Multi-label Image Classification with Visual Attention and Handcrafted Features

Siyi Tang
siyitang@stanford.edu

Mingkun Chen
maxcmk@stanford.edu

Ruge Zhao
rugezhao@stanford.edu

Abstract

We explored CNN-extracted and handcrafted features for multi-label classification of museum object images. We applied RNN and a visual attention mechanism to these features to learn the inter-dependency of image labels. Moreover, unlike traditional sequence models, our approach does not require a pre-defined order of ground truth labels. Our experiments on the "iMet Collection" challenge suggest the effectiveness of using handcrafted features as well as RNN with visual attention in multi-label image classification of museum objects. Finally, we show that despite trained on noisy ground truth labels, our CNN-RNN model with visual attention is able to attend to relevant image regions, and predict reasonable labels associated with the images.

1. Introduction

The Metropolitan Museum of Art in New York (The Met) has a diverse collection of over 1.5M objects, of which over 200K have been digitized with imagery. The online cataloguing information is generated by Subject Matter Experts (SME) and includes a wide range of data such as multiple object classifications, artist, title, period, date and culture etc. Currently, the SME-generated annotations describe the object from an art history perspective but not from a museum-goer's point of view. Thus adding fine-grained attributes to the museum objects from the museum-goers understanding is of interest, in which case visitors can search for visually related objects online.

In this paper, we work on the "iMet Collection" challenge on Kaggle, which is to label the museum object images provided by The Met with multiple attribute labels, each describing the *cultural* style and/or *tags* for objects appearing in the artwork. For example, shown in the left panel of Figure 1 is an image of an oil painted man's portrait. This sample has a *culture* label "American" that describes its style, and two *tag* labels "men" and "portraits" which depict subjects that can be observed from this picture. Apart from paintings, objects shown in the image can also be other types of artworks. Take as an instance the photograph of a decorated ceramic bowl shown in the right panel of Figure 1: this image comes with a *culture* label "Chinese with European decoration" and five *tag* labels such as "bowls", "buildings", "human figures", etc.

We approach this multi-label image classification problem as follows. First, we incorporated additional handcrafted features into the architecture of a convolutional neural network (CNN). Secondly, we implemented a visually attended recurrent neural network (RNN) on top of these features so that the inter-dependencies among labels can be learned and the label-associated image regions can be attended to. The inputs to our model are images of museum objects. The outputs to our model are predicted labels corresponding to the fine-grained attributes of the images.

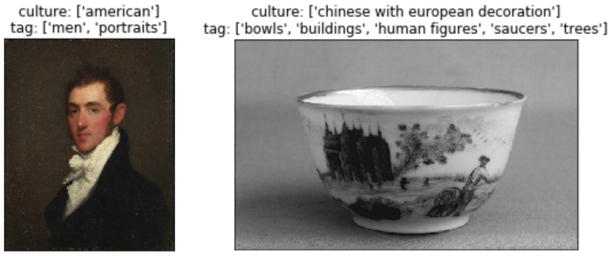

Figure 1: Examples of artwork image with *culture* and *tag* labels

The main contributions of this paper are as follows:

- We show that incorporating handcrafted features into CNN provides richer information for multi-label image classification.
- Our visual attention mechanism allows our model to focus on local regions-of-interests in the images associated with the labels.
- The use of RNN enables the model to learn inter-dependencies among labels.
- Unlike traditional sequence models, our proposed method does not require pre-defined orders of ground truth labels at training time.

2. Related Work

Multi-Label classification problem has been studied extensively during the past decade or so, and both classic machine learning techniques and deep learning have been employed to deal with this problem. Some of the earlier works adopted non-neural network techniques. Boutell et al. examine several training and scoring paradigms for multi-label classification in general [1]. One of the most common strategies for image labelling in particular is a nearest neighbor based approach which computes distance metrics on various features of the images [2, 3]. Despite the success these approaches achieved in a few restricted situations, they quickly lost popularity over time because of the emerging deep learning-based techniques.

With the development of deep learning and neural network architectures, the work done by Zhang et al. is among the first to apply neural network techniques to multi-label classification, which views each output node as a binary classification task, and utilizes a simple feed forward network to exploit the dependency across labels in computational biology [4]. It was later extended by deep neural networks based multi-label algorithms which proposed different loss functions or architectures [5, 6, 7]. For example, Gong et al. [5] study how a rank-based loss functions influenced the training of an AlexNet-like CNN architecture, Wei et al. [6] investigate a model based on combining single-label classified candidates from a series of proposal regions, and Hu et al. [7] put forth a model which uses concept layers modeled with label graphs by a structured inference neural network. While these approaches produce promising results, the techniques still have their weaknesses as they are not designed for learning the inter-dependency between labels.

Recurrent neural networks (RNNs) are primarily used to learn the sequential relationships in sequence data. Long Short-Term Memory (LSTM) [8], a variant of RNN, is an effective method which has been applied to exploit the long-term dependency in a sequence. In the field of image captioning, Xu et al. present a representative model combining CNN, RNN and visual attention to generate text captions for images [9]. This approach requires a pre-defined order of ground truths for learning. Adapting CNN-RNN models to multi-label image classification tasks, [10, 11] propose variants of CNN-RNN models to deal with image labels that do not have specific orders. In particular, [10] explores multiple ways of ordering the labels such as random ordering and ordering from the easiest labels, while [11] proposes an order-free CNN-RNN framework for multi-label classification.

In artwork domain, features can either be learned through optimizing for a specific task (e.g. CNN for image classification) or handcrafted based on semantic-level algorithms like HOG [12]. [13, 14] show the effectiveness of integrating handcrafted features with neural networks features by incorporating handcrafted features into the neural network architecture before the final prediction layer.

3. Methods

3.1. Baseline

Our baseline model is ResNet50 [15] pre-trained on ImageNet. We loaded ResNet50 pre-trained weights from torchvision, and modified the last fully connected (FC) layer's output size to 1,077, corresponding to the number of labels in our task. We treated the task as a one-vs-rest classification problem assuming class independence, and used binary cross-entropy loss as the loss criterion. We only trained the last FC layer, while kept all the pre-trained weights in the previous layers frozen. At test time, because we generated six images of each validation/test image by resizing and cropping (see Section 4.2 below), the final prediction was obtained by averaging the predicted probabilities over all the six samples, and thresholding the averaged probabilities to obtain the predicted labels. In addition, we performed threshold search by grid search on range $(0, 0.5)$ to find the best fixed threshold (applied to all classes) to convert predicted probabilities to labels.

3.2. ResNet50 + HOG

We investigated jointly using features from CNN and handcrafted Histogram of Oriented Gradients (HOG) features [12]. HOG has been proven to be one of the most effective handcrafted features according to [14], where the authors added HOG features to the last FC layer of a neural network for painting style categorization.

At data preprocessing phase, HOG features were extracted using skimage's implementation of the HOG algorithm (Algorithm 1), producing a vector of size 10,368 for each 224×224 image. Figure 2 shows a sample image and its HOG descriptor.

Algorithm 1: Overview of HOG

Input : Image I
Output: HOG descriptor vector $\mathbf{v}_H$
Apply normalization algorithm to each RGB channel;
Compute gradients;
Weighted vote into spatial and orientation cells;
Contrast normalize over overlapping spatial blocks;
Collect HOG's over detection window and concatenate into an output vector;

Building upon baseline model, the HOG features $\mathbf{v}_H$ were concatenated with the CNN features extracted before the last FC layer of pre-trained ResNet50. Similar to baseline, we replaced the last FC layer in ResNet50 with a FC

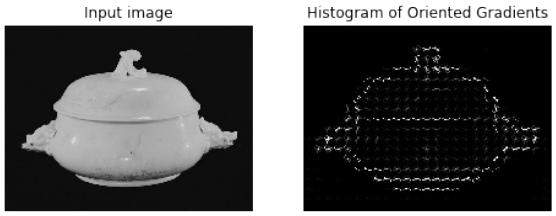

Figure 2: Example of an image and its HOG descriptor

layer having an output size of 1,077 corresponding to the number of classes.

Similar to baseline, we froze all pre-trained weights up to the last average pooling layer in ResNet50, and only trained the FC layer at training time. The same threshold search and prediction averaging strategy at test time were used as explained in Section 3.1.

3.3. CNN-RNN with Visual Attention

One drawback of the baseline is that it assumes independence among the labels. However, we observed that many of the attributes have similar meanings and are thus highly correlated. To capture the label dependencies, we combine RNN with CNN-extracted features and visual attention as proposed in [9] and [11], which we refer to as AttnCNN-RNN in this paper. There are three components in AttnCNN-RNN.

Encoder CNN. The first component is the encoder CNN. We used the same CNN architecture as baseline. We computed the following from the CNN: (a) features $\mathbf{V}_{feat} \in R^{2048\times14\times14}$ from the average pooling layer (second last layer); (b) predicted probabilities $\mathbf{v}_{prob} \in [0,1]^{1077}$ from the last FC layer. We then reshaped $\mathbf{V}_{feat}$ into $(\mathbf{v}_1, ..., \mathbf{v}_{196})$ where $\mathbf{v}_i \in R^{2048}, \forall i \in \{1, ..., 196\}$, i.e. there are 196 features, each having dimension 2048.

Attention Layer. One challenge of this image classification task is that some attributes, especially those belonging to *tags*, might be extremely small in the image. For instance, a painting labeled with "women" and "landscape" might have the woman appearing in a small corner. Hence, to encourage the model to focus on specific regions in the input image, we applied the same visual attention mechanism as detailed in [9] and [11]. Specifically, at each time step of the RNN (see next section), the attention function takes as input the feature vector $\mathbf{v}_i$ and the previous hidden state $\mathbf{h}_{t-1}$ from the RNN, and generates a weight $\alpha_{i,t}$:

$$\epsilon_{i,t} = f_{att}(\mathbf{v}_i, \mathbf{h}_{t-1}) \tag{1}$$

$$\alpha_{i,t} = \frac{e^{\epsilon_{i,t}}}{\sum_{j=1}^{m} e^{\epsilon_{j,t}}} \tag{2}$$

where f_{att} is a two-layer perceptron (Linear-ReLU-Linear). Finally, the attention layer produces a context vector $\mathbf{z}_t$, a weighted sum of the features:

$$\mathbf{z}_t = \sum_{i=1}^{196} \alpha_{i,t}\mathbf{v}_i \tag{3}$$

Decoder RNN. Lastly, we applied an LSTM to predict image labels one at a time. At each time step t, the LSTM takes as input predicted probability $\mathbf{v}_{pred}$ from previous time step, $\mathbf{z}_t$ from the attention layer, predicted labels $\tilde{\mathbf{y}}_{t-1}$ accumulated from past time steps, as well as previous hidden state $\mathbf{h}_{t-1}$ to produce the current hidden state $\mathbf{h}_t$. To predict a label at time step t, we passed $\mathbf{h}_t$ through two FC layers f_{pred} (with 0.5 probability dropout in between) and a sigmoid function to generate probability $\mathbf{p}_t$:

$$\mathbf{h}_t = f_{LSTM}(\mathbf{v}_{pred}, \mathbf{z}_t, \tilde{\mathbf{y}}_{t-1}, \mathbf{h}_{t-1}) \tag{4}$$

$$\mathbf{p}_t = \sigma\big(f_{pred}(\mathbf{h}_t)\big) \tag{5}$$

At training time, RNN generally requires a pre-defined order of ground truths to compute loss between prediction and ground truth at each time step. Unlike image captioning in [9] where the generated words have pre-defined orders, image labels do not have specific orders. In [10], the authors explored multiple ways of ordering the labels such as random ordering, ordering from the easiest labels etc. In contrast, we adopted the order-free approach in [11]. The training procedure of our AttnCNN-RNN model is summarized in Algorithm 2, where $T = 10$ is the maximum number of labels in the training set.

At test time, we terminated the label generating process at the end of time step t if the normalized path probability P_t is below a certain threshold, where P_t is computed as follows:

$$P_t = \sqrt[t]{Pr(l_1|I) \times ... \times Pr(l_t|l_1, ..., l_{t-1})} \tag{6}$$

Because of the large number of rare occurring classes, we used class-specific threshold for each class. Concretely, let $\gamma\mathbf{f}$ be the threshold vector for all classes, where $\mathbf{f} \in R^{1077}$ is the proportions of the class occurrences in the training set, and γ is a hyperparameter to be tuned on the validation set. If $P_t < \sqrt[t]{\gamma f_{l_1} \times ... \times \gamma f_{l_t}}$, we would terminate the label generation at the end of time step t. Moreover, because we generated six images for each validation/test image by resizing and cropping (see Section 4.2), the final predicted labels for the original image were obtained by the union of the predictions over all the six samples.

Beam Search. We note that at test time during each time step, it is possible to have multiple closely-scored predicted labels, and only picking the one with the maximum probability is likely to result in wrong prediction. To make things worse, prediction error at an earlier time step can propagate as the predicting process move on and eventually significantly degrade the final model performance. We applied

the technique of beam search at test time [10, 11] to alleviate this error propagation problem by keeping the best K prediction paths at each time step. More precisely, the algorithm records K predicted labels with the highest probability at time step t; then at time step $t+1$, the algorithm search from all $K \times \mathscr{C}$ successor paths generated from the K previous paths, where $\mathscr{C}$ is the number of classes (i.e. 1,077 in our case). The K candidates for the following time step $t+2$, is then selected by choosing the best K prediction paths with the highest cumulative probability from all $K \times \mathscr{C}$ successor paths.

We implemented AttnCNN-RNN by ourselves, with some references of attention layer and beam search from the Github tutorial on using PyTorch for image captioning.

Algorithm 2: Training of AttnCNN-RNN

Input : Image I, label $\mathbf{y}$
Output: Predicted labels $\tilde{\mathbf{y}}$
Load pre-trained ResNet50 parameters;
Randomly intialize parameters: ResNet50 last FC layer θ_R, attention layer θ_a, LSTM θ_L and final prediction layer θ_p;
while $i \leq N_{epochs}$; **do**
  Extract $\mathbf{V}_{feat}$ and compute $\mathbf{v}_{prob}$ from ResNet50;
  Initialize loss $L = 0$, $\mathbf{v}_{pred} = \mathbf{v}_{prob}$, predicted labels $\tilde{\mathbf{y}}_0 = \{\}$, and label candidate pool $C_0 = \{1, ..., 1077\}$;
  for ($t = 1$; $t \leq T$; $t++$) {
    Compute the context vector $\mathbf{z}_t$ by equation 3 ;
    Compute the hidden state $\mathbf{h}_t$ and the predicted probability $\mathbf{p}_t$ by equations 4 and 5;
    Obtain current time step predicted label $l_t = \text{argmax}_{C_t} \mathbf{p}_t$;
    Update the accumulated predicted labels $\tilde{\mathbf{y}}_t = \tilde{\mathbf{y}}_{t-1} \cup \{l_t\}$;
    Update the label candidate pool $C_t = C_{t-1} \setminus \{l_t\}$;
    Compute cross-entropy loss $L+ = -\sum_{j=1}^{1077} y_j \log(p_{t,j}) + (1-y_j)\log(1-p_{t,j})$
    $\mathbf{v}_{pred} = \mathbf{p}_t$
  }
  $L = \frac{L}{T}$;
  Stochastic gradient descent on $\theta_R, \theta_a, \theta_L$ and θ_p ;
  $i++$;
end

3.4. AttnCNN-RNN + HOG

Finally, we combined HOG features and convolutional features extracted from CNN with AttnCNN-RNN architecture. Specifically, $\mathbf{v}_{prob}$ in Algorithm 2 is now the output vector from the last FC layer in ResNet50+HOG (Section 3.2), while $\mathbf{V}_{feat}$ remains the same. At training time, we froze all the layers except for the last FC layer in ResNet50+HOG. At test time, we applied the same threshold search and beam search strategies as described in Section 3.3.

4. Dataset and Features

4.1. Dataset

The dataset we use is a collection of artwork images and their associated attributes from The Met, publicly available on the Kaggle competition page. The official dataset provided in the Kaggle competition includes training (with attribute labels) and test sets. Each training data sample consists of one image with at least one label. Because the labels were annotated by a single annotator without a verification step, the labels are *noisy*, and some might have similar meanings. There are in total 109,237 artwork images associated with a total number of 1,103 labels in the training set. The dimension of each image is normalized such that the shorter dimension is 300 pixels. The labels belong to two general categories: *culture* and *tag*. Since the labels in the test set are hidden, we do not use the official test set provided by Kaggle in this project. Instead, we split the official training set into our own train/validation/test sets.

4.2. Data Preprocessing

We first discarded samples that contain labels appearing less than 3 times out of the 109,237 images from the official training set. This resulted in 109,200 images and 1,077 labels left, which were then split into our own train/validation/test sets with a ratio of 8:1:1, resulting in 87,360, 10,920 and 10,920 samples in train, validation and test sets respectively.

We observed that *tag* attributes are generally related to specific objects appearing in local regions in the images, while *culture* attributes are abstract properties of the entire images. Furthermore, there are many images with large aspect ratios (e.g. 4428×300) in the dataset. Motivated by these observations, we performed a six-fold increase on the entire dataset by resizing and cropping. For each image in the train set, if the sample comes with both *culture* and *tag* labels, we generated one sample by resizing the original image into 224×224 associated with the original labels, and five other samples by performing five random crops of size 224×224 with the *culture* labels only; if the sample comes with *tag* label only, we generated six samples, all resized into 224×224 associated with the original labels. Resizing prevents cropping out local regions associated with *tag* labels, whereas cropping prevents shape distortion of objects in the images. We then applied further data augmentation on the train set by performing random horizontal flip and

Model	F2	F1	Precision	Recall
Baseline (val)	0.327	0.233	0.186	0.587
ResNet50 + HOG (val)	0.337	0.247	0.204	0.563
AttnCNN-RNN (val)	**0.391**	0.263	0.174	0.619
AttnCNN-RNN + HOG (val)	0.390	0.262	0.174	0.617

Table 1: Model performance on validation set

Model	F2	F1	Precision	Recall
AttnCNN-RNN (test)	0.393	0.264	0.175	0.622

Table 2: Best model performance on test set

random color jitter on each of the samples. For all the images in validation/test sets, we simply generated one sample by resizing and five samples by random cropping. Lastly, all samples in the train/validation/test sets were normalized with per-channel mean and standard deviation of the train set.

4.3. HOG Features

We also extracted HOG features from all the preprocessed images. Details on how HOG features are incorporated into our models' architecture are explained in sections 3.2 and 3.4.

5. Experiment Results and Discussions

For all the models described above, we used Adam optimizer with cosine annealing of learning rate and batch size of 32. Due to memory constraint, 32 is the largest batch size we could use. Furthermore, to prevent over-fitting, we used L2 regularization with a L2 weight decay rate of 1×10^{-5}. In addition, to account for extreme imbalance of positive and negative examples for the rare classes, we incorporated a weight for each class when computing the binary cross-entropy loss. More specifically, the loss including the class weight is as follows:

$$L = -\sum_{j=1}^{1077} p_j y_j \log(p_{t,j}) + (1 - y_j) \log(1 - p_{t,j}) \quad (7)$$

where $p_j = \dfrac{\text{number of negative examples}}{\text{number of positive examples}}$ is the weight for the j-th class. Moreover, we clipped the maximum class weight at 10 to prevent extremely large weights. In addition, for AttnCNN-RNN model, there is an extra hyperparameter γ to threshold the predicted labels. All the above-mentioned hyperparameters were chosen based on the best model performance on the validation set.

As this is a multi-label classification problem, we used averaged F2 score (averaged over samples) as the main evaluation metric. F2 score provides more robustness against noisy labels than F1 score as it favors recall more than precision. F2 score is defined as follows:

$$F2 = \frac{(1 + \beta^2) \times \text{precision} \times \text{recall}}{\beta^2 \times \text{precision} + \text{recall}} \quad (8)$$

where $\beta = 2$ is used to weigh recall higher than precision. We also looked at metrics such as sample-averaged F1 score, precision and recall. The reason why we did not use averaged accuracy as an evaluation metric is that accuracy would likely be inflated because of the large number of classes and the sparsity of labels.

Table 1 shows the performance of baseline (ResNet50), ResNet50+HOG, AttnCNN-RNN as well as AttnCNN-RNN+HOG on the validation set.

5.1. ResNet50 + HOG

Using HOG features jointly with CNN features outperforms baseline by 1% for F2 score, and achieves a F2 score of 0.337 on validation set.

5.2. AttnCNN-RNN

The prediction results of AttnCNN-RNN model on validation set are shown in Table 1. AttnCNN-RNN greatly outperforms baseline and ResNet50+HOG models, suggesting the effectiveness of using RNN and visual attention mechanism for multi-label image classification.

5.3. AttnCNN-RNN + HOG

Since combining HOG features with CNN-extracted features improves baseline model performance, we also experimented on integrating HOG features into AttnCNN-RNN model architecture. As shown in Table 1, the model performance on validation set is similar to AttnCNN-RNN.

Based on F2 scores, we conclude that our best model is AttnCNN-RNN. Table 2 shows the best model's performance on test set.

5.4. AUC-ROC for Individual Labels Predicted by ResNet50+HOG

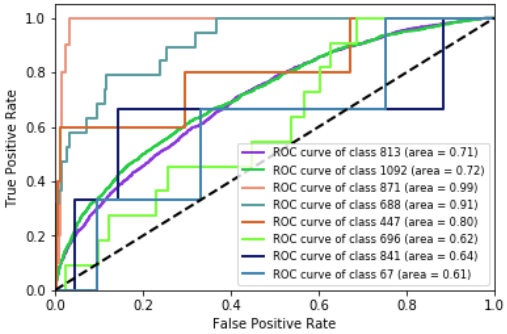

Figure 3: AUC-ROC of ResNet50+HOG on validation set

In Figure 3, we show the ROC curve and the micro-averaged area under the ROC curve (AUC-ROC) predicted by ResNet50+HOG model on validation set. The ROC curve is not illustrated for the best model, since AttnCNN-RNN directly outputs hard labels instead of probabilities.

Because of the large number of classes, we only show 8 classes here by choosing 2 classes from each quartile (sorted by the class proportions in the training set). From the fourth quartile, class 813 and class 1092 correspond to the two largest classes, "tag:men" and "tag:women" respectively, and our model achieves around 0.7 AUC-ROC on these two classes. For class 871 ("tag:pediments") and 688 ("tag:games") from the third quartile, our model achieves decent accuracy. Model performance deteriorates significantly for smaller classes as seen for class 696 ("tag:george washington"), 841 ("tag:necktie") and 67 ("culture:caucasian").

5.5. Attention Visualization

In this section, we visualize some label-associated image regions that are attended to when the best model AttnCNN-RNN predicts a specific label.

Figure 4 shows four examples of visually attended image regions in the validation set. The first column shows the original raw image with ground truth labels (black texts), the second column shows one of the six cropped and normalized images obtained from the raw image, and the last column shows the local regions of the cropped image where the model attend to for the corresponding predicted label (blue texts for correct predictions or red texts for wrong predictions). To visualize the visually attended regions, we overlay α in equation 2 with the cropped image in the second column. Importantly, we can see that the model is able to attend to the relevant regions when making predictions. For instance, in the third example (third row), even if the woman in the image is largely cropped out, the model is able to infer the label "tag:women" from the woman's dress. Note that even though the predicted label in the last example (last row) is not one of the ground truth labels, the attended region and the prediction do make sense. This suggests that despite trained on noisy ground truth labels, our model is able to capture important visual information and predict associated labels in the images.

Figure 4: Examples of visually attended regions

5.6. Analyses of Predictions

Our AttnCNN-RNN model has learned inter-dependencies among labels. Some of the predictions are quite reasonable even if they do not exist in the noisy ground truth labels.

Figure 5 shows two examples of predictions in the test set. Black texts are the ground truth labels, while blue texts are the predicted labels. In the first example (top panel), labels such as "tag:landscapes" and "tag:rivers" are reasonable labels of the image even if the ground truth labels do not contain these two labels. Note that certain predicted labels such as "tag:houses" and "tag:boats" do not appear in the image. However, we can expect such errors to occur because houses and boats are highly relevant to the context of this image. Similarly, in the second exam-

Figure 5: Prediction examples from test set

ple (bottom panel), predictions such as "tag:human figures" and "tag:women" are both reasonable labels. The model wrongly predicts "tag:houses" and "tag:soldiers", which might be because the model confuses tents in the image with houses, and men with soldiers.

5.7. Confusion Matrix

Here we analyze the performance of our AttnCNN-RNN model by looking at the confusion matrix of an example label "tag:trees". As this is a multi-label classification problem, when plotting the confusion matrix for a single label, we think of our algorithm as a binary classifier: 1 means that the image has the label "tag:trees" while 0 means no "tag:trees" label. In other words, we only look at the prediction on this specific label and ignore the others.

In Figure 6, for instance, the confusion matrix of the largest class label "tag:men" is plotted. We see the following relationship: true negatives (TN) >> false positives (FP) > true positives (TP) >> false negatives (FN). Note that due to the large number of classes and the low prevalence of labels, i.e. (FN+TP)/total is as small as 0.17, it is not surprising to have FP > TP, i.e. wrongly predicting "tag:men" label more often than correctly predicting it. In this example, the true positive rate is calculated as TP/(TP+FN) = 0.92; the true negative rate is calculated as TN/(TN+FP) = 0.60. This means that our model tends to make correct predictions when the ground truth is 1. In contrast, when the ground truth is 0, the algorithm has a relatively degraded performance. In other words, our model tends to predict more labels than the ground truth for an image, which agrees with what we have seen in Section 5.6. The reason of this can be an improper threshold that is not harsh enough to filter out unnecessary predictions. We note that the observations discussed here is a common case for large classes.

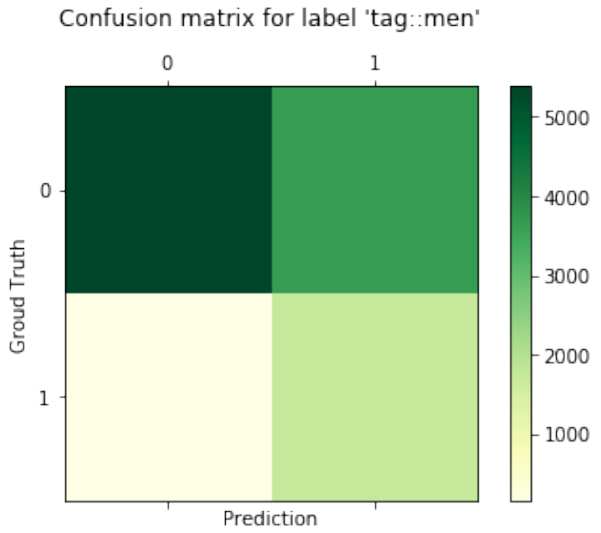

Figure 6: Example confusion matrix for a single label "tag:men". The confusion matrix is plotted by treating the model as a binary classifier.

6. Conclusions and Future Work

In this paper, we explore CNN-extracted and handcrafted features for large-scale multi-label image classification of museum objects. We show that incorporating handcrafted features into CNN provide richer information than CNN-

extracted features alone. Moreover, by applying a visually attended RNN on top of the CNN features, our best model AttnCNN-RNN is able to focus on label-associated image regions and capture inter-dependencies among labels, without the need of pre-ordering the ground truth labels at training stage.

After analyzing the prediction results, we noticed that at prediction time, the model made better predictions on cropped versions of the images than the resized versions. This might be because the aspect ratio of objects in the cropped images are preserved. Therefore, one future direction could be to only crop test images, and combine the predicted labels over these cropped images to obtain the final predictions. In addition, as we put together RNN with visual attention in AttnCNN-RNN, we did not know which one contributes more to the performance boost compared to the baseline model. Hence, another future direction would be to perform ablation test to investigate the relative contributions of RNN and visual attention mechanism.

7. Contributions and Acknowledgements

Siyi set up the starter code for the entire project.

Siyi and Mingkun implemented AttnCNN-RNN and beam search. Ruge worked on extracting handcrafted feature and combined the features into baseline models in the last FC layer and threshold search strategy at evaluation time.

All authors wrote proposal, milestone and final reports together.

We would like to thank The Met, Kaggle and FGVC6 Workshop at CVPR 2019 for hosting this "iMet Collection" competition and providing the data. We also thank the authors of this Github repository, where we referenced the implementations of the attention layer and beam search algorithms for our AttnCNN-RNN model.

References

[1] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.

[2] A. Makadia, V. Pavlovic, and S. Kumar, "A new baseline for image annotation," in *European conference on computer vision*, pp. 316–329, Springer, 2008.

[3] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid, "Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation," in *2009 IEEE 12th international conference on computer vision*, pp. 309–316, IEEE, 2009.

[4] M.-L. Zhang and Z.-H. Zhou, "Multilabel neural networks with applications to functional genomics and text categorization," *IEEE transactions on Knowledge and Data Engineering*, vol. 18, no. 10, pp. 1338–1351, 2006.

[5] Y. Gong, Y. Jia, T. Leung, A. Toshev, and S. Ioffe, "Deep convolutional ranking for multilabel image annotation," *arXiv preprint arXiv:1312.4894*, 2013.

[6] Y. Wei, W. Xia, J. Huang, B. Ni, J. Dong, Y. Zhao, and S. Yan, "Cnn: Single-label to multi-label," *arXiv preprint arXiv:1406.5726*, 2014.

[7] H. Hu, G.-T. Zhou, Z. Deng, Z. Liao, and G. Mori, "Learning structured inference neural networks with label relations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2960–2968, 2016.

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[9] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," *arXiv preprint arXiv:1502.03044*, 2015.

[10] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, "Cnn-rnn: A unified framework for multi-label image classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2285–2294, 2016.

[11] S.-F. Chen, Y.-C. Chen, C.-K. Yeh, and Y.-C. F. Wang, "Order-free rnn with visual attention for multi-label classification," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[12] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 886–893 vol. 1, June 2005.

[13] B. Saleh and A. M. Elgammal, "Large-scale classification of fine-art paintings: Learning the right metric on the right feature," *CoRR*, vol. abs/1505.00855, 2015.

[14] S. Bianco, D. Mazzini, P. Napoletano, and R. Schettini, "Multitask painting categorization by deep multibranch neural network," *CoRR*, vol. abs/1812.08052, 2018.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.