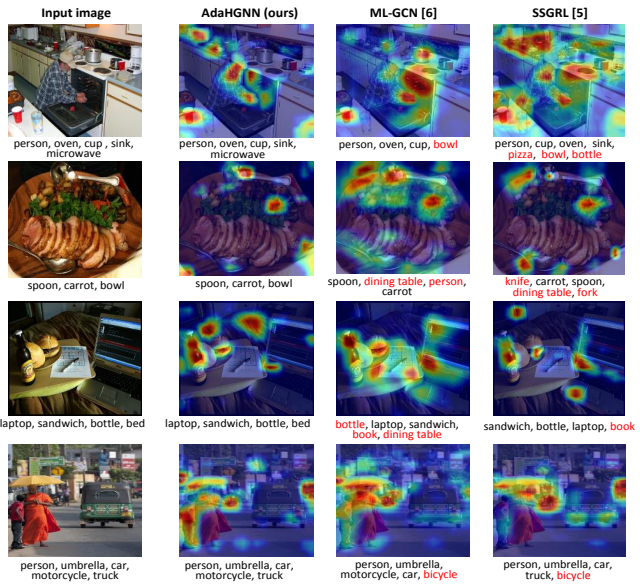

Figure 6: Compare the visualization results of different methods on MS-COCO dataset. Column 1 is the input image and ground truth labels, and column 2-4 are the predicted labels and the corresponding semantic feature maps. Red text indicates classification errors.

4.7.3 High-order Semantic Association Analysis. We further analyze the ability of AdaHGNN to capture high-order semantic associations. Fig. 5 shows the predicted labels of AdaHGNN, ML-GCN [6], and SSGRL [5] on MS-COCO dataset, which are sorted by confidence from high to low. According to the probability of statistics from the training set in Fig. 1, we know that the co-occurrence probability of "sports ball" and "person + tennis racket" is 0.53. And "chair" co-occurs with "dining table + cup" is as high as 0.45. However, when the conditions are met, ML-GCN [6] and SSGRL [5] can't correctly recognize "sports ball" and "chair" while AdaHGNN can. It demonstrates that AdaHGNN has the ability to capture high-order semantic associations.

5 CONCLUSION

In this paper, to capture the high-order semantic relations among multi-labels, we propose a novel hypergraph neural networks based model for multi-label image classification. To overcome the limitation of the hand-crafted hypergraph constructing method, an automatic hypergraph learning mechanism is also proposed based on label embeddings. Extensive experiments are conducted on four public benchmarks, and new state-of-the-art performances are achieved on three of them. Further analysis demonstrates the efficacy of AdaHGNN on modeling high-order semantic associations.

ACKNOWLEDGMENTS

This work was supported by the Natural Science Foundation of China (Grant No. 61872113), and Strategic Emerging Industry Development Special Funds of Shenzhen (Grant No. XMHT20190108009, JCYJ20190806112210067).

REFERENCES

[1] Aaliyah Alshehri, Yakoub Bazi, Nassim Ammour, Haidar Almubarak, and Naif Alajlan. 2019. Deep Attention Neural Network for Multi-Label Classification in Unmanned Aerial Vehicle Imagery. *IEEE Access* 7 (2019), 119873–119880.

[2] Long Chen, Ronggui Wang, Juan Yang, Lixia Xue, and Min Hu. 2019. Multi-label image classification with recurrently learning semantic dependencies. *The Visual Computer* 35, 10 (2019), 1361–1371.

[3] Long Chen, Wujing Zhan, Wei Tian, Yuhang He, and Qin Zou. 2019. Deep Integration: A Multi-Label Architecture for Road Scene Recognition. *IEEE Transactions on Image Processing* 28, 10 (2019), 4883–4898.

[4] Shang-Fu Chen, Yi-Chen Chen, Chih-Kuan Yeh, and Yu-Chiang Frank Wang. 2018. Order-free RNN with visual attention for multi-label classification. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[5] Tianshui Chen, Muxin Xu, Xiaolu Hui, Hefeng Wu, and Liang Lin. 2019. Learning semantic-specific graph representation for multi-label image recognition. In *Proceedings of the IEEE International Conference on Computer Vision*. 522–531.

[6] Zhao-Min Chen, Xiu-Shen Wei, Peng Wang, and Yanwen Guo. 2019. Multi-label image recognition with graph convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 5177–5186.

[7] C Chow and Cong Liu. 1968. Approximating discrete probability distributions with dependence trees. *IEEE transactions on Information Theory* 14, 3 (1968), 462–467.

[8] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yantao Zheng. 2009. NUS-WIDE: a real-world web image database from National University of Singapore. In *Proceedings of the ACM international conference on image and video retrieval*. 1–9.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 248–255.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 4171–4186.

[11] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2010. The pascal visual object classes (voc) challenge. *International journal of computer vision* 88, 2 (2010), 303–338.

[12] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 3558–3565.

[13] Weifeng Ge, Sibei Yang, and Yizhou Yu. 2018. Multi-evidence filtering and fusion for multi-label classification, object detection and semantic segmentation based on weakly supervised learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1277–1286.

[14] Hao Guo, Kang Zheng, Xiaochuan Fan, Hongkai Yu, and Song Wang. 2019. Visual attention consistency under image transforms for multi-label image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 729–739.

[15] Yuhong Guo and Suicheng Gu. 2011. Multi-label classification using conditional dependency networks. In *Twenty-Second International Joint Conference on Artificial Intelligence*.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[17] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[18] Yuansheng Hua, Lichao Mou, and Xiao Xiang Zhu. 2019. Recurrently exploring class-wise attention in a hybrid convolutional and bidirectional LSTM network for multi-label aerial image classification. *ISPRS journal of photogrammetry and remote sensing* 149 (2019), 188–199.

[19] Xiao-Yuan Jing, Fei Wu, Zhiqiang Li, Ruimin Hu, and David Zhang. 2016. Multi-label dictionary learning for image annotation. *IEEE Transactions on Image Processing* 25, 6 (2016), 2712–2725.

[20] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision* 123, 1 (2017), 32–73.

[21] Liang Li, Shuhui Wang, Shuqiang Jiang, and Qingming Huang. 2018. Attentive recurrent neural network for weak-supervised multi-label image classification. In *Proceedings of the 26th ACM international conference on Multimedia*. 1092–1100.

[22] Qing Li, Xiaojiang Peng, Yu Qiao, and Qiang Peng. 2019. Learning Category Correlations for Multi-label Image Recognition with Graph Networks. *arXiv preprint arXiv:1909.13005* (2019).

[23] Qiang Li, Maoying Qiao, Wei Bian, and Dacheng Tao. 2016. Conditional graphical lasso for multi-label image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2977–2986.

[24] Xin Li, Feipeng Zhao, and Yuhong Guo. 2014. Multi-label Image Classification with A Probabilistic Label Enhancement Model. In *UAI*, Vol. 1. 3.

[25] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer, 740–755.

[26] Feng Liu, Tao Xiang, Timothy M Hospedales, Wankou Yang, and Changyin Sun. 2017. Semantic regularisation for recurrent image annotation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2872–2880.

[27] Yongcheng Liu, Lu Sheng, Jing Shao, Junjie Yan, Shiming Xiang, and Chunhong Pan. 2018. Multi-label image classification via knowledge distillation from weakly-supervised detection. In *Proceedings of the 26th ACM international conference on Multimedia*. 700–708.

[28] Fan Lyu, Qi Wu, Fuyuan Hu, Qingyao Wu, and Mingkui Tan. 2019. Attend and Imagine: Multi-Label Image Classification With Visual Attention and Recurrent Neural Networks. *IEEE Transactions on Multimedia* 21, 8 (2019), 1971–1981.

[29] Quanling Meng and Weigang Zhang. 2019. Multi-Label Image Classification with Attention Mechanism and Graph Convolutional Networks. In *Proceedings of the ACM Multimedia Asia on ZZZ*. 1–6.

[30] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[31] Sameera Ramasinghe, CD Athuraliya, and Salman H Khan. 2018. A context-aware capsule network for multi-label classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 0–0.

[32] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2015. FitNets: Hints for Thin Deep Nets. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[33] Jing Shao, Kai Kang, Chen Change Loy, and Xiaogang Wang. 2015. Deeply learned attributes for crowded scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4657–4666.

[34] Chang Tang, Xinwang Liu, Pichao Wang, Changqing Zhang, Miaomiao Li, and Lizhe Wang. 2019. Adaptive hypergraph embedded semi-supervised multi-label image annotation. *IEEE Transactions on Multimedia* 21, 11 (2019), 2837–2849.

[35] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. 2016. Cnn-rnn: A unified framework for multi-label image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2285–2294.

[36] Liqin Wang, Aofan Zhang, Peng Wang, and Yongfeng Dong. 2019. Automatic image annotation using model fusion and multi-label selection algorithm. *Journal of Intelligent & Fuzzy Systems* 37, 4 (2019), 4999–5008.

[37] Meng Wang, Changzhi Luo, Richang Hong, Jinhui Tang, and Jiashi Feng. 2016. Beyond object proposals: Random crop pooling for multi-label image recognition. *IEEE Transactions on Image Processing* 25, 12 (2016), 5678–5688.

[38] Ya Wang, Dongliang He, Fu Li, Xiang Long, Zhichao Zhou, Jinwen Ma, and Shilei Wen. 2020. Multi-Label Classification with Label Graph Superimposing. In *AAAI*. 5177–5186.

[39] Zhouxia Wang, Tianshui Chen, Guanbin Li, Ruijia Xu, and Liang Lin. 2017. Multi-label image recognition by recurrently discovering attentional regions. In *Proceedings of the IEEE international conference on computer vision*. 464–472.

[40] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. 2019. MMGCN: Multi-modal Graph Convolution Network for Personalized Recommendation of Micro-video. In *Proceedings of the 27th ACM International Conference on Multimedia*. 1437–1445.

[41] Yunchao Wei, Wei Xia, Min Lin, Junshi Huang, Bingbing Ni, Jian Dong, Yao Zhao, and Shuicheng Yan. 2015. HCP: A flexible CNN framework for multi-label image classification. *IEEE transactions on pattern analysis and machine intelligence* 38, 9 (2015), 1901–1907.

[42] Jiaxin Wu, Sheng-Hua Zhong, and Yan Liu. 2019. MvsGCN: A Novel Graph Convolutional Network for Multi-video Summarization. In *Proceedings of the 27th ACM International Conference on Multimedia*. 827–835.

[43] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*. 2048–2057.

[44] Zheng Yan, Weiwei Liu, Shiping Wen, and Yin Yang. 2019. Multi-label image classification by feature attention network. *IEEE Access* 7 (2019), 98005–98013.

[45] Hao Yang, Joey Tianyi Zhou, Yu Zhang, Bin-Bin Gao, Jianxin Wu, and Jianfei Cai. 2016. Exploit bounding box annotations for multi-label object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 280–288.

[46] Renchun You, Zhiyao Guo, Lei Cui, Xiang Long, Yingze Bao, and Shilei Wen. 2020. Cross-Modality Attention with Semantic Graph Embedding for Multi-Label Classification. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

[47] Jun Yu, Dacheng Tao, and Meng Wang. 2012. Adaptive hypergraph learning and its application in image classification. *IEEE Transactions on Image Processing* 21, 7 (2012), 3262–3272.

[48] Sergey Zagoruyko and Nikos Komodakis. 2017. Paying More Attention to Attention: Improving the Performance of Convolutional Neural Networks via Attention Transfer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

[49] Junjie Zhang, Qi Wu, Chunhua Shen, Jian Zhang, and Jianfeng Lu. 2018. Multilabel image classification with regional latent semantic dependencies. *IEEE Transactions on Multimedia* 20, 10 (2018), 2801–2813.

[50] Feng Zhu, Hongsheng Li, Wanli Ouyang, Nenghai Yu, and Xiaogang Wang. 2017. Learning spatial regularization with image-level supervisions for multi-label image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 5513–5522.

[51] Jianqing Zhu, Shengcai Liao, Zhen Lei, and Stan Z Li. 2017. Multi-label convolutional neural network based pedestrian attribute classification. *Image and Vision Computing* 58 (2017), 224–229.

[52] Xiaofeng Zhu, Yonghua Zhu, Shichao Zhang, Rongyao Hu, and Wei He. 2017. Adaptive Hypergraph Learning for Unsupervised Feature Selection.. In *IJCAI*. 3581–3587.