Blending the Past and Present of Automatic Image Annotation

Thesis submitted in partial fulfillment
of the requirements for the degree of

*Master of Science in **Computer Science and Engineering** by Research*

by

Ayushi Dutta
201507668
ayushi.dutta@research.iiit.ac.in

International Institute of Information Technology, Hyderabad
(Deemed to be University)
Hyderabad - 500 032, INDIA
Dec 2019

International Institute of Information Technology, Hyderabad
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "**Blending the Past and Present of Automatic Image Annotation**" by **Ayushi Dutta**, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date	Adviser: Prof. C.V. Jawahar
Date	Co-Adviser: Assistant Prof. Yashaswi Verma

To Sujit & Madhumita

Acknowledgments

For this thesis, I sincerely thank Prof C.V. Jawahar for giving me the opportunity to work with him, and for his subsequent guidance and patience. My special regards to my co-adviser Dr. Yashaswi Verma, who has mentored and supported me throughout my research. I am very grateful towards them. I am also thankful to the IIIT Hyderabad administration for fostering a good research culture. I would like to specially mention the dedication of Prof C.V. Jawahar, Prof P J Narayanan, Prof Jayanthi Sivaswamy, Prof Anoop M. Namboodiri and other faculty members, for bringing CVIT lab to its present form. I have learned a lot from my fellow lab mates who have all sincerely contributed to the success of CVIT.

At IIIT, I have made new friends and discovered the mutual love towards research. I am thankful to all of them, and finally towards a loving family, who I know shall always be there as the backbone to all my endeavours.

Abstract

Real world images depict varying scenes, actions and multiple objects interacting with each other. We consider the fundamental Computer Vision problem of image annotation, where an image needs to be automatically tagged with a set of discrete labels that best describe its semantics. As more and more digital images become available, image annotation can help in the automatic archival and retrieval of large image collections. Being at the heart of image understanding, image annotation can also assist in other visual learning tasks, such as image captioning, scene recognition, multi-object recognition, etc..

With the advent of deep neural networks, recent research has achieved ground-breaking results in single-label image classification. However, for images representing the real world, containing different objects in varying scales and viewpoints, modelling the semantic relationship between images and all of their associated labels continues to remain a challenging problem. Additional challenges are posed from class-imbalance, incomplete labelling, label-ambiguity and several other issues that are commonly observed in the image annotation datasets.

In this thesis, we study the image annotation task from two aspects. First, we bring to attention some of the core issues in the image annotation domain related to dataset properties and evaluation metrics that inherently affect the annotation performance of existing approaches to a significant extent. To examine these key aspects, we evaluate ten benchmark image annotation techniques on five popular datasets using the same baseline features, and perform thorough empirical analyses. With novel experiments, we explore possible reasons behind variations in per-label versus per-image evaluation criteria and discuss when each one of these should be used. We investigate dataset specific biases and propose new quantitative measures to quantify the degree of image and label diversity in a dataset, that can also be useful in developing new image annotation datasets. We believe the conclusions derived in this analysis would be helpful in making systematic advancements in this domain.

Second, we attempt to address the annotation task by a CNN-RNN framework that jointly models label dependencies in an image while annotating it. We base our approach on the premise that labels corresponding to different visual concepts in an image share rich semantic relationships among them (e.g., "sky" is related to "cloud"). We follow recent works that have explored the CNN-RNN style models due to RNN's capacity to model higher-order dependencies, but are limited in their approach to train the RNN in a pre-defined label order sequence. We overcome this limitation and propose a new method to learn multiple label prediction paths. We evaluate our proposed method on a number of popular and relevant datasets and achieve superior results compared to existing CNN-RNN based approaches. We also discuss the scope of the CNN-RNN framework in the context of image annotation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Automatic analysis of useful information from images has been the core objective of Computer Vision, a scientific field that aims to give machines human-like visual sensing capability and automate human vision tasks. One such fundamental problem in Computer Vision is that of tagging images with discrete semantic labels, also called as image annotation.

The present world has witnessed a huge growth in the availability of digital images due to the rapid technological advancements in photo capturing devices and reach of the Internet. By automatic labelling of images with relevant tags, image annotation can help to efficiently archive and access these large image collections, which otherwise would have been infeasible. This also enables search engines, which are quite efficient in doing text-based search, to perform image retrieval [13, 44, 59]. As labels associated with an image correspond to different visual concepts/objects/things present in the image, image annotation lies at the heart of image understanding and is often a precursor to other visual learning problems such as image captioning [54], scene recognition [4], multi-object recognition [35], etc.

The difficulty in automatic image label assignment comes from the well known "semantic gap". There is a fundamental difference between how a machine/computer perceives an image versus how humans do. While humans carry an inherent intuition for recognizing objects in a given context, machines/computers see images as simply a chunk of numbers and work based on precise instructions. Recent research on deep neural networks has achieved ground-breaking results in single-label image classification, and has brought about significant advancements in other Computer Vision problems including image annotation. However, it ignores the fact that a real-world image usually contains multiple objects and may represent complex scenes.

Successfully identifying all of the related concepts/objects/things in an image is still an unsolved problem, and there is need to develop and evaluate new solutions. Challenges arise from the scale of objects present in the image, occlusions, image quality, unconventional viewpoints such as rotations, abstract representations such as sketches, etc. Modelling and learning the semantic association between images and labels require datasets with image examples and their corresponding labels. Since these datasets are annotated by humans, problems like "class-imbalance" (large variations in the frequency of different labels), "incomplete-labelling" (many images are not annotated with all the relevant labels from the vocabulary) and "label-ambiguity" (when there are labels conveying the same meaning, due to

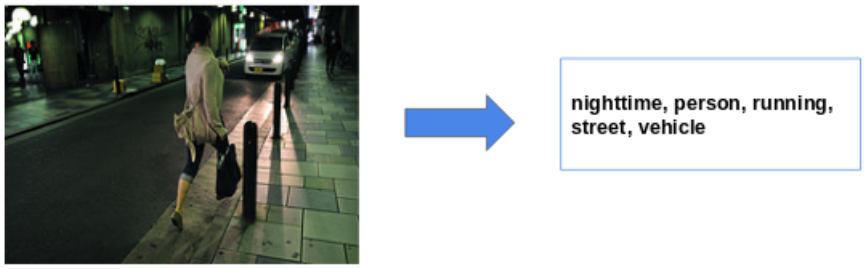

Figure 1.1 An overview of the Image Annotation problem (best viewed in colour).

which usually only one of them is assigned by an annotator) are commonly observed in them. Moreover, the labels are tagged to the image as a whole and there is no information as to which which part of the image corresponds to which label for learning. All of these add to the practical issues in training an image annotation model.

Image annotation approaches thus have to work beyond image classification to tackle these challenges and improve the autonomous understanding of real world images. What seemed to be a basic problem of assigning textual labels to an image, is non-trivial and lessons learned from it can help in the broader scope of problems in Computer Vision.

1.1 Problem Statement

We briefly describe the problem of image annotation.

- **Image Annotation:** The aim is to predict a set of semantic labels for an image from a given vocabulary. The associated labels represent a first-hand understanding of the image. They may refer to the different objects/things in the image, describe the scene/context, and specify varying concepts, attributes or actions as relevant to the image. (Figure 1.1)

1.2 The Context

In the past, several methods have been proposed for image annotation. It was initially addressed by translation [11] and relevance [14, 24, 40] based models. This was followed by nearest neighbour based methods [19, 29, 37, 53] which received tremendous success in image annotation domain. They were based on the simple idea that similar looking images would have similar labels. Then, the advent of deep convolutional neural networks (CNNs) [48] achieved ground-breaking results in single-label image classification and opened new directions of research. This subsequently guided efforts towards using multi-label loss functions [17] to train neural networks for the multi-label image annotation task. In order to further improve its performance, recent methods have since then explored the use of side

information such as user tags/groups from image metadata, or the WordNet hierarchy [23]. Another interesting new line of work has followed, which is based on modelling the inter-label correlations with image word embeddings [15] or CNN-RNN frameworks [25, 55].

All of these methods have helped advance the image annotation domain with their encouraging results on the various annotation datasets. Yet when compared to single label image classification, achieving this improvement has been quite challenging. In this thesis, we thus study the image annotation task from two aspects. First, with empirical analysis we bring to attention some of the fundamental issues mainly related to dataset properties and evaluation metrics, that have an impact on the annotation performance of existing approaches. Second, we follow the recent line of work on CNN-RNN-style models that exploit label dependencies to address the annotation task. We attempt to overcome the training limitation of existing CNN-RNN models that train on a predefined label order sequence, as well as derive useful insights into the learning scope of such models. To summarize, our key contributions are:

- To empirically analyze the core issues/aspects in the image annotation domain, we consider ten benchmark image annotation techniques. We evaluate their results using the same baseline CNN features on five popular image annotation datasets which can be useful in comparing future techniques addressing this task.

- We thoroughly explore the aspects related to the per-label versus per-image performance evaluation metrics, and discuss their impacts on quantitative performance of annotation approaches. As per our knowledge, this is the first study of this nature that is related to the image annotation task.

- We propose novel measures to quantify the degree of diversity (both image as well as label) in image annotation datasets. These are shown to relate with the performance of annotation methods, and can also be useful in developing new image annotation datasets.

- We propose a new CNN-RNN-style model that overcomes the limitation of earlier CNN-RNN models regarding training in a pre-defined label order. We illustrate how our CNN-RNN model can learn multiple label prediction paths, and evaluate its performance for comparison on relevant datasets.

- We derive useful insights into the scope and effectiveness of CNN-RNN-style models in the context of image annotation.

1.3 Organization

In this chapter, we have introduced the image annotation problem, and our motivation and objectives towards addressing some of its challenges. The rest of the thesis is divided into four chapters.

- In Chapter 2, we briefly introduce some of the fundamental concepts in machine learning used in this thesis.

- In Chapter 3, we describe our empirical analysis on the core issues related to the image annotation problem and introduce new measures to counteract them.
- In Chapter 4, we describe and evaluate our proposed CNN-RNN model based solution to address the image annotation task.
- In Chapter 5, we discuss the summary, some directions for future research and conclusions based on this thesis.

Chapter 2

Background

In this chapter, we present a brief background of some machine learning techniques that were adapted in our approach and analysis.

2.1 K-Nearest Neighbour (KNN)

KNN is a non-parametric supervised method that is based on similarity in the feature space. In its simplest form, the label of an unknown test sample is assigned by the majority vote of its K-nearest neighbours from the training data (whose labels are known). If K = 1, then the test data is simply assigned the class of the single nearest neighbour. The assignment for K=1 and K=4 is shown in figure 2.1.

The performance of KNN depends on the value of the hyperparameter K, and also the distance metric used. If the value of K is very small, the test sample ends up with a small neighborhood and this could result in poor performance because of sparse, noisy, ambiguous or poorly labeled data. If we try to increase the value of K, it results in introduction of outliers from other classes. Advanced KNN algorithms also use various weighting schemes to assign weights to the contributions of the neighbours, so that the nearer neighbours contribute more to the majority vote than the distant ones. For example, a common weighting scheme can be to give each neighbour a weight of $1/d$, where d is the distance to the test sample.

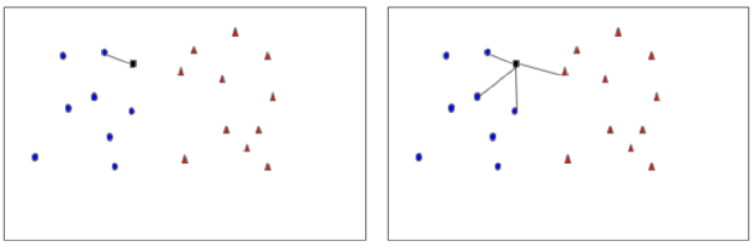

Figure 2.1 An example of K-nearest neighbour assignment with K = 1 (left) and K = 4 (right) (best viewed in colour). Figure reference: [43]

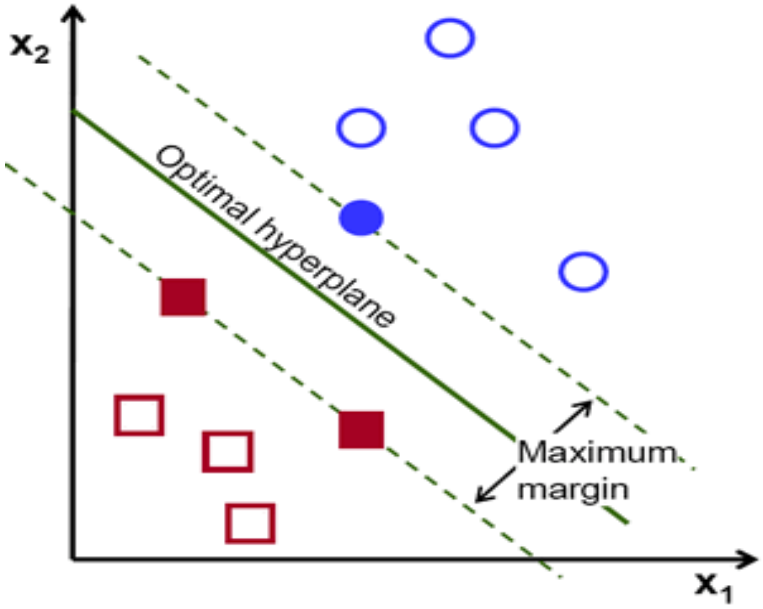

Figure 2.2 Maximum-margin SVM classifier separating samples from two classes (best viewed in colour). Figure reference: [12]

2.2 Support Vector Machine (SVM)

A Support Vector Machine (SVM) is a discriminative classifier defined by a separating hyperplane with maximum margin. In other words, given labeled training data, the algorithm outputs a hyperplane that optimally separates the data according to their labels (positive/negative) with maximum margin.

Let us assume a training set $\{x_i, y_i\}_{i=1}^{n}$ of n examples, where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$ denotes whether it belongs to the class or not. In SVM, the goal is to learn a hyperplane, characterized by the parameters w and b, such that the following constraints are satisfied for all data points:

$$w \cdot x_i + b >= 1, \text{if } y_i = 1 \tag{2.1}$$

$$w \cdot x_i + b <= -1, \text{if } y_i = -1 \tag{2.2}$$

These constraints can be re-written as:

$$y_i(w \cdot x_i + b) >= 1 \tag{2.3}$$

Since this is a hard constraint, we can approximate it by introducing non-negative slack variables ξ_i.

$$y_i(w \cdot x_i + b) >= 1 - \xi_i, \forall i \tag{2.4}$$

This leads to the following optimization problem:

$$\min \frac{\lambda}{2}||w||^2 + \frac{1}{n}\sum_{i=1}^{n}\xi_i \tag{2.5}$$

$$\text{s.t. } y_i(w \cdot x_i + b) >= 1 - \xi_i,\ \xi_i >= 0\ \forall i \tag{2.6}$$

Here, $||\cdot||^2$ is the squared L_2 norm that acts as a regularizer on w and ensures a hyperplane with the maximum margin separation between the two classes. $\lambda > 0$ is a hyperparameter that handles the trade-off between the two terms, the regularization term and the loss function (also called hinge-loss), that penalizes violation of the constraints. Given a new sample x, we predict whether it belongs to the given class or not based on $y = \text{sign}(w \cdot x_i + b)$.

SVM provides several practical advantages such as a convex optimization problem, large margin guarantees, good generalization, scalability, and fast testing time, and is often used as the de facto baseline in classification tasks.

2.3 Kernel Canonical Correlation Analysis (KCCA)

Given different views of the data, such as visual features, audio features, textual features etc., we can construct a common multi-modal representation by a technique called Canonical Correlation Analysis (CCA). We first briefly describe CCA and then move to explain the extended KCCA algorithm.

Let I be an image, $\phi^v(I)$ and $\phi^\tau(I)$ be its two different views of visual and textual features respectively. Given N training pairs of visual and textual features $(\phi^v(I_1), \phi^\tau(I_1)), ..., (\phi^v(I_N), \phi^\tau(I_N))$ of images $I_1 ... I_N$, the goal of CCA is to simultaneously find directions z_v^* and z_τ^* that maximize the correlation of the projections of feature spaces ϕ^v onto z_v^* and ϕ^τ onto z_τ^*. Formally,

$$z_v^*, z_\tau^* = \arg\max_{z_v, z_\tau} \frac{E[< \phi^v, z_v >< \phi^\tau, z_\tau >]}{\sqrt{E[< \phi^v, z_v >^2 < \phi^\tau, z_\tau >^2]}} \tag{2.7}$$

$$= \arg\max_{z_v, z_\tau} \frac{z_v^T C_{v\tau} z_\tau}{\sqrt{z_v^T C_{vv} z_v z_\tau^T C_{\tau\tau} z_\tau}} \tag{2.8}$$

where $E[\cdot]$ denotes the empirical expectation, C_{vv} and $C_{\tau\tau}$ denote the auto-covariance matrices for ϕ^v and ϕ^τ respectively, and $C_{v\tau}$ denotes the between-sets covariance matrix.

As the CCA algorithm can model only linear relationships, KCCA was introduced to model non-linear relationships and let data projections onto higher-dimensional feature spaces by the kernel trick. The problem then is to search for solutions of z_v^* and z_τ^* that lie in the span of the N training instances $\phi^v(I_i)$ and $\phi^\tau(I_i)$:

$$z_v^* = \sum_{i=1}^{N} \alpha_i \, \phi^v(I_i), \;\; z_\tau^* = \sum_{i=1}^{N} \beta_i \, \phi^\tau(I_i) \tag{2.9}$$

The objective of KCCA is to identify the weights $\alpha, \beta \in \mathcal{R}^N$ that maximize:

$$\alpha^*, \beta^* = \arg\max_{\alpha, \beta} \frac{\alpha^T K^v K^\tau \beta}{\sqrt{\alpha^T (K^v)^2 \alpha \beta^T (K^\tau)^2 \beta}} \tag{2.10}$$

where K^v and $K^\tau \in \mathcal{R}^{NXN}$ denote the kernel matrices calculated over N image pairs, of visual and textual features respectively.

This leads to solving an eigen value problem with top M eigen vectors, and yields the coefficients $A = [\alpha_1, ..., \alpha_M]$ and $B = [\beta_1, ..., \beta_M]$. For each pair (α_j, β_j) of the given bases, the corresponding eigen-value r_j measures the correlation between the projected input pairs. Let $R = \text{diag}([r_1, ..., r_M])$, thus the final features can be obtained by projecting the visual features onto a common subspace as:

$$\psi(I) = (K^v A)R \tag{2.11}$$

2.4 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) is a feed-forward artificial neural network with learnable weights and biases, that can operate on input volumes such as multi-channeled images. Inspired by biological processes, its connectivity pattern between neurons is analogous to the organization of the animal visual cortex. While artificial neural networks have been in use for a variety of tasks, it was CNN that first demonstrated the capability of incorporating a large number of hidden layers in a network. Since then, it has led to a monumental growth in the ability to solve Computer Vision problems.

A CNN is typically a sequence of layers having neurons, that transform one volume of activations to another through a differentiable function starting from the raw image pixels on one end to class scores at the other. The neurons in a layer are connected to a small region of the layer before it, except at the fully connected layer(s) towards the end of the network. The parameters of the network are end-to-end trainable using a loss function on the output of the last layer. A CNN architecture (Figure 2.3) is mainly composed of an input layer, a stack of convolutional, ReLU and pooling layers, and finally the fully-connected layers, as described below.

Input Layer:

Input is an image of dimension *height* $\times$ *width* $\times$ *depth*, containing raw pixel values and *depth* denoting the three color channels (R,G and B).

Convolutional Layer:

The convolutional layer is the core building block of a CNN, consisting of a set of learnable parameters called filters, which are 3-dimensional volumes small spatially (along width and height) but with same depth as the input volume (Figure 2.4). Every filter is convolved spatially across the input during the forward pass, to compute dot products between the filter (weights and biases) and the local input region at any spatial position of the input volume. This produces a 2-dimensional activation map showing filter responses at different spatial positions. In other words, the network learns filters that activate when it detects some specific type of feature (e.g., edges of some orientation, specific patterns, etc.) at some spatial position in the input. The output volume of the convolution layer is the activation maps for all filters stacked along the depth dimension.

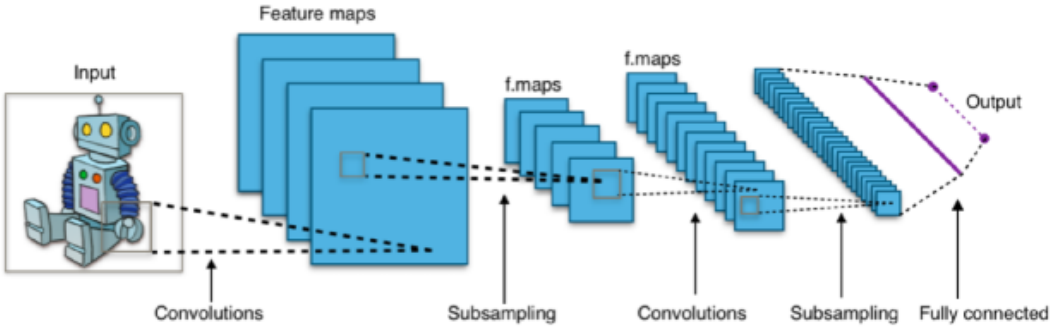

Figure 2.3 Example of a typical convolutional neural network architecture (best viewed in colour). Figure reference: [1]

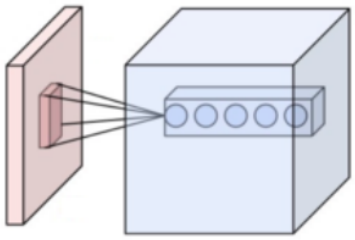

Figure 2.4 Connections between neurons of a Convolutional Layer (blue) and the input volume(red) (best viewed in colour). Figure reference: [2]

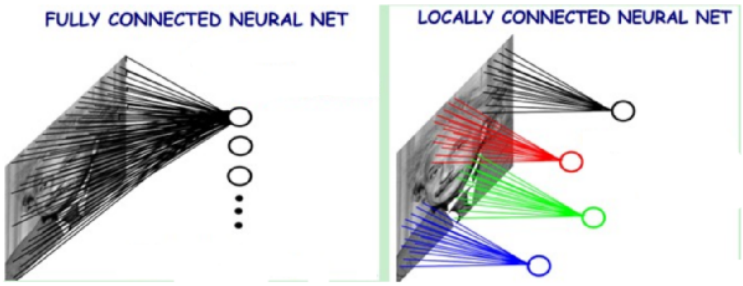

Figure 2.5 Local Connectivity of Convolutional Layers (best viewed in colour). Figure reference: [42]

(a) Max Pooling Operation. (b) Average Pooling Operation.

Figure 2.6 Examples of Pooling Layers (best viewed in colour). Figure reference: [42]

Pooling Layer:

The pooling layer performs a downsampling operation on the input along its spatial dimensions (width, height), to reduce the spatial size of the representation progressively in the network. It is done on every depth slice of the input independently, thus keeping the depth dimension unchanged. Pooling reduces the amount of network parameters, controls overfitting and provides a form of translation invariance. Two most common forms of it used are max pooling and average pooling. Max pooling replaces the input region it is connected to with the maximum value, whereas average pooling replaces it with the mean (or average) value of the input region (Figure 2.6).

ReLU Layer:

The ReLU layer applies an element-wise activation function $\max(0, x)$, that thresholds the neurons' outputs at zero. This layer adds non-linearity to the CNN.

Fully-Connected Layer:

As the name implies, each neuron in a fully-connected layer is connected to all of the neurons in the previous layer (Figure 2.5). Generally for classification task, the final fully-connected layer of any CNN consists of C hidden-units, where C is the number of classes. The output of the C classes is passed through a softmax or sigmoid activation function to obtain class-probability scores corresponding to each class.

2.5 Recurrent Neural Network (RNN)

Recurrent Neural Network (RNN) is a type of artificial neural network where the output from the previous step can be fed as input to the current step. Thus RNN has a memory that captures information of what has been calculated (seen) so far, which helps it in processing sequential information, such as generating words of a sentence.

Figure 2.7 shows the example of an RNN unrolled over three time-steps; with an input x_t and a hidden state ("memory") s_t, at time-step t. $s_t = f(Ux_t + Ws_{t-1})$ where f is a non-linear activation

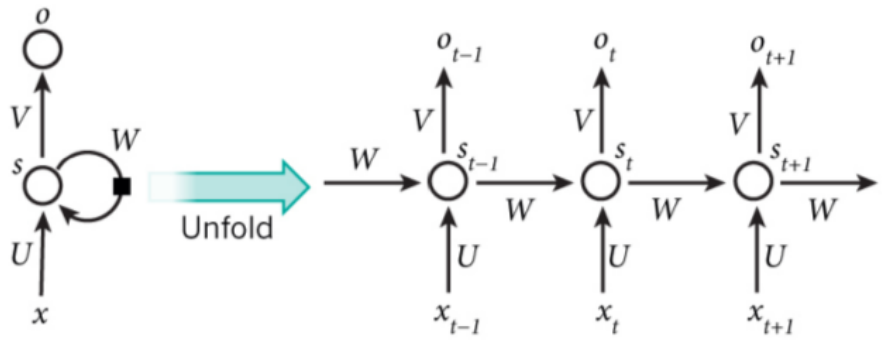

Figure 2.7 A Recurrent Neural Network and the unfolding in time of the computation involved in its forward computation. Figure reference: [42]

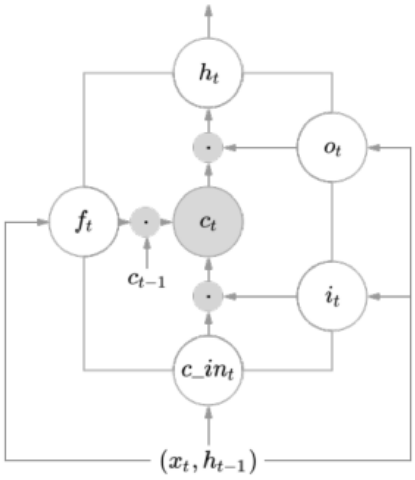

Figure 2.8 A Long Short Term Memory Cell. Figure reference: [42]

function and U, W are the network weights. The output o_t at time t is obtained as $o_t = \text{softmax}(Vs_t)$. The parameters U, V and W are shared across all time-steps, which reflects the recurrent nature of the RNN.

Long Short Term Memory (LSTM)

In theory, RNNs can make use of information in arbitrarily long sequences, but in practice they are limited to looking back only a few steps. This is due to the difficulty in training long sequences on account of vanishing gradients. Long Short Term Memory network (LSTM) is a special type of RNN with a memory cell unit capable of learning long-term dependencies. It does so by the use of specialized neurons made generally of *sigmoid* layer(s) called "gates", that regulate the information flow in its memory cell. There are mainly three gates: an " "input" gate which controls the extent to

which the current input influences the value of the memory cell, a "forget" gate which decides what information to throw away from the old cell state, and an "output" gate which selectively propagates the cell's new value to the output.

There are different variants of LSTM, but in general a LSTM cell can be described by the following equations:

$$\begin{aligned}
\text{Input gate } i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) && (2.12)\\
\text{Forget gate } f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) && (2.13)\\
\text{Output gate } o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) && (2.14)\\
\text{Input transform } c_in_t &= \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_{c_in}) && (2.15)\\
\text{Cell state update } c_t &= f_t \cdot c_{t-1} + i_t \cdot c_in_t && (2.16)\\
\text{Output } h_t &= o_t \cdot \tanh(c_t) && (2.17)
\end{aligned}$$

where x_t is the input vector, h_t is the output vector and c_t is the cell state at time t; $\sigma(\cdot)$ is the sigmoid activation function; $W_{h,\cdot}$ are the recurrent weights, $W_{x,\cdot}$ are the input weights and $b\cdot$ are the biases.

2.6 Summary

We have briefly described some of the machine learning techniques that will be used in subsequent chapters. We began by introducing some of the traditional methods such as KNN, SVM and KCCA that are still relevant in the image annotation context as shall be seen in Chapter 2. We then presented a short note on CNN, that provides a modern strong representation and has been used as the de-facto feature in all of our experiments. We finally described another modern representation called RNN, which we shall use in chapter 3 in conjunction with CNN, to help design a unified framework for classifying images while learning its label dependencies.

Chapter 3

The Quirks of What Works

3.1 Introduction

During the last decade, a large number of non-deep-learning based [5, 6, 14, 16, 19, 28, 33, 34, 37, 38, 40, 51, 52, 53, 57, 58] and deep-learning based [17, 23, 27, 30, 32, 45, 55] image annotation techniques have been proposed that have been shown to achieve encouraging results on various annotation datasets. However, their scope has mostly remained restricted to advancing quantitative results on the test data, without giving much attention to some of the core issues related to evaluation criteria and dataset properties.

By convention, the performance of image annotation methods has been reported using both per-label and per-image metrics. Per-image metrics indicate the completeness of labels assigned to an image, and per-label is a measure of image retrieval performance of a label. Though quantitatively one shall wish to improve both per-label and per-image results, the two evaluation criteria with their specific biases can give different interpretations regarding the performance of the annotation method, and often one may be preferred over the other.

Alternatively, an image annotation method's performance is indicated by the dataset on which it has been trained and evaluated. While it is known that problems like "class-imbalance" (large variations in the frequency of different labels) and "incomplete-labelling" (many images are not annotated with all the relevant labels from the vocabulary) are commonly observed in image annotation datasets, the amount of diversity in images and labels present in these datasets is also an important factor that needs to be considered.

These key aspects in the image annotation domain can have a significant impact while working in real-world settings. In this chapter, we aim to bring close attention to them via thorough empirical analyses. We explore the possible reasons behind variations in per-label and per-image evaluation metrics with novel experiments. Further, we also investigate the effect of dataset specific biases, and propose new quantitative measures that can be useful in developing new datasets.

To perform our analysis, we consider ten benchmark image annotation techniques (five deep-learning based and five non-deep learning based), and use the same baseline features. We evaluate our models using features from the state-of-the-art deep convolutional neural network (CNN) models

(GoogLeNet [49] and ResNet [21]), which were initially trained on the ImageNet [46] 1000-class classification dataset. These results can be useful in comparing future techniques addressing this task. In the chapter, we illustrate our findings using the GoogleLeNet [49] features, and report corresponding results using the ResNet [21] features in appendix A. The conclusions derived in our study are independent of the features used, as can be observed from the results using both GoogLeNet [49] and ResNet [21].

3.2 Label Prediction Models

We first describe the benchmark annotation methods we have considered in our experiments, before proceeding to our experiments and analysis. As per our understanding, these techniques have made notable contributions in advancing this challenging area, and have also reported some of the best results on standard datasets.

3.2.1 Non-deep learning based methods

The non-deep learning based methods can be applied on both handcrafted features as well as features extracted from a pretrained deep neural network. Unlike a deep learning based method, they are not end-to-end trainable; i.e., the algorithm cannot refine the features which it takes as an input.

Here we use the features extracted from the penultimate layers of the GoogLeNet [49] and ResNet [21] convolutional neural network(CNN) models, which were initially trained on the ImageNet [46] 1000-class classification dataset. Following this, these features are transformed into an embedding space learned using Kernel Canonical Correlation Analysis (KCCA) [20] (refer section 2.3). This embedding is expected to provide a semantically richer representation than using raw features. We consider five non-deep learning methods namely, JEC [37, 38], TagRel [29], TagProp [19], 2PKNN [51, 53] and SVM [9].

3.2.1.1 Joint Equal Contribution (JEC)

Based on the idea that similar images should share similar labels, Makadia *et al.* [37, 38] proposed a greedy nearest-neighbour based approach for image annotation. Given a test image, initially it transfers the labels occurring in its first nearest (or visually the most similar) training image in the order of their frequencies in the training set. Then it picks labels from additional neighbors, and considers their frequencies and co-occurrence with the initially assigned labels for further assignment. Formally, the algorithm is described as follows.

Let I be the query image to which n keywords need to be assigned using the query's K nearest neighbors in the training set. Let $I_i, i = 1, ..., K$ be the K nearest neighbors ordered by increasing distance (i.e., I_1 is the most similar image). The number of keywords associated with I_i is denoted by $|I_i|$.

1. Rank the keywords of I_1 according to their frequencies in the training set.

2. Of the $|I_1|$ keywords of I_1, transfer the n highest ranking keywords to query I. If $|I_1| < n$, proceed to step 3.

3. Rank the keywords of neighbors I_2 through I_K according to two factors: 1) co-occurrence in the training set with the keywords transferred in step 2, and 2) local frequency (i.e., how often they appear as keywords of images I_2 through I_K), and then select the highest ranking $n - |I_1|$ keywords to transfer to I.

3.2.1.2 Tag Relevance (TagRel)

Li *et al.* [29] proposed an annotation approach based on the idea that for a given test image, the degree of relevance of a label is proportional to its frequency in the neighbor set of that image. However, in order to penalize very high frequency of a given label occurring in the neighborhood, it also takes into account the overall frequency of that label in the complete training set.

Given an image I, tag τ, number of neighbors K, the number of images with tag τ in the visual neighborhood k_τ, and n_τ as the number of images labeled with τ in the entire collection S, the tag relevance score of τ with respect to I is calculated as:

$$f_{TagVote}(I, \tau) = k_\tau - K \cdot \frac{n_\tau}{|S|}$$

3.2.1.3 TagProp

Guillaumin *et al.* [19] proposed a weighted nearest neighbour model for image annotation. Given an image, the model takes weighted average of labels occurring in the neighbour set of that image. To estimate the parameters that control the weights, it tries to maximize the likelihood of true labels during training.

Let $y_{i\tau} \in \{-1, +1\}$ denote the absence/presence of keyword τ for image i. The tag presence prediction $p(y_{i\tau} = +1)$ for image i is a weighted sum over the training images, indexed by j:

$$p(y_{i\tau} = +1) = \sum_j \pi_{ij} p(y_{i\tau} = +1|j) \tag{3.1}$$

$$p(y_{i\tau} = +1|j) = \begin{cases} 1 - \epsilon & \text{for } y_{j\tau} = +1, \\ \epsilon & \text{otherwise} \end{cases} \tag{3.2}$$

where $\pi_{ij} >= 0$ denotes the weight of image j for predicting the tags of image i, $\pi_{ii} = 0$, $\sum_j \pi_{ij} = 1$ and $\epsilon = 10^{-5}$.

$$\pi_{ij} = \frac{\exp(-d_\theta(i, j))}{\sum_{j'} \exp(-d_\theta(i, j'))} \tag{3.3}$$

where d_θ is a distance metric with parameters θ that we want to optimize. To estimate the parameters that control the weights π_{ij}, we maximize the log-likelihood of the predictions of training annotations.

$$\mathcal{L} = \sum_{i,\tau} c_{i\tau} \ln p(y_{i\tau}) \tag{3.4}$$

where $c_{i\tau}$ is a cost that takes into account the imbalance between keyword presence and absence. Additionally, to boost the performance of rare labels, sigmoid functions with parameters $\{\alpha_w, \beta_w\}$ are learned per label τ.

$$p(y_{i\tau} = +1) = \sigma(\alpha_\tau x_{i\tau} + \beta_\tau) \tag{3.5}$$

$$x_{i\tau} = \sum_j \pi_{ij} y_{j\tau} \tag{3.6}$$

3.2.1.4 2PKNN

Inspired by the success of JEC [37, 38] and TagProp [19], Verma and Jawahar proposed a 2-Pass k-Nearest Neighbour (2PKNN) [51, 53] algorithm which is a two-step approach. Given a test image, the first step constructs a balanced neighbourhood that ensures a certain minimum number of occurrences of each label, thus addressing class-imbalance. Then in the second step, labels are propagated from the initially picked neighbours by computing a weighted average of their relevance based on visual similarity with the corresponding neighbouring images.

Let $\mathcal{T}_l \subseteq \mathcal{T}, \forall l \in \{1, ..., L\}$ be the subset of training data that contains all the images annotated with the label y_l. The sets $\mathcal{T}_l$, also called semantic groups, are not disjoint as an image usually has multiple labels and hence belongs to multiple semantic groups. Given a test image I, from each semantic group we pick K_1 images that are most similar to I and form corresponding sets $\mathcal{T}_{I,l} \subseteq \mathcal{T}_l$. Once $\mathcal{T}_{I,l}$(s) are determined, we merge them to form a set $\mathcal{T}_I = \{\mathcal{T}_{I,1} \cup ... \cup \mathcal{T}_{I,l}\}$.

The second pass of 2PKNN is a weighted sum over the samples in $\mathcal{T}_I$ to assign importance to labels based on image similarity. This gives the posterior probability for I given a label $y_l \in \mathcal{Y}$ as

$$P(I|y_l) \propto \sum_{(I_j, Y_j) \in \mathcal{T}_I} \theta_{I,I_j} \cdot P(y_l|I_j) \tag{3.7}$$

where, $\theta_{I,I_j} = \exp(-\pi D(I, I_j))$ denotes the contribution of image I_j in predicting the label y_l for I depending on their visual similarity, with π being a scalar that controls the decay of θ_{I,I_j} and $D(I, I_j)$ being the distance between I and I_j in the feature space. $P(y_l|I_j)$ is 1 or 0 depending on the presence or absence of the label y_l in image I_j.

3.2.1.5 Support Vector Machines (SVM)

A simple binary SVM classifier [9] (refer section 2.2) can be extended for the multi-label case by learning a classifier for each label (one-versus-rest). Precisely, to learn a classifier for a given label, all the training images annotated with that label are considered as positive samples and the rest as negative

samples. Since this approach addresses the problem of multi-label image annotation on an individual label basis, it ignores the co-existence of other labels.

3.2.2 Deep Learning based methods

Deep convolutional neural networks (CNNs), have achieved great success on single-label image classification task. For multi-label prediction, we can directly train the CNN with loss functions tailored for it. Loss functions penalize the differences in ground-truth and predicted labels. Precisely, we consider five loss functions: Softmax [17], Sigmoid, Pairwise Ranking (or simply Ranking) [17], WARP [17, 57], and the recently proposed LSEP [30]. We adopt GoogLeNet [49] and ResNet [21] pre-trained trained on the ImageNet [46] 1000-class classification dataset as our CNN models, and fine-tune on our datasets.

We assume that we have a set of n images with label vector $y \in \mathbb{R}^c$, with y_j being 1 if the image has the label j else 0, and c being the number of classes. Let $f_j(x_i)$ be the activation value from the CNN for image x_i and class j.

3.2.2.1 Softmax Cross Entropy

Since each image has multiple labels as denoted by its label vector y_i, we can obtain its ground-truth probability p_i by normalizing y_i as $y_i/||y_i||_1$. The posterior probability of an image x_i and class j can be expressed as:

$$\hat{p}_{ij} = \frac{\exp(f_j(x_i))}{\sum_{k=1}^{c} \exp(f_k(x_i))} \tag{3.8}$$

The cost function to be minimized is the standard cross entropy loss between the predictions and the ground-truth probabilities:

$$J = -\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{c} p_{ij} \log(\hat{p}_{ij}) \tag{3.9}$$

3.2.2.2 Sigmoid Cross Entropy

The posterior probability of an image x_i and class j is obtained by the sigmoid function $\hat{p}_{ij} = 1/(1 + \exp(-f_j(x_i)))$. The ground truth probability p_i is the label vector y_i. The cost function to be minimized is the standard cross entropy loss between the predictions and the ground-truth probabilities:

$$J = -\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{c} (p_{ij} \log(\hat{p}_{ij}) + (1 - p_{ij}) \log(1 - \hat{p}_{ij})) \tag{3.10}$$

3.2.2.3 Pairwise Ranking

We want to rank the positive labels to always have higher scores than negative labels, which leads to the following loss function:

$$J = \sum_{i=1}^{n} \sum_{j=1}^{c+} \sum_{k=1}^{c-} \max(0, 1 - f_j(x_i) + f_k(x_i)) \tag{3.11}$$

where $c+$ is the count of positive labels and $c-$ is the count of negative labels.

3.2.2.4 Weighted Approximate Ranking(WARP)

Gong *et al* [17, 57] proposed the WARP loss to specifically optimize the top-k accuracy for annotation. Weights are assigned to the loss terms of the pairwise ranking loss as per the label ranks such that, if a positive label is ranked top in the label list, then we assign a small weight to the loss, but when a positive label is not ranked top, we assign a much larger weight which pushes the positive label to the top.

$$J = \sum_{i=1}^{n} \sum_{j=1}^{c+} \sum_{k=1}^{c-} L(r_j) \max(0, 1 - f_j(x_i) + f_k(x_i)) \tag{3.12}$$

where $L(\cdot)$ is a weighting function, r_j is the rank for the jth class.

$$L(r) = \sum_{j=1}^{r} \alpha_j, \text{with } \alpha_1 \geq \alpha_2 \geq ... \geq 0 \tag{3.13}$$

α_j is equal to $1/j$. The ranks r_j are estimated by using a stochastic sampling approach. For a positive label, we continue to sample negative labels until we find a violation; then we record the number of trials s we sampled for negative labels.

$$r_j = \lfloor \frac{c-1}{s} \rfloor \tag{3.14}$$

for c classes and s sampling trials.

3.2.2.5 Log-Sum-Exp Pairwise Ranking(LSEP)

Recently, Li *et al.* [30] introduced the LSEP loss, which is a differentiabe pairwise ranking loss. It minimises:

$$J = \sum_{i=1}^{n} \log(1 + \sum_{v \notin y_i} \sum_{u \in y_i} \exp(f_v(x_i) - f_u(x_i))) \tag{3.15}$$

3.3 Model Comparison

In this section, we present the datasets, evaluation metrics and quantitative results of the benchmark annotations methods described in the previous section 3.2.

3.3.1 Datasets

We use the following benchmark datasets in our analyses, a subset of which has been used by almost all the existing techniques in their evaluations.

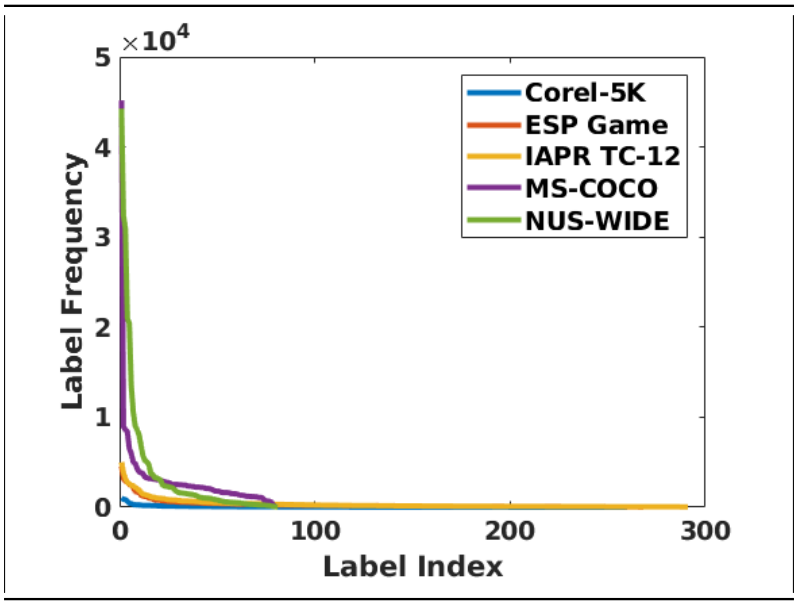

Figure 3.1 Frequency of labels in the training sets of each of the five datasets sorted in decreasing order (best viewed in colour).

1. **Corel-5K** [11]: It contains $4,500$ training and 499 testing images. Each image is annotated with up to 5 labels, with 3.4 labels per image on an average. This is one of the oldest image annotation datasets, and was considered as the de facto benchmark for evaluation until recently. Since most of the recent image annotation techniques are based on deep neural networks and require large amount training data, there has been a decline in the usage of this dataset.

2. **ESP Game** [3]: This dataset contains $18,689$ training and $2,081$ testing images, with each image being annotated with up to 15 labels and 4.7 labels on an average. It was formed using an online game where two mutually unknown players are required to assign labels to a given image, and score points for every common label. This way, several participants perform the manual annotation task, thus making this dataset quite challenging.

3. **IAPR TC-12** [18]: It contains $17,665$ training and $1,962$ testing images. Each image is annotated with up to 23 labels, with 5.7 labels per image on an average. In this dataset, each image is associated with a long description in multiple languages. Makadia *et al.* [37, 38] extracted nouns from the descriptions in the English language and treated them as annotations. Since then, it has been used extensively for evaluating image annotation methods.

4. **NUS-WIDE** [8]: This is the largest publicly available image annotation dataset, containing $269,648$ images downloaded from Flickr. The vocabulary contains 81 labels, with each image being annotated with up to 3 labels. On an average, there are 2.40 labels per image. Following the earlier papers [17, 50], we discard the images without any label. This leaves us with $209,347$

images, that we split into ~ 125K images for training and ~ 80K for testing by adopting the split originally provided by the authors of this dataset.

5. **MS-COCO** [35]: This is the second largest popular image annotation dataset, and is primarily used for object recognition in the context of scene understanding. It contains $82,783$ training images and 80 labels, with each image being annotated with 2.9 labels on an average. For this dataset, the ground-truth of the test set is not publicly available. Hence, we consider the validation set containing $40,504$ images as the test set in our experiments.

Figure 3.1 shows the frequencies of the labels in these datasets sorted in descending order. The graphs of these datasets drop rapidly, and almost level out near the tail. This shows the aspect of "class-imbalance" i.e, each of these datasets contains only a small number of high frequency labels, and a huge number of labels with very low frequencies. The graph also shows that the Corel-5K ,ESP Game and IAPR TC-12 datasets suffer more from the "class-imbalance" problem than the newer datasets of NUS-WIDE and MS-COCO. "Class-imbalance" has an effect on the overall annotation performance as explained in section 3.4.1.

3.3.2 Evaluation Metrics

To analyze annotation performance, we consider both per-label as well as per-image evaluation metrics. Below we describe the metrics in these two categories:

3.3.2.1 Per-label evaluation metrics

Here, we consider per-label precision, recall and mean average precision (mAP). Given a label, let it be present in the ground-truth of m_1 images, and during testing let it be predicted for m_2 images out of which m_3 predictions are correct ($m_3 \leq m_1$ and $m_3 \leq m_2$). Then the precision for this label will be m_3/m_2, and recall will be m_3/m_1. These values are averaged over all the labels in the vocabulary to get average (percentage) per-label precision ($\mathrm{P_L}$) and average per-label recall ($\mathrm{R_L}$) respectively. From these, we compute average per-label F1 score ($\mathrm{F1_L}$), which is the harmonic mean of $\mathrm{P_L}$ and $\mathrm{R_L}$; i.e., $\mathrm{F1_L} = 2 \times \mathrm{P_L} \times \mathrm{R_L} / (\mathrm{P_L} + \mathrm{R_L})$. We also consider the N+ metric, that counts the number of labels with positive recall (or, how many labels in the vocabulary are correctly predicted for at least one test image). Additionally, we compute label-centric mAP ($\mathrm{mAP_L}$) that measures the quality of image-ranking corresponding to each label.

3.3.2.2 Per-image evaluation metrics

Here, we consider per-image precision, recall and mAP. Given an image, let there be n_1 labels present in its ground-truth, and during testing a model predicts n_2 labels out of which n_3 predictions are correct ($n_3 \leq n_1$ and $n_3 \leq n_2$). Then the precision for this image will be n_3/n_2, and recall will be n_3/n_1. These values are averaged over all the images in the test set to get average (percentage)

Method	Per-label metrics					Per-image metrics			
	P_L	R_L	$F1_L$	mAP_L	N+	P_I	R_I	$F1_I$	mAP_I
Johnson[27]	54.74	57.30	55.99	61.88	–	53.46	75.10	62.46	80.27
Hu[23] (1)	57.02	59.82	58.39	67.20	–	56.84	78.78	66.04	89.99
Hu[23] (2)	58.30	60.63	59.44	69.24	–	57.05	79.12	66.30	82.53
Liu [36]	71.73	61.73	66.36	–	–	77.41	76.88	77.15	–
Gong[17]	31.65	35.60	33.51	–	80	48.59	60.49	53.89	–
Ren[45]	37.74	40.15	38.91	–	81	52.23	65.03	57.93	–
Wang[55]	40.50	30.40	34.73	–	–	49.90	61.70	55.18	–
Liu [36]	55.65	50.17	52.77	–	–	70.57	71.35	70.96	–
SoftMax*	45.16	51.72	48.22	46.45	81	52.98	74.92	62.07	79.95
Sigmoid*	45.91	52.18	48.85	53.97	81	53.84	75.69	62.92	81.19
Ranking*	44.49	51.70	47.82	45.41	81	52.84	74.27	61.75	79.34
WARP*	43.91	53.17	48.09	46.04	81	53.03	74.56	61.98	79.54
LSEP*	44.29	53.46	48.45	49.32	81	53.64	75.54	62.73	80.91
JEC	37.15	40.91	38.94	21.63	80	29.36	69.32	41.25	61.68
TagRel	39.75	59.27	47.58	49.15	81	49.87	70.71	58.49	73.55
TagProp	48.84	58.10	53.07	53.81	80	51.52	73.16	60.46	76.90
2PKNN	52.49	52.28	52.38	51.96	81	45.30	64.77	53.31	67.82
SVM	46.56	52.38	49.30	51.84	81	53.31	74.96	62.31	79.87

Table 3.1 Performance comparison of various annotation models (deep-learning based models are marked by '*') on the NUS-WIDE dataset using *GoogLeNet* features.

per-image precision P_I and average per-image recall R_I. From these two scores, we compute per-image F1 score ($F1_I$), which is the harmonic mean of P_I and R_I. We also compute image-centric mAP (mAP_I) that measures the quality of label-ranking corresponding to each image.

3.3.2.3 Label Assignment

Unless stated otherwise, we follow the earlier papers [14, 17, 19, 50, 51] and assign the top 5 labels to each test image in the Corel-5K, ESP Game and IAPR TC-12 datasets, and the top 3 labels in the NUS-WIDE and MS-COCO datasets for evaluating all the metrics, except for mAP_L and mAP_I that are evaluated on the complete ranked list of all the (test) images and labels respectively.

3.3.3 Results

In Table 3.1 (for NUS-WIDE), Table 3.2 (for MS-COCO) and Table 3.3 (for Corel-5K, ESP Game and IAPR TC-12), we compare the annotation performance of different label prediction models. In general, we can observe significant variations in the results on different datasets. This is because of the differences in how these datasets were created and their vocabularies. For datasets with large vocabularies (Corel-5K, ESP Game and IAPR TC-12), the results are usually lower than those with small

Method	Per-label metrics					Per-image metrics			
	P_L	R_L	$F1_L$	mAP_L	N+	P_I	R_I	$F1_I$	mAP_I
SoftMax*	58.87	57.41	58.13	59.47	80	58.32	71.76	64.35	80.34
Sigmoid*	61.14	58.87	59.98	67.74	80	60.16	73.22	66.05	82.37
Ranking*	58.84	57.18	57.99	61.48	80	58.65	71.48	64.43	80.22
WARP*	59.21	56.44	57.79	61.07	80	58.27	71.00	64.01	79.80
LSEP*	60.16	59.24	59.70	63.92	80	59.80	73.05	65.77	82.04
JEC	53.03	42.68	47.30	29.63	80	49.52	61.52	54.87	55.82
TagRel	52.36	57.61	54.86	62.87	80	54.20	67.07	59.95	73.35
TagProp	60.35	56.82	58.53	63.13	80	57.23	70.13	63.02	77.97
2PKNN	71.00	49.25	58.16	61.03	80	50.20	61.50	55.28	69.74
SVM	61.71	59.07	60.36	68.33	80	60.11	73.03	65.94	81.75

Table 3.2 Performance comparison of various annotation models (deep-learning based models are marked by '*') on the MS-COCO dataset using *GoogLeNet* features.

vocabularies (NUS-WIDE and MS-COCO). Another reason is the diversity in these datasets (in terms of both images as well as labels), that we will analyze and discuss in Section 3.4.2.

In Table 3.1, the first two blocks show the performances reported by some recent techniques, all of which are based on some end-to-end trainable deep neural network. Note that while the techniques in the second block make use of only the available training data (images and their labels), those in the first block also make use of additional meta-data such as either social tags [27, 36] or the WordNet hierarchy [23]. Due to this, while these are able to achieve significantly higher results than others, these can not be compared directly. From Table 3.1, we observe that both non-deep-learning methods as well as deep-learning based methods (that do not use additional meta-data) give comparable results.

In Table 3.2, we compare the performance of various methods on the MS-COCO dataset. Here, we observe that generally TagProp and SVM achieve the best results among the non-deep-learning based methods, and Sigmoid and LSEP achieve the best results among the deep-learning based methods. Similar to the NUS-WIDE dataset, the best results using both deep as well as non-deep methods are comparable.

In Table 3.3, we compare the performance of the five non-deep-learning based methods on small-scale datasets using GoogLeNet features. Here, we do not consider deep-learning based methods since they require large amount of data for proper training. From the results, we observe that 2PKNN generally achieves the best performance.

In general, for all the datasets and methods, we can observe that the scores corresponding to per-image metrics are higher than per-label metrics. We will study this trend in Section 3.4.1, where we will empirically show that per-image metrics show some bias towards good performance on frequent labels.

Method	Per-label metrics					Per-image metrics			
	P_L	R_L	$F1_L$	mAP_L	N+	P_I	R_I	$F1_I$	mAP_I
Corel-5K									
JEC	41.70	44.95	43.27	37.29	161	45.97	64.92	53.76	50.87
TagRel	40.64	46.43	43.34	41.81	167	45.29	63.76	52.96	58.56
TagProp	37.88	42.79	40.19	43.15	155	46.05	65.27	54.00	60.13
2PKNN	46.10	52.85	49.25	53.18	197	44.48	62.60	52.01	57.89
SVM	36.64	46.29	40.90	53.15	158	48.42	68.77	56.83	67.30
ESP Game									
JEC	45.15	31.39	37.03	21.66	239	41.85	47.06	44.31	35.83
TagRel	38.89	42.05	40.41	38.81	252	43.57	48.52	45.92	49.91
TagProp	44.48	41.23	42.79	38.95	250	44.17	49.77	46.80	51.44
2PKNN	45.48	42.20	43.78	41.40	260	43.89	49.43	46.50	51.57
SVM	44.21	36.07	39.73	41.33	245	47.13	52.97	49.88	55.70
IAPR TC-12									
JEC	44.52	27.77	34.20	20.08	226	49.62	47.92	48.76	38.08
TagRel	45.07	42.21	43.59	39.71	267	50.66	49.01	49.82	52.40
TagProp	49.13	41.73	45.13	44.18	270	50.39	49.18	49.78	55.40
2PKNN	50.77	41.64	45.75	46.39	275	50.41	48.72	49.55	56.39
SVM	51.13	30.81	38.45	46.67	235	54.41	52.63	53.50	60.60

Table 3.3 Performance comparison of various annotation methods on Corel-5K, ESP Game and IAPR TC-12 datasets using *GoogLeNet* features.

3.4 Analysis

Now we analyze various aspects of image annotation datasets and performance evaluation metrics by considering the ten annotation methods discussed in Section 3.2 as the working examples wherever required.

3.4.1 Per-label versus Per-image Evaluation

In per-image metrics each test image contributes equally, and thus they tend to get biased towards performance on frequent labels. A good performance on frequent labels, will automatically translate to the performance of majority images. In contrary, each label contributes equally in per-label metrics, due to which they tend to get affected by performance on rare labels. Rare labels correspond to retrieving few images, and getting them correct leads to high performance scores for that label, thus impacting the per-label average.

It is important to note that the issue of imbalance in label frequencies (also called class-imbalance) in image annotation datasets has attained some attention in the past [19, 51, 53]. Here, we study the relative trade-off between per-label and per-image metrics, with respect to various such factors.

	Method	Per-label F1 score ($F1_L$)				Per-image F1 score ($F1_I$)			
		True	Rare	Freq.	Rand	True	Rare	Freq.	Rand
Corel-5K	Ground	100.00	99.25	99.55	71.59	100.00	82.61	82.61	82.61
	JEC	43.27	52.07	51.74	33.87	53.76	53.80	56.42	54.33
	TagRel	43.34	54.53	53.61	34.53	52.96	53.09	57.01	53.42
	TagProp	40.19	50.59	49.47	32.46	54.00	54.14	56.52	54.14
	2PKNN	49.25	62.27	61.98	38.15	52.01	52.11	55.61	52.51
	SVM	40.90	52.90	52.15	34.72	56.83	56.92	59.08	57.11
ESP Game	Ground	100.00	99.26	99.56	81.50	100.00	88.60	88.60	88.60
	JEC	37.03	47.38	46.81	29.02	44.31	44.36	49.37	44.92
	TagRel	40.41	58.92	58.44	36.10	45.92	45.98	52.40	46.62
	TagProp	42.79	57.99	57.43	35.82	46.80	46.87	52.22	47.25
	2PKNN	43.78	59.22	59.14	36.10	46.50	46.54	52.15	47.02
	SVM	39.73	52.76	51.92	32.99	49.88	49.95	53.69	50.47
IAPR TC-12	Ground	100.00	99.26	99.71	87.58	100.00	92.83	92.83	92.83
	JEC	34.20	41.30	41.11	28.21	48.76	48.88	52.72	49.28
	TagRel	43.59	57.83	58.06	38.33	49.82	49.89	55.97	50.27
	TagProp	45.13	57.70	57.81	38.37	49.78	49.87	55.65	50.32
	2PKNN	45.75	57.90	57.95	38.07	49.55	49.63	54.10	50.04
	SVM	38.45	44.95	44.64	31.10	53.50	53.62	55.91	54.05
NUS-WIDE	Ground	100.00	98.56	99.11	62.33	100.00	79.88	79.90	79.88
	SoftMax*	48.22	68.33	67.97	36.41	62.07	62.08	65.12	62.48
	Sigmoid*	48.85	68.57	68.30	36.49	62.92	62.94	65.36	63.33
	Ranking*	47.82	68.34	67.94	36.25	61.75	61.76	64.82	62.20
	WARP*	48.09	69.44	69.19	36.99	61.98	61.99	65.12	62.43
	LSEP*	48.45	69.58	69.40	37.17	62.73	62.75	65.51	63.16
	JEC	38.94	57.17	56.62	28.81	41.25	56.74	61.01	57.37
	TagRel	47.58	74.31	74.42	39.42	58.49	58.50	65.80	59.07
	TagProp	53.07	73.61	72.96	39.51	60.46	60.48	65.13	60.95
	2PKNN	52.38	68.03	68.64	33.88	53.31	53.32	58.65	54.06
	SVM	49.30	68.95	68.48	36.60	62.31	62.32	64.89	62.72
MS-COCO	Ground	100.00	98.36	99.19	82.9	100.00	85.46	85.46	85.46
	SoftMax*	58.13	72.90	72.73	52.06	64.35	64.41	66.75	64.84
	Sigmoid*	59.98	73.98	73.82	53.41	66.05	66.12	67.92	66.47
	Ranking*	57.99	72.74	72.52	51.96	64.43	64.51	66.49	64.94
	WARP*	57.79	72.09	71.96	51.28	64.01	64.08	66.26	64.52
	LSEP*	59.70	74.23	74.13	53.61	65.77	65.83	67.88	66.22
	JEC	51.05	63.34	62.93	42.21	57.22	57.30	59.80	57.99
	TagRel	58.20	74.77	74.94	52.85	61.55	61.61	67.70	62.09
	TagProp	61.44	73.71	73.68	52.90	64.21	64.27	66.54	64.74
	2PKNN	62.05	72.33	72.22	50.77	61.47	61.54	66.85	62.07
	SVM	60.37	73.11	73.02	52.59	65.28	65.34	66.94	65.72

Table 3.4 Comparing the actual per-label and per-image performance (using *GoogLeNet*) of various label prediction models (deep-learning based models are marked by *) with those obtained by replacing incorrect predictions with rare/frequent/random incorrect labels. (Refer Section 3.4.1.1 for details.)

3.4.1.1 Experiment with upper-bounds of various methods

Recall that as discussed in Section 3.3.2.3, we assign a fixed number of labels to each test image during evaluation. However, since several test images may have either more or less labels in the ground-truth, no method can achieve perfect performance. In order to study the relative trade-off between per-label and per-image metrics, we try to evaluate the best performance achievable by each method. For this, we assume to know what labels are incorrect predictions for each test image using a given annotation method, and then replace these labels by either the most frequently occurring, or the least frequently occurring, or randomly chosen incorrect labels to satisfy the requirement of $3/5$ label assignment. Note that in this analysis, we can not evaluate mAP_L and mAP_I.

In Table 3.4, we show the performance of various methods in terms of F1 scores using GoogLeNet. Here, "True" denotes the actual performance obtained by each method, and "Rare", "Freq.", and "Rand" denote the scores obtained by filling the empty slots with rare, frequent and random incorrect labels respectively. In case of "Ground" (ground-truth), the "True" performance is 100% since here we relax the constraint of assigning exactly $3/5$ labels and evaluate over the ground-truth labels themselves. In one sense, these results can be thought of as upper-bounds achievable using various methods. In case of F1_L, we can observe that the performance of each method improves significantly when we replace the incorrect predictions by either the most rare or the most frequent (incorrect) labels. This is expected because very few labels tend to get highly penalized. Due to this, while their performances drop, that of each of the remaining labels improves, thus significantly improving the overall performance. However, when we randomly assign incorrect labels, the penalty spreads across all the labels and this leads to significant drop in the performance. In contrast, in case of F1_I, we can observe that the performance of each method improves significantly when we replace the incorrect predictions by the most frequent labels. However, the performance of each method generally remains close to its actual ("True") performance when we replace incorrect predictions by either the most rare or randomly chosen (incorrect) labels.

These results show that per-image metrics are biased towards rewarding correct predictions of frequently occurring labels. Moreover, as long as the *number* of incorrect predictions remains the same, the performance will not get seriously affected *irrespective* of what labels are incorrectly assigned. In contrast, per-label metrics are expected to provide comparatively better insights about annotation performance for both rare as well as frequently occurring labels, even though the performance scores corresponding to rare labels might be somewhat noisy in the sense that they are based on only a handful of test images.

3.4.1.2 Experiment with rare / frequent / random label assignments

To further study the bias of per-image metrics towards frequent labels, we assign to all the test images in a dataset (i) the same $3/5$ most rare labels, (ii) the same $3/5$ most frequent labels, and (iii) $3/5$ randomly chosen labels. The fixed set of frequent/rare labels is chosen based on the frequencies of labels in the training subset of a given dataset. While assigning random labels, we pick different

Dataset	Method	Per-label metrics				Per-image metrics		
		P_L	R_L	$F1_L$	N+	P_I	R_I	$F1_I$
Corel-5K	Rare	0.00	1.92	0.00	5	0.24	0.33	0.28
	Frequent	0.34	1.92	0.57	5	17.59	25.50	20.82
	Random	1.16	0.99	1.07	23	1.28	1.68	1.45
ESP Game	Rare	0.00	1.86	0.00	5	0.15	0.13	0.14
	Frequent	0.31	1.86	0.53	5	16.51	18.72	17.55
	Random	1.90	1.80	1.84	96	1.83	1.94	1.88
IAPR TC-12	Rare	0.00	1.72	0.01	5	0.32	0.46	0.37
	Frequent	0.33	1.72	0.55	5	19.10	17.04	18.01
	Random	2.04	1.75	1.88	105	2.05	1.84	1.94
NUS-WIDE	Rare	0.00	3.70	0.00	3	0.04	0.06	0.05
	Frequent	1.06	3.70	1.65	3	28.71	39.37	33.21
	Random	2.95	3.68	3.28	80	2.96	3.71	3.30
MS-COCO	Rare	0.01	3.75	0.02	3	0.33	0.37	0.35
	Frequent	0.93	3.75	1.49	3	24.87	24.29	24.57
	Random	3.48	3.61	3.54	80	3.46	3.59	3.52

Table 3.5 Performance by assigning the three most rare, the three most frequent, and three randomly chosen labels to each test image.

randomly chosen labels for each image. Table 3.5 shows the performance obtained using these label assignment techniques. We can observe that the performance is negligibly low in all the cases, except in the case of per-image metrics with frequent label assignment. Precisely, simply by assigning the same three/five most frequent labels to all the test images, we can achieve quite significant $F1_I$ scores.

From the above results, we arrive at the conclusion that for the image annotation problem, per-label metrics should be preferred over per-image metrics in general. Additionally, this analysis also suggests another issue in the evaluation schemes that have been followed for over a decade in the image annotation domain, that require each test image to be annotated with a pre-defined fixed number of labels rather than doing variable number of label assignments, and thus warrants more discussion in future work.

3.4.2 Dataset Diversity

Here we introduce the notion of measuring diversity in image annotation datasets.

3.4.2.1 Label Diversity

To study this, we look at the list of label-sets(a label-set is the set of labels assigned to an image) in the test data, instead of individual labels (Figure 3.2). We define two measures: percentage unique label-sets and novel label-sets. The former computes what percentage of labels-sets are unique in the

			
animal, grass, mountain, sky	animal, grass, mountain, sky	animal, grass, mountain, sky	animal, grass, mountain, sky

Figure 3.2 Example of images belonging to the same label-set from NUS-WIDE dataset, both training (first two from left) and test (last two from left) images (best viewed in colour)

Dataset	Corel-5K	ESP Game	IAPR TC-12	NUS-WIDE	MS-COCO
Unique	86.8	95.2	95.3	12.3	27.2
Novel	48.9	82.5	73.2	6.7	17.3

Table 3.6 Label diversity in test data in terms of percentage "unique" and "novel" label-sets.

ground-truth of the test data, and the latter computes what percentage of label-sets are novel (i.e., not seen in the training data). Note that while computing the percentage of novel label-sets, we omit the uniqueness criterion; i.e., multiple test images can have the same novel label-set.

$$\text{unique}_{\text{test}} = \frac{|\text{unique label-sets}_{\text{test}}|}{|\text{test images}|} * 100 \tag{3.16}$$

$$\text{novel}_{\text{test}} = \frac{|\text{label-sets}_{\text{test}} - \text{label-sets}_{\text{train}}|}{|\text{test images}|} * 100 \tag{3.17}$$

Table 3.6 shows the values of these two measures for various datasets. We observe that ESP Game and IAPR TC-12 datasets offer maximum diversity in terms of both unique as well as novel label-sets. However, for the NUS-WIDE dataset, we observe that there are only 12.3% of unique label-sets in the test set, and only 6.7% of test images that have novel label-sets in their ground-truth that are not seen in the training set. This indicates that there is a lack of label diversity in the NUS-WIDE dataset. This is because though it has a large number of images, its vocabulary contains only 81 labels with just around 2.40 labels per image, indicating a low degree of *multi-labelness* in this dataset. In table we show the five most common label-sets observed in the NUS-WIDE dataset.

3.4.2.2 Image Diversity

One natural expectation from an image annotation dataset is that its test set should contain compositionally novel images that are not seen in entirety in the training set [10]. To study this phenomenon, we identify compositionally novel as well as compositionally similar images in the test sets of various

datasets and evaluate the performance of different methods on these subsets. To do so, we bin the images in the test set of a given dataset based on their feature similarity with the training images. For each test image, we compute its Euclidean distance with every training image, and then take the mean of the distance with the 50 closest images. This mean distance measures the degree of visual overlap of each test image with the training images, with larger mean distance denoting less overlap and vice-versa. Using this, we pick the 20% most overlapping and 20% least overlapping images from the test set of each dataset.

Performances of various methods on these two sets are shown in Table 3.7 and 3.8. From these results, we can make following observations: (i) The performances of all the methods significantly improve on the "20% most" set, and significantly reduce on the "20% least" set compared to that on the full test set. While this is the case in all the datasets, the degree of relative variations in performance is minimum in the ESP Game dataset. This indicates that though all the datasets lack compositional diversity in their images, the ESP Game dataset seems to suffer the least from this. However, since it is an order of magnitude smaller than the NUS-WIDE dataset, this also motivates to create new large-scale datasets that would contain compositionally novel images in their test sets. (ii) The reduction in performance corresponding to per-image metrics on the "20% least" set is much less compared to per-label metrics. This again demonstrates that per-label metrics may be more informative than per-image metrics for evaluation in the image annotation task. Figure 3.3 some example images from the NUS-WIDE dataset along with their ground-truth and predicted labels. Here, we observe that for the images that are more similar to training images, the number of predicted labels that match with the ground-truth labels is higher than those for the images that are less similar.

3.5 Summary

While it is close to two decades since the problem of image annotation has been studied [41], improving upon the quantitative results has always remained the key focus. In the chapter, through detailed experimental analyses on five popular image annotation datasets, we have made an attempt to highlight some of the core yet mostly overlooked issues related to dataset construction and popularly used evaluation metrics in this domain. Our two key observations are: (i) among all the datasets, the ESP Game dataset offers the maximum label and image diversity, and and is least influenced by the impact of frequent labels on the performance, and (ii) per-label metrics should be preferred over per-image metrics for comparing image annotation techniques in general. Based on these observations, we would like to emphasize the importance of taking careful considerations with respect to these aspects when developing new datasets and techniques for the image annotation task in the future.

	Method	Per-label metrics				Per-image metrics			
		P_L	R_L	$F1_L$	mAP_L	P_I	R_I	$F1_I$	mAP_I
Corel-5K	JEC	67.42	74.61	70.83	68.47	62.20	84.42	71.63	71.53
	TagRel	68.02	78.20	72.76	75.58	63.20	85.50	72.68	83.58
	TagProp	70.23	78.25	74.02	76.27	64.60	87.50	74.33	85.42
	2PKNN	71.27	83.22	76.78	83.46	65.80	89.33	75.78	87.06
	SVM	66.32	81.22	73.02	83.81	65.00	88.42	74.92	86.30
ESP Game	JEC	59.77	51.88	55.55	45.97	55.92	64.79	60.03	51.78
	TagRel	55.46	63.50	59.21	64.05	53.91	60.96	57.22	64.49
	TagProp	61.04	64.74	62.84	65.84	56.83	64.96	60.63	69.18
	2PKNN	62.16	64.05	63.09	64.52	57.84	66.46	61.85	68.94
	SVM	56.90	61.44	59.08	64.93	59.47	68.02	63.46	71.12
IAPR TC-12	JEC	55.24	46.62	50.56	43.10	64.27	65.57	64.92	56.84
	TagRel	66.27	65.45	65.85	71.11	65.09	66.19	65.63	73.57
	TagProp	68.75	67.35	68.04	76.03	65.14	66.12	65.62	76.01
	2PKNN	68.22	66.27	67.23	74.18	66.26	67.02	66.63	77.09
	SVM	63.57	56.74	59.96	76.24	68.50	69.55	69.02	79.78
NUS-WIDE	SoftMax*	59.80	65.98	62.74	63.53	61.75	81.68	70.33	87.64
	Sigmoid*	60.18	65.76	62.84	69.02	62.48	82.33	71.05	88.93
	Ranking*	57.75	66.30	61.73	62.91	61.79	81.54	70.30	87.64
	WARP*	58.08	67.26	62.34	62.94	61.94	81.69	70.46	87.48
	LSEP*	59.23	66.77	62.77	65.60	62.51	82.38	71.08	88.86
	JEC	56.54	49.29	52.67	38.02	56.55	75.62	64.71	71.22
	TagRel	58.44	70.99	64.12	66.16	59.69	79.20	68.08	82.76
	TagProp	65.64	71.83	68.59	68.93	60.72	80.44	69.20	85.18
	2PKNN	65.26	69.97	67.53	66.94	58.74	77.97	67.00	81.63
	SVM	63.90	67.79	65.79	66.67	61.57	81.16	70.02	86.95
MS-COCO	SoftMax*	60.62	65.89	63.15	64.98	57.36	89.61	69.95	93.96
	Sigmoid*	59.34	66.92	62.90	69.64	57.70	89.87	70.28	94.47
	Ranking*	56.62	66.74	61.26	65.55	57.19	89.34	69.74	93.97
	WARP*	58.15	65.86	61.77	65.30	56.89	89.01	69.41	93.71
	LSEP*	60.40	67.08	63.57	66.50	57.71	89.93	70.30	94.51
	JEC	60.91	56.91	58.84	44.74	52.23	83.64	64.31	78.14
	TagRel	56.44	65.63	60.69	65.83	53.45	85.07	65.65	87.68
	TagProp	69.09	63.50	66.18	66.14	56.27	88.23	68.72	92.13
	2PKNN	66.74	65.39	66.06	66.48	54.40	86.24	66.71	91.41
	SVM	69.09	63.80	66.34	65.01	56.20	88.03	68.60	91.99

Table 3.7 Performance comparison (using *GoogLeNet*) of various label prediction models (deep-learning based models are marked by *) over the 20% most overlapping test subsets of various datasets (refer Section 3.4.2.2 for details.)

	Method	Per-label metrics				Per-image metrics			
		P_L	R_L	$F1_L$	mAP_L	P_I	R_I	$F1_I$	mAP_I
Corel-5K	JEC	27.32	30.82	28.96	27.84	32.40	48.92	38.98	37.74
	TagRel	25.74	32.03	28.54	31.34	31.60	47.67	38.01	45.44
	TagProp	25.02	31.41	27.85	34.74	33.60	51.92	40.80	44.67
	2PKNN	26.51	34.97	30.16	51.54	30.40	47.25	37.00	38.99
	SVM	24.88	37.27	29.84	49.67	36.80	56.17	44.47	53.21
ESP Game	JEC	17.97	14.61	16.12	12.15	27.34	31.36	29.21	23.36
	TagRel	21.14	22.18	21.65	22.18	31.22	35.77	33.34	34.51
	TagProp	20.57	19.87	20.22	22.09	31.51	35.99	33.60	34.97
	2PKNN	21.25	20.60	20.92	24.14	29.93	33.64	31.68	34.12
	SVM	18.60	15.69	17.02	23.39	33.67	38.13	35.76	39.18
IAPR TC-12	JEC	16.21	11.65	13.56	10.70	33.28	31.46	32.34	22.77
	TagRel	24.24	22.16	23.16	22.80	36.03	34.48	35.24	34.67
	TagProp	23.45	19.88	21.52	27.43	36.95	36.40	36.67	38.79
	2PKNN	22.06	16.52	18.89	29.33	35.06	33.60	34.32	38.50
	SVM	15.99	11.83	13.60	28.93	39.64	38.26	38.94	43.46
NUS-WIDE	SoftMax*	23.32	25.14	24.19	18.86	42.30	67.14	51.90	67.45
	Sigmoid*	23.02	26.17	24.49	22.49	42.86	67.72	52.49	68.26
	Ranking*	20.40	24.06	22.08	16.86	41.31	64.89	50.48	65.15
	WARP*	20.94	25.80	23.11	17.74	41.59	65.53	50.89	65.76
	LSEP*	21.95	27.31	24.34	19.79	42.68	67.53	52.30	67.88
	JEC	16.91	24.06	19.86	10.10	37.29	58.77	45.63	48.18
	TagRel	17.53	35.18	23.40	20.07	38.12	60.38	46.74	58.92
	TagProp	25.17	27.97	26.50	22.78	41.72	66.01	51.12	65.02
	2PKNN	29.08	19.87	23.61	21.02	34.02	53.66	41.64	50.03
	SVM	21.41	24.16	22.70	20.90	42.53	66.96	52.02	67.17
MS-COCO	SoftMax*	39.52	35.72	37.52	32.87	48.57	54.17	51.21	62.78
	Sigmoid*	42.50	38.53	40.42	39.44	51.17	56.38	53.65	65.58
	Ranking*	39.35	33.93	36.44	32.67	48.22	52.49	50.26	61.33
	WARP*	38.69	33.19	35.73	32.70	47.75	51.70	49.65	60.79
	LSEP*	42.17	39.32	40.69	37.04	50.61	56.03	53.18	64.90
	JEC	34.46	27.82	30.79	17.23	43.14	47.25	45.13	45.01
	TagRel	36.95	41.32	39.01	37.27	46.90	52.40	49.49	56.81
	TagProp	42.77	38.42	40.48	37.18	49.69	54.76	52.10	61.56
	2PKNN	49.74	34.05	40.43	39.15	46.03	50.42	48.13	58.22
	SVM	42.37	37.32	39.69	39.13	51.00	55.83	53.31	63.93

Table 3.8 Performance comparison (using *GoogLeNet*) of various label prediction models (deep-learning based models are marked by *) over the 20% least overlapping test subsets of various datasets (refer Section 3.4.2.2 for details.)

	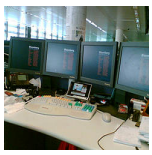	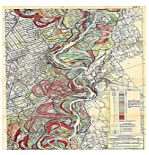		
GT: {animal, road, zebra}	**GT:** {computer}	**GT:** {map}	**GT:** {person, soccer}	**GT:** {railroad, sky}
TagProp: {animal, sky, zebra}	**TagProp:** {computer, sky, water}	**TagProp:** {map, sky, water}	**TagProp:** {person, soccer, sports}	**TagProp:** {clouds, railroad, sky}
Sigmoid: {animal, grass, zebra}	**Sigmoid:** {computer, person, window}	**Sigmoid:** {map, tree, window}	**Sigmoid:** {grass, person, soccer}	**Sigmoid:** {plants, railroad, sky}
				
GT: {nighttime, sky}	**GT:** {window}	**GT:** {toy}	**GT:** {road, sand}	**GT:** {leaf}
TagProp: {person, sky, wedding}	**TagProp:** {reflection, sky, water}	**TagProp:** {plants, road, sky}	**TagProp:** {flags, sand, sky}	**TagProp:** {sky, tree, water}
Sigmoid: {book, nighttime, sky}	**Sigmoid:** {reflection, water, window}	**Sigmoid:** {animal, clouds, military}	**Sigmoid:** {ocean, sand, snow}	**Sigmoid:** {leaf, sky, water}

Figure 3.3 Examples from the "most" overlapping images (top) and the "least" overlapping images (bottom) from the NUS-WIDE dataset. For each image, its ground-truth labels (GT) and the labels predicted using TagProp and Sigmoid methods are shown. The labels in blue are the ones that match with the ground-truth labels (best viewed in colour).

Chapter 4

Revisiting CNN-RNN

4.1 Introduction

Real world images can be associated with multiple labels corresponding to different visual concepts present in that image. Such labels share rich semantic relationships among them (e.g., "sky" is related to "cloud"). Understanding of the inter-label relationship, can in turn improve the understanding of the image. In a particular line of work in image annotation, different techniques have been proposed to model these label dependencies, such as Cross Correlation Analysis based methods [20], Structured Inference Neural Network [23], Recurrent Neural Networks (RNNs) [25, 55] and Spatial Regularization Network [31]. Among these, with the motivation to create a deep unified framework, CNN-RNN-style models have received increasing attention in the recent years [7, 25, 36, 55], particularly due to RNN's capacity to capture higher-order label relationships while keeping the computational complexity tractable.

Since RNN training requires sequential input, existing works imposed a pre-defined ordering on image labels based on the frequencies of labels observed in the training data (e.g. rare-to-frequent or frequent-to-rare). Jin and Nakayama [25] showed that the order of labels in the training phase has an impact on the annotation performance, and found the rare-to-frequent order to work the best. However, such ordering introduces a hard constraint on the model; e.g. if we impose rare-to-frequent label ordering, the model would be forced to identify the rare labels first, which is difficult since rare labels often represent small objects. Since future labels are predicted based on previously predicted labels, any error in the first step would increase the likelihood of errors in subsequent predictions. Similarly if we impose frequent-to-rare label ordering, it is still hard for the model as it has to make several correct predictions before predicting the correct rare label. A frequency based pre-defined label ordering may not reflect the true label dependencies and defining such an order is dependent on dataset-specific statistics.

In the chapter, we attempt to address the above limitations by proposing to learn label dependencies by learning multiple label paths instead of a fixed pre-defined order. Specifically, while training at any given time-step, we enforce the model to predict all the correct labels instead of a particular label as in a pre-defined order, except for the label it selected as the most probable one in the previous time-step. During testing, we obtain the final label prediction by max-pooling the prediction scores across

all time-steps. In this way, the best prediction of a label can be obtained from its individual prediction path. Additionally, allowing the model to learn and predict from multiple label paths also provides the advantage that in reality there may not be just a single correct order that reflects the proper label dependency.

We evaluate our proposed approach on two large-scale multi-label image annotation datasets (NUS-WIDE and MS-COCO), and demonstrate that it performs better than the state-of-the-art CNN-RNN models. Further, we also present insights into the utility and scope of CNN-RNN-style models in the context of image annotation. The outline of the chapter is as follows: In section 4.2, we present a brief overview of some related work in image annotation, in section 4.3 we describe our proposed approach, in section 4.4 we present our experiments, results and analysis and conclude the chapter in section 4.5.

4.2 Related Work

Deep CNNs, which have achieved great success on single-label image classification task, can provide a strong baseline for multi-label classification too. Gong *et al*. [17] evaluated several loss functions on a pre-trained deep CNN, and found the WARP loss [57] to perform the best. Recently, Li *et al*. [30] introduced the LSEP loss, which is a differentiable pairwise ranking loss. In parallel, advanced nearest neighbour based methods such as TagProp [19] and 2PKNN [51] have also been shown to perform well on multi-label annotation using CNN features [50, 53]. They are based on the fact that images similar to each other in the visual space are likely to share labels. While Tagprop is a distance based parametric nearest-neighbour model, 2PKNN tries to improve the neighbourhood with the goal of achieving a uniform label distribution to counter the class-imbalance problem.

There have also been attempts to explicitly model label dependencies. Hu *et al*. [23] proposed a deep structured inference neural network that uses different concept layers analogous to the WordNet [39] hierarchy, and captures relationships between the layers. Uricchio *et al*. [50] applied statistical approach KCCA to embed image and labels into a common semantic space. Andrea *et al*. [15] proposed DeviSE, which can jointly embed and train visual and word embeddings through a neural network. Feng *et al*. [31] proposed a Spatial Regularization Network (SRN) that generates attention maps for all labels and captures the underlying relations between them via learnable convolution filters.

Recent approaches based on CNN-RNN encoder-decoder framework have also been popularly used to model label correlations, and have been motivated by image captioning approaches [54]. These approaches treat multi-label prediction as a sequence prediction problem, where the RNN model is trained to predict the labels for a given image in a sequential manner, analogous to predicting a caption. As discussed in Section 4.1, such approaches require a pre-defined ordering among the labels at the time of training, and thus constrain the model to predict the labels in that order. In another recent work, Liu *et al*. [36] demonstrated the utility of giving CNN label probability vector as an input to RNN rather than giving visual features from the penultimate layer. Our work is closely related to that of Shang-Fu *et al*. [7], who made a similar attempt to propose an order-free RNN model for multi-label prediction.

However, unlike their approach, our model tries to implicitly learn the dependencies between the input and output labels.

Another line of work uses side information, such as user tags/groups from image metadata, or external information such as WordNet hierarchy. Side information can provide rich contextual information and thus leads to better understanding of visual data. Johnson *et al*. [26] use a non-parametric approach to find image neighbours according to the metadata, and then aggregate visual information of an image and its neighbours with a deep network to improve label prediction. Recently, a few works such as [56] also make use of a pre-trained region proposal network to generate confident object proposals.

4.3 Approach

Let there be N training images $\{I_1, \ldots, I_N\}$, such that each image I has its ground-truth binary label vector $y = (y^1, y^2, \ldots, y^C)$, where y^c ($\forall c \in \{1, 2, \ldots, C\}$) is 1 if the image has the label c in its ground-truth and 0 otherwise, $\hat{y}$ is the corresponding predicted label score vector, and C is the total number of labels.

We use a CNN-RNN framework which consists of two components: (i) a deep CNN that extracts semantic representation from images, and (ii) an RNN that models image-label relationships and label-label dependencies. For a given image I, the CNN component predicts its feature vector representation, and the RNN component predicts a label score vector $\hat{y}_t$ at each time-step t in a sequential manner, and generates a prediction path $\pi = (a_1, a_2, ..., a_T)$, where a_t denotes the label corresponding to the maximum score from $\hat{y}_t$ at time t, and T is the total number of time steps. While existing methods train the RNN model using a specific prediction path based on a pre-defined label order (e.g., rare-frequent), we propose an order-free approach to train the RNN model that allows it to learn multiple feasible prediction paths based on the given ground-truth set of labels. During inference, we fuse scores across the prediction paths to obtain the final label scores. Below, first we give an overview of the base CNN-RNN model, and then present our order-free learning approach. Figure 4.1 gives an overview of our framework.

4.3.1 CNN-RNN

We employ the current state-of-the-art semantically regularised CNN-RNN (S-CNN-RNN) [36] as our base CNN-RNN model. This model introduced a semantically regularised embedding layer as an interface between the CNN and RNN to decouple the learning problems of the CNN and RNN, thus allowing a more efficient joint training. The CNN takes as an input an image I and produces a probabilistic estimate of the labels $\hat{s} \in \mathbb{R}^{C \times 1}$.

In other words, it uses the label prediction layer of CNN instead of the feature vector from the penultimate fully-connected layer, with the CNN model being trained using the standard cross entropy loss. The RNN decoder takes $\hat{s}$ as an input and then generates a sequence of labels $\pi = (a_1, a_2, ..., a_{n_I})$, where a_i is the label predicted at the t^{th} time-step, and n_I is the total number of labels predicted.

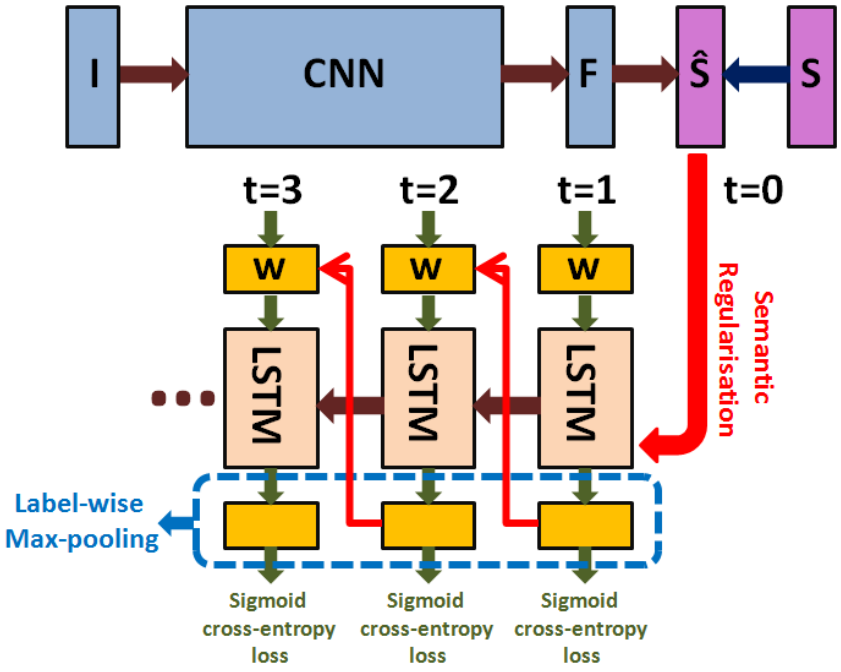

Figure 4.1 Overview of the proposed approach. In the base we have a Semantic regularised CNN-RNN model, where RNN initial state is set by predictions from the CNN. The model is trained at every time step with a cross entropy loss over all true labels except the label from previous time step given to it as input. Final predictions are obtained by max-pooling across time.

Analogous to the contemporary approaches, it uses the Long Short-Term Memory (LSTM) [22] network as the RNN decoder, which controls message passing between time-steps with specialized gates. A forward pass at time t with input x_t is computed as follows:

$$
\begin{aligned}
h_0 &= W_{h_0,\hat{s}} \cdot \hat{s} + b_{h_0} \\
c_0 &= W_{c_0,\hat{s}} \cdot \hat{s} + b_{c_0} \\
i_t &= \sigma(W_{i,h} \cdot h_{t-1} + W_{i,c} \cdot c_{t-1} + W_{i,x} \cdot x_t + b_i) \\
f_t &= \sigma(W_{f,h} \cdot h_{t-1} + W_{f,c} \cdot c_{t-1} + W_{f,x} \cdot x_t + b_f) \\
o_t &= \sigma(W_{o,h} \cdot h_{t-1} + W_{o,c} \cdot c_{t-1} + W_{o,x} \cdot x_t + b_o) \\
\\
g_t &= \sigma(W_{g,h} \cdot h_{t-1} + W_{g,c} \cdot c_{t-1} + W_{g,x} \cdot x_t + b_g) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot \delta(c_t)
\end{aligned}
$$

where c_t and h_t are the model's cell and hidden states respectively at time t; i_t, f_t and o_t are the activations of input gate, forget gate and output gate respectively; $W_{\cdot,\hat{s}}$ is the initial weight to set initial cell state and hidden state; $W_{\cdot,h}$, $W_{\cdot,c}$ are the recurrent weights, $W_{\cdot,x}$ is the input weight, and $b_{\cdot}$ are the

biases; $\sigma(\cdot)$ is the sigmoid function, and δ is the output activation function. At time step t, the model uses its last prediction a_{t-1} as input, and computes a distribution over the possible outputs:

$$x_t = E \cdot a_{t-1} \tag{4.1}$$

$$h_t = LSTM(x_t, h_{t-1}, c_{t-1}) \tag{4.2}$$

$$\hat{y}_t = W \cdot h_t + b \tag{4.3}$$

where E is the label embedding matrix, W and b are the weight and bias of the output layer, a_{t-1} is the *one-hot encoding* of the last prediction, and $LSTM(\cdot)$ is a forward step of the unit. The output vector $\hat{y}_t$ defines the output scores at t, from which the next label a_t is sampled.

4.3.2 Proposed Training

We now introduce our approach to train the RNN *without* a pre-defined label order. Again, let there be N training images and C labels. Each image I has its ground-truth binary label vector $y = (y^1, y^2, \ldots, y^C)$. Also, let y_t denote the ground-truth label vector at time t and $\hat{y}_t$ be the corresponding predicted label score vector. The RNN is trained using the standard sigmoid cross entropy between y_t and $\hat{y}_t$ at each time-step.

Recall that in CNN-RNN based image captioning approaches, y is a *sequence* of words $y = (w_1, w_2, ..., w_T)$, thus y_t is the *one-hot encoding* of the actual word at the specified time index. In multi-label image annotation, there is no sequence of words but a *set* of labels. Thus, previous works define a frequency based sequence and y_t would be the *one-hot encoding* of the label as defined in this assumed sequence. , let us assume that an image has the labels {*sky, clouds, person*} in its ground-truth, and a rare-to-frequent frequency based ordering of these labels is {*clouds, person, sky*}. Then the model is forced to learn to predict *clouds* in the first time-step, *person* in the next time-step based on *clouds* predicted previously, then *sky*, and finally an end-of-sequence token. Here, y_1 is a label vector with 1 for *clouds* and 0 for all other labels, y_2 has 1 for *person* and 0 for all other labels, and so on.

In practice, since the ground-truth y_t at time-step t is unknown, one could assume that y_t at each time-step is the original label vector y. This would enforce the model to predict all ground-truth labels instead of one particular ground-truth label at each time-step; *i.e.*, from the previous example, the model would be forced to predict all the three labels {*sky, clouds, person*} at every time-step based on the most confident label predicted in previous time-step. However, this poses the problem that if the most confident label predicted at time-step t is l, the model may end-up learning a dependency from l to l in the next time-step along with the dependency from l to other labels. In other words, there is a high chance that a label which is easiest to predict would be the most confident prediction by RNN at every time-step, and thus the same label would then be repeatedly chosen for feedback to the RNN.

Shang-Fu *et al*. [7] tried to alleviate this problem by explicitly sampling a different label at every time-step to give as feedback to the RNN (discussed in detail in Section 4.3.5)). Unlike them, here we let the model implicitly learn to predict a different label, by proposing a greedy approach. If l is the most

confident prediction at time-step time t, we mask out l in the next time step; *i.e.*, in the next time-step, we treat l as a negative label rather than positive and learn a dependency from l to all other labels except itself. We explain this mathematically below.

Let l_t be the hard predicted label for the image I at time-step t (*i.e.*, the most confident label with the highest prediction score):

$$l_t = \underset{c \in \{1,2,...,C\}}{\arg\max} \quad \hat{y}_t^c \tag{4.4}$$

Let a_{t-1} be the *one-hot encoding* of the corresponding hard predicted label l_{t-1}. We define a label mask $\tilde{a}_t$ at time t as:

$$\tilde{a}_t = \neg a_{t-1} \tag{4.5}$$

In other words, this label mask is a negation of the *one-hot encoding* of the hard predicted label of previous time step. The mask contains a 0 corresponding to the previously selected label index, and 1 for the rest. Using this, we define a modified ground-truth label vector at time-step t as:

$$y_t = \tilde{a}_t \odot y \tag{4.6}$$

where $\odot$ represents element-wise multiplication. At time-step $t = 0$, $\tilde{a}_0$ will be a vector with all ones.

We can now train the model using the standard sigmoid cross-entropy loss at each time-step t, between the modified ground-label vector and the predicted label scores at that particular time-step:

$$\mathcal{L}_t = y_t \cdot log(\sigma(\hat{y}_t)) + (1 - y_t) \cdot log(1 - \sigma(\hat{y}_t)) \tag{4.7}$$

At time-step t, the above loss is computed for each training image (in the given batch), and their summation gives the total loss at that time-step. These loss values over all the time-steps are finally added to get the total loss.

4.3.3 Learning multiple label prediction paths

The proposed training approach can inherently learn multiple prediction paths instead of a fixed prediction path. Precisely, in the first time-step, the LSTM is trained to predict all the true labels by the loss $\mathcal{L}_1$, analogous to a CNN that can be trained to predict all the true labels by a single look at an image. As the LSTM strives to predict all the labels, the most confident label predicted by it is given as the feedback for the next time-step. In the second time-step, the model is trained to predict all the true labels except the most confident label predicted by it in the previous time- step, and this continues for a fixed number of time-steps.

Let us try to understand through an example. Let $\{l_1, l_2, l_3, l_4, l_5\}$ be the complete set of labels, and let $\{l_1, l_3, l_5\}$ be the true labels for a given image. At $t = 1$, the model is forced to predict all the true labels. Suppose l_1 is selected by the model at time t_1 as the most confident label, then it is required to learn the dependency from l_1 to both l_3 and l_5 by the loss $\mathcal{L}_2$, which now penalizes l_1 by setting it as a negative label for time-step $t = 2$. This gives a chance to labels $\{l_3, l_5\}$ to be predicted at $t = 2$ on the basis of both the image *and* the confidence that l_1 is present in the image. Now, let us say that the label l_5

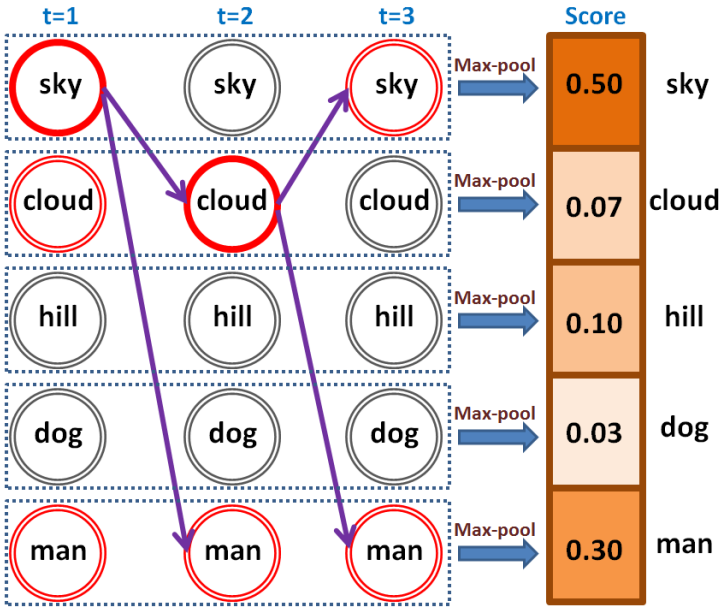

Figure 4.2 An example of multiple label paths in the proposed approach. Let {*sky, cloud, hill, dog, man*} be the complete label set (vocabulary), and let {*sky, cloud, man*} be the ground-truth set of labels for a given image. **During training**, at $t = 1$, the model is trained to predict all the positive labels correctly. Suppose it selects *sky* as the most confident label. Then at $t = 2$, the model is trained to learn dependencies from *sky* to both *cloud* and *man*, while keeping *sky* as a negative label. Suppose the model now selects *cloud* as the label with the highest confidence. Then at $t = 3$, the model will be trained to predict *sky* and *man*. In this way, the model learns multiple dependency paths *sky* $\rightarrow$ *cloud* $\rightarrow$ *sky* and *sky* $\rightarrow$ *cloud* $\rightarrow$ *man*. **During testing**, given an input image, the most confident label at a each time-step is fed as input to the LSTM, and this is repeated for a fixed number of time-steps. At the end, the predictions scores across all the time-steps are label-wise max-pooled, and the labels above a threshold are assigned.

is the most confident label predicted at time-step $t = 2$, then the loss $\mathcal{L}_3$ trains the model to try predict $\{l_1, l_3\}$ at time-step $t = 3$. The model thus learns the dependency paths $\{l_1, l_5, l_3\}$ and $\{l_1, l_5, l_1\}$. In other words, it learns not only to go back to l_1 from l_5, but also learns to predict l_3 based on the confidence that the image already has $\{l_1, l_5\}$. In this way, the labels that are hard to predict initially are learned to get predicted at later time-steps based on their dependencies on other labels present in the ground-truth.

The proposed training approach is a greedy approach that is self-guided by the training procedure itself. If the model is most confident at predicting l_1 at the first time-step, we train it to predict $\{l_3, l_5\}$ in the next time-step. If the model had predicted l_3 first, the training at the next time-step would have been for $\{l_1, l_5\}$. In case a model predicts an incorrect label, say l_2, as the most confident prediction, it will be given as the feedback to the LSTM for the next time-step. However, since we penalize the negative labels in every time step, the model learns to try not predict them.

By self-learning multiple label prediction/dependency paths, the model inherently tries to learn label correlations. , if l_3 is chosen immediately after l_1 by the model, there is strong likelihood that $\{l_1, l_3\}$ co-occur strongly in practice, and it makes more sense to predict l_3 when l_1 is present. In contrast, in a pre-defined order based training, the label dependencies are learned in some specified order, which may not be an appropriate order in practice. Also, there may not be a single specific order that reflects the label dependencies in an image. Since the proposed approach can learn multiple label dependency paths, it can mitigate both these issues. Figure 4.2 illustrates this idea.

4.3.4 Label Prediction

Once the model is trained, for a given test image, the LSTM network is iterated for T time-steps, resulting in T prediction score vectors $(\hat{y}_1, \hat{y}_2, ..., \hat{y}_T)$, where each $\hat{y}_t = (\hat{y}_t^1, \hat{y}_t^2, \ldots, \hat{y}_t^C)$ denotes the score vector over the C labels at times-step t. We employ label-wise max-pooling to fuse the scores across all the time-steps into the final result $\hat{y} = (\hat{y}^1, \hat{y}^2, ..., \hat{y}^C)$, where:

$$\hat{y}^c = max(\hat{y}_1^c, \hat{y}_2^c, ..., \hat{y}_T^c), \forall c = 1, \ldots, C. \tag{4.8}$$

The final predicted label probability distribution $\hat{p}$ is:

$$\hat{p}_i = \sigma(\hat{y}) \tag{4.9}$$

We threshold the probability scores by $\alpha = 0.5$ for final assignment; *i.e.*, if the probability score of a label for an image is greater than 0.5, we assign that label to the image.

It is worth noting that unlike pre-defined label order based training, our model does not predict an "end-of-sequence" token. We carry out the LSTM iterations for a fixed number of time-steps 'T' determined experimentally. Also, during prediction, since we do not require to have labels in a specific order, we max-pool through time to get the best prediction score for each label. This also helps in obtaining a probability distribution over all the labels.

4.3.5 Comparison with Order-Free RNN

Our work is closely related to that of Shang-Fu *et al*. [7], who proposed an Order-Free RNN framework for image annotation. They train the model to predict *all* the true labels at *each* time step. To achieve order-free learning and prediction, their model uses the concept of a "candidate label pool" which would initially contain all the labels. At each time-step, the most confident label l_t for feedback to the RNN would be selected from the candidate pool, and the pool would be updated by removing l_t from it. In our approach, we train our model to predict all but one true labels at every time-step (except the first time-step), and penalize on the label predicted in previous time-step as explained in Section 4.3.2. We also show a simple inference technique based on max-pooling the scores across all time steps.

Dataset→	NUS-WIDE						MS-COCO					
Metric→	Per-label			Per-image			Per-label			Per-image		
Method↓	P_L	R_L	$F1_L$	P_I	R_I	$F1_I$	P_L	R_L	$F1_L$	P_I	R_I	$F1_I$
CNN@Top-K	47.66	55.95	48.26	55.66	78.20	59.67	64.59	62.59	61.65	64.07	78.01	64.96
CNN@0.5	68.82	49.01	55.35	80.82	70.58	69.85	81.97	61.97	69.04	87.40	73.96	76.5
CNN-RNN [55]	40.50	30.40	34.70	49.90	61.70	55.20	66.00	55.60	60.40	69.20	66.40	67.80
RIA [25]	52.90	43.60	47.80	68.90	66.70	67.80	64.30	54.10	58.70	74.20	64.50	60.00
Order-Free [7]	59.40	50.70	54.70	69.00	71.40	70.20	71.60	54.80	62.10	74.20	62.20	67.7
S-CNN-RNN [36]	60.35	54.40	54.22	73.22	75.34	71.25	78.89	62.92	68.19	83.03	57.91	64.34
Proposed	60.85	54.43	55.32	76.50	73.06	70.62	77.09	64.32	69.63	84.90	75.83	77.13

Table 4.1 Results evaluated by per-label and per-image metrics on the two datasets.

4.4 Experiments

Now we present the experimental analysis of the proposed approach and comparison with competing methods.

4.4.1 Datasets

We experiment using two popular and large-scale image annotation datasets: NUS-WIDE [47] and MS-COCO [35]. The NUS-WIDE dataset contains $269,648$ images downloaded from Flickr. Its vocabulary contains 81 labels, and each image is annotated with up to 3 labels. On an average, there are 2.40 labels per image. Following the earlier papers, we discard the images without any label. This leaves us with $209,347$ images, that we split into ~ 125K images for training and ~ 80K for testing by adopting the split originally provided by the authors of this dataset.

The second dataset, MS-COCO, is popularly used for various object recognition tasks in the context of natural scene understanding. The training set has $82,783$ images, and the objects are categorized into 80 classes, with about 2.9 object labels per image. Since the ground-truth labels of the test set are not available, we consider the validation set of $40,504$ images to evaluate the results, as followed in the earlier papers.

4.4.2 Evaluation Metrics

To analyze and compare prediction results, we consider both per-label as well as per-image evaluation metrics. We report mean precision, mean recall and mean F1. For each label (image), the precision is defined as $p(\hat{y}, y) = |y \cap \hat{y}|/|\hat{y}|$, and recall is defined as $r(\hat{y}, y) = |y \cap \hat{y}|/|y|$, where y and $\hat{y}$ are the set of ground-truth and predicted labels (images), and $|\cdot|$ is the cardinality of a set. For each label (image), F1 score is the harmonic mean of the per-label (per-image) precision and per-label (pre-image) recall metrics; *i.e.*, $F1 = 2 \times P \times R/(P + R)$, where P and R are the precision and recall values respectively. When y denotes the set of images that have a particular label in their ground-truth and

$\hat{y}$ denotes the set of images to which that label is assigned after prediction, these correspond to mean per-label metrics obtained after averaging over all the labels (denoted by P_L, R_L and $F1_L$). When y denotes the set of labels present in the ground-truth of an image and $\hat{y}$ denotes the labels assigned to that image after prediction, these correspond to mean per-image metrics obtained after averaging over all the images (denoted by P_I, R_I and $F1_I$).

4.4.3 Implementation Details

We use ResNet-101 [21] as the CNN model, which is pre-trained on the ILSVRC12 1000-class classification dataset [46] and fine-tuned for both NUS-WIDE and MS-COCO datasets separately. For fair comparison with the rare-to-frequent Semantic Regularisation method [36], we have implemented their method using the same CNN model. For our LSTM network, we use 512 cells with the $tanh$ activation function and 256-dimensional label-embedding. We train the model using a batch size of 32 with RMSProp Optimiser and learning-rate of $1e-4$ for 50 epochs. Also, we iterate over LSTM for $T = 5$ time-steps.

4.4.4 Compared Methods

We compare with two CNN baselines and various existing CNN-RNN-style image annotation models: **(1) ResNet-101@Top-K [21]**: ResNet-101 model fine-tuned with the cross-entropy loss and top-3 predictions are taken from the classification scores. This model has been used as the baseline by all the above mentioned CNN-RNN style methods for image annotation. **(2) ResNet-101@α [21]**: ResNet-101 model fine-tuned with the cross-entropy loss and probability scores are obtained on which a threshold of $\alpha = 0.5$ is applied to get the label prediction. **(3) CNN-RNN [55]**: A CNN-RNN model which performs joint embedding of the CNN penultimate fully-connected layer features and RNN outputs, and does training using rare-to-frequent label order. **(4) RIA [25]**: A CNN-RNN model, in which the CNN penultimate-layer features set the LSTM hidden state. This paper experimented with various frequency based label orders, and rare-to-frequent label order was reported to perform the best. **(5) Order-Free RNN [7]**: A CNN-RNN model that does not require any pre-defined label order for training. **(6) S-CNN-RNN [36]**: A CNN-RNN model where the state of the RNN is set with the CNN output probabilities instead of the penultimate fully-connected layer features, and training is done using rare-to frequent label order. For fair comparison, we have implemented this model with ResNet-101 as the CNN component.

4.4.5 Results

Table 4.1 summarizes the performance of various methods on the two datasets. From the results, we observe that our model performs better than the Rare-to-Frequent S-CNN-RNN model on both the datasets in terms of the $F1_L$ score. Precisely, it improves by 1.10% and 1.44% absolute on the NUS-WIDE and MS-COCO datasets respectively. In terms of $F1_I$ score, our model is comparable to the

NUS-WIDE		MS-COCO	
S-CNN-RNN [36]	Proposed	S-CNN-RNN [36]	Proposed
68.06	84.05	81.44	93.49

Table 4.2 Comparison between S-CNN-RNN [36] and the proposed approach based on top-1 accuracy of the predicted label (averaged over all the images) on the two datasets.

S-CNN-RNN model on the NUS-WIDE dataset, and significantly outperforms it (by 12.79% absolute) on the MS-COCO dataset. From the results, we also observe that our while model performs significantly better than the ResNet-101@Top-K, which has been used as the baseline by all the previous papers on CNN-RNN-style image annotation, its performance is comparable to ResNet-101@α. It is worth noting that while ResNet-101@α performs well in terms of the precision metric, our model performs well in terms of the recall metric. This demonstrates the capability of our model in learning label dependencies, which in turn improves the recall of true labels.

4.4.6 Analysis

We provide further analysis on the performance of the proposed model and that of the competing models; *i.e.*, S-CNN-RNN and CNN.

4.4.6.1 Comparison with S-CNN-RNN

As discussed before, earlier CNN-RNN-syle image annotation approaches [25, 36, 55] have advocated the use of rare-to-frequent label ordering at the time of training. Here, we analyze the performance of our model against a pre-defined label order based training. For this, we consider another metric called "top-1 accuracy" in Table 4.2, which denotes the percentage of images with the correct top-1 predicted label. The top-1 label is the first predicted label in the sequence, and is obtained by running a single iteration of LSTM for both the methods. As we can see, our top-1 accuracy is much higher as compared to when a pre-defined (rare-to-frequent) label ordering is used for training. This is so because in practice, several labels often correspond to small/specific objects/concepts that are difficult to identify. In such a scenario, if the first predicted label is wrong, it increases the likelihood of the subsequent ones also being wrong. However, our model implicitly learns to choose a salient label based on which it can predict other labels, and thus achieves a higher top-1 accuracy.

Another reason why previous CNN-RNN based methods have used rare-to-frequent label ordering is to give more emphasis on the correct prediction of rare labels, since they have higher influence on per-label evaluation metrics. To analyze how our model performs for labels in different frequency range, we sort the labels based on their frequencies in ascending (rare-to-frequent) order. We divide them equally into four groups, thus the first group contains the 25% most rare labels and the last group contains the 25% most frequent labels. We compare the $F1_L$ scores using both S-CNN-RNN and our method, in each of the label groups in Figure 4.3. As we can observe, our model performs better than S-CNN-RNN for the rarest labels, in case of both NUSWIDE and COCO, even though it has not been explicitly

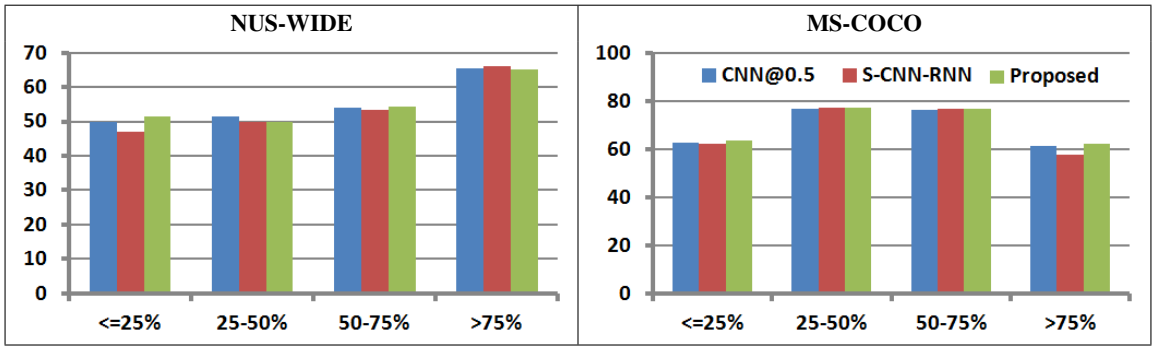

Figure 4.3 Comparison between CNN@0.5, S-CNN-RNN [36] and the proposed model for $F1_L$ scores in different label bins sorted by frequency. (Best viewed in color.)

				
clouds, sky	clouds, sky, water, beach, buildings	clouds, sky, window	ocean, water, waves	animal, dog
clouds, house, sky	clouds, sky, water	clouds, sky, vehicle, window	animal, ocean, water, waves	animal, dog, sky

Figure 4.4 Annotations for example images from the NUS-WIDE dataset. The second row shows the ground-truth labels and the third row shows the labels predicted using the proposed model. The labels in blue are the ones that match with the ground-truth, and the labels in red are the ones that are depicted in the corresponding images but are missing in their ground-truth annotations. (best viewed in colour)

emphasized to predict rare labels. In general, our model achieves either better or comparable $F1_L$ score for all the four label partitions on both the datasets. This shows that our model is capable of learning label importance better than the models that try to do so explicitly by enforcing a pre-defined ordering on labels.

4.4.6.2 Comparison with CNN

Traditionally, image annotation methods [17, 19, 53] would assign the top-k labels from the prediction scores, even when in practice an image can have any number of labels. To address this, two approaches have been proposed in the literature. First is to predict a label sequence using a CNN-RNN based model [25, 55], that would predict the "end-of-sequence" token after predicting a variable number

of labels. Second is to apply a threshold on the prediction scores of a CNN [30]. We compare our model with both CNN@Top-K and CNN@α, where $\alpha \in [0, 1]$ is the threshold and is usually set as 0.5.

As observed in the Table 4.1, CNN-RNN models in general perform better than CNN@Top-K. They improve upon the precision since they predict variable number of labels, and also have a higher recall because of their ability to learn label dependencies. CNN@α tries to mitigate this by restricting the number of labels predicted based on their prediction scores, and thus achieves higher precision than CNN@Top-K, though at the cost of reduced recall. When compared to CNN@α, the performance of our model is comparable in terms of the $F1_L$ score, but it has a much higher recall, and its performance is limited by the decrease in precision. This can be attributed to the fact that there may not be enough examples in the training data to capture the noisy label correlations. As shown in Figure 4.3, our model performs the best in case of rare labels compared to both CNN@α and S-CNN-RNN. In real-world scenarios, different applications have different requirements in terms of precision and recall, however since our model is able to maintain higher F1 score, it may be preferred over other models in general.

4.4.7 Qualitative Results

We present some qualitative results on the NUSWIDE dataset in Figure 4.4. From these results, we can observe that our model correctly predicts most of the ground-truth labels. Moreover, the additional labels being predicted are depicted in the corresponding images, but are missing in their ground-truth (also called incomplete-labelling). As it is well-acknowledged in the image annotation domain, real-world multi-label datasets suffer from class-imbalance and incomplete-labelling issues [19, 53]. While the quantitative comparisons in Figure 4.3 demonstrated the effectiveness of the proposed model in addressing the class-imbalance issue, these results qualitatively demonstrate its capability in addressing incomplete-labelling.

4.5 Summary

We have presented a new approach to train CNN-RNN-style model for the multi-label image annotation task, that allows the RNN component of the model to learn semantic relationships among the labels without the need of providing a pre-defined label ordering. Experiments on NUS-WIDE and MS-COCO datasets validate the efficacy of the proposed model, and also show its ability to learn relationships among labels better than the competing label sequence prediction models.

Chapter 5

Conclusion

5.1 Summary

Automatic image annotation is a challenging problem as it involves real world images that can be associated with multiple labels. It is also a fundamental problem in computer vision with numerous applications, and that can assist in other visual learning tasks such as image captioning, scene recognition, etc.. In this thesis, we began by introducing the automatic image annotation problem, motivating the reader as to its importance and explaining the difficulties involved. We then undertook a study from two different aspects.

In chapter 3, through thorough empirical analysis we made some key dataset and evaluation metrics related observations, that are significant in order to make systemic advancements in the image annotation domain. We concluded that per-label metrics are a better indicator of performance of an annotation method than per-image metrics. With proposed measures, we showed the lack of diversity in existing image annotation datasets. We have marked the necessary considerations to be made when developing new datasets and techniques for the image annotation task in the future. The approaches we have taken in performing our experiments also provide new ways of carrying out analysis in various areas of research.

In Chapter 4, we addressed the image annotation task based on the premise that labels corresponding to different visual concepts in an image share rich semantic relationships among them. Following recent trends, we adopted a unified CNN-RNN framework that can classify images as well as model label dependencies. As previous CNN-RNN-style models followed a predefined label sequence to train their RNN, we showed how a CNN-RNN framework can learn multiple label prediction paths instead of a fixed predefined path. We showed that this model performs better than a predefined ordering imposed on the labels. We also derived further insights into the CNN-RNN framework by showing how it improves the recall of labels, but gets limited in performance by the corresponding decrease in precision which may be due to the lack of enough labelled data for modelling label dependencies. With this analysis, new models of a unified framework and CNN-RNN incorporation will continue to invite new directions of research in the image understanding domain.

5.2 Future Directions

In this final section, we discuss some possible directions in which the proposed methods can be extended for further improvement, as well as new areas of research in the image annotation domain based on our own experience while working on this problem.

- In chapter 4 we proposed a CNN-RNN framework where the RNN component learns label dependencies. An alternate track of CNN-RNN based frameworks utilizes the RNN component to sequentially look into different regions of the image and generate candidate region proposals for classification. We can look into combining the two ideas, where the RNN can have a semantic understanding while generating object proposals.

- In chapter 4 we discussed the importance of learning semantic relationships between the discrete labels of an image. In automatic image annotation, these labels can be nouns, verbs, adjectives etc.. We may like to investigate if the nature of the labels can help us in learning better label correlations.

- During our analysis, we observed that the difficulties which CNNs face in recognizing small objects is applicable to the automatic image annotation problem as well. This encourages us to look into multi-scaled object proposals for further improvement in performance.

- During our analysis, we also observed that due to missing labels in the test data, our model gets penalized during evaluation even when it predicts a correct label. This reduces the actual performance of a method and hence demands our attention in the future.

Appendix A

The Quirks: Additional results with ResNet

In this appendix, we present the results of the experiments as described in chapter 3 using ResNet [21] features. In Table A.1, Table A.2 and Table A.3, we compare the annotation performance of different label prediction models (Refer section 3.3.3). In Table A.4 we show the per-label vs per-image evaluation criteria related experimental results as described in section 3.4.1.1. Image diversity related quantitative analysis as described in section 3.4.2.2 is presented in Tables A.5 and A.6.

Method	Per-label metrics					Per-image metrics			
	P_L	R_L	$F1_L$	mAP_L	N+	P_I	R_I	$F1_I$	mAP_I
Johnson[27]	54.74	57.30	55.99	61.88	–	53.46	75.10	62.46	80.27
Hu[23] (1)	57.02	59.82	58.39	67.20	–	56.84	78.78	66.04	89.99
Hu[23] (2)	58.30	60.63	59.44	69.24	–	57.05	79.12	66.30	82.53
Liu [36]	71.73	61.73	66.36	–	–	77.41	76.88	77.15	–
Gong[17]	31.65	35.60	33.51	–	80	48.59	60.49	53.89	–
Ren[45]	37.74	40.15	38.91	–	81	52.23	65.03	57.93	–
Wang[55]	40.50	30.40	34.73	–	–	49.90	61.70	55.18	–
Liu [36]	55.65	50.17	52.77	–	–	70.57	71.35	70.96	–
SoftMax*	44.36	51.88	47.82	47.94	80	53.61	75.68	62.76	80.73
Sigmoid*	46.97	53.11	49.85	55.72	81	54.64	76.74	63.83	82.20
Ranking*	45.73	52.82	49.03	49.30	81	54.09	75.96	63.19	81.28
WARP*	44.74	52.44	48.28	48.96	81	53.81	75.48	62.83	80.65
LSEP*	43.30	53.98	48.05	51.09	81	54.34	76.41	63.52	81.86
JEC	39.71	40.60	40.15	22.05	80	49.27	69.84	57.78	62.83
TagRel	40.27	60.26	48.28	50.87	81	50.86	71.95	59.60	74.85
TagProp	49.45	59.13	53.86	54.55	81	52.37	74.21	61.41	78.03
2PKNN	47.94	55.76	51.55	52.97	81	51.90	73.00	60.67	77.46
SVM	46.35	54.02	49.89	53.55	81	54.23	76.00	63.30	80.97

Table A.1 Performance comparison of various annotation models (deep-learning based models are marked by '*') on the NUS-WIDE dataset using *ResNet* features. Refer section 3.3.3 for details

Method	Per-label metrics					Per-image metrics			
	P_L	R_L	$F1_L$	mAP_L	N+	P_I	R_I	$F1_I$	mAP_I
SoftMax*	56.77	56.20	56.49	57.24	80	57.88	71.05	63.79	79.69
Sigmoid*	58.82	57.92	58.37	66.74	80	59.78	72.62	65.58	81.72
Ranking*	56.75	55.75	56.25	58.75	80	58.00	70.54	63.66	79.42
WARP*	57.09	55.31	56.19	58.11	80	57.54	70.03	63.18	78.93
LSEP*	57.35	58.66	57.99	61.41	80	59.52	72.61	65.41	81.51
JEC	56.06	45.53	50.25	32.53	80	51.64	64.29	57.27	59.77
TagRel	55.67	59.67	57.60	63.30	80	55.66	68.67	61.49	74.40
TagProp	63.11	58.29	60.61	63.46	80	58.17	71.07	63.98	78.52
2PKNN	63.77	55.70	59.46	62.72	80	54.13	66.95	59.86	75.48
SVM	60.27	57.67	58.94	65.52	80	59.33	72.08	65.09	80.38

Table A.2 Performance comparison of various annotation models (deep-learning based models are marked by '*') on the MS-COCO dataset using *ResNet* features. Refer section 3.3.3 for details

Method	Per-label metrics					Per-image metrics			
	P_L	R_L	$F1_L$	mAP_L	N+	P_I	R_I	$F1_I$	mAP_I
Corel-5K									
JEC	37.87	43.04	40.29	34.94	151	46.41	65.80	54.43	51.64
TagRel	37.01	45.23	40.71	40.13	157	46.25	65.43	54.20	59.16
TagProp	36.54	42.62	39.34	49.23	145	47.81	67.92	56.12	62.10
2PKNN	47.21	54.95	50.78	54.6	201	45.41	64.30	53.23	59.08
SVM	36.79	43.84	40.01	54.32	155	48.94	69.64	57.48	68.13
ESP Game									
JEC	47.14	30.43	36.99	21.92	235	42.69	48.22	45.29	37.20
TagRel	43.12	41.84	42.47	39.97	255	44.55	49.87	47.06	51.37
TagProp	48.48	41.78	44.88	41.04	253	45.70	51.60	48.47	53.48
2PKNN	46.17	44.22	45.17	42.89	262	46.04	52.25	48.95	54.22
SVM	47.11	35.02	40.17	43.46	236	48.99	55.55	52.06	58.61
IAPR TC-12									
JEC	48.69	27.16	34.87	20.52	226	51.34	49.49	50.40	39.86
TagRel	48.29	41.35	44.55	41.70	265	53.03	51.31	52.15	55.50
TagProp	52.27	40.97	45.93	45.82	266	52.88	51.28	52.07	57.72
2PKNN	53.54	42.58	47.43	48.88	281	53.21	51.43	52.30	59.32
SVM	51.97	28.16	36.53	48.71	224	54.41	52.37	53.37	61.05

Table A.3 Performance comparison of various annotation methods on Corel-5K, ESP Game and IAPR TC-12 datasets using *ResNet* features. Refer section 3.3.3 for details

	Method	Per-label F1 score ($F1_L$)				Per-image F1 score ($F1_I$)			
		True	Rare	Freq.	Rand	True	Rare	Freq.	Rand
Corel-5K	Ground	100.00	99.26	99.55	70.65	100.00	82.62	82.62	82.62
	JEC	40.29	49.92	49.06	33.19	54.43	54.57	56.64	54.99
	TagRel	40.71	52.19	51.37	34.95	54.20	54.33	57.38	54.98
	TagProp	39.34	49.16	47.87	32.53	56.12	56.31	57.73	56.56
	2PKNN	50.78	64.36	63.96	39.39	53.23	53.32	57.01	53.84
	SVM	40.01	50.89	50.13	34.06	57.48	57.58	59.21	57.72
ESP Game	Ground	100.00	99.26	99.56	81.74	100.00	88.60	88.60	88.60
	JEC	36.99	46.26	45.50	28.44	45.29	45.35	49.80	45.85
	TagRel	42.47	58.71	58.42	35.92	47.06	47.11	53.09	47.56
	TagProp	44.88	58.83	58.08	36.61	48.47	48.53	52.99	49.14
	2PKNN	45.17	61.38	61.06	37.83	48.95	49.00	53.43	49.34
	SVM	40.17	50.90	50.20	32.18	52.06	52.12	55.00	52.62
IAPR TC-12	Ground	100.00	99.26	99.71	87.75	100.00	92.83	92.83	92.83
	JEC	34.87	40.77	40.41	28.21	50.40	50.53	53.67	51.03
	TagRel	44.55	56.95	57.01	38.45	52.15	52.24	56.89	52.63
	TagProp	45.93	56.71	56.73	38.56	52.07	52.16	56.85	52.52
	2PKNN	47.43	59.19	59.12	39.43	52.30	52.39	55.39	52.80
	SVM	36.53	41.87	41.23	29.04	53.37	53.52	55.14	53.86
NUS-WIDE	Ground	100.00	98.57	99.12	62.36	100.00	79.88	79.88	79.88
	SoftMax*	47.82	68.43	67.75	36.74	62.76	62.77	65.33	63.17
	Sigmoid*	49.85	69.44	69.08	37.26	63.83	63.84	66.18	64.20
	Ranking*	49.03	69.25	68.84	37.07	63.19	63.20	65.74	63.60
	WARP*	48.28	68.86	68.51	36.72	62.83	62.84	65.33	63.21
	LSEP*	48.05	70.14	69.82	37.70	63.52	63.53	66.10	63.90
	JEC	40.15	58.14	57.57	29.43	57.78	57.80	61.60	58.37
	TagRel	48.28	74.85	75.12	40.19	59.60	59.61	66.25	60.13
	TagProp	53.86	74.14	74.10	40.19	61.41	61.42	65.93	61.84
	2PKNN	51.55	71.15	71.31	37.94	60.67	60.68	63.71	61.12
	SVM	49.89	70.19	69.82	37.64	63.30	63.31	65.60	63.68
MS-COCO	Ground	100.00	98.36	99.20	82.94	100.00	85.46	85.46	85.46
	SoftMax*	56.49	72.08	71.76	51.26	63.79	63.88	66.26	64.33
	Sigmoid*	58.37	73.35	73.06	52.81	65.58	65.65	67.35	66.02
	Ranking*	56.25	71.65	71.37	50.87	63.66	63.74	65.76	64.20
	WARP*	56.19	71.43	71.04	50.34	63.18	63.26	65.68	63.70
	LSEP*	57.99	73.84	73.68	53.14	65.41	65.48	67.47	65.90
	JEC	50.25	63.27	62.66	42.01	57.27	57.37	59.77	58.04
	TagRel	57.60	74.70	74.68	52.78	61.49	61.56	67.47	62.12
	TagProp	60.61	73.66	73.42	52.54	63.98	64.05	66.36	64.45
	2PKNN	59.46	71.70	71.44	49.77	59.86	59.94	66.06	60.49
	SVM	58.94	73.10	72.87	52.44	65.09	65.16	66.86	65.53

Table A.4 Comparing the actual per-label and per-image performance (using *ResNet*) of various label prediction models (deep-learning based models are marked by *) with those obtained by replacing incorrect predictions with rare/frequent/random incorrect labels. (Refer Section 3.4.1.1 for details.)

	Method	Per-label metrics				Per-image metrics			
		P_L	R_L	$F1_L$	mAP_L	P_I	R_I	$F1_I$	mAP_I
Corel-5K	JEC	73.50	78.88	76.10	74.73	64.00	86.58	73.59	72.79
	TagRel	72.38	79.02	75.55	77.48	64.20	86.83	73.82	85.70
	TagProp	70.93	79.55	74.99	85.88	64.80	87.83	74.58	86.87
	2PKNN	77.51	87.53	82.22	88.44	68.40	92.83	78.76	90.02
	SVM	68.22	81.64	74.33	88.81	66.60	90.17	76.61	89.42
ESP Game	JEC	58.54	51.67	54.89	45.86	55.68	65.63	60.25	52.94
	TagRel	57.23	62.79	59.88	65.68	54.53	63.23	58.56	66.56
	TagProp	62.59	64.83	63.69	67.21	56.83	66.68	61.37	69.59
	2PKNN	62.03	65.87	63.89	65.90	58.18	68.39	62.87	70.44
	SVM	57.26	59.61	58.41	67.73	60.09	70.90	65.05	74.60
IAPR TC-12	JEC	57.97	46.19	51.42	42.49	66.21	67.04	66.62	58.61
	TagRel	65.99	62.63	64.27	71.58	67.53	67.74	67.63	75.93
	TagProp	68.15	64.74	66.40	76.52	66.61	67.24	66.92	77.98
	2PKNN	69.27	64.97	67.05	75.58	68.29	68.70	68.49	78.97
	SVM	61.25	50.91	55.60	77.15	68.85	68.95	68.90	80.76
NUS-WIDE	SoftMax*	59.96	67.45	63.49	66.84	62.87	81.68	71.05	88.51
	Sigmoid*	61.95	68.61	65.11	72.55	63.93	82.69	72.11	90.30
	Ranking*	60.27	68.24	64.01	67.37	63.49	82.26	71.66	89.64
	WARP*	59.83	67.86	63.59	67.14	63.30	82.01	71.45	89.06
	LSEP*	59.34	68.82	63.73	69.23	63.76	82.54	71.94	90.01
	JEC	59.38	51.27	55.02	40.04	58.37	76.62	66.26	72.47
	TagRel	61.54	73.08	66.82	70.13	61.37	79.83	69.39	84.72
	TagProp	67.10	74.23	70.49	72.15	62.05	80.78	70.19	86.53
	2PKNN	67.33	70.76	69.00	70.41	61.50	79.91	69.51	85.55
	SVM	64.72	69.88	67.20	69.72	63.32	81.88	71.42	88.69
MS-COCO	SoftMax*	55.55	66.04	60.34	66.06	57.61	89.90	70.22	94.23
	Sigmoid*	55.85	66.54	60.73	71.42	57.92	90.16	70.53	94.69
	Ranking*	53.86	66.07	59.34	65.44	57.55	89.77	70.14	94.26
	WARP*	53.98	66.00	59.39	65.17	57.24	89.45	69.81	93.95
	LSEP*	56.08	66.99	61.06	67.82	58.02	90.30	70.65	94.79
	JEC	61.84	55.72	58.62	45.09	52.80	84.71	65.05	79.22
	TagRel	59.11	65.26	62.03	67.08	54.25	85.96	66.52	88.57
	TagProp	68.28	63.73	65.93	67.31	56.69	88.73	69.18	92.40
	2PKNN	63.95	64.70	64.32	66.29	53.93	85.81	66.23	90.44
	SVM	63.85	63.40	63.63	66.84	56.96	88.99	69.46	92.92

Table A.5 Performance comparison (using *ResNet*) of various label prediction models (deep-learning based models are marked by *) over the 20% most overlapping test subsets of various datasets (refer Section 3.4.2.2 for details.)

	Method	Per-label metrics				Per-image metrics			
		P_L	R_L	$F1_L$	mAP_L	P_I	R_I	$F1_I$	mAP_I
Corel-5K	JEC	19.68	25.69	22.29	23.31	27.80	45.75	34.58	35.15
	TagRel	19.25	28.24	22.89	25.59	27.80	45.42	34.49	37.62
	TagProp	20.32	29.35	24.01	38.17	32.20	52.58	39.94	44.31
	2PKNN	21.03	28.24	24.11	47.12	25.80	43.08	32.27	38.44
	SVM	18.86	29.40	22.98	43.29	32.60	53.25	40.44	49.64
ESP Game	JEC	20.91	16.11	18.20	12.92	29.21	33.06	31.02	24.32
	TagRel	21.26	23.29	22.22	22.90	31.99	36.29	34.01	35.52
	TagProp	22.31	22.36	22.34	25.18	33.91	38.43	36.03	38.19
	2PKNN	24.00	25.22	24.60	27.36	32.76	37.27	34.86	38.23
	SVM	18.39	16.86	17.59	26.54	37.27	42.65	39.78	43.39
IAPR TC-12	JEC	20.12	14.44	16.81	11.89	36.39	35.86	36.12	26.43
	TagRel	25.81	24.19	24.97	25.35	38.78	38.66	38.72	38.40
	TagProp	24.59	20.78	22.52	29.85	39.39	39.55	39.47	41.42
	2PKNN	28.58	22.01	24.87	31.80	38.68	38.06	38.36	42.39
	SVM	16.72	12.12	14.05	31.01	39.95	39.49	39.72	44.75
NUS-WIDE	SoftMax*	20.33	25.84	22.76	18.37	41.82	67.57	51.66	67.38
	Sigmoid*	23.46	27.01	25.11	22.96	42.71	68.74	52.69	68.73
	Ranking*	21.84	25.99	23.73	18.90	41.94	67.32	51.69	67.12
	WARP*	21.78	27.21	24.20	19.18	41.45	66.26	51.00	66.03
	LSEP*	20.79	28.89	24.18	20.17	42.22	68.07	52.12	68.02
	JEC	16.96	24.82	20.15	11.24	37.25	59.87	45.93	48.28
	TagRel	17.62	37.71	24.02	20.05	38.07	61.19	46.94	59.21
	TagProp	24.16	30.37	26.91	22.11	41.25	66.42	50.90	65.01
	2PKNN	22.63	28.05	25.05	20.10	40.14	63.74	49.26	63.75
	SVM	19.31	24.40	21.56	20.70	42.17	67.39	51.88	67.41
MS-COCO	SoftMax*	38.04	34.46	36.16	30.70	47.43	52.92	50.02	61.57
	Sigmoid*	41.22	37.91	39.50	38.28	50.13	55.20	52.54	64.32
	Ranking*	38.82	33.03	35.69	31.06	46.81	50.93	48.78	60.05
	WARP*	39.82	31.65	35.27	30.53	46.13	49.85	47.92	59.09
	LSEP*	40.18	39.50	39.83	34.87	49.66	55.27	52.32	63.92
	JEC	42.37	37.32	39.69	39.13	51.00	55.83	53.31	63.93
	TagRel	36.46	42.08	39.07	37.19	45.97	51.47	48.56	56.02
	TagProp	42.18	39.15	40.61	37.06	48.59	53.68	51.01	60.65
	2PKNN	45.89	33.71	38.87	36.59	43.76	48.35	45.94	56.09
	SVM	41.01	37.98	39.44	39.19	49.82	54.83	52.20	63.32

Table A.6 Performance comparison (using *ResNet*) of various label prediction models (deep-learning based models are marked by *) over the 20% least overlapping test subsets of various datasets (refer Section 3.4.2.2 for details.)

Related Publications

Journal:

- Ayushi Dutta, Yashaswi Verma, C. V. Jawahar. Automatic Image Annotation: The Quirks and What Works. *In Multimedia Tools and Applications, 2018*

Bibliography

[1] Typical cnn architecture. https://en.wikipedia.org/wiki/Convolutional_neural_network#/media/File:Typical_cnn.png,.

[2] Neurons of a convolutional layer (blue), connected to their receptive field (red). https://en.wikipedia.org/wiki/Convolutional_neural_network#/media/File:Conv_layer.png,.

[3] L. V. Ahn and L. Dabbish. Labeling images with a computer game. In *ACM SIGCHI conference on Human factors in computing systems*, 2004.

[4] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown. Learning multi-label scene classification. *Pattern recognition*, 39(9):1757–1771, 2004.

[5] G. Carneiro, A. B. Chan, P. J. Moreno, and N. Vasconcelos. Supervised learning of semantic classes for image annotation and retrieval. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(3):394–410, 2007.

[6] M. Chen, A. Zheng, and K. Q. Weinberger. Fast image tagging. In *ICML*, 2013.

[7] 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 *AAAI*, 2018.

[8] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng. NUS-WIDE: A real-world web image database from National University of Singapore. In *ACM CIVR*, 2009.

[9] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines: And Other Kernel-based Learning Methods*. Cambridge University Press, 2000.

[10] J. Devlin, H. Cheng, H. Fang, S. Gupta, L. Deng, X. He, G. Zweig, and M. Mitchell. Language models for image captioning: The quirks and what works. In *ACL*, 2015.

[11] P. Duygulu, K. Barnard, J. F. G. de Freitas, and D. A. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *ECCV*, 2002.

[12] D. L. Eliot. Support vector machines (svm) for ai self-driving cars. https://www.aitrends.com/ai-insider/support-vector-machines-svm-ai-self-driving-cars/, 2018.

[13] H. J. Escalante, C. A. Hérnadez, L. E. Sucar, and M. Montes. Late fusion of heterogeneous methods for multimedia image retrieval. In *MIR*, 2008.

[14] S. L. Feng, R. Manmatha, and V. Lavrenko. Multiple Bernoulli relevance models for image and video annotation. In *CVPR*, 2004.

[15] A. Frome, G. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov. Devise: A deep visual-semantic embedding model. In *Neural Information Processing Systems (NIPS)*, 2013.

[16] H. Fu, Q. Zhang, and G. Qiu. Random forest for image annotation. In *ECCV*, pages 86–99, 2012.

[17] Y. Gong, Y. Jia, T. K. Leung, A. Toshev, and S. Ioffe. Deep convolutional ranking for multilabel image annotation. In *ICLR*, 2014.

[18] M. Grubinger, P. D. Clough, H. Müller, and T. Deselaers. The IAPR benchmark: A new evaluation resource for visual information systems. In *International Conference on Language Resources and Evaluation*, 2006. URL `http://www-i6.informatik.rwth-aachen.de/imageclef/resources/iaprtc12.tgz`.

[19] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid. TagProp: Discriminative metric learning in nearest neighbour models for image auto-annotation. In *ICCV*, 2009.

[20] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Comput.*, 16(12):2639–2664, 2004.

[21] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[22] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(9):1735–1780, Nov. 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `http://dx.doi.org/10.1162/neco.1997.9.8.1735`.

[23] H. Hu, G.-T. Zhou, Z. Deng, Z. Liao, and G. Mori. Learning structured inference neural networks with label relations. In *CVPR*, 2016.

[24] J. Jeon, V. Lavrenko, and R. Manmatha. Automatic image annotation and retrieval using cross-media relevance model. In *SIGIR*, 2003.

[25] J. Jin and H. Nakayama. Annotation order matters: Recurrent image annotator for arbitrary length image tagging. In *ICPR*, 2016.

[26] J. Johnson, L. Ballan, and L. Fei-Fei. Love thy neighbors: Image annotation by exploiting image metadata. In *ICCV*, 2015.

[27] J. Johnson, L. Ballan, and L. Fei-Fei. Love thy neighbors: Image annotation by exploiting image metadata. In *ICCV*, 2015.

[28] M. M. Kalayeh, H. Idrees, and M. Shah. NMF-KNN: Image annotation using weighted multi-view non-negative matrix factorization. In *CVPR*, 2014.

[29] X. Li, C. G. M. Snoek, and M. Worring. Learning social tag relevance by neighbor voting. *Trans. Multi.*, 11(7):1310–1322, 2009.

[30] Y. Li, Y. Song, and J. Luo. Improving pairwise ranking for multi-label image classification. In *CVPR*, 2017.

[31] Y. Li, Y. Song, and J. Luo. Learning spatial regularization with image-level supervisions for multi-label image classification. In *CVPR*, 2017.

[32] Z. Li and J. Tang. Weakly supervised deep matrix factorization for social image understanding. *IEEE Transactions on Image Processing*, 26(1):276–288, 2016.

[33] Z. Li, J. Liu, C. Xu, and H. Lu. Mlrank: Multi-correlation learning to rank for image annotation. *Pattern Recognition*, 46(10):2700–2710, 2013.

[34] Z. Li, J. Liu, J. Tang, and H. Lu. Robust structured subspace learning for data representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(10):2085–2098, 2015.

[35] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnic. Microsoft COCO: Common objects in contex. In *ECCV*, 2014.

[36] F. Liu, T. Xiang, T. M. Hospedales, W. Yang, and C. Sun. Semantic regularisation for recurrent image annotation. In *CVPR*, 2017.

[37] A. Makadia, V. Pavlovic, and S. Kumar. A new baseline for image annotation. In *ECCV*, 2008.

[38] A. Makadia, V. Pavlovic, and S. Kumar. Baselines for image annotation. *Int. J. Comput. Vision*, 90(1):88–105, 2010.

[39] G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM (CACM)*, 38 (11):39–41, 1995.

[40] S. Moran and V. Lavrenko. A sparse kernel relevance model for automatic image annotation. *International Journal of Multimedia Information Retrieval*, 3(4):209–219, 2014.

[41] Y. Mori, H. Takahashi, and R. Oka. Image-to-word transformation based on dividing and vector quantizing images with words. In *MISRM'99 First International Workshop on Multimedia Intelligent Storage and Retrieval Management*, 1999.

[42] K. Mullick. *Learning Deep and Compact Models for Gesture Recognition*. MS Thesis, IIIT Hyderabad, 2017.

[43] M. B. R. *Exemplar based approaches on Face Fiducial Detection and Frontalization*. MS Thesis, IIIT Hyderabad, 2017.

[44] N. Rasiwasia, J. C. Pereira, E. Coviello, G. Doyle, G. R. G. Lanckriet, R. Levy, and N. Vasconcelos. A new approach to cross-modal multimedia retrieval. In *ACM MM*, 2010.

[45] Z. Ren, H. Jin, Z. L. Lin, C. Fang, and A. L. Yuille. Multi-instance visual-semantic embedding. *CoRR*, abs/1512.06963, 2015.

[46] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vision*, 115(3):211–252, 2015.

[47] T. seng Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng. Nus-wide: A real-world web image database from national university of singapore. In *In CIVR*, 2009.

[48] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[49] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[50] T. Uricchio, L. Ballan, L. Seidenari, and A. D. Bimbo. Automatic image annotation via label transfer in the semantic space. *CoRR*, abs/1605.04770, 2016.

[51] Y. Verma and C. V. Jawahar. Image annotation using metric learning in semantic neighbourhoods. In *ECCV*, 2012.

[52] Y. Verma and C. V. Jawahar. Exploring SVM for image annotation in presence of confusing labels. In *BMVC*, 2013.

[53] Y. Verma and C. V. Jawahar. Image annotation by propagating labels from semantic neighbourhoods. *Int. J. Comput. Vision*, 121(1):126–148, 2017.

[54] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015.

[55] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu. CNN-RNN: A unified framework for multi-label image classification. In *CVPR*, 2016.

[56] Y. Wei, W. Xia, J. Huang, B. Ni, J. Dong, Y. Zhao, and S. Yan. Order-free rnn with visual attention for multi-label classification. *arXiv:1406.5726*, 2014.

[57] J. Weston, S. Bengio, and N. Usunier. WSABIE: Scaling up to large vocabulary image annotation. In *IJCAI*, 2011.

[58] S. Zhang, J. Huang, Y. Huang, Y. Yu, H. Li, and D. N. Metaxas. Automatic image annotation using group sparsity. In *CVPR*, pages 3312–3319, 2010.

[59] Y. Zhuang, Y. Yang, , and F. Wu. Mining semantic correlation of heterogeneous multimedia data for cross-media retrieval. *IEEE Transactions on Multimedia*, 10(2):221–229, 2008.