Multi-Label Image Recognition in Anime Illustration with Graph Convolutional Networks

Pengfei Deng[1,*], Jingkai Ren[1], Shengbo Lv[1], Jiadong Feng[1], Hongyuan Kang[1]
Department of School of Information, University of Xiamen,
{dengpengfei, 23320201154015}@stu.xmu.edu.cn

Abstract

Multi-label Recognition is a major challenge in the field of computer vision. Its task is to recognize the set of label objects contained in an image. The key to this problem is how to build the dependency between the labels. The early method is to use Recurrent Neural Networks(RNNs) and Long Short-Term Memory networks(LSTMs) to capture the dependency between the two labels, but this method is too complicated. The current popular method is to use graph neural networks to model the relationship between the labels, such as Graph Convolutional Networks(GCNs) or Hypergraph Neural Networks(HGNNs). This paper uses a relatively simple GCN to model the labels of anime illustrations, where each label is regarded as a vertex in the graph, and constructe a directed graph. And then capture the dependencies between labels from the graph. The experimental results show that applying this method to our anime illustration dataset, we obtaine better mAP then the vanilla DenseNet and ResNet.

Introduction

Multi-Label Recognition is the most basic task in computer vision. Its goal is to recognize a set of labels contained in an image. Different from single-label image recognition, the multi-label task is more challenging, that is, how to associate multiple labels with image regions and the correlation between multiple labels, as shown in 1.

Regarding the problem of how to associate multiple labels with image regions, some works (Wei et al. 2015; Yang et al. 2016) use object detection technology to extract the corresponding image regions and enhance the feature learning of the regions. These tasks usually require additional object boundary annotations in training, which will limit the scope of practical applications. Other work(Yan et al. 2019; Zhu et al. 2017) uses the Attention Mechanism(Xu et al. 2015) to capture the association between image regions and labels under image-level supervision. However, this does not consider the dependencies between different labels (Chen et al. 2019b).

As for how to determine the correlation between multiple labels, a popular method is to use Recurrent Neural Networks(RNNs) or Long Short-Term Memory networks(LSTMs) to construct the correlation between labels (Chen et al. 2019a). Although it has achieved good performance, its problem is that it can only model sequential label relationships (Chen et al. 2019c). Another method is to capture the dependency of two labels based on a probability graph model, such as ChowLiu Tree(Chow and Liu 1968), PLEM(Li, Zhao, and Guo 2014), etc. These methods use label symbiosis pairs to construct a maximum spanning tree structure for Multi-Label Image Classification Tasks(MLIC). However, the probability graph model has a high computational complexity.

By modeling the dependence of labels, the recognition accuracy of neural networks can be effectively improved. In recent years, Graph Neural Networks(GNNs) have been introduced into the task of multi-label recognition. A simple and effective way is use Graph Convolutional Networks(GCNs) to extract the correlation between nodes and neighbor nodes, then calculate it with the image features extracted by CNNs model to enhance the information transmission of the system of labels and feature learning(Chen et al. 2019c). However, this method only captures the dependency relationship between the two labels, and cannot model high-order semantic dependencies(Wu et al. 2020).

Some works have used the hypergraph structure to model the high-order relationships from data(Tang et al. 2019). These methods treat each sample as a vertex and iteratively optimize each variable by fixing other variables. In the task of multi-label image recognition, by treating each label as a vertex, and integrating the adaptive hypergraph into HGNN(Feng et al. 2019) for end-to-end training, using label embeddings to use any number of hyperedges directly to initialize the adaptive hypergraph, which can automatically model high-order semantic relations. This method not only avoids the rigidity of the correlation graph manually, but also avoids the statistical bias caused by the imbalance of the labels in the training set(Wu et al. 2020).

In this paper, we compared the graph convolutional networks and the hypergraph neural networks. The hypergraph neural network model is too complex, difficult to implement and train, and the effect achieved is similar to the graph convolutional networks in our dataset. Therefore, we choose

[*]Work done during the deep learning course.

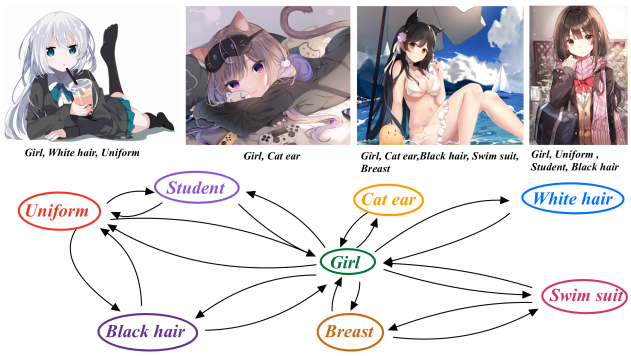

Figure 1: We can discover the relationship between labels by constructing a directed graph. Where $Label_A \rightarrow Label_B$ means that when $Label_A$ appears, $Label_B$ is likely to appear. in this figure, we can see that in the anime illustration dataset, some label pairs actually have a strong correlation, such as "uniforms" and "students", "swimsuit" and "breast".

simple and effective graph convolutional networks to capture the correlation between labels. The model learn the interdependent object classifiers from the previous label representation through the GCN mapping function, and then apply the generated object classifiers to the image features extracted by another CNN model to finish the task of multi-Label image recognition. And we apply the model to the anime illustration dataset we built, and get better experimental results.

In this paper, we do the following work:

- By writing a Python script, grabbing the anime illustrations on the Pixiv and categorizing them by labels, constructing our own dataset(currently there are about 200,000 illustrations and nearly 2,00 labels).
- Use GCN model to model the correlation between labels, and then combine the image features extracted by the CNN model to achieve multi-label recognition.
- Evaluate our model on the dataset we constructed and compare it with other traditional CNN models.

Related Works

With the rapid development of deep learning, the performance of image classification is getting better and better, and it has already reached the limit in single-label image classification. In order to extend the convolutional neural network for multi-label recognition of images, many efforts have been made.

One of the most direct methods of multi-label recognition is to train an independent two-classifier for each label, but this method does not consider the relationship between the labels. When the number of labels increases, the number of predicted labels will increase exponentially.

Therefore, many researchers try to capture the dependencies in labels to improve the recognition accuracy of neural networks. (Wang et al. 2016) used Recurrent Neural Networks(RNNs) to convert labels into embedded label vectors, so that the correlation between labels can be obtained. In addition, the attention mechanism has also been widely used to find correlations in multi-label recognition tasks. (Wang et al. 2017) introduced a spatial transformer layer and a Long Short-Term Memorys(LSTMs) unit to capture the correlation between labels.

Compared with the above-mentioned structure learning method, the graph-based method is proved to be more effective in the modeling of label correlation. (Li et al. 2016) reated a tree-structured graph in the label space by using the maximum spanning tree algorithm. (Lee et al. 2018) introduced a knowledge graph used to describe the relationship between multiple labels.

With the great success of Graph Neural Network(GNN) on visual tasks, GNN has been introduced to MLIC and achieved impressive progress. For instance, (Chen et al. 2019c) propose a novel graph convolutional network based model(ML-GCN) to learn the label relationships. (Wang et al. 2020) add lateral connections between GCN and CNN at different stages to enhance the information transmission of feature learning and label system. Despite the significant improvements have been achieved, these methods only capture the relations of two label, which can't model high-order semantic dependencies.

Some works introduced the hypergraph structure to model high-level relationships between data. Recently, HGNN(Feng et al. 2019) was proposed to learn multi-modal and complex data through hyper-edge convolution operations. It satisfies the characteristics of high-order relationships in multiple labels. A-GCN(Li et al. 2019) uses two 1×1 convolutional layers and a dot product operation to learn a correlation matrix with fixed-size paired labels. AdaHGNN(Wu et al. 2020) uses label embedding to directly initialize the adaptive hypergraph with any number of hyperedges, which can automatically model high-order semantic relations.

Hypergraph-based models can model high-order semantic information of labels well, but the model structure is often very complicated. Therefore, in this paper, we use the relatively simple GCN to capture the correlation and dependence between labels. First, pre-build a label relationship graph from dataset. And then use GCN model to spread information between multiple labels, so as to build a mutually dependent classifier for each label. These classifiers continuously absorb the information of neighbor labels in the process of image convolution. Finally, we apply them to the feature vectors extracted by CNN for multi-label recognition.

Method

Graph Convolutional Network Overview

The basic idea of GCN is to update the node representation by spreading information between nodes. Unlike the convolution operation on the image, the goal of GCN is to learn the function $f(.,.)$ on the graph G, and describe the characteristics $H^l \in \mathbb{R}^{n \times d}$ and the corresponding correlation matrix $A \in \mathbb{R}^{n \times n}$ as input (Where n represents the number of nodes, and d represents the dimension of node features), and update the node characteristics to $H^{l+1} \in \mathbb{R}^{n \times d}$. Each

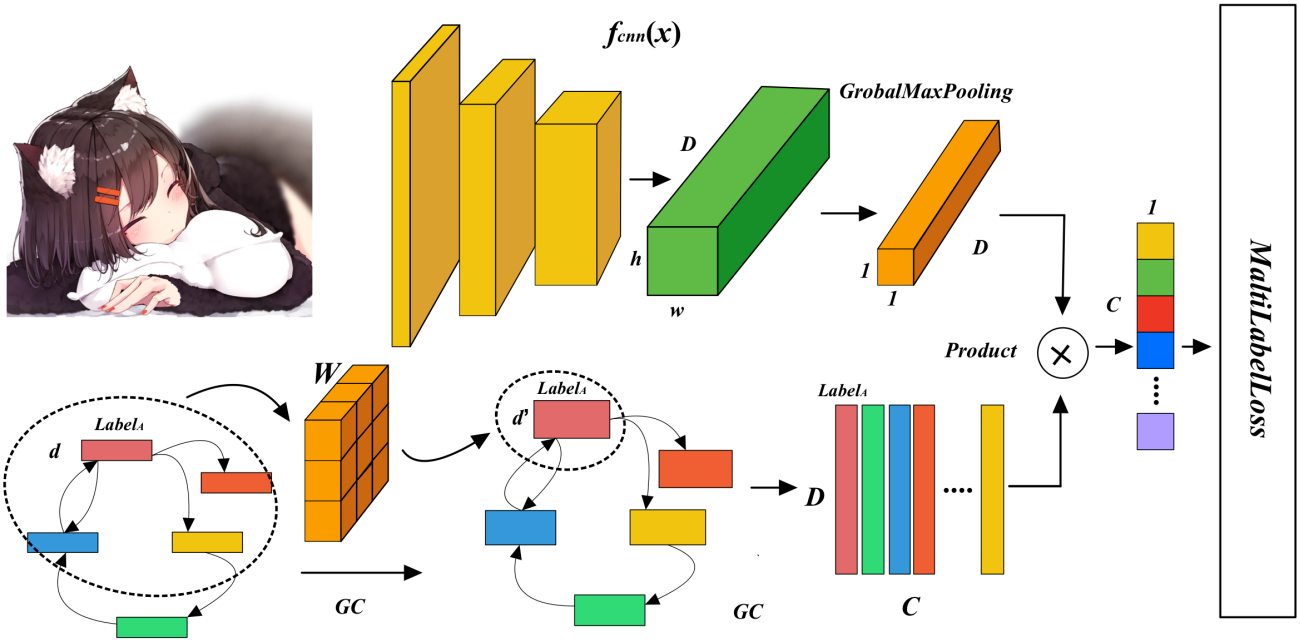

Figure 2: The model structure of multi-label recognition task is composed of CNN image feature extraction module and GCN multi-label classifier learning module. For the CNN image feature extraction module, we use the CNN to extract image features, and the final output feature map of $D \times h \times w$ and use the global max pooling to obtain a feature vector of $x \in \mathbb{R}^D$. For the GCN multi-label classifier learning module, we use GCN to convolve the pre-built label map from the dataset, and finally output the matrix of $W \in \mathbb{R}^{D \times C}$. Then do multiplication of the feature vector x and matrix W, namely $\widehat{y} = Wx \in \mathbb{R}^D$, and finally calculate the loss of $\widehat{y}$ and the real y.

GCN layer can be written as a nonlinear function, which is the following equation

$$H^{l+1} = f(H^l, A). \tag{1}$$

After convolution operation, $f(.,.)$ can be expressed as

$$H^{l+1} = h(\widehat{A}H^l W^l) \tag{2}$$

where $W^l \in \mathbb{R}^{d \times d'}$ is the transition matrix that needs to be learned, and $\widehat{A} \in \mathbb{R}^{n \times n}$ is the normalized correlation matrix, $h(.)$ is a non-linear operation, here we use LeakyReLU. Therefore, we can use multiple GCN layers to learn and construct complex correlations of nodes.

GCN for Multi-Label Recognition

GCN is proposed for semi-supervised classification, where the node-level output is the prediction score of each node. Therefore, it is necessary to perform certain processing on the output of GCN, which is, design the final output of each GCN node as a classifier of the corresponding label in the task. In addition, because the multi-label image recognition task does not provide the corresponding label map structure, that is, the correlation matrix, it is necessary to construct the correlation matrix from scratch. The method we use is shown in the figure. It is divided into two modules, CNN feature extraction module and GCN classifier learning module, as shown in 2.

CNN feature extraction module We can use the CNN model to learn an image feature, so DenseNet121 is used in this paper. Therefore, for any image I with an input size of 512×512, we can get feature maps of $1024 \times 16 \times 16$. Then perform global max-pooling on the obtained features to obtain image-level features x

$$x = f_{GMP}(f_{cnn}(I; \theta_{cnn})) \in \mathbb{R}^D \tag{3}$$

where θ_{cnn} are model parameters, and $D = 1024$.

GCN classifier learning module Based on the mapping function of GCN, we learn interdependent object classifiers from the label representation, that is, $W = w_i{}_{i=1}^{C}$, where C represents the number of labels. Multiple GCN layers are used, where each GCN layer takes the node representation of the previous layer as input. For the first layer, the input is a matrix of $Z \in \mathbb{R}^{C \times d}$, where d is the word-level embedding dimension. For the last layer, the output is $W \in \mathbb{R}^{C \times D}$, where D represents the dimension of the image. By applying the learned classifier to the image features, we can get the predicted score

$$\widehat{y} = Wx \tag{4}$$

We assume that the true label of the image is $y \in R^C$, where $y_i = 0, 1$ indicates whether the label i appears in the image. The final loss function uses the traditional multi-label classification loss function for training

$$Loss = \sum_{i=1}^{C} y^i log(\delta(\widehat{y}^i)) + (1 - y^i) log(1 - \delta(\widehat{y}^i)) \tag{5}$$

Figure 3: Image of conditional probability between two labels. As usual, when "Cat ear" appears in the image, "Girl" will also occur with a high probability. However, in the condition of "Girl" appearing, "Cat ear" will not necessarily occur.

where $\delta(.)$ is the sigmod function.

Calculation of Correlation Matrix

GCN back-propagates information is based on the correlation matrix A, so how to construct the correlation matrix A has become a crucial issue. We traverse the dataset to construct the correlation matrix, the specific steps are as follows:

First calculate the conditional probabilities of the two labels $P_{ij} = P(L_j|L_i)$, which means the probability of the L_j label appearing under the condition that the L_i label appears. By traversing the dataset, we can get all conditional probability between labels, namely P. As shown in 3, $P(L_i|L_j)$ not equal $P(L_j|L_i)$. Thus, the correlation matrix is asymmetrical.

Considering that there may be a small amount of noise in the dataset, such as images of the wrong class, the conditional probability of two irrelevant labels still exists, that is, a wrong edge will appear in the graph. Therefore, we define a hyper parameter γ to filter out some edges

$$A_{ij} = \begin{cases} 0, & P_{ij} < \gamma \\ 1, & P_{ij} \geq \gamma \end{cases} \tag{6}$$

where A is the binary correlation matrix. Using a binary correlation matrix may cause excessive smoothing, so that nodes from different classes become difficult to distinguish. Therefore, we use the following weighting scheme

$$A'_{ij} = \begin{cases} \dfrac{p}{\sum_{j=1,i\neq j}^{C} A_{ij}}, & i \neq j \\ 1-p, & i = j \end{cases} \tag{7}$$

where A' is the new correlation matrix, and p determines the weights assigned to the node itself and other related nodes.

When updating node features, we only modify the inherent weight of the node itself, and the weight of related nodes will be determined by the domain distribution.

Experiment

In this section, we use several common multi-label evaluation methods, such as mAP(mean average precision), OF1(average overall F1), CF1(average per-class F1) to evaluate our model. Then we describe how to obtain the implementation details of the anime illustration dataset and model. Finally, we compare our model with the vanilla Resnet and DenseNet, and display the comparison and multi-label classification result of anime illustration respectively.

Datasets

We write a Python script to grab the anime illustrations on the Pixiv, a famous Japanese manga website, and finally classify the images through the labels of each illustration. Since the size and dimensions of each illustration are inconsistent, the illustrations are also preprocessed, and the black edges are filled to make the size of each illustration consistent. Finally, we also open sourced our crawling script and dataset to github.

Implementation details

Different from general neural networks, our model consists of a CNN image feature extraction module and a GCN label object classifier learning module. For the CNN image feature extraction module, we use DenseNet121 to extract image features, and finally use global max pooling on the output feature map of $1024\times16\times16$ to turn it into a 1024×1 vector. For the GCN module, we use two layers of GCN to capture the dependency relationship of the labels, and the final output is adjusted to a $C \times 1024$ matrix. Then we use the onehot vector to represent the multi-label information of an anime illustration, that is, an anime illustration has labels $\{l_1, l_2, ..., l_n\}$, then the value of 1 is assigned to the first l_i of the vector, as follows

$$labels_i^{(l_1,l_2,...,l_n)} = [0, 0, 1_{l_1}, .., 1_{l_i}, ..., 0]_{1\times C}, n \leq C \tag{8}$$

For the correlation matrix, by traversing the dataset, for any two labels L_i and L_j, calculate the number of images that appear at the same time, and the number of images that only appear in L_i or L_j, and divide that you can get the corresponding $P(L_i|L_j)$ or $P(L_j|L_i)$, and then set $\gamma = 0.002$ and $p = 0.2$.

In the training phase, we set the size of each image to 512×512, and we use random horizontal flips for data enhancement operations. In the optimization phase, we use the Adam optimization method, and the learning rate is set to 0.001, and the final number of training epochs is set to 30 respectively.

Finally, we use Pytorch to implement and train our model.

Experimental results

We apply the GCN model to the anime illustration dataset and compare it with the traditional CNN models of Resnet101 and Densenet121, and then evaluate our model from several key aspects.

We extract the 40 most representative labels(about 40,000 anime illustrations) from the original anime illustration dataset, of which 3/4 are used as the training set and the remaining 1/4 are used as the validation set.

Table 1 and Table 2 show the multi-label recognition results of our model and Resnet101 and Densenet121. Our model finally got 79.3% of mAP, 73.2% of CF1 and 75.2% of OF1, which is better than the vanilla Resnet101 and Densenet121. In addition, the recognition accuracy of each

Table 1: Comparisons of AP and mAP with state-of-the-art methods on ours dataset.

Method	gril	brest	uniform	black stockings	black hair	swimsuit	cat ears	maid	mAp
Resnet101	80.1	68.1	66.3	62.6	60.4	63.2	55.4	64.0	65.1
Densenet121	82.0	69.5	66.7	65.3	61.2	64.1	60.2	58.3	66.4
GCN(Ours)	**95.3**	**79.4**	**75.8**	**74.4**	**74.1**	**70.3**	**72.1**	**75.5**	**79.3**

Figure 4: The predicted results on the top are based on ours model, while the results on the bottom are DenseNet121.

Table 2: Comparisons with state-of-the-art methods on ours dataset.

Method	All			Top3	
	mAP	CF1	OF1	CF1	OF1
Resnet101	65.1	62.3	60.2	62.6	59.2
Densenet121	66.4	64.5	63.7	63.3	60.2
GCN(Ours)	**79.3**	**73.2**	**75.2**	**69.8**	**70.4**

label is also better than the results obtained by vanilla Resnet101 and Densenet121.

Classifier Visualization

We selected a few representative anime illustrations, and compared our model with Densenet121. From the Figure 4, we can find that our model can recognize almost all the labels of an image, especially the dependent labels. For example, "uniform" and "student", "swim suit" and "breast", but Densenet121 can only recognize some labels, and even miss some dependent labels.

Conclusion

The key problem of multi-label image recognition is how to model the dependence between labels. In this paper, we use GCN to model the label dependence of anime illustrations. Compared with the method of directly using CNN models to get top k, our model makes full use of the information of labels to predict the labels contained in an anime illustration more accurately. Although it is not possible to model high-order semantic relations directly, this method is simple and effective, and the model is easy to train. In addition, the accuracy of the result is not high due to some anime illustration labels have insufficient or incorrect information for the small number of our datasets and the complete use of scripts for the division of labels, and the number of illustrations contained between labels is inconsistent.

References

Chen, L.; Wang, R.; Yang, J.; Xue, L.; and Hu, M. 2019a. Multi-label image classification with recurrently learning semantic dependencies. *The Visual Computer* 35(10):1361–1371.

Chen, T.; Xu, M.; Hui, X.; Wu, H.; and Lin, L. 2019b. Learning semantic-specific graph representation for multi-label image recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, 522–531.

Chen, Z.-M.; Wei, X.-S.; Wang, P.; and Guo, Y. 2019c. Multi-label image recognition with graph convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5177–5186.

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

Feng, Y.; You, H.; Zhang, Z.; Ji, R.; and Gao, Y. 2019. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3558–3565.

Lee, C.-W.; Fang, W.; Yeh, C.-K.; and Frank Wang, Y.-C. 2018. Multi-label zero-shot learning with structured knowledge graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1576–1585.

Li, Q.; Qiao, M.; Bian, W.; and Tao, D. 2016. Conditional graphical lasso for multi-label image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2977–2986.

Li, Q.; Peng, X.; Qiao, Y.; and Peng, Q. 2019. Learning category correlations for multi-label image recognition with graph networks. *arXiv preprint arXiv:1909.13005*.

Li, X.; Zhao, F.; and Guo, Y. 2014. Multi-label image classification with a probabilistic label enhancement model. In *UAI*, volume 1, 1–10.

Tang, C.; Liu, X.; Wang, P.; Zhang, C.; Li, M.; and Wang, L. 2019. Adaptive hypergraph embedded semi-supervised multi-label image annotation. *IEEE Transactions on Multimedia* 21(11):2837–2849.

Wang, J.; Yang, Y.; Mao, J.; Huang, Z.; Huang, C.; and Xu, W. 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.

Wang, Z.; Chen, T.; Li, G.; Xu, R.; and Lin, L. 2017. Multi-label image recognition by recurrently discovering attentional regions. In *Proceedings of the IEEE international conference on computer vision*, 464–472.

Wang, Y.; He, D.; Li, F.; Long, X.; Zhou, Z.; Ma, J.; and Wen, S. 2020. Multi-label classification with label graph superimposing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 12265–12272.

Wei, Y.; Xia, W.; Lin, M.; Huang, J.; Ni, B.; Dong, J.; Zhao, Y.; and Yan, S. 2015. Hcp: A flexible cnn framework for multi-label image classification. *IEEE transactions on pattern analysis and machine intelligence* 38(9):1901–1907.

Wu, X.; Chen, Q.; Li, W.; Xiao, Y.; and Hu, B. 2020. Adahgnn: Adaptive hypergraph neural networks for multi-label image classification. In *Proceedings of the 28th ACM International Conference on Multimedia*, 284–293.

Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhudinov, R.; Zemel, R.; and Bengio, Y. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, 2048–2057.

Yan, Z.; Liu, W.; Wen, S.; and Yang, Y. 2019. Multi-label image classification by feature attention network. *IEEE Access* 7:98005–98013.

Yang, H.; Tianyi Zhou, J.; Zhang, Y.; Gao, B.-B.; Wu, J.; and Cai, J. 2016. Exploit bounding box annotations for multi-label object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 280–288.

Zhu, F.; Li, H.; Ouyang, W.; Yu, N.; and Wang, X. 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.