July 2019

DATA-DRIVEN APPROACH TO IMAGE CLASSIFICATION

Venkatesh NarasimhaMurthy

DATA-DRIVEN APPROACH TO IMAGE CLASSIFICATION

A Dissertation Presented

by

VENKATESH N. MURTHY

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2019

College of Information and Computer Sciences

DATA-DRIVEN APPROACH TO IMAGE CLASSIFICATION

A Dissertation Presented

by

VENKATESH N. MURTHY

Approved as to style and content by:

R. Manmatha, Chair

James Allan, Member

Subhransu Maji, Member

Patrick A. Kelly, Member

James Allan, Chair of the Faculty
College of Information and Computer Sciences

to my grandparents, especially to my maternal grandmother Chandramma

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor, R. Manmatha, for helping and guiding me throughout my graduate career. He taught me to conduct research with practical aspects, made me realize the importance of writing a good code, most importantly he was always there for me even after moving to the other end of the country. I would also like to thank my committee members: James Allan, Subhransu Maji and Patrick A. Kelly. Especially James and Subhransu for high-level and interdisciplinary guidances allowing me to see beyond my otherwise limited research field.

I thank Vivek Singh (my mentor during an internship at Siemens) for his valuable discussions and suggestions on my research work. His brilliant ideas and unrelenting passion have been very inspiring. I would also like to thank A.G. Ramakrishnan (my MS thesis advisor at IISc), Pallapa Venkatram (mentor at IISc) and M. Girish Chandra (mentor at Tata Consultancy Services Innovation labs) for nurturing me as a researcher early on, and instilling in me the curiosity to pursue a PhD.

Special thanks goes to all the graduate students/alumni for their friendship and collaboration: Ethem F. Can, Ismet Zeki Yalniz, Harshal Pandya, Abhishek Roy, Sandeep Kalra, Pranav Mirajkar, Sunil Kumar, Weize Kong, Shiri Dori-Hacohen, Anand Seetharam, Henry Feild, and David Wemhoener. I am sincerely indebted to all CSCF and CIIR staff members for their support of my work, especially Kate Morruzzi , Dan Parker, Jean Joyce and Leeanne M. Leclerc.

Finally, I would like to thank the most important people in my life, my family. I thank my parents, Padma Govindaraj and Narasimha Murthy Muddaiah, for their tremendous sacrifices, unconditional love, encouragement and support. I thank my in-laws Ramamma, Prema and Chikkegowda for their constant support. I thank my wife, Pavitra and two beautiful sons, Vihan and Advik, for always being there for me and giving me positive energy. Particularly, thank Pavitra for being so understanding, especially with respect to my travels and internships; I truly appreciate it - **we did this**. I wouldn't have been able to make it without their support.

This work was supported in part by the Center for Intelligent Information Retrieval, in part by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11-PC20066, and in part by NSF grant #IIS-0910884. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DOI/NBC, or the U.S. Government.

ABSTRACT

DATA-DRIVEN APPROACH TO IMAGE CLASSIFICATION

MAY 2019

VENKATESH N. MURTHY

B.E., SIR M. VISVESVARAYA INSTITUTE OF TECHNOLOGY, BANGALORE

M.E., INDIAN INSTITUTE OF SCIENCE, BANGALORE.

M.Sc., UNIVERSITY OF MASSACHUSETTS, AMHERST

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: R. Manmatha

Image classification has been a core topic in the computer vision community. Its recent success with convolutional neural network (CNN) algorithm has led to various real world applications such as large scale management of photos/videos on cloud/social-media, image based search for online retailers, self-driving cars, building robots and healthcare. Image classification can be broadly categorized into binary, multi-class and multi-label classification problems. Binary classification involves assigning one of the two class labels to an instance. In multi-class classification problem, an instance should be categorized into one of more than two classes. Multi-label classification is a generalized version of the multi-class classification problem where each image is assigned multiple labels as opposed to a single label.

In this work, we first present various methods that take advantage of deep representations (fully connected layer of pre-trained CNN on the ImageNet dataset) and

yield better performance on multi-label classification when compared to methods that use over a dozen conventional visual features. Following the success of deep representations, we intend to build a generic end-to-end deep learning framework to address all three problem categories of image classification. However, there are still no well established guidelines (in terms of choosing the number of layers to go deeper, the number of kernels and the size, the type of regularizer, the choice of non-linear function, etc.) to build an efficient deep neural network and often network architecture design is specific to a problem/dataset. Hence, we present some initial efforts in building a computational framework called Deep Decision Network (DDN) which is completely data-driven. DDN is a tree-like structured built stage-wise. During the learning phase, starting from the root network node, DDN automatically builds a network that splits the data into disjoint clusters of classes which would be handled by the subsequent expert networks. This results in a tree-like structured network driven by the data. The proposed approach provides an insight into the data by identifying the group of classes that are hard to classify and require more attention when compared to others [146, 49, 117, 76]. This feature is crucial for people trying to solve the problem with little or no domain knowledge, especially for applications in medical domain [97, 155]. Initially, we evaluate DDN on a binary classification problem and later extend it to more challenging multi-class and multi-label classification problems. The extension of DDN to multi-class and multi-label involves some changes but they still operate under the same underlying principle. In all the three cases, the proposed approach is tested for its recognition performance and scalability on publicly available datasets providing comparison to other methods.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure **Page**

CHAPTER 1
INTRODUCTION

In recent times, we have witnessed an exponential growth in visual data, creating a huge demand for automatic digital image analysis. For example, this growth includes diverse sources such as the millions of photos/videos uploaded everyday to social networks as well as the increase in medical images due to advancements in imaging technologies. Further, the free unlimited cloud storage service for multimedia data offered by companies like Google and Amazon suggests a strong continuing growth of data. Such large volumes of data require an advanced recognition/classification system to automatically organize and summarize them. Several attempts [150, 44, 78] have been made toward this goal and recent advances in the field of deep learning technology have made it possible to achieve satisfactory performance in real world applications. However, since each classification problem/application is unique, there is still a requirement to build an efficient and effective network architecture to further improve the performance. For instance, Convolutional Neural Network (CNN) based methods have consistently been the top performers on large scale image classification benchmarks such as the ImageNet Large-Scale Visual Recognition challenge (ILSVRC 2012, 2013 and 2014) [64, 116, 125]. The success of CNNs is partly due to the availability of large datasets and high-performance computing systems and partly due to the recent technical advances in learning methods and regularization techniques like dropout [122], dropconnect [139], maxout [38] and batch normalization [56]. However, there are still no well established guidelines to train a performant network. One way to overcome this is by cross-validation, but that becomes too expensive since there are too many design choices to make. Some of the recent work tried to address this using bayesian optimization for tuning the hyperparameters of the network [119] but still the design choice (in terms of number and type of layers) has to be in place. Hence, we take a small step towards addressing this issue and propose a data-driven computational framework for building the neural networks.

Figure 1 shows an example of how understanding/summarizing an image problem is defined in the computer vision community. The problem is categorized based on the use case and its complexity. For instance, the binary classification problem can be defined as distinguishing whether it's an indoor or outdoor image [128, 68, 131] or in the medical/machine-vision applications, it's mostly focused on separating positive samples from negative ones [80]. In fact, most of the classical machine learning algorithms such as logistic regression, neural network, linear discriminant analysis and Support Vector Machines (SVM) [94] were initially formulated to solve the binary classification and were later extended to the multiclass classification problem. Following the same trend, we validate our data-driven approach of building a neural network for the binary classification problem first and later extend it for more complex problems. The multiclass classification can be defined as a problem of assigning one label chosen from a large predefined vocabulary that summarizes or identifies the main interest in an image [45, 11], in Figure 1 it could be "swimming pool". In case of the multi-label classification, the problem involves assigning multiple labels to an image that identifies all the objects of interest in an image [78, 150]. In increasing order of complexity, we have binary, multiclass and multilabel classification. Multi-label classification is more challenging and interesting because of the following a) Manual annotation of all the objects of interest in an image is labor intense and expensive, b) there exists a strong correlation and dependencies between labels, c) annotated images are scarce resulting in data imbalance and d) user defined tags on social media data or Flickr are incomplete and they often expresses individuality or emotions (which isn't of much help for solving image annotation problem). Multi-label classification is also known by other names such as image annotation and image tagging, hence we use them interchangeably.

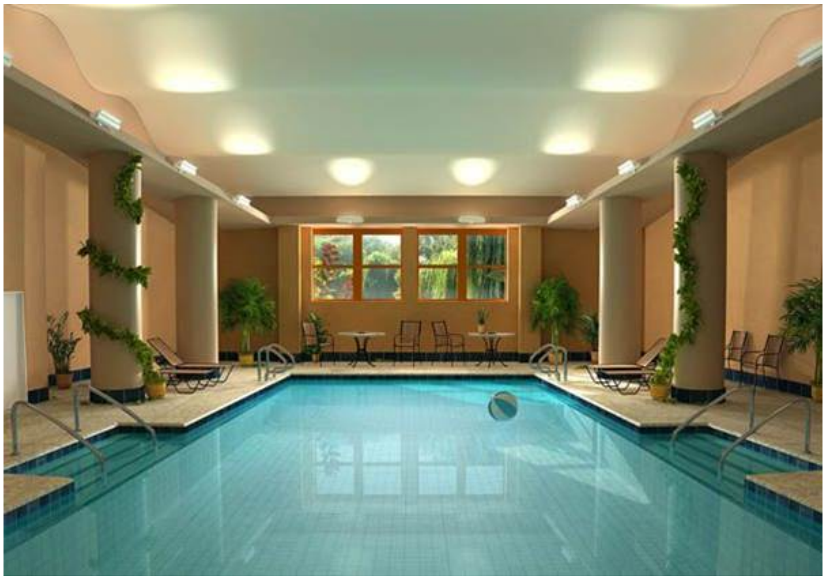


Figure 1.1: Example showing various ways of addressing the image understanding problem. **Binary classification**: [Indoor, Outdoor]. **Multi-class classification**: [Kitchen, Dining area, Bedroom, Swimming pool, . . .]. **Multi-label classification**: [Water, Windows, Chairs, plants, Ball, Sky, Grass, Car, Beach, . . .].

In this dissertation, we start off by studying the effectiveness of deep representations versus multiple handcrafted features with our proposed models for addressing the multi-label classification problem. Following the advantages of deep learning features, we aim to design an end-to-end deep learning network to solve the image annotation problem. But since network architecture design can be problem/dataset specific, we propose a data driven approach for designing an efficient network. Initially, we test our data-driven hypothesis on a relatively simpler problem such as binary classification and later extend them to more complex multi-class and multi-label (image annotation/tagging) classification problems.

The contributions of this dissertation are as follows:

- Three different models for image annotation - a hybrid model (SVM-DMBRM), a Canonical Correlation Analysis (CCA) model and a hypergraph model, all of them use deep representations and yield similar or better performance than most of the existing methods that use over dozens of handcrafted features. One of the proposed approaches (CCA-KNN) is shown to outperform all other existing techniques on four standard publicly available datasets: Corel-5K, ESP-Game, IAPRTC-12 and NUS-WIDE.

- A novel deep learning architecture known as a Cascaded Deep Decision Network (CDDN) for addressing the binary classification of endoscopic images for diagnosing medical images.

- A novel data-driven deep learning architecture called a Deep Decision Network (DDN) that provides an alternative approach towards building an efficient deep learning network for multi-class classification. This is shown to yield state-of-the art performance (at the time of publication of this work [92]) on two publicly available datasets: CIFAR-10 and CIFAR-100. The proposed approach is tested for scalability, yielding competitive results on large scale ImageNet dataset.

- DDN principle is extended to address the more complex multi-label classification problem. The proposed approach is tested for both scalability and performance on publicly available image annotation datasets.

The contributions and scope of the work presented in this thesis reflect the trends in the computer vision community (particularly image classification task) around late 2017 or before that time period. Since then there have been many advances because the field is moving rapidly (partially due to the success of deep learning techniques). For example, the number of paper submissions for one of the top conferences in the computer vision community (CVPR) grew from 1724 (461 accepted) in 2010 to 5100 (1300 accepted) in 2019, that's more than double [1]. In terms of progress made, the current state-of-the-art (Gpipe [55]) yields a top-5 error rate of 3% in image classification task on ILSVRC 2012 dataset [23], which is almost a 33% (relative) error reduction when compared to ResNet-152 (single model testing) [48]. That's a significant amount of error reduction in less than 2 years when compared to pre-deep learning period (before 2012) that only reduced the top-5 error rate by 7% (28% to 26%) [106]. Overall, the trend suggests that building deeper networks seems to help improve the classification accuracy. Moreover, the best results in the competition seems to use ensemble of models to improve the performance, but training ensemble of networks is both time consuming and computationally expensive. Hence, as an alternative we provide a data-driven framework for designing an efficient deep neural network built on the idea of mixture of experts (is based on the divide-and-conquer principle). Our generic framework can be applied to any state-of-the-art network (as root node), and hence it remains applicable even with the current research trends in the field. It has the potential to further boost the performance along with providing some valuable insights into the data.

The outline of the dissertation is as follows:

Chapter 1 is about introduction to the problem that we are trying to solve.

Chapter 2 provides the literature overview along with some required background.

Chapter 3 introduces three kind of models to address the multi-label (Image Annotation) classification problem. In the first section, a hybrid model which combines a discriminative (SVM) and a generative model (DMBRM) is proposed and we show its effectiveness compared to a model with fourteen handcrafted features. In the second section, we present a CCA based model which incorporates both a visual feature (deep learning feature) and a text feature (word embedding vector) to build a better image annotation system. Following that, we propose a novel multi-scale hypergraph heat diffusion framework for the automatic image annotation task. This last technique enables us to model the higher order relationship among images in the feature space and provides a multi-scale label diffusion mechanism to address the class imbalance problem in the data.

Chapter 4 presents a novel data driven framework CDDN that builds a deep neural network on the fly for classifying endoscopic images. During the learning phase, CDDN automatically builds a network which discards samples that are classified with high confidence scores by a previously trained network and concentrates only on the challenging samples which are handled by the subsequent expert shallow networks. CDDN is validated on a publicly available ISBI 2014 Polyp challenge dataset.

Chapter 5 introduces a novel Deep Decision Network (DDN) that provides an alternative approach towards building an efficient deep learning network. During the learning phase, starting from the root network node, DDN automatically builds a network that splits the data into disjoint clusters of classes which are handled by the subsequent expert networks. The proposed method provides an insight into the data by identifying the group of classes that are hard to classify and require more attention when compared to others. In DDN's evaluation on standard public datasets (CIFAR-10 and CIFAR-100) for the multi-class classification problem is shown to yield state-of-the-art performance (at the time of publication of this work [92])). The system is also shown to be scalable to one of the largest publicly available ImageNet dataset yielding competitive results. Later, the data-driven deep neural network idea is extended to solve the multi-label classification problem and we call it as DDN-annot. Following the underlying principle of the DDN, the root node strategy remains the same, the subsequent expert network or cluster of samples are identified by applying unsupervised K-means to the feature space (preferably the last layer before the soft-max). The hope is that the clusters would capture the coexistence of labels and thus giving a better chance for the expert network to predict more meaningful labels based on the visual content. DDN-annot is evaluated on two publicly available datasets: IAPRTC-12 and relatively large NUS-WIDE dataset. DDN-annot yielded comparable results to the state-of-the-art results when compared to deep learning based methods in 2017.

CHAPTER 2
RELATED WORK

Before we present the related work, we would like to provide background about statistical methods that get used in building the image annotation models. In the first section, we discuss relevance models concept which were inspired by its application in information retrieval, followed by Support Vector Machines (SVM) and Canonical Correlation Analysis (CCA). We then discuss the recently popular deep learning based feature representation techniques for both images and text. The second section discusses related work on (multi-label classification). This is followed by binary classification work that is specific to endoscopic images and finally multiclass image classification work that includes the latest developments in deep learning techniques.

2.1 Background

In this section, some of the techniques that are found to be useful for the image annotation task are briefly explained. First, we talk about relevance models in the immediate subsection, followed by SVM and CCA. In the final subsection, we briefly discuss image/text embeddings (created using neural network) that are used as features and they are shown to be quite successful in recent times for various computer vision and NLP tasks.

2.1.1 The Relevance Model

The relevance model is used for determining the probability $P(w|R)$ of observing a word w in a document that is relevant to a query, where R represents the class

Table 2.1: Summary of generative models

Method	Word distribution $P(w\vert J)$	Image distribution$P(I\vert J)$
CMRM	$(1-\alpha_j)\frac{\#(w,\ j)}{\vert j\vert} + (\alpha)\frac{\#(w,\ \tau)}{\vert\tau\vert}$	$(1-\beta_j)\frac{\#(b,\ j)}{\vert j\vert} + (\beta)\frac{\#(b,\ \tau)}{\vert\tau\vert}$
CRM	$\frac{\mu p_{w,j} + N_w}{\mu + N}$	$\frac{1}{n}\sum_{i=1}^{n}\frac{exp{-}(g-g_i)^T\Sigma^{-1}(g-g_i)}{\sqrt{2^k\pi^k\vert\Sigma\vert}}$
MBRM	$\frac{\mu\delta_{w,j} + N_w}{\mu + N}$	$\frac{1}{n}\sum_{i=1}^{n}\frac{exp{-}(g-g_i)^T\Sigma^{-1}(g-g_i)}{\sqrt{2^k\pi^k\vert\Sigma\vert}}$

of documents that are relevant to the query. In the context of image annotation, it involves determining $P(w|I)$, probability of a word given an image. The relevance models used for image annotation are CMRM [58], CRM [67] and MBRM [27] that produced strong baselines in the early years of image annotation research.

In order to obtain $P(w|I)$, we need a good estimate of the joint distribution $P(w, I)$ of observing image features with possible annotation words. As shown in [27] and [67], one possible way of determining it is by computing the expectation over training images. The annotations for a test image are obtained by maximizing the expectation:

$$w^* = \arg\ \max_{w\in V} P(w, I_t) \tag{2.1}$$

$$w^* = \arg\ \max_{w\in V} \sum_{J\in T} P(w, I_t|J)P(J) \tag{2.2}$$

where J is an image in the training set T, w is a word or a set of words in the vocabulary V, and I_t is the test image.

Assuming that the probabilities of observing word w and the image I_t are mutually independent then the above equation can be rewritten as:

$$w^* = \arg\ \max_{w\in V} \sum_{J\in T} P(w|J)P(I_t|J)P(J) \tag{2.3}$$

where $P(J)$ indicates the prior distribution of an image, and usually assumed to be uniform. $P(w|J)$ is the likelihood of w given the training image J, in other words, it models the word distribution in the training set. For modeling the words, a multinomial distribution was used in case of CRM [67] and a multiple Bernoulli distribution was used for MBRM [27]. $P(I_t|J)$ represents the likelihood of the test image given the training image J. This is estimated using visual similarity. Both $P(I_t|J)$ and $P(w|J)$ are estimated using a maximum likelihood estimate smoothed with the background model. The summary of these models are captured in table 2.1.

From table 2.1, in case of CMRM, $\#(w, J)$ denotes the actual number of times the word/tag/label w occurs in the caption of image J. $\#(w, \tau)$ is the total number of times w occurs in all captions in the training set τ . Similarly, $\#(b, J)$ reflects the actual number of times some region of the image J is labeled with visual word b, and $\#(b, \tau)$ is the cumulative number of occurrences of visual word b in the training set. $|w|$ stands for the aggregate count of all tags and visual words occurring in image J, and $|\tau|$ denotes the total size of the training set. The smoothing parameters α_J and β_J determine the degree of interpolation between the maximum likelihood estimates and the background probabilities for the tags and the visual words respectively.

In the case of CRM and MBRM models as shown in table 2.1, Gaussian kernel is used for representing feature g_i of every region of image J. Σ represent feature covariance matrix. For approximating the word distribution, CRM and MBRM differs only by parameter $p_{w,J}$ (representing the probability of occurrence of word)and $\delta_{w,J}$ (it takes a value 1 if that word is present or else 0). μ is the smoothing parameter, N_w is the number of training images that contain w in the annotation and N is the total number of training images.

2.1.2 Support Vector Machines

For a classification problem with labels y and and features x, a support vector machine (SVM) is one of the popular machine learning algorithms for classifying the samples into one of the finite set of discrete categories. Here we provide a brief description of a binary class SVM, but for more details please refer to [20].

Let (x_i, y_i) be the training samples, where $i = 1, ...N$, $y_i \in \{-1, 1\}$ is the class label (binary-class problem). SVM tries to find a hyperplane w that better separates the positive and negative samples with a maximum margin. In order to find the hyperplane one has to solve the following optimization problem:

$$
\begin{aligned}
min_w \frac{1}{2} \parallel w \parallel^2 \qquad (2.4)\\
s.t \;\; y_i(w^T x_i + b) \geq 1, i = 1, ..., N
\end{aligned}
$$

When the training samples are not linearly separable, we introduce a slack variable $\xi \geq 0$ to relax the constraint on every sample.

Now the optimization involves, minimizing the following equation

$$\frac{1}{2} \parallel w \parallel^2 + \frac{1}{N} C \sum_{i}^{N} \xi_i \qquad (2.5)$$

subject to $\xi \geq 0$ and the following equation

$$y_i(w^T x_i + b) \geq 1 - \xi_i \quad \forall_i \qquad (2.6)$$

Here, C is used to control the tradeoff between the two terms and $\parallel . \parallel^2$ denotes the L2 norm squared.

Having found the optimal hyperplane, the decision function for classifying a sample is given by

$$f(x) = sign(w^T x + b) \tag{2.7}$$

This yields the distance from the hyperplane and the sign indicates the category type (positive implies class label 1 and negative implies class label -1).

2.1.3 Canonical Correlation Analysis

Let's assume, $\mathbf{X}$ and $\mathbf{Y}$ are p-dimensional and q-dimensional random vectors respectively, such that the joint variate $\mathbf{Z} = (\mathbf{X}^T, \mathbf{Y}^T)^T$ has a joint distribution with mean zero and positive definite covariance matrix $\sum$. Without loss of generality, assume that $p \leq q$.

Consider an arbitrary linear combination $U = \alpha^T X$ of the components of X and an arbitrary linear combination $V = \gamma^T Y$ of the components of Y.

The correlation between U and V is given by

$$\rho = \frac{cov(U, V)}{\sqrt{var(U)var(V)}} = \frac{cov(\alpha^T X, \gamma^T Y)}{\sqrt{var(U)var(V)}} = \frac{\alpha^T cov(X, Y)\gamma}{\sqrt{var(\alpha^T X)var(\gamma^T Y)}} \tag{2.8}$$

Canonical correlation analysis seeks to find the U and V that have maximum correlation. The basic theory of the analysis was developed by Hotelling [52].The following derivation is obtained from [53], for more details please refer to it.

Since scaling does not change correlation, one typically normalizes the arbitrary vectors such that U and V have unit variance, that is,

$$1 = var(U) = cov(\alpha^T X, \alpha^T X) = \alpha^T \Sigma_{xx} \alpha \equiv \alpha^T \Sigma_{xx} \alpha \tag{2.9}$$

$$1 = var(V) = cov(\gamma^T Y, \gamma^T Y) = \gamma^T \Sigma_{yy} \gamma \equiv \gamma^T \Sigma yy \gamma \tag{2.10}$$

Here, Σ denotes the covariance.

Since $E(U) = E(V) = 0$ and with 2.9 and 2.10, the correlation between U and V Eq. 2.8 reduces to

$$\rho = E(UV) = \alpha^T \Sigma_{xy} \gamma. \tag{2.11}$$

The problem then becomes one of finding α and γ to maximize Eq. 2.11 subject to the constraints in Eq. 2.9 and Eq. 2.10. Let,

$$\phi = \alpha^T \Sigma_{xy} \gamma - \frac{1}{2} \delta (\alpha^T \Sigma_{xx} \alpha - 1) - \frac{1}{2} w (\gamma^T \Sigma_{yy} \gamma - 1), \tag{2.12}$$

where δ and w are Lagrange multipliers. Differentiating ϕ with respect to the elements of α and γ and setting the result to zero, we obtain

$$\frac{\partial \phi}{\partial \alpha} = \Sigma_{xy} \gamma - \delta \Sigma_{xx} \alpha = 0 \tag{2.13}$$

$$\frac{\partial \phi}{\partial \gamma} = \Sigma'_{xy} \alpha - w \Sigma_{yy} \gamma = 0 \tag{2.14}$$

Hence we have,

$$\Sigma_{xx}^{-1} \Sigma_{xy} \gamma = \delta \alpha \tag{2.15}$$

and

$$\Sigma_{yy}^{-1} \Sigma_{xy}^{T} \alpha = w \gamma \tag{2.16}$$

Substituting Eq. 2.16 in 2.15 yields

$$\Sigma_{xx}^{-1} \Sigma_{xy} \Sigma_{yy}^{-1} \Sigma_{xy}^{T} \alpha = M_\alpha \alpha = \lambda \alpha. \tag{2.17}$$

with $\lambda = \delta w$. Similarly, substituting Eq. 2.15 in 2.16 gives

$$\Sigma_{yy}^{-1} \Sigma_{xy}^{T} \Sigma_{xx}^{-1} \Sigma_{xy} \gamma = M_\gamma \gamma = \lambda \gamma. \tag{2.18}$$

Both these equations can be viewed as eigenvalue equations, with M_α and M_γ sharing the same non-zero eigenvalues λ.

As M_α and M_γ are known from the data, α can be found by solving the eigenvalue problem Eq. 2.17.

$w\gamma$ can then be obtained form Eq. 2.16. Since w is unknown, the magnitude of γ is unknown, and the normalization conditions Eq. 2.9 and 2.10 are used to determine the magnitude of α and γ.

The Matrix M_α is of dimension $n_x \times n_x$, while M_γ is $n_y \times n_y$, so generally we pick the smaller of the two to solve the eigenvalue problem.

From Eq.2.11,

$$\rho^2 = \alpha^T \Sigma_{xy} \gamma \gamma^T \Sigma_{xy}^T \alpha = \delta w (\alpha^T \Sigma_{xx} \alpha)(\gamma^T \Sigma_{yy} \gamma) \tag{2.19}$$

From Eq. 2.9 and 2.10, the above equation reduces to

$$\rho^2 = \lambda \tag{2.20}$$

The eigenvalue problems Eq. 2.17 and 2.18 yield n λs, with $n = \min(n_x, n_y)$.

Assuming the $\lambda' s$ to be all distinct and nonzero, we have for each λ_j $(j = 1, .., n)$, canonical variates, u_j and v_j, with correlation $\rho_j = \sqrt{\lambda_j}$ between the two, and eigenvectors, α_j and γ_j.

2.1.4 Image and Text Embeddings

In this section, we look at recently proposed techniques to efficiently represent images and text using neural networks. Though we don't provide all the technical details here, we do try provide a overview of how image and text features are extracted using pre-trained neural networks. In recent times, there is a large improvement in various tasks in both computer vision [64] [82] [34] [110] and natural language processing (NLP) fields [87] [62, 100] [43] due to the recent advances in deep learning techniques. Deep learning's success can be attributed to high performance computing (like GPU's and clusters), massive parallelization and publicly available large datasets.

2.1.4.1 Image Embeddings

From 90's until the deep learning era (2012), many vision related tasks involved a two step approach [51, 61, 11]. The first step involved finding a good representation for images and the second involved choosing the best machine learning algorithm to either classify or segment (pixel-wise classification) them. The performance of the system largely depends on a better representation of the images and these are handcrafted and task-specific. To overcome this, [70] proposed to build an end-to-end system which operates directly on the raw input data to yield the end result (class-label or labeling a region). In [70] , they made use of the back-propagation algorithm to effectively train the convolutional neural network (CNN) showing it's effectiveness on MNIST (isolated handwritten digits) dataset when compared to the traditional two step approach.

Even though CNNs were shown to be successful for relatively smaller tasks, they weren't able to effectively recognize generic objects that required deeper/larger CNNs. Larger CNNs were hard to train because it required a lot of computation time and memory and during 1990's the resources (lack of powerful CPU's or GPU's) were also limited. In addition, larger networks also suffered from overfitting problems (limited availability of labeled data) and there was no efficient technique to overcome this problem.

Thankfully the above mentioned limitations are being overcome to an extent by the following advancements in the field. Since 2010, the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [23] is being conducted annually. ILSVRC consists of around 1.4 million images publicly made available, and they are all manually labeled with the presence or absence of 1000 object categories. This really benefited most of the machine learning algorithms that requires large data. Especially CNNs were able to take advantage of this and as a result in ILSVRC 2012, Krizhevsky et al.,[64] used graphical processor units (GPU) for fast and efficient implementation of larger CNNs and won the competition by a large margin (achieved 15.3% top-5 error rate compared to the second best top-5 error rate of 26.2%). The following are some of the factors that contributed for CNN to be successful: large dataset, parallel computations (GPU's), ReLu (non-linear activation function), data augmentation (affine transformations) and dropout [122] (to avoid overfitting). Neural networks are in general trained using backpropagation algorithm. It involves updating the weights of the network based on the gradients of the loss function.

Following by the success of CNNs at ILSVRC, [104] showed that the output of fully-connected layer in CNNs (such as AlexNet) trained on ILSVRC 2012 dataset can be effectively used as a feature representation for images. They empirically showed its effectiveness for most of the computer vision tasks and in many cases exceeded the performance of state-of-the-art algorithms that used hand-crafted features.

In our work, we use the variant of a CNN known as Visual Geometry Group 16 layered network (VGG-16) [116]. This was designed mainly for competing in ImageNet ILSVRC-2014, and eventually VGG secured the first and the second places in the localization and classification tasks respectively. Recent advances include many variants of CNNs that have large number of layers [116, 34]. Recently CNNs with more than 150 layers [48] have been proposed achieving a top-5 error rate of 4.49% on the imagenet classification task.

The main contribution in VGG-16/VGG-19 work involves rigorous evaluation of networks of increasing depth, which shows that a significant improvement on the prior-art configurations (AlexNet) can be achieved by increasing the depth to 16-19 weight layers, which is substantially deeper than what has been used in the prior art. Further they reduce the number of parameters in such deep networks by using small 3x3 filters in all convolutional layers (the convolution stride is set to 1) as opposed to 5x5 filters. Extensive use of non-linear layers might have been one more factor in the increased efficiency. For more details please refer to their paper [116]. Figure 2.1

and Table 2.2 provides the network architecture for VGG-16. We use the output of fc-4096, a layer before fc-1000 as a feature representation for images.

Following is a brief description of some of the important layers in the VGG network:

Convolutional layer: It consists of a set of learnable filters (weights and biases). As the name suggest, the filters are convolved (dot product) with every spatial point in the input region producing a 2-dimensional activation/feature maps. Though the filters are small in dimension (width x height), the number of filters (depth) are usually large in number. Intuitively, these filters learn to fire when they see a distinctive visual features such as edges, curves and colors in the beginning layers and more part like or object like features deep down the network [149].

Pooling layer: This is mainly used for downsampling, in other words to reduce the spatial dimension. Hence, it helps in reduced computations (because of reduced parameters) and also avoids overfitting. It will also provide some translation invariance.

Fully connected layer: In this layer, each neuron is connected to all the activation units (neurons) of the previous layer. The spatial information is lost but these features are more invariant. This layer maybe used as a feature from a pre-trained network for various applications.

Softmax layer: The softmax function typically remains as the last layer in the network for classification problems. It provides a probabilistic values over all the K classes to make a decision as defined in equation 2.21. It's generally accompanied by cross-entropy (log) loss for computing the gradients and updating the weights using backpropagation algorithm.

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \tag{2.21}$$

Rectified Linear Unit (ReLU) layer: This is used as an activation function

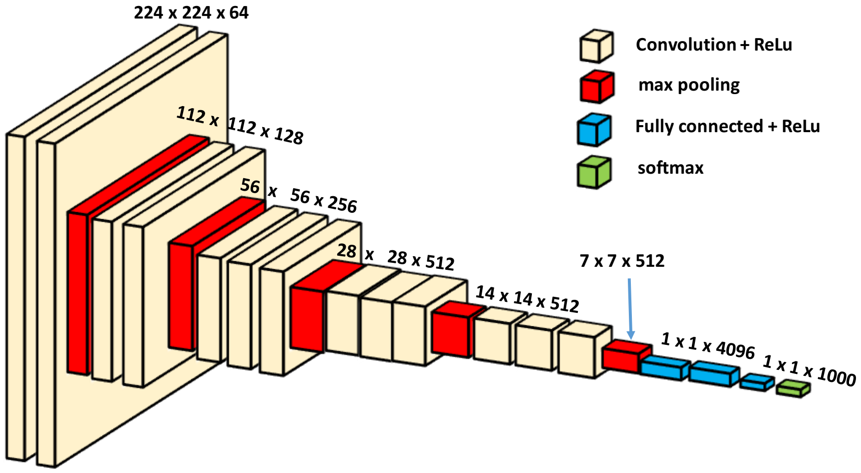


Figure 2.1: Macroarchitecture of VGG16.

which is non-linear in nature. A rectified linear unit has output 0 if the input is less than 0, and it takes the same value as the input otherwise (as defined in equation 2.22).

$$\text{ReLu}(x) = \max(0, x) \tag{2.22}$$

Table 2.2: Network architecture for VGG-16. The convolutional layer parameters are denoted as conv(receptive field size)-(number of channels). The ReLU activation function is not shown for brevity.

Image	conv3-64	conv3-64	max-pool	conv3-128	conv3-128	max-pool	conv3-256	conv3-256	conv1-256	max-pool	conv3-512	conv3-512	conv1-512	max-pool	conv3-512	conv3-512	conv1-512	max-pool	fc-4096	fc-4096	fc-1000	softmax

2.1.4.2 Text Embeddings

Classical approaches for modelling words and sentences uses n-gram conditional probabilities based on the co-occurrence frequencies of words in a document. The main drawback of this approach is the curse of dimensionality, as n grows, context grows exponentially thus making the models intractable. As an alternative, [141] proposed a neural network approach to learn a distributed representation of words which is also known as a word embedding. Here the goal was to jointly learn the word feature vectors and the parameters of the probability function using a common global objective function (maximizing log likelihood). As a result of this, each word in the vocabulary is represented by a real valued feature vector that provides a syntactic and semantic meaning. Recently, the most popular and widely used word representation known as word2vec [87] was proposed. They took an unsupervised approach and proposed a skip gram model which optimizes the objective function of maximizing the average log probability of the neighboring words conditioned on the center word. The most interesting feature of this model is that it preserves linear regularities among the word representations. For example, the word vectors: vec(king)-vec(man)+vec(woman) is close to vec(queen) in feature space. In some of our proposed work, we represent all tags/labels by a real valued vector using the word2vec tool.

2.2 Image Annotation and Retrieval Models

A large number of models have been proposed for automatic image annotation task by researchers in the past decade. Early work involved a machine translation based approach [26], in which the images were represented as vocabulary of blobs (through image segmentation) and the task was to translate it into set of words. Subsequent models further improved the results and they may be broadly divided into three groups - generative models, discriminative models and nearest neighbor based models. Generative model approaches consists of both mixture and topic based models. Examples of mixture models are the Cross Media Relevance Model (CMRM) [58], the Continuous-space Relevance Model (CRM) [67] and the Multiple Bernoulli Relevance Model (MBRM) [27] . These models estimate the joint probability of words and visual features. Given visual features of a test image, the model is then used to compute conditional probability scores for words. Visual features are represented either by discretizing and clustering them or by using a kernel density estimate. The words in the vocabulary maybe modeled using a multinomial distribution but the best results are obtained by modeling the words using a multiple Bernoulli distribution. The parameters of these models are estimated using smoothed maximum likelihood estimates. Along the same lines, a Markov Random Field [143] based approach was proposed that could boost the potential of traditional generative model approaches by modeling context relationship among semantic concepts. Recently an attempt was made to improve the performance of CRM using Sparse Kernel Learning (SKL-CRM) [90] , in which they try to learn the optimal combination of kernels for over a dozen visual features. In the case of topic models, each annotated image is modeled as a mixture of topics over visual and tag features, where the mixture proportions are shared between different features or views. Some of the work related to this include latent Dirichlet allocation [9], probabilistic latent semantic analysis [89], hierarchical Dirichlet processes [144].

Most of the earlier proposed models [26] [58] [67] took block/region based approach for extracting visual features. The images were segmented using segmentation algorithm such as normalized cuts [113]. Since most of the images were unconstrained the segmentation results were inconsistent and this had an impact on the final results. Later, researchers [27] showed that features computed over a simple rectangular region on the image instead of segmentation can yield superior performance. And in most recent models [83] [41] [132], it was shown that simply using multiple global features can yield better performance over all the existing methods.

Discriminative based approaches like support vector machines (SVM) [102], supervised multi-class learning (SML) [14] and multiple instance learning [39] involve building a classifier for each annotation tag by treating them as multi-class multi-labelling problem (either one-versus-all or one-versus-one). In the passive aggressive model for image retrieval (PAMIR) [39] they optimize an image ranking loss inspired by ranking SVM. A random forest based approach was also explored in [31]. In a recent paper [133], they modified the hinge-loss in SVM to gain tolerance against confusing labels. Scalability is an issue in this type of setup as it requires pre-defined set of models per word.

Several successful nearest neighbor based models have been inspired by MBRM including the Joint Equal Contribution (JEC) model [83], TagProp [41] and the 2PKNN model [132]. JEC was the first to utilize nearly dozens of local and global features. JEC proposed to consider equal contributions from different features (mean of distances) while transferring annotations from the nearest neighbors to the test image. Followed by JEC, the authors of tag propagation (Tag-prop) introduced the standard multiple-feature (15 local and global visual features) image annotation dataset.

They also introduced weighted K-Nearest Neighbor (KNN) which assign labels to the test image based on the learned weights of the tags from neighboring training images. In addition, to address the class-imbalance problem and to effectively combine multiple features, metric learning along with label-specific models in the nearest neighbor setup was proposed. The state-of-the-art 2PKNN (two-pass K-Nearest-Neighbor) technique is a two-step approach. Given a test image, it tries to find K neighboring images from each semantic group (images are grouped according to their annotation labels thus resulting in L overlapping clusters for a vocabulary of size L) to form a subset of training set. In the second step, the tags are predicted based on the weighted combination of distances from multiple features of images in the training subset. The optimal weights to combine base distances and features was determined via metric learning, which involves a large margin set-up by generalizing a single-label classification metric learning algorithm for multi-label prediction. 2PKNN yielded better performance partly because it was able to effectively handle the data imbalance problem by finding a subset of training images for every test image. But, however it adds extra computation at test time making it a weak candidate for practical applications. In a recent paper [7], the performance of the nearest neighbor based methods was significantly improved using kernelized canonical correlation analysis (KCCA) embeddings of both visual and textual features. NMF-KNN approach in [60] fused multiple features using weighted multi-view nonnegative matrix factorization in conjunction with NN approach.

Most of the recent techniques are based on deep learning, CNN with Weighted Approximate Ranking (WARP) [36] loss was introduced to leverage the CNN feature representations and rank the candidate tags for a given test image using ranking objective. Fast0Tag [151] involved end-to-end learning for embedding both visual and textual information but their main focus was to solve zero-shot multi-label classification problem (labels for which there exists no training images). Recent attempt [140] was made to combine semantic representation obtained from CNN with RNN (capturing label dependency) model to capture both image-label relationship and label dependency.

2.3 Binary Classification of Endoscopic images

The automatic analysis of endoscopic images plays a vital role in visual diagnosis of medical conditions. In most cases, the medical conditions are completely curable if they are detected early. Towards this end, conventional computer vision based techniques have demonstrated reasonable success [32, 138, 137]. Most of the methods [75, 6, 5, 152, 74] (dealing with endoscopic images) typically involve extraction of features from images, followed by a vector quantization step based on a pre-defined visual vocabulary (usually constructed by k-means clustering) which results in an intermediate compact representation of an image that can be ingested as a training sample for supervised classifiers. While these methods are effective, they consistently fail to leverage the data-driven aspect of the problem as all three steps - feature extraction, generation of intermediate representation, and finally the classification, are mutually independent.

Recently, deep learning based approaches [64] have demonstrated a significant performance boost on generic image classification tasks [23] by addressing the final classification objective in an integrated framework using layered neural networks. This has motivated many researchers to apply deep neural network based methods in the field of medical image analysis [17, 15, 95, 16, 33, 153]. In an early work [69] pertinent to classification, the authors introduce a two-layer network which utilizes independent subspace analysis to reconstruct a natural representation of tumor images captured through cell-microscopy.

With that being said, training networks for medical image classification tasks is a challenging task as it often requires thorough experimentation on large datasets. Due to the lack of large amount of good quality training data, the trained network architecture often overtly optimizes itself for only training data, and performs poorly on unseen test samples. The authors of [8] avoid this issue by employing a pre-trained convolutional neural network [64] whose parameters are learned from a large database of images from non-medical cases [23]. Their research demonstrates high performance on a medical application of chest pathology detection in X-ray images. While such a pre-trained architecture has demonstrated success in a specific cross-domain exercise, the generalization aspect is still inconclusive.

2.4 Multiclass Image Classification using CNN

Deep Learning in general has shown to be an effective framework advancing the state-of-the art performance on various tasks [108] in the field of computer vision [82, 116, 130, 93, 109, 34]. In particular, CNN based models have been the top performers in computer vision related tasks till date and recent work show that Recurrent Neural Networks [RNN] [101] with CNN (as the image recognition backend) can be effective for some of the tasks dealing with sequence data. This is made possible due to publicly available large image datasets such as ImageNet [23] and also due to high performance computing systems like GPU's and large scale distributed clusters. For any given problem, the real challenge exists in coming with an effective architecture, and this could vary based on the domain requirements (medical-field, autonomous-driving, machine-vision etc.). Various attempts have been made to come up with an effective CNN network architecture either by going deeper (see [116, 125] which were the top performers in the 2014 ImageNet challenge) or by introducing new components:- **1. Activation units** such as (a) rectified linear unit (ReLu) [64] helped in accelerating the learning and have a great influence on the performance of large models trained on large datasets, (b) Parametric rectified linear units (PReLu) [47] which replace the parameter-free ReLU activation by a learned parametric activation unit to further improve the classification performance; **2. Regularizers** like (a) dropout [122] randomly set some activation units to zero in a given layer and provides the effect of model averaging, (b) dropconnect [139] sets the weights to zero instead of activation units, (c) maxout [38] outputs the maximum of a set of inputs and this can be used as an alternative to dropout; **3. Normalization** such as batch normalization [56] that normalizes the layer inputs providing an accelerated learning and improved performance. We propose an alternative generic deep learning framework which helps in improving the classification performance by leveraging any of the existing networks (with a mixture of new components) as the starting root node in our proposed tree structured deep decision network. Our work is inspired by decision trees [103] and the idea of sample partitioning [13], which are both classical approaches in machine learning. There have been a few related papers that are tree-like structured CNNs, starting with [123] aimed towards improving the classification performance of classes with limited training dataset by transferring knowledge among similar classes. A recent paper [50] attempted to build a hierarchical CNN but the main objective was to transfer knowledge from a large network to a small network to achieve scalability but without compromising on the performance. In our proposed work, we aim to provide a generic framework that automatically discovers data-hierarchy and improve the performance by separating out the easily separable data from the hard ones. The hard confusion cases will be routed deep down the tree to be handled by the expert network nodes. This framework reflects the well established idea of mixture of experts [86] machine learning algorithm. Mixture of experts is developed based on the divide-and-conquer principle. Here the input space is divided and handled by the respective experts (can be any learner). The final decision may be based on a weighted experts decision or it can also be a gated function. In our case, we use error based (confusion matrix to be specific) for partitioning the input data into different clusters. For each cluster we assign an expert network and during inference, the final decision is based solely on the expert.

CHAPTER 3

MULTI-LABEL CLASSIFICATION (IMAGE ANNOTATION) AND RETRIEVAL MODELS USING DEEP REPRESENTATIONS.

3.1 Introduction

Automatic image annotation is a labeling problem wherein the task is to predict multiple textual labels for an image describing its contents or visual appearance. Automatic image/video annotation plays an important role in managing the exponentially increasing number of images/videos being uploaded to the internet. For instance in the year 2017, people uploaded 350 million photos on an average to Facebook each day [2] and more than 400 hours of videos were uploaded to YouTube every minute [3]. Some of this data is tagged by the users, but these tags may be ambiguous or incomplete. Researchers have tried to make use of the metadata associated with the images/videos to build a better image classification or object detection system. Image classification system has also been used to transfer the tags to unannotated images based on their similarities and vice versa [83, 27, 41, 91, 67]. Our objective here is to predict a fixed number of tags for a given test image which accurately describes the visual content.

Most existing techniques are based on supervised learning which involve learning a mapping function between low level visual features (color, local descriptors, etc) and high-level semantic concepts (sun, sky, etc). However, the problem of poor annotation (training images not being annotated with all relevant keywords) and class-imbalance (large variations in the number of positive samples/class) make automatic image annotation a difficult problem to solve. Existing methods use dozens of handcrafted features such as quantized Scale Invariant Feature transform (SIFT), quantized color histograms in different color spaces (RGB, LAB, HSV) to build a tag prediction model. These models may be generative [27, 67, 142], discriminative [14, 133, 41, 132] or nearest neighbor-based ones; among these, nearest neighbor based models are shown to be the most successful [83, 41, 132].

Multiple features with the right type of model are shown to improve the annotation performance significantly in the current state of the art system [132]. Yet, these dozens of handcrafted features serve as a bottleneck in designing scalable realtime systems. Hence, we propose a set of models in the following sections which uses a deep learning representation (based on a single CNN based feature representing an image) yielding better results. CNN features are shown to be successful for many vision tasks (object detection, classification and segmentation) producing significantly improved results on the most challenging datasets such as PASCAL VOC and ILSVRC2012 [35, 105].

The rest of the chapter is organized as follows: First, the proposed models are described in the following order - SVM-DMBRM (a hybrid) model that combines both discriminative and generative approaches in Section 3.2, a method based on Canonical Correlation Analysis (CCA) with deep learning embeddings (for both images and labels) in Section 3.3 and a hypergraph based model in Section 3.4. A brief introduction to all the datasets that are being used in our experiments are provided in Section 3.5; the details of evaluation metrics are given in Section 3.6. Section 3.7 presents the experimental results of all our proposed models and the findings. Finally in Section 3.8, we provide additional experimental results on one of the largest dataset (NUS-WIDE) using our best performing models, which helps in validating the scalability factor.

3.2 Hybrid (SVM-DMBRM) Model

Here we present a hybrid model combining a generative and a discriminative model for the image annotation task. A Support Vector Machine (SVM) is used as the discriminative model and a Discrete Multiple Bernoulli Relevance Model (DMBRM) is used as the generative model. The idea of combining both the models is to take advantage of the distinct capabilities of each model. The SVM tries to address the problem of poor annotation (images are not annotated with all relevant keywords), while the DMBRM model tries to address the problem of data imbalance (large variations in the number of positive samples). In practice, DMBRM does not work well with high-dimensional data, hence a Latent Dirichlet Allocation (LDA) model is used to reduce the dimensionality of vector quantized feature before using it. The results of the hybrid model compare well with the state-of-the-art results on three standard datasets: Corel-5k, ESP-Game and IAPRTC-12 (see section 3.5).

In addition, we show that deep learning (DL) features when combined with the SVM-DMBRM model yield comparable results to the state-of-the-art system. Image features are extracted using Convolutional Neural Networks (CNN) pre-trained on the ImageNet dataset (ISVRC12). The SVM-DMBRM model with powerful deep learning features results in a simple, efficient and scalable system because it uses a single feature as opposed to dozens of handcrafted features used in the state-of-the-art system. We also provide the SVM-DMBRM performance results with DL feature on a large dataset (NUS-WIDE).

3.2.1 Discriminative Model

Image annotation may be viewed as a variation of a multi-class problem in which a number of words are employed to annotate a test image. However, in the case of images sharing the same annotations, the creation of multi-class models is very difficult because different classes sharing the same descriptors yield noisy discriminative hyper-planes; here we focus on binary models rather than a multi-class model. In the case of binary models the intra-class dependencies are ignored unlike the multi-class models. Here we create a binary classification model per word in the vocabulary and then make use of its responses for annotation. While creating a model M_{w_i} for word w_i we assume that the images (in the training set) annotated with w_i are positive examples (i.e. $y_i = +1$) and similarly the images that are not annotated with w_i are assumed to be negative examples (i.e. $y_i = -1$). Employing binary classification models for words enables us to deal with the issue of images sharing the same word annotations.

If our vocabulary consists of a number of words $W = \{w_1, w_2, ..., w_n\}$ then we create n binary models each of which provides a discriminative model for its corresponding word. For a test image we get n responses representing the probability of having an annotation of each word. The standard evaluations [67, 58, 27, 41, 132] require five word annotations per image, hence we annotate a test image with the five words having the highest responses. Imbalanced positive examples might be a problem for the image annotation task since every word might have a different number of annotated images. We normalize the responses of each binary model to deal with this imbalance problem. We first take the normal inverse cumulative distribution of the responses (i.e. the probabilities of having a word as an annotation) and then we map them back to [0,1].

3.2.2 Generative Model

We use a discrete MBRM model as opposed to the continuous model proposed in [27]. The reason for the discrete version is due to the fact that it helps in reducing the computational complexity. The following derivation follows [27, 58]. Let V represent the annotation vocabulary and W be any arbitrary set of words. Also, Let J be an image in the training dataset $\mathcal{T}$. Each image is associated with a set of dimensionality reduced feature vector and annotation words where, each feature vector f has a dimension m and annotation words have dimension n ($W = w_1, w_2 \cdots w_n$). For a test image, we extract its features and its distribution is known but we need to predict the words associated with it, formally given by $P(w|f)$. From Bayesian theory,

$$P(w|f) = \arg\max_{w} \frac{P(w,f)}{P(f)} \tag{3.1}$$

One possible solution to computing the joint distribution $P(w,f)$ is by taking an expectation over the entire training set of images. Following the formulation in [58], the joint probability is given by:

$$\begin{aligned} P(w,f) = \sum_{J \in \mathcal{T}} \{ P_{\mathcal{T}}(J) \prod_{i=1}^{m} P(f_i|J) \prod_{w_i \in w} P(w_i|J) \times \\ \prod_{w_i \notin w} (1 - P(w_i|J)) \} \end{aligned} \tag{3.2}$$

$P_{\mathcal{T}}(J)$ is kept uniform for all images in the training dataset. $P(f_i|J)$ are estimated using smoothed maximum likelihood estimates [58] as follows:

$$P(f_i|J) = (1 - \alpha_J) \frac{n(f_i, J)}{n(f, J)} + (\alpha_J) \frac{n(f_i, \mathcal{T})}{n(f, \mathcal{T})} \tag{3.3}$$

Here $n(f_i, J)$ represents the number of times the visterm (quantized feature value) occurs in the training image J, $n(f, J)$ denotes the total number of visterms in image J, $n(f_i, \mathcal{T})$ denotes the number of times the visterm occurs in the entire training dataset $\mathcal{T}$ and $n(f, \mathcal{T})$ indicates the total number of visterms in the entire training dataset $\mathcal{T}$. The smoothing parameter α is estimated using a validation dataset.

$P(w_i|J)$ for each word is estimated using a Bayes estimate given by [27]

$$P(w_i|J) = \frac{\beta * 1_{w_i, J} + N_{w_i}}{\beta + N} \tag{3.4}$$

Here, $1_{w_i,J}$ is a indicator function for word w_i occurring in image J. The smoothing parameter β is estimated using a validation dataset. N_w is the number of training images containing w_i and N is the total number of training images.

3.2.2.1 LDA for Dimensionality Reduction

In our experiment, the feature sets are vector quantized and generally are large dimensional vectors. One of the main limitation of generative models such as CMRM, CRM or MBRM model is that their performance is limited by the dimensionality of the feature vector. Consider equation (3.2), in order to compute $P(f_i|J)$ we take a product over all the feature values because of the independence assumption. Even though we use the log-sum-exp trick, its performance gets degraded. In order to overcome this we used a Latent Dirichlet Allocation model [66] to reduce the dimensionality. We treat each feature value in an image as a word and summarize the words in the document using fewer topics. In other words, the LDA model gives us a compact representation of feature vectors. Experimentally, we fixed the dimensionality of the feature vectors to be around 100 for all 14 features. These dimensionality-reduced features were used only in the case of the generative model whereas the feature dimensionality remained unchanged for the SVM model.

3.2.3 Fusion of Models

To get the best of both the techniques, we combine discriminative and generative models as follows. Let $F = \{f_1, f_2, ..., f_m\}$ be a set of descriptors that we use in this work. Let $P_d(f_i)$ be the response of a discriminative model in terms of probabilities created for the descriptor f_i. Please note that we create separate models for each of the descriptors. Similarly let $P_g(f_i)$ be the response of a generative model in terms of probabilities created over the descriptor f_i. Then the final response P_D for discriminative models is provided below;

$$P_D = \frac{1}{m} \sum_{i}^{m} P_d(f_i) \tag{3.5}$$

Similarly, the final response G for generative models is as follows:

$$P_G = \frac{1}{m} \sum_{i}^{m} P_g(f_i) \tag{3.6}$$

The final response P_R is based on the linear combination of discriminative and generative scores as follows:

$$P_R = (1 - \lambda) P_D + \lambda P_G \tag{3.7}$$

Here, λ is determined empirically.

3.3 Canonical Correlation Analysis model

Here we propose simple and effective models for image annotation that make use of Convolutional Neural Network (CNN) features extracted from an image and word embedding vectors to represent their associated tags. Our first set of models is based on the Canonical Correlation Analysis (CCA) framework that helps in modeling both visual features (CNN feature) and textual features (word embedding vectors) of the data. Results on all three variants of the CCA models, namely linear CCA, kernel CCA and CCA with k-nearest neighbor (CCA-KNN) clustering are reported. The best results are obtained using CCA-KNN which outperforms previous results on the Corel-5k and the ESP-Game datasets and achieves comparable results on the IAPRTC-12 dataset. In our experiments, we evaluate CNN features on existing models which bring out the advantages of it over dozens of handcrafted features. We also demonstrate that word embedding vectors perform better than binary vectors as a representation of the tags associated with an image. In addition we compare the

CCA model to a simple CNN based linear regression model, which allows the CNN layers to be trained using back-propagation.

3.3.1 Feature Extraction

Here, we provide details about how the CNN features are extracted from images, followed by details on how to use word embedding vectors to represent the tags.

3.3.1.1 CNN features

Given an image, we extract a 4096-dimensional feature vector (X) (last fully connected layer before softmax) using a pre-trained CNN on the ILSVRC-2012 dataset as described in Simonyan et al. [116]. We explored both VGG-16 and VGG-19 layered architecture features (more details in section 2.1.4). Since both of them gave similar results, we used VGG-16. Features extracted from Caffe-Net provided by Caffe [59] (similar to AlexNet [65]) did not work as well as VGG-16, hence we used VGG-16 features for all our experiments. The features are computed by forward propagating a mean-subtracted 224x224 RGB image through eight convolutional layers and three fully connected layers. In our case, we resize all the images irrespective of their aspect ratio to 224x224 to make it compatible with the CNN.

3.3.1.2 Word embeddings

For each tag associated with an image, we represent the tag (word) by a 300 dimensional real valued feature vector using a Word2Vec tool and we call it as a word embedding vector ($E \in R^{l \times q}$), where l is the number of labels and $q = 300$ dimensions. These word vectors are obtained from a pre-trained skip-gram text modeling architecture introduced by Mikolov et al. [87]. It was shown that the model learns similar embedding vectors for semantically related words. Therefore, we use it to represent the annotations. We take the average of all the word embedding vectors (Y) associated with multiple tags representing an image. Formally, if there are k tags associated with an image I then $Y = \frac{1}{k}\sum_{i=1}^{k} E_i$ and their association is represented as $\{I, Y\}$. While reporting the result, we refer to word embedding vectors as W2V.

3.3.2 Proposed Method

Here we present the details of our proposed model and its variant.

3.3.2.1 Canonical Correlation Analysis (CCA)

Given a pair of views for an image – a visual feature (X, i.e., CNN feature) and a textual feature (Y, word embedding vector), CCA computes projections w_x and w_y for X and Y respectively to maximize their correlation. Concretely, for M samples, let $X \in R^{m\times p}$ and $Y \in R^{m\times q}$ be the two views of the data, then, the projection vectors w_x *and* w_y are computed by maximizing the correlation coefficient ρ

$$\rho = \underset{w_x, w_y}{\arg\max} \frac{w_x^T XY^T w_y}{\sqrt{(w_x^T XX^T w_x)(w_y^T YY^T w_y)}} \tag{3.8}$$

The dimensionality of these new projection vectors is less than or equal to the smallest dimensionality of the two variables. The canonical correlations are invariant to affine transformations of the variables. The solution is found by formulating it as a generalized eigenvalue problem [46]:

$$XY^T(YY^T)^{-1}YX^T w_x = \eta XX^T w_x \tag{3.9}$$

where η is the eigenvalue corresponding to the eigenvector w_x. Thus, multiple projection vectors can be found which form a projection matrix $W_x \in R^{p\times l} \in$ and similarly $W_y \in R^{q\times l}$. where, l is the number of eigenvectors corresponding to the top l eigenvalues. In the case of regularized CCA (rCCA), L2 regularization is used and it constrains the norms of canonical weights w_x and w_y and thus avoids overfitting. Thus for rCCA we have:

$$\rho = \arg\max_{w_x,w_y} \frac{w_x^T XY^T w_y}{\sqrt{(w_x^T XX^T w_x + \|\lambda w_x\|^2)(w_y^T YY^T w_y + \|\lambda w_y\|^2)}} \tag{3.10}$$

The following is the generalized eigenvalue problem of rCCA:

$$XY^T(YY^T)^{-1}YX^T w_x = \eta(XX^T + \lambda I)w_x \tag{3.11}$$

3.3.2.2 Kernel CCA (KCCA)

Since CCA can only capture linear relationships, we propose to use a χ^2 kernel for exploiting non linear relationships. The χ^2 kernel was found to be well suited in our experiments. The visual feature X is mapped to a high dimensional feature space $\mathcal{H}_x$ using a function ϕ_x. The ϕ_x mapping is achieved using a positive definite kernel function $K_x = \langle \phi_x, \phi_x \rangle \in R^{m\times m}$, where $\langle :, : \rangle$ is an inner product in $\mathcal{H}_x$. Similarly, the word embedding vector Y is mapped to $\mathcal{H}_y$ using the kernel function $K_y = \langle \phi_y, \phi_y \rangle \in R^{mxm}$. Kernel CCA finds the solution of w_x and w_y as a linear combination of the training data:

$w_x = \sum_{i=1}^{m} \alpha_i \phi_x(x_i)$ and $w_y = \sum_{i=1}^{m} \beta_i \phi_y(y_i)$. Since feature vector dimensions are large, overfitting is an issue. To avoid this, we used a regularized kernel CCA [46] which finds $\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}$ by maximizing the following objective function that involves penalizing the norms of the projection matrix:

$$\arg\max_{\boldsymbol{\alpha},\boldsymbol{\beta}} \frac{\boldsymbol{\alpha}^T K_x K_y \boldsymbol{\beta}}{\sqrt{(\boldsymbol{\alpha}^T K_x^2 \boldsymbol{\alpha} + r_x \boldsymbol{\alpha}^T K_x \boldsymbol{\alpha})(\boldsymbol{\beta}^T K_y^2 \boldsymbol{\beta} + r_y \boldsymbol{\beta}^T K_y \boldsymbol{\beta})}} \tag{3.12}$$

where, $r_x \boldsymbol{\alpha}^T K_x \boldsymbol{\alpha}$ and $r_y \boldsymbol{\alpha}^T K_x \boldsymbol{\alpha}$ are the additional partial least square term added to the KCCA for regularization. The solution yields top l eigenvectors $\mathcal{W}_x = [\boldsymbol{\alpha}^1 \ldots \boldsymbol{\alpha}^l]$ and $\mathcal{W}_y = [\boldsymbol{\beta}^1 \ldots \boldsymbol{\beta}^l]$ which form the projection matrix.

3.3.2.3 Implementation details

CCA and KCCA with regularization were implemented as explained in [46]. Regularization was found to be important to avoid overfitting resulting in better performance. In the case of linear CCA, we project X onto W_x, project Y onto W_y and project E onto W_y:

$$U = (X - \mu_X)W_x \, , \, V = (Y - \mu_Y)W_y \text{ and } Z = EW_y \tag{3.13}$$

Given a test image I_t, we extract deep learning visual features X and project it using W_x as $U = (X - \mu)W_x$ and compute the correlation distance to V using $d = 1-(\frac{(U-\mu_U)(V-\mu_V)}{\sqrt{(U-\mu_U)(U-\mu_U)}\sqrt{(V-\mu_V)(V-\mu_V)}})$. The corresponding tags associated with the closest matching V_i (i^{th} training sample with lowest distance value) are assigned to the test image (tags are also ranked according to their frequency in the training dataset). If the tags are less than the fixed annotation length, we pick the next closest match and transfer the tags, we repeat this until we obtain the required set of tags - in our case its five (to compare with previous work).

Similarly, in the case of KCCA, we kernelize X, Y and Z and later project onto $\mathcal{W}_x, \mathcal{W}_y, \mathcal{W}_x$ respectively. For a test image, we kernelize the visual features and follow the same procedure as above.

In CCA with KNN clustering (CCA-KNN) setup, we initially create clusters of images grouped according to its labels. The resulting clusters will be overlapping and the number of clusters will be equal to the number of labels. The only difference from our CCA/KCCA implementation is that, after finding the correlation distance of U with V, we choose K semantic neighbor samples from each cluster for that particular test image and now all their associated tags form a subset of tags Z_k (potential candidates for a test image). Later, we rank the words w for a test image I_t according to the probability score:

$$P(I_t|w) = \sum_k \exp(-D(U, Z_k))1_k(w) \tag{3.14}$$

where, $D(U, Z_k)$ is the correlation distance between U and Z_k and $1_k(w)$ is an indicator function which takes a value 1 if the tag is present among neighbors and 0 otherwise.

3.3.3 CNN-based regression model

Inspired by the success of deep CNN architectures [65, 116, 35] on the large scale image classification task, we use it to solve the task of automatic image annotation. To the best of our knowledge, this is the first attempt to formulate this problem based on a CNN. The idea is to formulate the problem as a linear regression. We achieve this by replacing the last layer of Caffe-Net (very similar to AlexNet except that pooling is done before normalization) with a projection layer (fully connected layer) and we call it as a CNN regressor (CNN-R). CNN provides the mapping function which regresses the fixed size of the input image to a word embedding vector. The network consists of five convolutional layers and two fully connected layers with some series of non-linear transformation (rectified linear unit) and pooling layers. Most importantly, it uses some dropout layers in addition to avoid overfitting. For further architecture details, please refer to [65]. In this setup, we increased the learning rate for the newly introduced layer while reducing it for all the other previous layers, the reason being that we are trying to fine-tune the network previously trained on 1.2 million images. The input image size was fixed to be 227x227 and the final regressed output was a 300 dimensional vector. The output dimensional vector is 300 because we chose to represent the tag by a 300 dimensional real valued feature vector using Word2Vec and multiple tags associated with an image were handled by taking their average. Since we have chosen to do linear regression, we use Euclidean loss (L2) instead of Softmax loss during the training phase. The prediction layer tries to predict the word embedding vector by minimizing the L2 loss depending on which, the model parameters are updated using the back-propagation algorithm.

3.4 Hypergraph model

We propose to solve the automatic image annotation task using a novel multi-scale hypergraph heat diffusion framework. This enables us to first capture the higher order similarity among multiple images in the feature space and subsequently exploit the topology of the underlying hypergraph. Such topological analysis enables us to perform simultaneous diffusion of the training labels at multiple scales in the transductive setup thereby addressing the key problem of class imbalance (by diffusing under-represented labels at relatively large scale).

This is realized as follows: First, we model the higher order feature similarity among images using the nearest-neighbour hypergraph modelling. We use Convolutional Neural Networks (CNN) features as visual features. Secondly, we compute the spectrum of the associated hypergraph Laplacian matrix and use it to derive the hypergraph heat-kernel matrix. Third, we diffuse the training image labels using the heat-kernel matrix at multiple scales and infer the test labels. Finally, we provide empirical validation of the proposed technique.

3.4.1 Proposed Method

Here, we provide details of the key steps of novel multi-scale hypergraph heat diffusion (HHD) framework which addresses the class imbalance problem in the automatic image annotation task using a single CNN feature.

3.4.1.1 Feature Extraction

CNN feature are extracted as described in section 3.3.1.1.

Let $\mathbf{x}_i$ be the 4096-dimensional feature vector representing the i^{th} image, the entire dataset consisting of n images (including both training and testing sets) can be represented as:

$$\mathbf{X} = \big[\mathbf{x}_1, \ldots, \mathbf{x}_n\big].$$

We can rewrite this by separately representing the training set of images with $\mathbf{X}^{train}$ and the test set of images with $\mathbf{X}^{test}$ such that:

$$\mathbf{X} = \{\mathbf{X}^{train} \cup \mathbf{X}^{test}\}.$$

3.4.1.2 Hypergraph Construction

A hypergraph enables capturing more information using hyperedges by linking multiple nodes as compared to simple graphs where only dyadic relationships are captured by the edges. In the context of the image annotation task, we have adopted the hypergraph construction from [147] where each image is considered as a node. Each node has one (or multiple) corresponding hyperedge(s) which connect k nearest-neighbor nodes in the feature space (for varying values of k). The set of hyperedges stacked together is known as the incidence matrix of the hypergraph. Let,

$$\mathbf{\Pi} = \big[\mathbf{he}_1, \ldots, \mathbf{he}_p\big] \tag{3.15}$$

be the incidence matrix of the nearest-neighbour hypergraph induced on the image feature set $\mathbf{X}$. Here, each hyperedge $\mathbf{he}_i = \big[he_i^1, \ldots, he_i^n\big]^T$ is an indicator vector of size n where each element $he_i^j = 1$ if node $\mathbf{x}_j$ participate in hyperedge $\mathbf{he}_i$ or else zero. Thus, multiple 1's suggest that the respective nodes contribute to the same hyperedge. The total number of hyperedges (p) is a multiple of n.

3.4.1.3 Hypergraph Heat Diffusion (HHD) Framework

Traditionally, a Gaussian function is used as a convolution kernel for scale-space analysis of a scalar function defined over Euclidean domains (eg., images), where the scale parameter is associated with the variance of this function. The reason being that the Gaussian corresponds to the closed-form solution of the heat diffusion equation on Euclidean domains [28].

A similar kernel framework exists for simple graphs in the transductive setup where both training and test data points are treated as graph nodes. The heat-kernel is a non-linear (exponential) family of kernels, and for simple graphs it is derived from the spectra (constituted by both eigenvalues & eigenvectors) of the Laplacian graph matrix [112]. Thus, the heat kernel is a symmetric kernel (analogous to a Gaussian kernel) for non-Euclidean spaces represented as graphs and is used as a diffusion tool for multi-scale label or information diffusion on graphs [127].

Interestingly, the Laplacian for hypergraph was derived in [154] where it was shown to be analogous to a simple graph Laplacian. We extend this work to define the heat diffusion framework for hypergraphs using the spectra of hypergraph Laplacian. This framework enables multi-scale (topological) analysis of hypergraphs. Using the definition of hypergraph incidence matrix in Eq.3.15, the hypergraph Laplacian is subsequently defined [154] as:

$$\mathbf{L} = \mathbf{I} - \left(\mathbf{D}_v^{-\frac{1}{2}} \mathbf{\Pi} \mathbf{W}_{he} \mathbf{D}_{he}^{-1} \mathbf{\Pi}^T \mathbf{D}_v^{-\frac{1}{2}}\right) \tag{3.16}$$

where,

- $\mathbf{\Pi}$ is $n \times p$ incidence matrix of the hypergraph with p hyperedges,
- $\mathbf{D}_v$ is $n \times n$ degree matrix of nodes defined as $\mathbf{D}_v = diag\left(\sum_p \mathbf{\Pi}\right)$,
- $\mathbf{D}_{he}$ is $p \times p$ degree matrix of hyperedges defined as $\mathbf{D}_{he} = diag\left(\sum_n \mathbf{\Pi}^T\right)$,

- $\mathbf{W}_{he}$ is $p \times p$ hyperedge weight matrix defined as $\mathbf{W}_{he} = diag(w_1, \ldots, w_p)$.

The $\mathbf{W}_{he}$ matrix can be used to enforce the relative significance of certain hyperedges over others by setting larger values but throughout our experiment, we set it to uniform values. The eigen-decomposition of $\mathbf{L}$ matrix is written as:

$$\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T \tag{3.17}$$

where, $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_n]$ be the matrix formed by the eigenvectors of $\mathbf{L}$ matrix and, $\boldsymbol{\Lambda} = diag(\lambda_1, \ldots, \lambda_n)$ be the diagonal eigenvalue matrix, these together define the Laplacian spectra.

The hypergraph heat diffusion framework can be derived using the associated Laplacian spectra. The $n \times n$ heat-kernel matrix for hypergraph can be computed as:

$$\mathbf{H}(t) = \mathbf{U}\exp(-\boldsymbol{\Lambda}t)\mathbf{U}^T \tag{3.18}$$

where, t is the scale parameter which govern the heat diffusion. One can show that the $\mathbf{H}(t)$ matrix is indeed a kernel matrix as it satisfies Mercer's kernel property, i.e., it is a real, positive semi-definite matrix with real and positive eigenvalues ($\exp(-\boldsymbol{\Lambda}t)$ for $t \geq 0$). This property is based on the proof presented in [154] that the hypergraph Laplacian matrix ($\mathbf{L}$) is a positive semi-definite matrix.

In practice, a low rank approximation of the heat kernel matrix in Eq. 3.18 is computed using a subset of ($\ll n$) smallest eigenvalues/eigenvectors of the $\mathbf{L}$ matrix. However, it is important to be cautious while choosing the rank size because the rank of the heat kernel matrix and the scale of diffusion have a complex relationship empirically analysed in [112].

3.4.1.4 Multi-scale Label Diffusion & Inference

Here, we present the formulation for multi-scale diffusion of training image labels using the HHD framework to infer the labels for test images. The label diffusion over a graph in the transductive setup is accomplished by computing the pseudo-inverse of the graph Laplacian matrix [81]. A similar approach is adopted for the hypergraph variant [147]. This kind of diffusion automatically decides a fixed scale at which labels are diffused in the local neighbourhood over the (hyper-)graph.

To overcome this limitation, in our HHD framework, the scale of diffusion is governed by the parameter t (see Eq. 3.18). This provides an explicit control over the scale of diffusion that is more powerful in the sense that, it would allow one to diffuse over-represented and under-represented labels separately. Thus, this kind of setup helps in addressing the prevalent class imbalance problem in real data. Here, the value of t can vary from zero to infinity (approximated by a relatively large value). A small scale diffusion (t closer to zero) would enforce the label diffusion in the smaller neighbourhood while a large scale diffusion (t closer to infinity) would enforce the label diffusion in a very large neighbourhood.

Let, the ground truth labels for both train and test be represented as:

$$\mathbf{Y} = \left[\mathbf{y}_1, \ldots, \mathbf{y}_m\right]$$

where each $\mathbf{y}_i$ is an n-dimensional indicator vector (0's and 1's) for the multi-label annotation setup with label vocabulary of size m. Let $\mathbf{Y}^{train} \subset \mathbf{Y}$ be the set of known labels (training set) and $\mathbf{Y}^{test}$ be the complementary set of unknown labels (testing set). Thus, a scale dependent label diffusion can be accomplished as:

$$\mathbf{Y}_t = \mathbf{H}(t)\mathbf{Y}. \tag{3.19}$$

Here, $\mathbf{Y_t}$ is the resultant matrix after diffusion of labels.

Let $\mathbf{Y}^{OR} \subset \mathbf{Y}$ be the subset of labels from over-represented class and $\mathbf{Y}^{UR}$ be the complementary set of under-represented class labels. The multi-scale (ms) diffusion to address the class imbalance problem can be achieved by diffusing over-represented labels at t_{small} and under-represented labels at t_{large} as:

$$\mathbf{Y}^{OR}_{t_{small}} = \mathbf{H}(t_{small})\mathbf{Y}^{OR}, \tag{3.20}$$

$$\mathbf{Y}^{UR}_{t_{large}} = \mathbf{H}(t_{large})\mathbf{Y}^{UR}, \tag{3.21}$$

These diffused label matrices can further be combined to

$$\mathbf{Y}_{ms} = \mathbf{Y}^{OR}_{t_{small}} \bigcup \mathbf{Y}^{UR}_{t_{large}}$$

.

Here, $\mathbf{Y_{ms}}$ is the resultant matrix after diffusing the labels at different scales.

Finally, we select the subset of multi-scale diffused labels for test set images (i.e., $\mathbf{Y}^{test}_{ms} \subset \mathbf{Y}_{ms}$) apply multi-label inference by taking the q largest entries of each row of $\mathbf{Y}^{test}_{ms}$ for inferring q labels for each test image. However, before inferring test image labels, we propose to normalize $\mathbf{Y}^{test}_{ms}$ with $L1$-normalization using $\mathbf{Y}^{train}_{ms}$. This type of normalization further helps in addressing the class imbalance problem.

3.4.1.5 Implementation Details

In regard to hypergraph Laplacian parameter, we set W_{he} (hyperedge weight matrix) to the identity matrix thereby giving equal importance to all hyperedges. Thus, we did not exploit the formulation completely by using a cross validation based tuning of W_{he} matrix which could probably yield better results. This was done intentionally, because we wanted to report generalized results and therefore a tuning which can be regarded as overfitting to datasets was avoided. Instead of using all eigenvectors of the Laplacian matrix, we used only 10% of the smallest eigenvectors for constructing a low rank heat kernel matrix for computation efficiency reasons. The scale parameters were empirically chosen for each dataset.

3.5 Dataset

We evaluate on four standard publicly available image annotation datasets - Corel-5k [26], ESP-Game [136], IAPRTC-12 [136] and NUS-WIDE [19]. These datasets contain a variety of images such as natural scenes, games, sketches, transportation vehicles, personal photos and so on, thus making image annotation a challenging task.

3.5.1 Corel 5k

The dataset consists of 5000 images, among which 4500 are used for training and the remaining 500 images are used for testing [25]. The label vocabulary consisted of 260 labels used for image annotation. Each image is annotated with a varying number of labels from 1 to 5 and an average of 3.5.

3.5.2 ESP Game

It consists of 20,770 images in total. Images are annotated via an online gaming setup [84]. If the images are annotated with the same key words by two distinct players, then they score a point. The training dataset consists of 18,689 images and the test set consists of 2081 images. The image annotation vocabulary consists of 268 labels and on an average each image is annotated with 4.7 labels.

3.5.3 IAPRTC-12

It is a collection of 19,627 images of natural scenes which are split into a training set consisting of 17665 images and a testing set consisting of 1962 images [40]. The label vocabulary consists of 291 labels with an average of 5.7 labels used for annotating each image.

3.5.4 NUS-WIDE

It's a web-image dataset that originally consists of 269,648 images and 5018 tags from Flickr. Amongst them, 223,821 image links were active on Flickr and hence we ended up using only them. We follow the exact same train (134,281 images) and test (89,603 images) split as provided by [151]. There are three sets of annotations/tags assigned to these images. The first set of tags are very noisy and consist of a lot of rare tags that account for nearly 5000 tags. The second set consists of 1000 tags after removal of some noisy and rare tags. The third set comprises 81 manually annotated tags with relatively less noise. These 81 concepts were carefully chosen from Flickr such that they are among the frequently occurring tags and they have both general concepts such as "vehicle" and specific concepts such as "statue" and "dancing". Each of these tags belong to different set of categories.

Table 3.1: Statistical details of the datasets used in this study. In labels per image column, the mean/max values are provided. In the distribution of labels column, label frequency greater than mean frequency (over-represented)/label frequency less than mean frequency (under-represented) are provided.

Dataset	Number of images	Vocab. size	Training images	Test images	Labels per image	Images per label	Distribution of labels
Corel-5K	5,000	260	4,500	500	3.4/5	58.6	195(75%)/65(25%)
ESP Game	20,770	268	18,689	2,081	4.7/15	362.7	201(75%)/67(25%)
IAPRTC-12	19,627	291	17,665	1,962	5.7/23	347.7	217(74.6%)/74(25.4%)
NUS-WIDE	223,821	81	134,281	89,603	1.7/20	2428.7	57(70%)/24(30%)

3.6 Evaluation Metric

We follow the standard evaluation metrics as reported in most of the previous work [91, 90, 132, 84, 27, 67]. Each test image is annotated with a fixed number of five labels. For a given label, let α be the number of images predicted, β be the number of images correctly predicted and let γ represent the number of images present in the ground truth set. Then, the recall and precision can be computed as $\frac{\beta}{\gamma}$ and $\frac{\beta}{\alpha}$ respectively. Finally, the average of recall (R) and precision (P) over all the labels are reported along with their first harmonic mean (F). In addition, $N+$ which represents the number of labels with non-zero recall value is also reported.

3.7 Experimental Results

In order to have a fair comparison with the previously reported results, we follow the same train and test split as reported in [41] and also fix the length of the annotations (five tags) for a test image. Table 3.2 provides the results of our proposed models on all three datasets- Corel-5K, ESP Game, and IAPRTC-12 in comparison with previously reported numbers. In addition, we also show the effectiveness of using CNN features in our proposed models as compared to their usage in some of the existing models, 2PKNN, JEC and TagProp. For reporting these results, we implemented the JEC method as described in [84], for 2PKNN [132] (reported results are obtained using the default parameter values, $k = 4$ and $w = 1$) and TagProp [42] we made use of the code provided by the authors. In the table 3.2, P represents the average precision, R represents the average recall, and N+ represents the non-zero recall (number of distinct words that are correctly assigned to the test image set).

3.7.1 Quantitative Analysis

SVM-DMBRM with handcrafted features performance is consistently better in terms of N+ measure and it is comparable to state of the art in F measure to all

Table 3.2: Experimental results of our proposed models with previously reported best scores on all three datasets. P: Average Precision, R: Average Recall, N+: Number of distinct words that are correctly assigned to at least one test image.

	Feature		Corel-5K				ESP Game				IAPRTC-12			
Method	Visual	text	P	R	F	N+	P	R	F	N+	P	R	F	N+
JEC [83]	HC	-	27	32	29	139	22	25	23	224	28	29	29	250
MBRM [27]	HC	-	24	25	25	122	18	19	19	209	24	23	24	223
TagProp(σML) [41]	HC	-	33	42	37	160	39	27	32	239	46	35	40	266
2PKNN [132]	HC	-	39	40	40	177	**51**	23	32	245	**49**	32	39	274
2PKNN+ML [132]	HC	-	**44**	46	45	191	**53**	27	36	252	54	37	**44**	**278**
KCCA-2PKNN [7]	HC	-	**42**	46	44	179	-	-	-	-	**59**	30	40	259
SKL-CRM [90]	HC	-	39	46	42	184	41	26	32	248	47	32	38	274
JEC	VGG-16	-	31	32	32	141	26	22	24	234	28	21	24	237
2PKNN	VGG-16	-	33	30	32	160	40	23	29	250	38	23	29	261
TagProp (σ)	VGG-16		30	35	32	149	31	28	30	246	38	30	34	260
Below are our models														
SVM-DMBRM	HC	-	36	48	41	**197**	55	25	34	**259**	56	29	38	**283**
SVM-DMBRM	VGG-16	-	42	45	43	186	51	26	35	251	58	27	37	268
HHD	VGG-16	-	31	49	38	194	35	**36**	34	257	32	**44**	36	280
CCA	VGG-16	W2V	35	46	40	172	29	32	30	250	33	32	33	268
KCCA	VGG-16	W2V	39	**53**	**45**	184	30	**36**	33	252	38	**39**	38	273
CCA-KNN	VGG-16	BV	39	51	44	192	44	32	**37**	254	41	34	37	273
CCA-KNN	VGG-16	W2V	**42**	**52**	**46**	**201**	46	**36**	**41**	**260**	45	38	**41**	**278**
CNN-R	Caffe-Net	W2V	32	41	37	166	45	29	35	248	**49**	31	38	272

other previous work. N+ is a measure of how well the system performs with the imbalanced positive example problem and also it indicates the number of distinct words that were used for annotating the test images.

The experimental results of using our proposed models (SVM-DMBRM, Hypergraph and CCA) with CNN features are comparable to the state of the art result (2PKNN) and in particular, CCA-KNN with single CNN feature yields a significant improvement over all the existing methods on Corel-5k and ESP-Game datasets. Our proposed methods with CNN features outperforms the TagProp's performance which specifically studied the usage of a bunch of hand crafted features and multiple metric learning (combining different weighted features) to yield better results. The current state of the art system (2PKNN) uses more than a dozen hand-crafted (HC) features which are computationally expensive and serve as a bottleneck for use in large scale applications. Here we show that one single CNN feature per image is scalable and effective for image annotation task. Also, we overcome the difficulty of choosing the best set of features and finding the best feature score fusion techniques. Interestingly, the CNN features used in JEC, TagProp and 2PKNN methods perform poorly in terms of both F and N+ measures when compared to even using the same techniques with traditional features. This is presumably because the strength of the techniques (JEC, TagProp and 2PKNN) depends on using metric learning with multiple features and in the case of single CNN feature, it lacks that advantage. This demonstrates that the improvement in the performance is attributed to our proposed models and not just the feature alone.

Table 3.2 also shows that word embedding vectors (W2V) work better than binary vectors (BV) with our best performing model CCA-KNN. This suggests that word embedding vectors provide a better representation for words than their binary form presumably because semantically related words tend to have similar word embedding vectors.

From Table 3.2, we see that CNN-R outperforms JEC and MBRM but not our best performing CCA-KNN model. CNN-R is competitive to TagProp and has a clear advantage over all the existing methods for the following reasons: (a) no need to extract multiple low level features and to incorporate high level semantics (b) no metric learning is required (c) can use transfer learning technique [111] for smaller datasets and (d) has the ability to generalize to unseen classes with the help of word embeddings vectors [120].

3.7.2 Results on NUS-WIDE dataset

To verify the scalability of our proposed models, we test it on one of the largest dataset (NUS-WIDE). Since the characteristics and distribution of NUS-WIDE is similar to the other three datsets (Corel-5K, IAPRTC-12, ESP-Game), we believe that different model and feature combinations would yield similar performance trends and hence we decided to experiment with only the best performing feature and model combination instead of trying out all other possibilities.

From Table 3.3, we clearly see that among our proposed models, CCA-KNN outperforms all other models. This is consistent with the earlier results on smaller datasets. In general, this data also suffers from skewed class label distribution and as a result finding a smaller subset of data (approx. balanced) in the first step of CCA-KNN method helps in boosting the performance as compared to CCA. Most popular methods like TagProp and 2PKNN underperform. One of the possible reason would be that, without multiple features and metric learning it just turns out to be a simple nearest neighbor technique. SVM-DMBRM and HHD model performance is better than TagProp and 2PKNN but it is still behind CCA-KNN method. This assures that our proposed model was able to make use of the CNN feature effectively and the improvement is not solely attributed to the CNN feature. Almost all the methods have same N+ score (using all the tags in the vocabulary) for annotating the test images, probably because it is relatively a small vocabulary.

The success of the CNN feature paved the way for us to further explore new deep learning architectures to solve the image annotation problem more effectively. In order to achieve this, we propose an alternative approach called Deep Decision Network (DDN) to build an efficient CNN architecture which is completely data-driven. We would like to first test its effectiveness in the order of difficulty of the problem - binary, multi-class classification and then further extend the framework to solve more challenging multi-label classification problem. The details are provided in the following chapters.

3.7.3 Qualitative Analysis

Figure 3.1 on page 53 provides some examples of randomly sampled images from all four datasets. These images are all automatically annotated with CCA-KNN (best among all our proposed models) method. The labels in green (bold) are correctly matched with the groundtruth labels, marked in blue are the semantically meaningful labels that are missing in the groundtruth, the labels marked in black (normal text) are the ones which our model failed to predict because of the fixed annotation length restriction and the labels in red color are predicted incorrectly by our model. We can clearly see that some images are poorly annotated (missing labels) but our method is still able to retrieve those semantically meaningful labels. In the other case, since we are restricted to a fixed length of annotation (five per image), our model might miss some of the labels present in the groundtruth.

Table 3.3: Experimental results of our proposed models with previously reported best scores on relatively large dataset (NUS-WIDE). P: Average Precision, R: Average Recall, N+: Number of distinct words that are correctly assigned to at least one test image.

	Feature		NUS-WIDE			
Method	Visual	text	P	R	F	N+
JEC	VGG-16	-	19	39	25	81
2PKNN	VGG-16	-	18	44	26	81
TagProp (σ)	VGG-16		18	41	25	81
Below are our proposed models						
SVM-DMBRM	VGG-16	-	23	58	33	81
HHD	VGG-16	-	21	49	29	81
CCA	VGG-16	W2V	22	55	31	81
CCA-KNN	VGG-16	W2V	**26**	**64**	**37**	81

3.8 Model parameters and its effectiveness

Model parameter settings largely affect the performance of the proposed models. Hence, in all our experiments the parameters are set using the validation dataset. Here we study the effects of varying different parameters in our proposed models and also try to understand it's significance.

3.8.1 Parameters of SVM-DMBRM model

3.8.1.1 Varying parameter λ

In Figure 3.2 we provide the precision, recall, and N+ scores for different values of λ. This study shows the unique capability of the model to get the desired performance for fixed number of annotation words just by varying the λ parameter. When the λ value is close to 0, SVM models dominate the final scores. On the other hand, if the parameter is close to 1, then the DMBRM models dominate the final scores. The precision score decreases when λ gets larger. The Recall scores increase when λ gets larger. In our experiments, we set the λ parameter to a value of 0.5 (provides good balance for recall and precision) which was determined based on the validation dataset.

3.8.2 Parameters of Hypergraph based model

3.8.2.1 Varying parameter k

Figure 3.3 provides the effects of varying k neighborhood parameter during the construction of the hypergraph. The plot shows the performance of our method in terms of F measure for varying sets of k. Parameter k on x-axis is varied in the set of powers of 2, like {4},{4,8},{4,8,32} and so on. The best results obtained here are the ones reported in the Table 3.2. We can see that our method is able to take advantage of multiple neighborhood sizes while constructing the hypergraph. However, the performance is too low for very small size of k and it reaches a plateau or decreases after a point when k is relatively large. In the former case, the hypergraph tends to capture less information (relationship between the images) while in the latter case, this might introduce large noise in the hypergraph construction.

3.8.3 Parameters of CCA based model

3.8.3.1 Varying parameter K

The effects of varying parameter K is shown in Figure 3.4. If the number of neighbors K chosen in the first stage of CCA-KNN model happens to be small, then the subset of images with their associated tags might not be sufficient for tagging the unknown image in the second stage. In other words, it would not generalize well to the unknown images and hence the test performance is low. In the case of K being too large, then it's equivalent to just the CCA method without KNN, which considers all the training images for annotating the test image. Hence, it's crucial to determine the right neighborhood size based on the validation dataset. In our experiments, we chose K to be 4, 2, and 2 for Corel-5k, ESP-Game and IAPRTC-12 datasets respectively.

3.8.3.2 Importance of word embeddings

Table 3.2 also shows that word embedding vectors (W2V) work better than the binary vectors (BV) with our best performing model CCA-KNN. This suggests that the word embedding vectors provide a better representation for words than their binary form, presumably because semantically related words tend to have similar word embedding vectors.

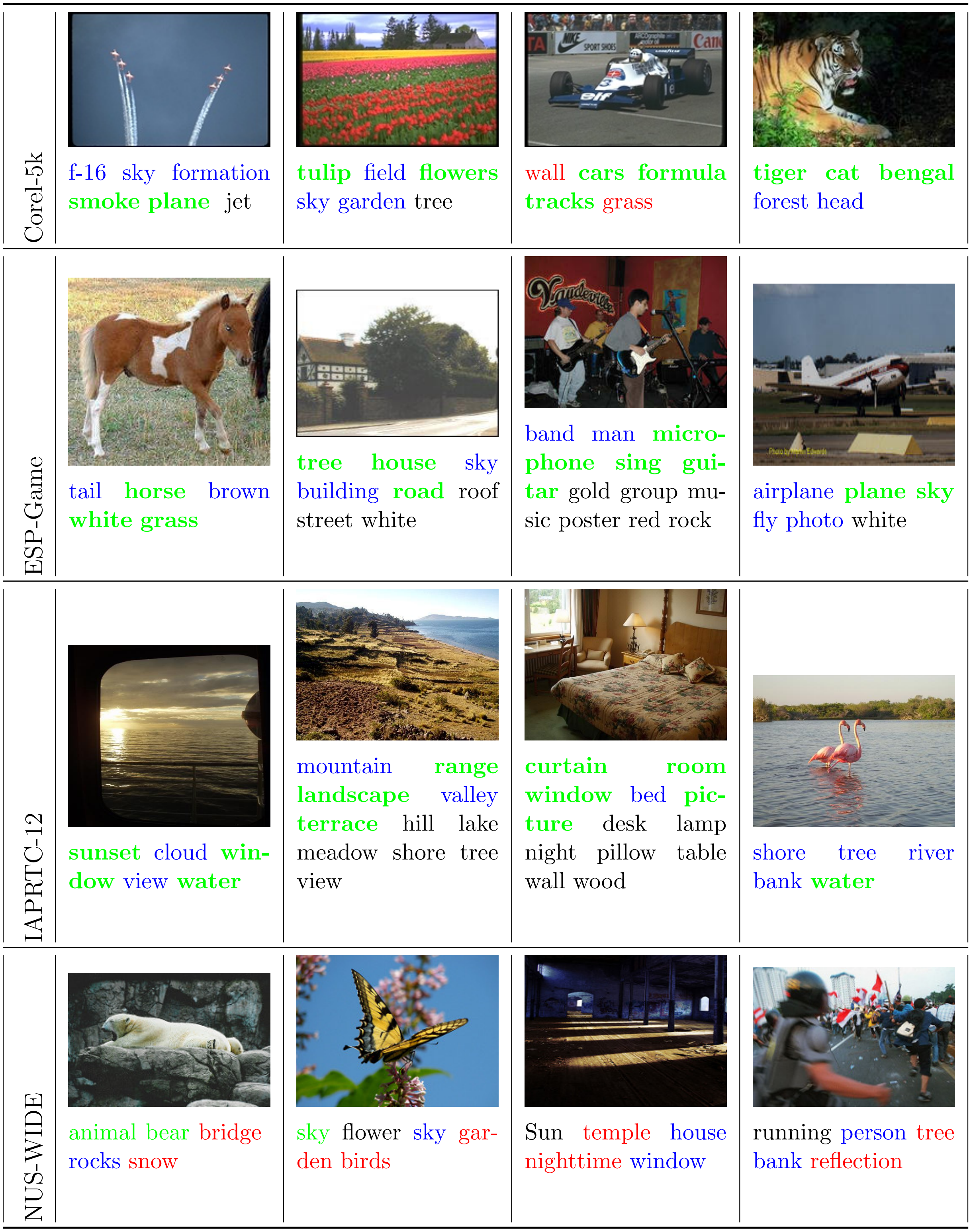

Corel-5k
f-16 sky formation smoke plane jet
tulip field flowers sky garden tree
wall cars formula tracks grass
tiger cat bengal forest head
ESP-Game
tail horse brown white grass
tree house sky building road roof street white
band man microphone sing guitar gold group music poster red rock
airplane plane sky fly photo white
IAPRTC-12
sunset cloud window view water
mountain range landscape valley terrace hill lake meadow shore tree view
curtain room window bed picture desk lamp night pillow table wall wood
shore tree river bank water
NUS-WIDE
animal bear bridge rocks snow
sky flower sky garden birds
Sun temple house nighttime window
running person tree bank reflection

Figure 3.1: Examples of randomly sampled images which are automatically annotated with CCA-KNN model. First row: Corel-5k, second row: ESP-Game, third row: IAPRTC‘-12 and fourth row: NUS-WIDE datasets. correctly matched with the groundtruth labels, semantically meaningful labels that are missing in the groundtruth, labels which our model failed to predict and predicted incorrectly.

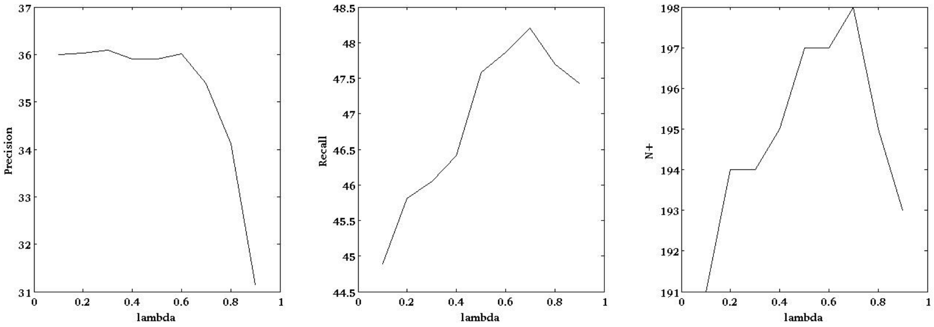


Figure 3.2: Precision, Recall, and N+ scores with different λ (lambda) values for SVM-DMBRM model.

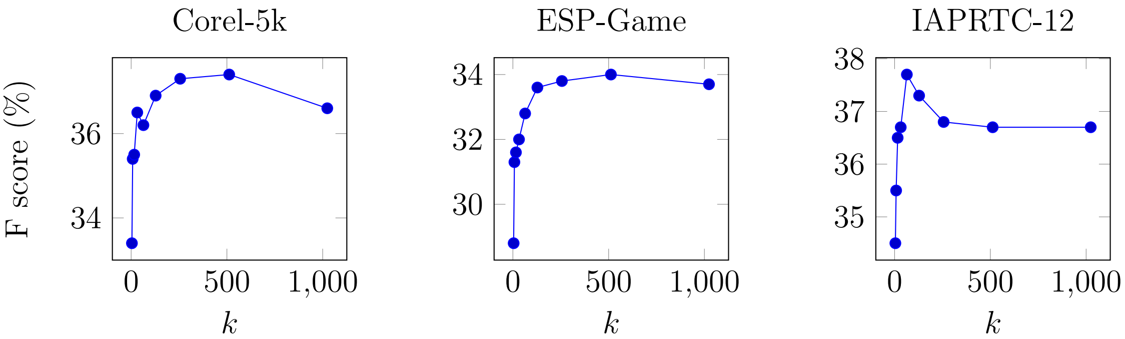


Figure 3.3: Performance of hypergraph based model for varying k neighborhood size parameter for all three datasets.

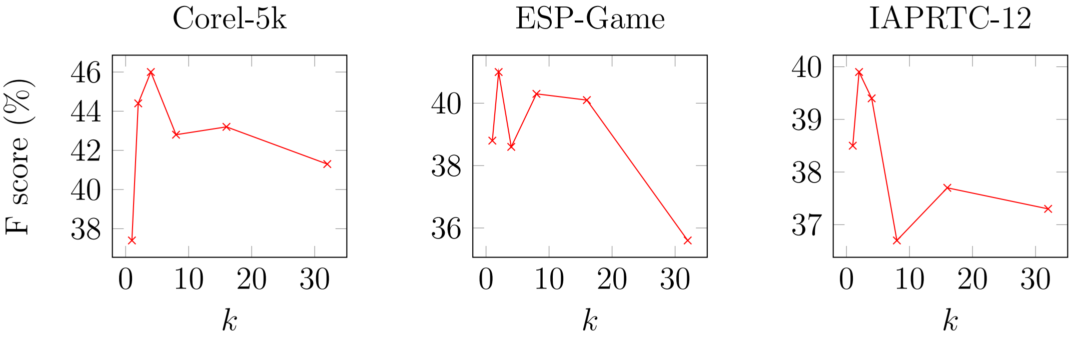


Figure 3.4: Performance of CCA-KNN model against varying K neighborhood size parameter (appears in the first stage of CCA-KNN) for all three datasets.

CHAPTER 4

CASCADED DEEP DECISION NETWORK (CDDN) FOR BINARY CLASSIFICATION - CLASSIFICATION OF ENDOSCOPIC IMAGES

Both traditional and wireless capsule endoscopes can generate tens of thousands of images for each patient. It is desirable to have the majority of irrelevant images filtered out by automatic algorithms during an offline review process or to have an automatic indication for highly suspicious areas during an online guidance. This also applies to the newly invented endo-microscopy, where online indication of tumor classification plays a significant role. Image classification is a standard pattern recognition problem and is well studied in the literature. However, performance on the challenging endoscopic images still leaves room for improvement. Here we present a novel Cascaded Deep Decision Network (CDDN) to improve image classification performance over standard deep neural network-based methods. During the learning phase, CDDN automatically builds a network which discards samples that are classified with high confidence scores by a previously trained network and concentrates only on the challenging samples that are to be handled by the subsequent expert shallow networks. We validate CDDN using two different types of endoscopic imagery- a polyp classification dataset and a tumor classification dataset. For both datasets we show that CDDN can outperform other methods by about 10%. In addition, CDDN can also be applied to other image classification problems.

4.1 Introduction

Endoscopic image analysis continues to play a quintessential role in visual diagnosis of medical conditions originating primarily in the gastrointestinal, respiratory, or other vital tracts of the human body. Early and precise detection of a plethora of these medical conditions can increase the chances of survival of an ailing patient through appropriate clinical procedures. For example, the relative 5-year survival rate for Colorectal Cancer when diagnosed at an early Polyp stage before it has spread, is about 90% [10]. Similarly, Meningioma, a benign intra-cranial tumor condition occurring in approximately 7 of every $100,000$ people [73], if detected early, can be treated surgically or by radiation, thereby drastically reducing the chances of growth and potential transformation to malignancy.

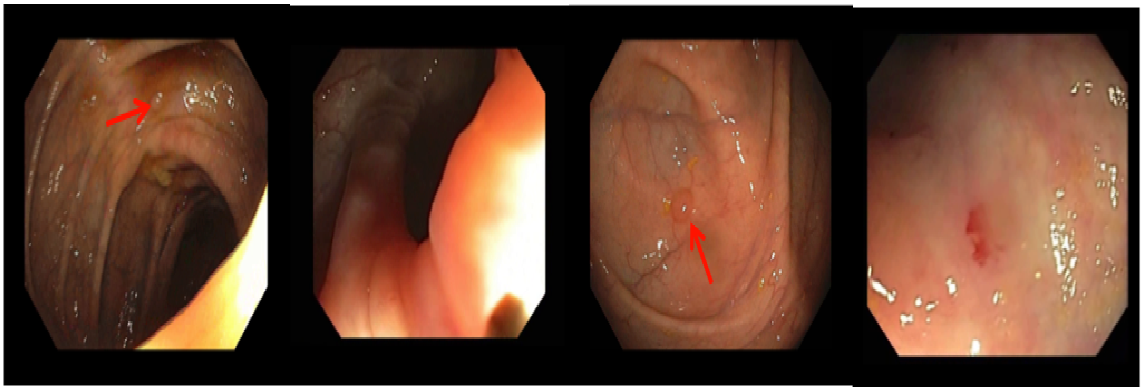


Figure 4.1: Sample images from the Polyp Classification Dataset obtained during a typical colonoscopic examination. Note the translucent blob-like shapes (pointed by arrows in red color) are colon polyps.

Currently, clinicians visually scan endoscopic images usually captured through electro-optical probes, for abnormal cell or tissue growth in the region under observation. Such manual screening procedures can often become tedious as a single probe typically generates a multitude of images. Furthermore, since the screening relies heavily on the skill sets of the clinician in charge, cases of missed detection are not uncommon. This emphasizes the need for Computer-aided Diagnostic (CAD) solutions that cannot only efficiently minimize human effort required while screening a large fraction of negative cases, but also provide a reliable reference to the clinicians. In this work, we focus only on eliminating negative images and all the experimental results are reported based on this.

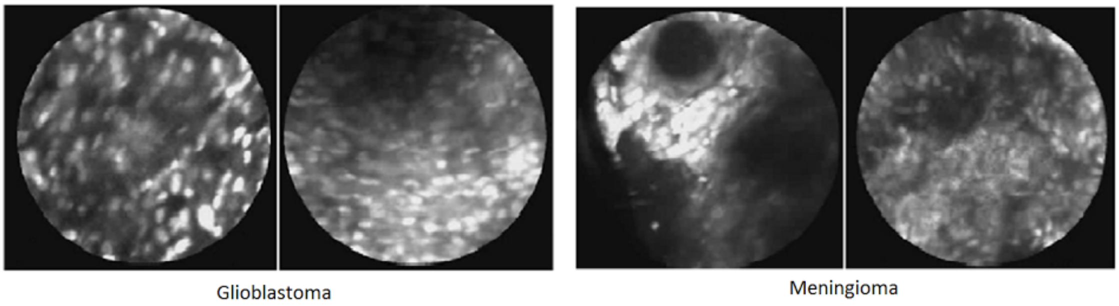


Figure 4.2: Sample Confocal LASER Endoscopic images from the Tumor Classification dataset with malignant Glioblastoma cases on the left and beningn Meningioma cases on the right. Note the sharp granular texture patterns in Glioblastoma cases.

In practice, each endoscopic procedure is specific to the medical condition and region of the body under observation. For example, within Capsule Endoscopy [88], an encapsulated wireless video camera is used to capture images from the gastrointestinal tract. In a different setting, neurosurgeons employ Confocal Laser Endomicroscopy (CLE) [99] probes as a surgical guidance tool to examine brain tissues for intracranial tumors. Although these application scenarios are vastly different, their fundamental objective involves searching for visually discriminative patterns that can be decisive for a binary classification task primarily to segregate positive from negative image samples.

More specifically, we focus on the following two tasks: 1) Filtering out images that do not contain polyps (polyps are visually translucent blobs in the GI tract as seen in Fig. 4.1). 2) Identifying malignant cases of brain tumors (Glioblastoma which is often identified by sharp granular patterns) from the benign ones (Meningioma which is characterized by smooth homogeneous patterns) in CLE images containing either of the two (refer to Fig. 4.2). Both of these scenarios have their own challenges - the former case has several challenges encountered by current computer vision systems like non-uniform illumination from light emitting diodes, noise from bubbles, bowel fluids, occlusion posed by anatomical complexity and large degrees of variation in shapes and size. The latter is limited by the low resolution of current CLE imagery, motion artifacts and often the presence of both kind of patterns in the probed area.

Automatic visual analysis of images pertaining to the aforementioned domains using conventional computer vision-based techniques has demonstrated reasonable success in the past. Most of these are based on variants of the Bag of visual Words (BoW) based computational frameworks owing to their simplicity of implementation. These methods [75, 6, 5, 152, 74] typically involve extraction of features from images, followed by a vector quantization step based on a pre-defined visual vocabulary (usually constructed by k-means clustering) which results in an intermediate compact representation of an image that can be ingested as a training sample for supervised classifiers. While these methods are effective, they consistently fail to leverage the data-driven aspect of the problem as all three steps of feature extraction, generation of intermediate representation, and finally the classification, are mutually independent.

Recently, deep learning based approaches [64], have demonstrated a significant performance boost on generic image classification tasks [23] by addressing the final classification objective in an integrated framework using layered neural networks. This has motivated many researchers to apply deep neural network-based methods in the field of medical image analysis [17, 15, 95, 16, 33, 153]. In an early work [69] pertinent to classification, the authors introduce a two-layer network which utilizes independent subspace analysis to reconstruct a natural representation of tumor images captured through cell microscopy.

In general, training networks for medical image classification tasks is a challenging task as it often requires thorough experimentation on large datasets. Due to the lack of a large amount of good quality training data, the trained network architecture often

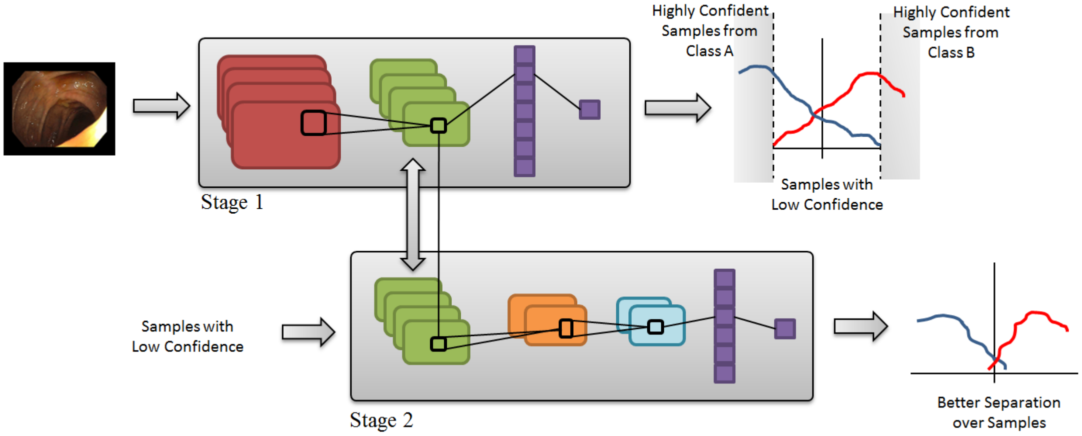

Highly Confident Samples from Class A
Highly Confident Samples from Class B
Samples with Low Confidence
Stage 1
Samples with Low Confidence
Stage 2
Better Separation over Samples

Figure 4.3: Cascaded Deep Decision Network (CDDN). For instance, stage-2 is built on top of conv layer(green color) of stage-1.

overtly optimizes itself for only training data, and performs poorly on unseen test samples. The authors of [8] avoid this issue by employing a pre-trained convolutional neural network [64] whose parameters are learned from a large database of images from non-medical cases [23]. Their research demonstrates high performance on a medical application of chest pathology detection in X-ray images. We argue that while such a pre-trained architecture has demonstrated success in a specific cross-domain exercise, the generalization aspect is still inconclusive. In this work, we propose a novel elegant computational framework called Cascaded Deep Decision Network (CDDN) to design an efficient network architecture with limited data but without over-fitting characteristics during the training process. In contrast to the existing deep learning based approaches, CDDN is built stage-wise during the learning phase. Our approach leverages a sampling strategy that discards samples classified with high confidence by a pre-trained network at the first-stage. Successive expert networks at different stages are trained, focusing on samples that are difficult to classify. This work is inspired by decision trees [103] and boosting[134, 107], which are both classical approaches in machine learning. Many variants of boosting trees have been explored and are shown to be successful for most of the vision tasks [134, 107, 135]. The fundamental concept of cascading is early rejection of the majority of test examples and it has been widely utilized to achieve real-time performance. Hence, we provide an efficient and effective approach to utilize this concept in the context of deep learning. Specifically our contributions are as follows: (a) piece-wise training strategy helps alleviate problems encountered by gradient based methods used heavily in contemporary deep learning research, (b) The proposed network architecture can help make an early decision thereby significantly reducing the computational time without compromising on the performance, (c) the data-driven design of CDDN offers an insight into the underlying structure of the data and finally (d) we demonstrate the effectiveness of our approach through rigorous experiments on two extremely challenging endoscopic image classification tasks.

From the philosophical perspective, our proposed approach has some similarity with ensemble methods commonly used in machine learning [29, 30]. However, a majority of these approaches encounter difficulties rejecting outliers in the presence of noisy training data. The sample selection strategy in CDDN, facilitates circumventing this issue early on, thereby not affecting the final performance of the network. To the best of our knowledge, this is the first work that introduces flavors of cascading deep networks [124, 22] into computer aided diagnosis of two crucial medical imaging applications.

4.2 Methodology

Given a classification problem, training a performant deep network is a difficult task since there are no well established guidelines to design the network architecture. Thus, training a network involves thorough experimentation and statistical analysis. Although going deeper in the neural network design has been shown to be effective [125], it increases the risk of over-fitting. Furthermore, as we experiment with the network architecture during the training process, it is difficult to leverage the results of the network trained in the previous iteration. To this end, we propose an alternate learning strategy to learn a deep neural network which allows building on and taking advantage of previous training experiments.

4.2.1 Cascaded Deep Decision Network (CDDN)

A cascaded deep decision network is a multi-stage deep neural network with decision stumps at each stage to classify easily separable data earlier in the network. An overview of the CDDN computational framework is provided in the Figure 4.3.

Given a dataset, a stage-1 (root) network is trained using the back propagation algorithm. Instead of optimizing the network to obtain the best performance, we only need to optimize until a reasonable performance is achieved e.g. 60-70%. Alternatively, a pre-trained network can be used as a stage-1 network if it achieves reasonable performance. The samples classified with high confidence are no longer considered for subsequent training. Further, a stage-2 network is trained to correctly classify the previously misclassified samples and/or the samples classified with low confidence; note that the stage-2 network is only optimized on a subset of the training data which was considered difficult by stage-1. This has the effect that as we go deeper we continue to “zoom-in” on resolving the problem cases. This stage-wise process is then continued until the desired performance is achieved.

There are several key differences between the CDDN architecture and the traditional deep networks. For instance, as we go deeper, the newly introduced layers gets trained only on a subset of the data. All the layers in previous stages are frozen while training the current stage. Furthermore, each subsequent stage builds on the feature space that was trained in the previous stage. Note that the subsequent stage can also be trained starting from any layer of the previous stage, which can be determined using a cross validation data set.

4.2.2 Piece-wise training for CDDN

The proposed architecture is trained in a unique fashion - Starting with a root network which is trained in a traditional way, we use the softmax layer to compute its performance and learn a threshold of confidence score for classification using cross-validation. The cross validation at each stage of the network is setup as follows: in the first stage, the training data is split into training (90%) and validation (10%) sets, while the network gets trained on the training set, the confidence score is determined using the validation set. For the next stage training, we combine both the training and validation sets of the previous stage and create a new split to continue the training process. This way we make sure that the entire training dataset gets utilized for training and the threshold value at each stage is determined based on the unseen samples which come into effect during testing.

At each stage, the samples with a confidence value below a threshold value are considered to be hard samples or confusion cases. These will be handled by the subsequent expert network which could be as simple as a single layer or a composition of multiple convolutional layers along with fully connected layers. In this work, we consider a shallow network as the expert network consisting of a convolutional and two fully connected layers along with some non-linearity and dropout layers. We continue to train the subsequent network layers using only the hard samples. While we do this, we completely freeze the previously trained layers. In other words, we set the learning rate of the previously trained network to zero and only train the newly added layers. This process can be recursively implemented until there are no more hard samples in the training dataset or until the desired depth of the network is met. This way, we are able to make use of the efforts of all the previous layers and also have the benefit of making an early decision based on the confidence score (provided by the softmax layer). The proposed training helps in overcoming the over-fitting problem during the training of expert shallow networks which concentrates only on the subset of the entire dataset. In addition, it also helps in avoiding local minima during gradient optimization and most importantly it provides better generalization, which is validated by our experimental evaluations.

4.2.3 Classification using CDDN

Given an image, we feedforward it through the first stage of the CDDN and obtain the confidence score from the softmax layer. If the score is higher than the threshold value (determined during the training process) then we declare it as the final output. If not, we move onto the next stage in the network and repeat the process until the last layer to get the final response. Mathematically,

$$f(I) = \begin{cases} y & \text{if } \ (\hat{I}_{s_{j=1}} = f_{s_{j=1}}(I)) > T_{s_{j=1}}\{i\} \\ y & \text{if } \ (\hat{I}_{s_{j=2}} = f_{s_{j=2}}(\hat{I}_{s_{j=1}})) > T_{s_{j=2}}\{i\} \\ & \vdots \\ y & \text{else } \ (\hat{I}_{s_{j=n}} = f_{s_{j=n}}(\hat{I}_{s_{j=n-1}})) \end{cases}$$

where the above mentioned parameters are defined as follows: I: input image, y: predicted label, s_j: different stages of the network and $j \in 1 \ldots n$, n: number of stage, $f(.)$: embedding function representing the network that predicts class labels with confidence, $\hat{I}$: embedded image and $T_{s_j}\{i\}$: threshold of a class label i at stage s_j.

4.2.4 Experimental Validation on MNIST digits

To validate CDDN and to provide more insight, we carried out a simple binary classification of digits '6' and '8' from the MNIST dataset [71]. The training set consists of 11769 samples and the testing set has 1932 images. Here we considered LeNet as our starting stage-1 network and for every subsequent stages we added a convolution layer and a fully connected layer (going deeper but to handle only a subset of the data which are considered to be the hard ones). In Figure 4.4 we can see that for the stage-1 network, 11,522 samples in the training set and 1884 samples in the testing set are classified with high probability (i.e., easy samples) and the remaining 247 samples of the training set and 48 of the testing set are harder to discriminate. For the harder samples, we build an expert network (i.e, stage-2) on stage-1's feature space. Since the resulting network is data-driven, the stopping criterion for network-growth is when the subsequent network fails to discriminate or there are very few training samples left out. The hard samples resulting from stage-1 and subsequent layers are shown in Figure 4.5. We can clearly see that stage-1 had some confusion cases which were resolved by the subsequent stage-2. Hence, in addition to improving the classification, the proposed approach provides some insight into the distribution of the samples.

4.3 Network Architecture and Implementation Details

In this section, we provide all the required implementation details of our proposed method along with the baselines setup such as a traditional deep network (TDN) using ImageNet Pre-trained features with SVM and a conventional approach using BOW representation for SIFT with SVM.

4.3.1 Bag-of-Words SIFT feature with SVM

For a given image, Dense SIFT (DSIFT) descriptors of 128 dimension are computed for every n_s pixels inside the region of interest R of each image. where R is the lens area and n_s are the sub-sampled pixels. Further, a modified vocabulary tree structure [96] is utilized to construct a visual vocabulary dictionary. The vocabulary tree defines a hierarchical quantization using a hierarchical k-means clustering. In this work, a complete binary ($k = 2$) search tree structure is utilized. 2^{n_d} leaf nodes are finally used as visual vocabulary words, where, n_d is the depth of the binary tree.

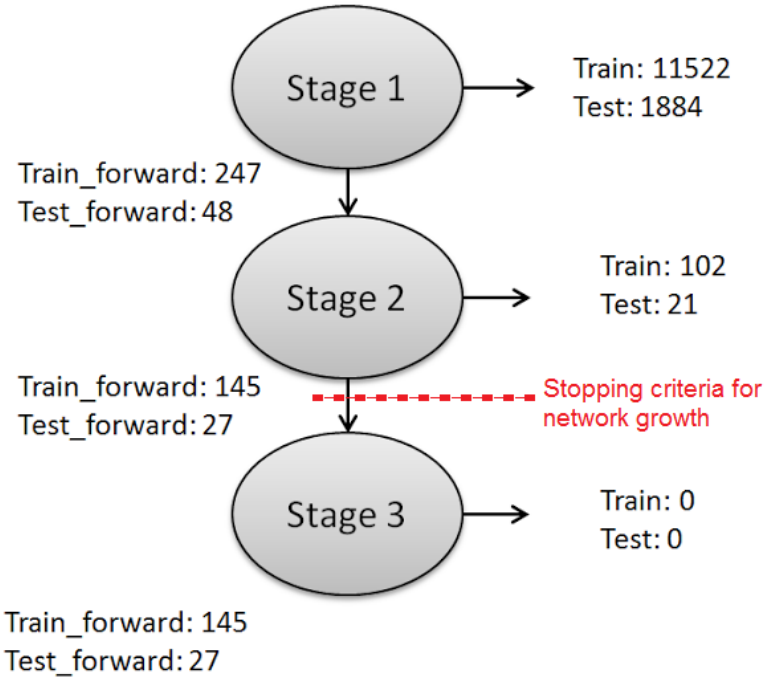


Figure 4.4: For validating the proposed method, CDDN was applied to the binary classification of digit '6' and '8' of MNIST dataset. A stopping criterion for network growth is when we see no improvement on the validation/training dataset performance, hence in this case it will result in a two-staged network.

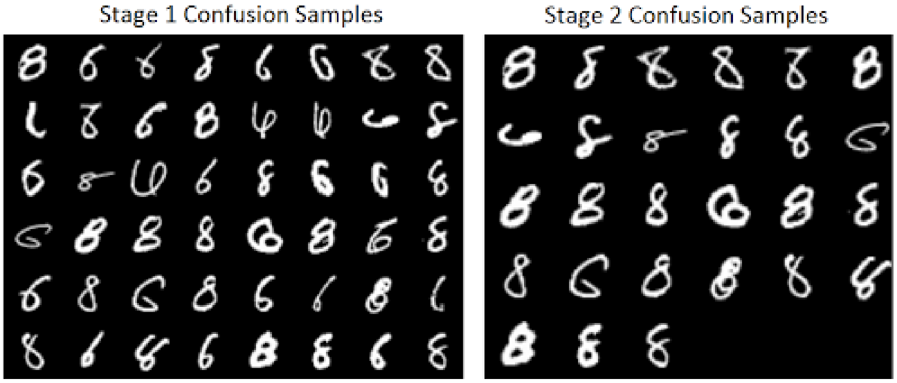


Figure 4.5: CDDN method idea validation on the classification of digit '6' and '8' of the MNIST dataset. The left image indicates some of the confusion classes at stage-1 and the right one indicates some confusion cases at stage-2. Observe that some of the confusion cases of stage-1 are resolved in stage-2.

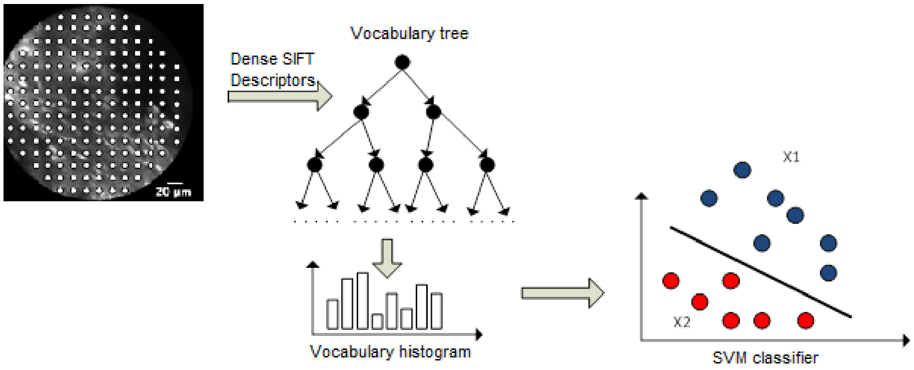


Figure 4.6: Workflow of the BOW representation for DSIFT with SVM classifier. White dots in the image represent the sampling points.

In the vocabulary tree learning stage, first the initial k-means algorithm is applied to the training data (a collection of SIFT descriptors is derived from the training data set and we randomly select a subset of the samples from these descriptors for final training) and then partition them into 2 groups, where each group consists of SIFT descriptors closest to the cluster center. This process is then recursively applied until the tree depth reaches the set value of n_d. In the online stage, a SIFT descriptor (a vector) is passed down the tree by each level via comparing this feature vector to the 2 cluster centers and choosing the closest one. The visual word histogram is computed for all the dense SIFT descriptors on each image. The resultant quantized representation is used to train an SVM classifier with a RBF kernel. The parameters of the SVM classifier are chosen using a coarse grid search algorithm. The entire workflow is depicted in Figure 4.6 for brain tumor classification data and we use a similar kind of setup for the polyp classification as well.

4.3.2 ImageNet Pre-trained Features with SVM

For an image, we extract feature vectors from all the layers of a pre-trained CNN on the ILSVRC-2012 dataset [59]. The dataset contains 1.2 million images which are manually annotated with labels from 1000 words vocabulary. Features are computed by forward propagating a mean-subtracted 224x224 RGB image through eight convolutional layers and three fully connected layers. In our case, we resize all the images irrespective of their aspect ratio to 224x224 to make them compatible with a pre-trained CNN. Features extracted from various layers were fed to the linear SVM classifier to evaluate its classification performance. This study was conducted to evaluate the performance of off-the-shelf pre-trained CNN features when applied to a couple of medical image classification problems and this also serves as a baseline.

4.3.3 Traditional Deep Network (TDN) and Cascaded Deep Decision Network (CDDN)

We used different deep network architectures to solve the polyp/no-polyp and meningioma/glioblastoma classification problems. The network architecture is summarized in Table 4.1. Notice that in the second stage, a convolution layer (Conv3) is introduced after the Conv2 layer, followed by fully connected (FC) layers. During stage 2 training, all the layers before Conv3 were frozen and the subsequent FC layers were randomly initialized. The final network architecture was determined based on the performance on a validation dataset. For all experiments, the step learning rate policy was adopted with the following parameters: learning rate set to 0.001, step size of 10000 and momentum of 0.9. The training loss converged well for both the datasets.

For comparison with traditional deep neural networks, we also train a deep network with similar model complexity as CDDN, in terms of number of layers and weight parameters. Thus, all the stage-1 and stage-2 layers of CDDN are combined to obtain a deep neural network, referred to as TDN in our experiments. This network (TDN) serves as a strong baseline to CDDN, because TDN can be interpreted as a classic ”going deeper” alternative to CDDN (which instead learns a stage 2 network on a subset of samples).

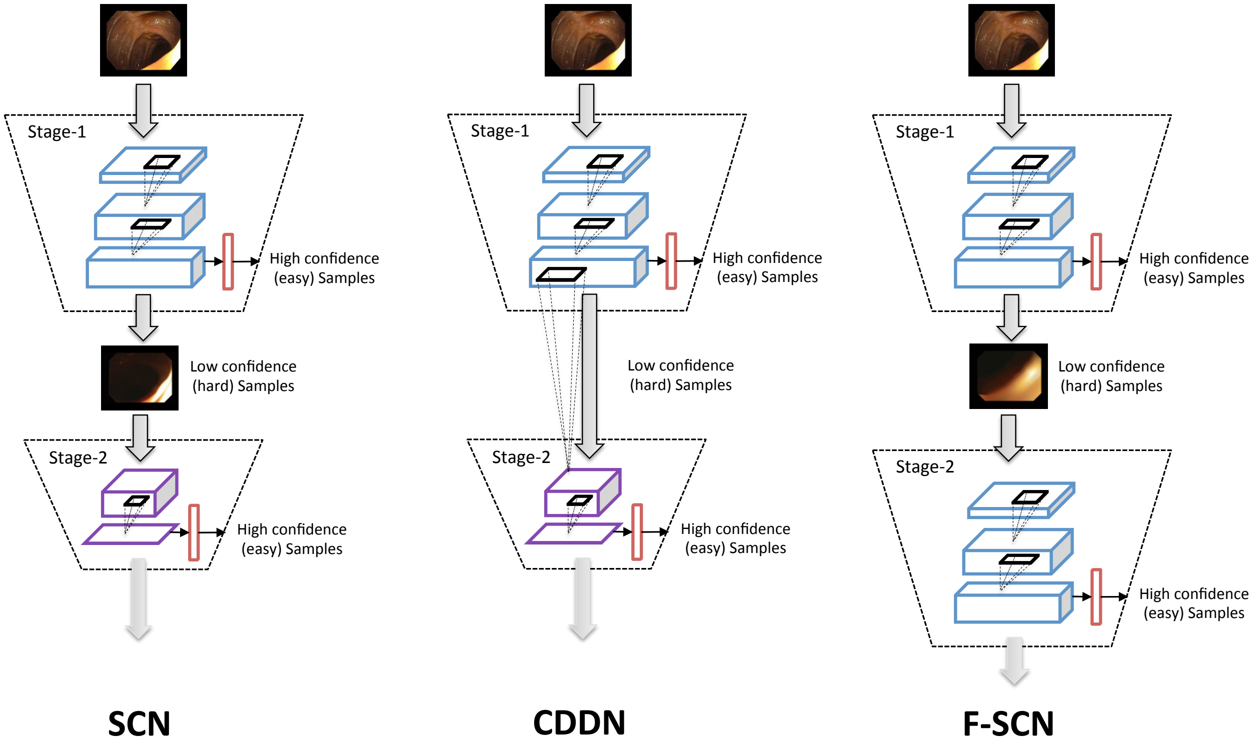

Stage-1
High confidence (easy) Samples
Low confidence (hard) Samples
Stage-2
High confidence (easy) Samples
SCN
Stage-1
High confidence (easy) Samples
Low confidence (hard) Samples
Stage-2
High confidence (easy) Samples
CDDN
Stage-1
High confidence (easy) Samples
Low confidence (hard) Samples
Stage-2
High confidence (easy) Samples
F-SCN

Figure 4.7: Comparison between Network Architectures. From left to right, Simple Cascaded Network (SCN), Cascaded Deep Decision Network (CDDN) and Fine-tuned Simple Cascaded Network (FSCN). Notice that CDDN's stage-2 is built on previous stage's feature space but for others cascade networks, the stage-2 is trained starting again from original image. For F-SCN, the first stage and second stage have the same architecture (depicted by color).

4.3.4 Cascaded Network

Cascading is a type of ensemble learning which involves concatenation of several classifiers. It is a multi-stage (classifier at each stage with same or different feature) approach, where the output information of the classifier is fed to the next classifier in the cascade. Since our approach bears similarities to cascading, we present alternate deep network architectures that directly embody cascading (ensemble of deep network classifiers), and provides a comparison with CDDN. We refer to these networks as Simple Cascaded Networks (SCN) and Fine-tuned Simple Cascaded Network (F-SCN). Figure 4.7 provides a comparison between CDDN and simple cascaded network architectures (SCN and F-SCN).

4.3.4.1 Simple Cascaded Networks (SCN)

This network is realized as a cascade of deep network classifiers, where each stage network is trained on only the misclassified samples from the previous stage. Unlike CDDN where each stage builds on the feature space of the previous stage, SCN trains the network in every stage starting from the original image; hence the correlation between the networks across stages is weaker in SCN. To enable a direct comparison, the size of the network (number of parameters) at each stage of SCN and CDDN is kept the same in all the experiments (ensuring similar model complexity).

4.3.4.2 Fine-tuned Simple Cascaded Networks (F-SCN)

Similar to SCN, this network is also realized as a cascade of deep network classifiers. However, instead of using a shallow network in subsequent stages, F-SCN duplicate the previous stage's network (including the parameters) and fine-tunes the parameters to correct the misclassified samples from the previous stage. In other words, a two stage F-SCN has two deep CNN networks with similar architecture (for network details in each stage please refer to Table 4.1). Similar to SCN, F-SCN trains each stage starting from the original image. The motivation behind this cascaded network design with fine-tuned networks at each stage is to help avoid over-fitting/under-fitting problem that are caused due to scarcity of training samples at each stage. Notice that the 2 stage F-SCN has almost twice the number of network parameters compared to CDDN (since stage-1 are generally much larger than stage-2) resulting in an increased model complexity and computational time.

Table 4.1: CDDN Configuration details. Conv: Convolutional layer, FC: Fully connected layer, AvePool: Average pooling and MaxPool: Max pooling. Each Conv layer is followed by a nonlinear function ReLU. Except for the last FC layer, rest of the FC layers are followed by ReLU and dropout layer with p=0.5.

Dataset	Convnet Configuration									
Polyp	stage-1	image (92x110x3)	Conv1 (64x11x11)	Maxpool (3x3)	Conv2 (128x5x5)	Avepool (3x3)	FC (512)	FC (2)		
	stage-2						Conv3 (256x3x3)	AvePool (3x3)	FC (512)	FC (2)
Brain Tumor	stage-1	image (110x110x1)	Conv1 (96x11x11)	MaxPool (3x3)	Conv2 (256x5x5)	MaxPool (3x3)	FC (4096)	FC (4096)	FC (2)	
	stage-2					Conv3 (384x3x3)	FC (4096)	FC (4096)	FC (2)	

4.4 Experiments

We report the performance of our proposed method in comparison to other methods on two different setups for endoscopic imaging - Brain tumor classification (classify images into Meningioma or Glioblastoma) and Polyp classification (to flag images containing a polyp). In both cases, we report results using a Bag Of visual Words (BOW) SIFT feature with SVM (RBF kernel) and ImageNet pre-trained features (best performing layer) with SVM. In addition, we report results using our proposed method CDDN and the strong baseline TDN (all the stages/network layers combined). Please note that in order to have a fair comparison, both the TDN and CDDN are designed to have the same complexity (number of layers and parameters).

4.4.1 Tumor Classification

4.4.1.1 Dataset

We use a commercially available clinical endo-microscope in the market called Cellvizio (Mauna Kea Technologies, Paris, France). Cellvizio is a probe-based CLE system -it consists of a laser scanning unit, proprietary software, a flat-panel display and fiber optic probes providing a circular field of view with a diameter of 160μm. The device is intended for imaging the internal micro-structure of tissues in the anatomical tract that are accessed by an endoscope. The system is clinically used during an endoscopic procedure for analysis of sub-surface structures of suspicious lesions, which are primarily referred to as optical biopsies [20]. In a surgical resection application, a neurosurgeon inserts a hand-held proof into a surgical bed to examine the remainder of the tumor tissue to be resected.

The equipment is used to collect 117 short videos, each from a unique patient suffering from Glioblastoma and relatively longer videos from patients with Meningioma. All videos are captured at 24 frames per second, under a resolution of 464x336. The collection of videos are hereafter being referred to as the Brain Tumor Dataset.

4.4.1.2 Pre-processing

Due to the limited imaging capability of CLE devices or intrinsic properties of brain tumor tissues, the resultant images often contain little categorical information and are not useful for recognition algorithms. Image entropy has been constantly used in the past [37] to quantitatively determine the information content of an image. Specifically, low-entropy images have very little contrast and large runs of pixels with the same or similar values.

In order to filter uninformative video frames, we empirically determine an entropy threshold by calculating the distribution of the individual frame entropy throughout the dataset (calculated over 34, 443 frames). In our case, this threshold is 4.15. This simple thresholding scheme allows us to select 14,051 frames containing Glioblastoma and 11,987 frames containing Meningioma cases. Experimental results are provided based on leaving a pair of patients (one with Glioblastoma and other with Meningioma) out. Further, we took a center crop of 220x220 square image inscribed in the circular lens region. Please note that for all the deep learning related experiments, images were resized to 110x110x1 to reduce the computational complexity.

4.4.1.3 Discussion

See table 4.2 for performance comparison. It is clearly evident that our proposed method CDDN significantly outperforms all the other methods. In comparison to TDN, CDDN improves the performance by around 9%, it does well on all the three measures of accuracy, sensitivity and specificity. This provides the evidence that our proposed method of building deeper networks is better than the traditional way of going deeper. Since CDDN makes early decisions on several samples, the average processing time for each sample for CDDN is lower when compared to TDN. We also provide the evaluation of different layers of the pre-trained network as features with an SVM classifier in Figure 4.8. We can see that on an average the 'Conv4' layer performs better across all the splits and hence to be consistent we report its results in the Table 4.2 as a baseline.

Table 4.2: Quantitative Performance Comparison on Tumor Classification Dataset; ImageNet pre-trained features were reported using 'Conv4' [[64]] layer with Linear SVM.

	SIFT+BOW +SVM(RBF)			ImageNet Pre-trained features			Traditional Deep Network			Deep Decision Network		
	Acc.	Sen.	Spec.	Acc.	Sen.	Spec.	Acc.	Sen.	Spec.	Acc.	Sen.	Spec.
split-1	81	0.96	0.71	67	0.90	0.50	78	0.91	0.69	81	0.87	0.76
split-2	63	0.97	0.49	61	0.94	0.47	66	0.93	0.69	73	0.97	0.63
split-3	82	0.91	0.75	89	0.97	0.86	77	0.77	0.77	89	0.90	0.88
split-4	98	0.98	0.97	95	0.96	0.94	93	0.93	0.93	97	0.95	1.0
split-5	77	0.70	0.84	83	0.73	0.92	74	0.79	0.69	85	0.70	0.99
Overall	**79**			**78**			**76**			**86**		

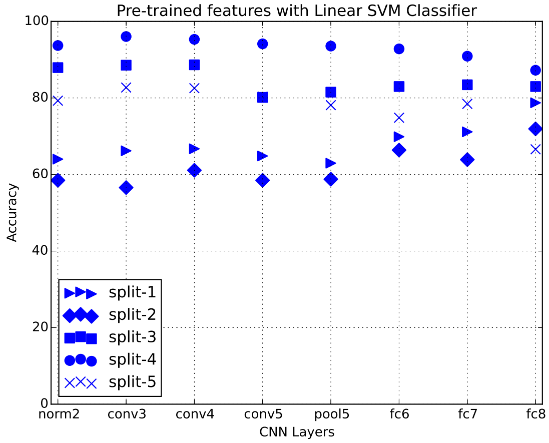


Figure 4.8: Classification Accuracy of different layers of pre-trained network as features with SVM classifier for Brain Tumor Classification

4.4.2 Polyp Classification for Colonoscopy

4.4.2.1 Dataset

Results are reported on a publicly available Polyp dataset from ISBI 2014 Challenge on Automatic Polyp Detection in Colonoscopy Videos [129]. The dataset consists of 21 short colonoscopy videos from ASU-Mayo Clinic polyp database, of which 11 videos have a unique polyp inside (positive shots) and the other 10 videos have no polyps (negative shots). Some videos are high resolution but some are recorded in a lower resolution, some videos display a careful colon examination while others show a hasty colon inspection; finally some videos have biopsy instruments in them. Please note that, even the videos containing a polyp will have a large number of frames where the polyp is absent and hence the groundtruth labels are provided at frame level. In our evaluation, we provide experimental results on four random splits (are at video level to avoid bias during train and test split) by reporting classification accuracy at frame level and also provide ROC curves.

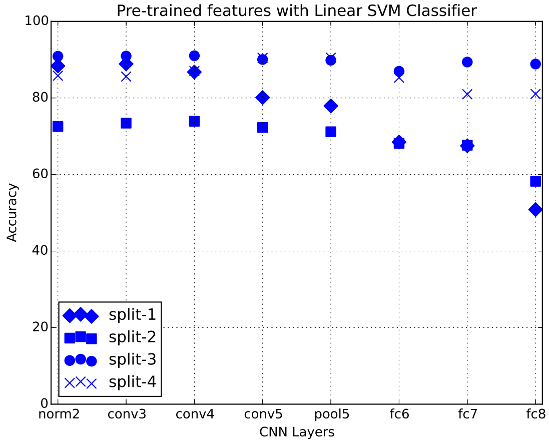


Figure 4.9: Classification Accuracy of different layers of pre-trained network as features with SVM classifier for Polyp/No-Polyp Classification

4.4.2.2 Pre-processing

Since the videos are of different resolutions and regions around the frames were varying, we fixed the final image size to be 636x530 (chosen based on the average resolutions of all the video frames. We identified the lens region separated from the rest of the black region and then resized (maintaining the aspect ratio) to fit the fixed window size of 636x530. Since frames containing the polyp were relatively very low, we chose to perturb only the positive (contains polyp) frames. Perturbation involved rotation by angles of 90,180 and 270 degrees followed by flipping and again rotating with the same set of angles. Please note that for all the experimentation the resulting images were later resized to 110x92x3 to handle the computational complexity.

4.4.2.3 Discussion

Table 4.3 demonstrates the performance comparison. We observe similar performance trends as reported for the brain tumor classification, where our proposed method CDDN outperforms all the other methods. In addition to the accuracy metric, we have also provided the ROC curve for all the splits in Figure 4.10. Overall, the area under the curve is significantly better for CDDN when compared to the

Table 4.3: Quantitative Performance Comparison on Polyp Classification Dataset

	SIFT+BOW +SVM(RBF)	ImageNet Pre-trained features (Conv3)	TDN	CDDN
	Acc.	Acc.	Acc.	Acc.
split-1	89.1	88.89	78.34	87
split-2	37.46	73.41	67.81	83
split-3	70.82	90.95	88.88	92.75
split-4	82.90	85.59	84.45	92.40
Overall	**70.08**	**81.66**	**80.67**	**87.43**

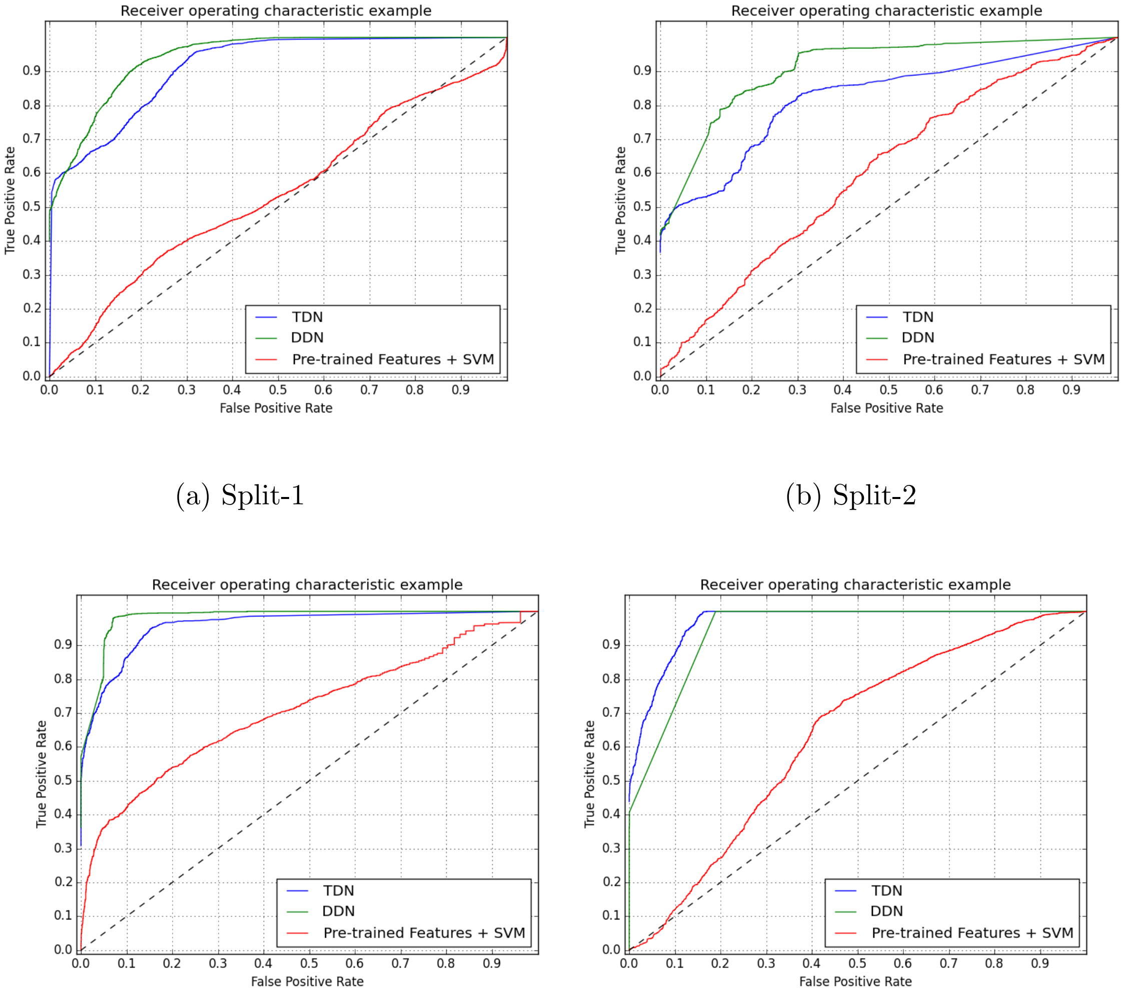
rest of the methods. All these experimental results convey that the proposed CDDN method is an efficient and effective alternative to the traditional way of building a deeper network. Considering a clinical use case, if we pick an operating point of false positive rate=17% with true positive rate=90% in Figure 4.10, then our system on an average is able to eliminate 84% of the negative images (do not contain polyp) but still be able to identify 90% of the positive cases (containing polyp) accurately. In Figure 4.9 we provide the effectiveness of different layers of the Pre-trained network as features when combined with SVM classifier. On an average across all the splits, we found that the 'Conv3' layer gives the best performance and thus their results are reported in the Table 4.3 as a baseline.

Table 4.4: CDDN Performance analysis on Polyp Classification Dataset. # - number of samples; Acc. - Accuracy (%).

	SCN					F-SCN					CDDN				
	Stage-1		Stage-2		Overall	Stage-1		Stage-2		Overall	Stage-1		Stage-2		Overall
	#	Acc.	#	Acc.	Acc.	#	Acc.	#	Acc.	Acc.	#	Acc.	#	Acc.	Acc.
split-1	3373	96.02	4090	67.41	67.41	4696	97.84	2767	58.07	83.1	3712	98.94	3751	86.36	**83.36**
split-2	1248	92.62	1595	51.28	51.28	1227	92.25	1616	50.99	68.8	1501	94.53	1342	83.07	**83.07**
split-3	1740	100	3665	62.49	62.49	3830	98.09	1575	63.23	87.93	4245	99.74	1160	92.74	**92.74**
split-4	3325	99.06	2629	66.67	66.67	3593	98.71	2361	75.91	89.46	3391	99.97	2563	92.40	**92.40**

4.4.3 Comparison with SCN and F-SCN

The stage-wise performance comparisons of our proposed method CDDN in comparison to SCN and F-SCN are provided in Table 4.4 and Table 4.5 for polyp and



(a) Split-1

(b) Split-2

(c) Split-3

(d) Split-4

Figure 4.10: ROC curves for Polyp dataset across all the splits.

tumor classification dataset respectively. We can observe that CDDN outperforms both SCN and F-SCN at each stage for all splits and its better even in terms of overall performance. We believe that SCN and F-SCN could not perform well because of over-fitting/under-fitting problem in the second stage due to limited number of samples (hard samples). This clearly indicates that our proposed method has the ability to dodge this prevalent problem while applying deep learning networks for medical related problems where the data is limited.

Table 4.5: CDDN Performance analysis on Tumor Classification Dataset. # -number of samples; Acc. -Accuracy (%).

	SCN					F-SCN					CDDN				
	Stage-1		Stage-2		Overall	Stage-1		Stage-2		Overall	Stage-1		Stage-2		Overall
	#	Acc.	#	Acc.	Acc.	#	Acc.	#	Acc.	Acc.	#	Acc.	#	Acc.	Acc.
split-1	540	96.14	791	25.53	54.99	333	92.79	998	69.13	75.05	340	77.35	991	82.44	**81.13**
split-2	203	91.03	481	16.83	39.47	255	92.94	429	40.79	60.23	121	100	563	67.49	**73.24**
split-3	1384	86.56	1962	46.48	63.06	444	88.51	649	62.71	73.19	445	97.75	648	82.87	**88.92**
split-4	445	100	237	67.93	88.85	544	100	138	78.26	95.6	537	100	145	87.58	**97.35**
split-5	1367	87.63	1979	47.85	64.1	1507	90.31	1839	60.95	74.17	1177	99.15	2169	80.26	**86.90**

CHAPTER 5

DEEP DECISION NETWORK (DDN)

5.1 Introduction

Convolutional Neural Network (CNN) based methods have consistently been the top performers on various computer vision tasks. But, there are still no well-established guidelines to train a performant deep network, and thus, training a deep network often involves thorough experimentation and statistical analysis. Although going deeper in the neural network design has shown to be effective [116, 125], it also increases the training duration as well as the risk of over-fitting.

Hence, we propose a novel computational framework called Deep Decision Network (DDN) to design an efficient deep network architecture without over-fitting the training process. This is an extension of the CDDN work (presented in the previous chapter) with the required changes for it to handle the relatively challenging multi-class classification problem. In contrast to existing deep learning-based approaches, DDN is built stage-wise during the learning phase (similar to CDDN). At each stage, the network introduces decision stumps to classify confident samples and partition the remaining data, which is difficult to classify, into smaller data clusters which are used for learning successive expert networks in the next stage. Note that data clusters at each stage are such that the samples within a cluster are difficult to distinguish using the trained classifier at that stage but the samples across clusters are easily distinguishable. This is achieved by fine tuning the trained classifier using a combination of softmax and weighted contrastive loss. A contrastive loss function helps in bringing the samples of the same group together and pushing apart the samples belonging to different classes. While the clustering is motivated by the divide-and-conquer principle, it has the added benefit of automatically discovering a data hierarchy based on appearance similarity. Notice that the DDN implicitly captures the intuition that hard samples require more computation.

Further, we introduce DDN-annot which is an extension of the DDN principle to address the more challenging multi-label classification (image tagging/annotation) problem. The idea here is to identify clusters of overlapping labels that capture the coexistence or dependency property. Then, the expert network is built for each cluster to handle the confusion between the subset of labels and assign them appropriately to the test image. The clusters are built based on the features extracted from the layer before the soft-max in the network.

Our contributions are as follows: (a) A proposed stage-wise training strategy for the DDN/DDN-annot helps alleviate problems encountered by gradient-based methods on deeper architectures, (b) In the case of DDN, a joint-loss (weighted contrastive and classification) optimization of the network is proposed to minimize errors during data partitioning, (c) A proposed data-driven design for the DDN/DDN-annot offers an insight into the underlying structure of the data, (d) The proposed network architecture can make early decision because of it's tree like structure as opposed to conventional deep neural network.

We demonstrate the following that shows the effectiveness of our proposed approach: (a) The DDN achieves state-of-the-art performance (at the time of publication of this work [92]) on CIFAR-10 and CIFAR-100 [63] public benchmarks and finally, (b) DDN (ResNet-50 with only a few additional expert layers) achieves performance that is equivalent to ResNet-101 (with nearly 100 layers) on the large scale publicly available ILSVRC 2012 (ImageNet) dataset [106]. (c) DDN-annot yields results comparable to the state-of-the-art on IAPRTC-12 and NUS-WIDE image annotation datasets.

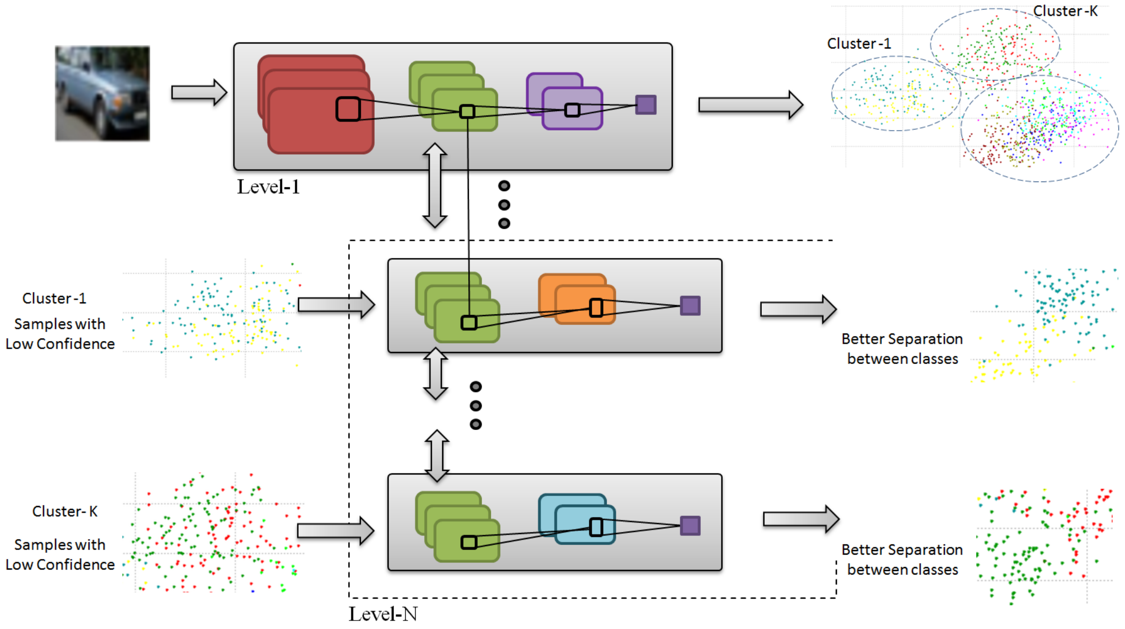


Figure 5.1: Overview of the Deep Decision Network (DDN) framework. We observe N levels in the DDN tree structured network and at each level there could be K clusters of confusion classes.

5.2 Deep Decision Network Framework

This section describes the Deep Decision Network and the algorithms involved in learning the deep decision network architecture and it's parameters.

5.2.1 Deep Decision Network

A deep decision network (DDN) is a tree structured deep neural network with decision stumps at each node to classify easily separable data earlier in the network and to determine the subsequent expert node for difficult cases. An overview of the DDN computational framework is provided in Figure 5.1.

Given a dataset, a root (level 1) network is trained using the back propagation algorithm. Instead of optimizing the network to obtain the best performance, we only need to optimize until a reasonable performance is achieved e.g. 60-70%. Alternatively, a pre-trained network can be used as a root network if it achieves reasonable performance. The confusion matrix, computed over the validation dataset is then used to identify clusters of object classes, such that each cluster may have large confusion among classes inside the cluster but the confusion across clusters is low. A subsequent expert network is trained for data within each cluster to correctly classify the previously misclassified samples and/or the samples classified with low confidence. This has the effect that as we go deeper we continue to "zoom-in" on resolving the problem cases. This process of building the network is continued until we see no further improvement on the validation data set. During testing, a sample is routed through DDN until it's class is determined (via early classification or at the leaf node).

There are a few key differences between the DDN architecture and the traditional deep networks. Firstly, all the layers in the previous levels are frozen while training the newly introduced network layers which forms a new node at the next level. Secondly, each node is built on the parent node's feature space to specifically handle a subset of classes. Note that each node can be trained starting from any layer of the parent node, and this choice of the layer can be determined using a cross validation data set.

5.2.2 Discovering data clusters

Here we discuss how the clusters are identified at each node of the Deep Decision Network using the spectral co-clustering algorithm [24]. The spectral co-clustering algorithm approximates the normalized cut of a bi-partite graph (symmetric matrix) to find heavy subgraphs (sub-matrices) thus resulting in block diagnolization of the matrix. We apply the spectral co-clustering algorithm over the co-variance of the confusion matrix; each block in the resulting block diagonal matrix forms a cluster. The resulting clusters may be disjoint (no overlapping classes) and the confusion among the classes within a cluster will be high because classes within a cluster will be closely related, for instance, classes of cats and dogs might form a cluster representing animal group. Furthermore, if there are any entries (in the confusion matrix) which are not within the diagonal blocks, then the samples contributing to those entries would get misclassified. Thus, to minimize the likelihood of such misclassifications, we fine tune the network parameters using a joint loss, combining softmax and weighted contrastive loss; this is explained in detail in the Section 3.3.

In order to determine the optimal clustering C^*, we define a fitness measure $fm(C)$, for a given clustering C computed using spectral co-clustering, as

$$fm(C) = \left(\epsilon + \frac{1}{K} \sum_{i=1}^{K} |C_i| \right) \tag{5.1}$$

where, ϵ is the misclassification error introduced due to the data-split, C_i is the i^{th} cluster (set of classes), $|.|$ is the size of a set. The optimal clustering C^* is then given by,

$$C^* = \arg\min_{C} fm(C) \tag{5.2}$$

Here, the first term in eqn. 5.1 will have a low value when there is just one cluster (K=1), and on the other hand, the second term in eqn. 5.1 will have a low value when K (number of clusters) is equal to number of class labels. Hence, the fitness measure tries to strike a balance between the two, so that the final error rate can be further minimized using the expert networks.

5.2.3 Minimizing errors during splitting

As mentioned earlier, errors due to incorrect assignment of samples to the clusters are irrecoverable - for example, assume airplane and ship form a cluster and the expert network is trained to distinguish only between these two classes. During testing, if an airplane sample gets misclassified (by the root network) as dog, then that sample will get assigned to a different cluster (cluster for which expert network is trained only to distinguish animal classes) and the corresponding expert network will have no idea about the airplane class, because the clusters are non-overlapping. Hence, there is no possibility for recovering the mistakes made by the root network.

But, we try to minimize such misclassification errors by augmenting the softmax-loss with an error-driven, weighted contrastive loss function that helps block diagonalize the confusion matrix; this is depicted in the Figure 5.2. The overall loss function is given by

$$L = \lambda_2 \times L_m + \lambda_1 \times L_{softmax} \tag{5.3}$$

where, L_m is the weighted contrastive loss function. The weights λ_1 and λ_2 were set to 1.0 based on performance on the validation dataset.

The weighted contrastive loss L_m can be interpreted as a set of soft constraints which impose a significantly higher penalty for misclassifying a sample to any class belonging to another cluster as compared to the penalty of misclassifying to a class that belongs to the same cluster. In other words, minimizing the weighted contrastive loss results in the distance between samples belonging to the same cluster to be small, and samples across different clusters to be large. The weighted contrastive loss function is given by

$$\begin{aligned} L_m = w_{ij} \times \Bigg(\frac{(1-Y)}{2} \times D^2 + \\ \frac{Y}{2} \times \{max(0, m - D)\}^2 \Bigg) \end{aligned} \tag{5.4}$$

where,

$$w_{ij} = \begin{cases} 0.1 & \text{if } i \subset C_k \text{ and } j \subset C_k \\ 1 & \text{otherwise} \end{cases}$$

where, w_{ij} is the weight corresponding to class labels i and j, D is the L_2-*norm* between a pair of samples. Y is the label representing whether samples belong to the

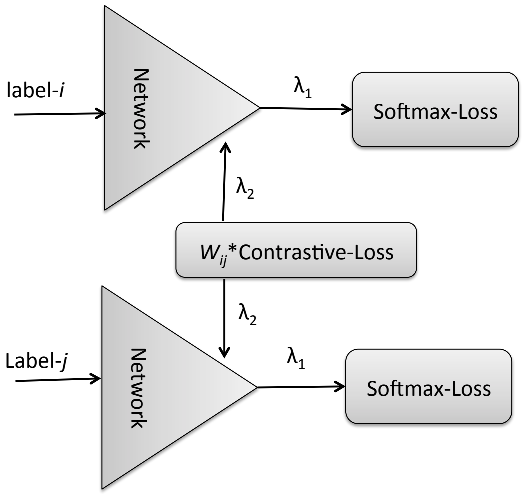


Figure 5.2: DDN optimization with weighted contrastive-loss function along with Softmax-loss.

same class (Y=0) or to a different class (Y=1), m is the margin (which is determined empirically on the validation dataset). C_k indicates the k^{th} cluster. w_{ij} values are found empirically.

5.2.4 Piece-wise training for DDN

The proposed architecture is trained in a unique fashion - starting with a root network (trained in the traditional way), we use it's softmax layer to compute the performance and learn a classification threshold for each class using cross-validation. This threshold is used during testing to make an early decision on samples. We then compute the confusion matrix on the validation dataset and use it to identify the clusters of confusion classes (as explained in Section 5.2.2). Next, the network is fine-tuned using the weighted contrastive loss (as explained in Section 5.2.3). The weights for the contrastive loss function are determined based on the confusion matrix. After fine-tuning, the samples are split according to their cluster ID's.

For each cluster, a node is added to the decision network. A node itself is a shallow network (or expert network) trained to distinguish between a subset of classes belonging to that cluster. For an expert network architecture, we utilize the micro networks (explained in 5.3.1) of Network In Network (NIN) [79]. Note that when we train the new layers, we freeze the previously trained layers by setting their learning rate to zero.

This process of adding a node to the decision network is continued recursively until there is no more improvement on the validation dataset and/or the maximum depth of the network is reached.

With this training scheme, DDN is able to make use of the efforts in the early layers for training the subsequent layers, and has the benefit of making an early decision. Furthermore, training expert networks (node) starting from parent feature spaces helps in avoiding the over-fitting problem. In addition, it also helps in avoiding getting stuck in poor solutions during the gradient optimization process, and converges to network parameters that provide better generalization; this is validated in our experiments.

5.3 Experimental Results on CIFAR dataset

We report the performance of the proposed method in comparison to other methods on publicly available benchmark datasets - CIFAR-10 and CIFAR-100 [63]. We implemented our method using Caffe [59] and all the experiments were carried out on a single Titan-X GPU. The train-test splits and data pre-processing are as provided in [38].

5.3.1 Network Details

In this work, we chose Network -in-Network (NIN) [79] as the root node of our DDN for experimenting on both CIFAR-10 and CIFAR-100 datasets. The root node could be any existing network but we chose NIN. NIN also has a nice property of being built with mlpconv (multi-layer perceptrons and convolutional layers) as a basic

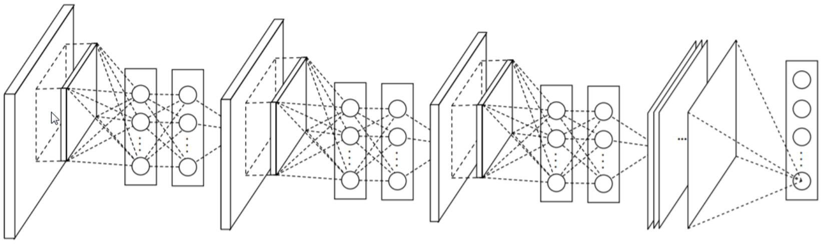


Figure 5.3: The overall structure of Network In Network (NIN). Adapted/borrowed from Lin et al. [79].

building block unit, see Figure 5.3. The original NIN consists of three micro network (mlpconv) consisting of convolutional layers and MLP unit as shown in figure. Each MLP layer is composed of a three-layer perceptron and a pooling layer. DDN consists of NIN as the root node and additional layers (shallow-network/branch nodes) are simply one mlpconv layer of NIN. Additional layers were introduced right after the second mlpconv unit of NIN to make use of the local feature response instead of the third node which seems to capture global class specific features. As in NIN, global average pooling was used instead of fully connected layers at the leaf nodes. All the network parameter settings, weights initialization and learning policy strictly follow the settings provided by NIN. The only change was during the addition of new layers (shallow-networks), the learning rate was set to 0.01 with a step size of 25K. In the current setup for both the datasets, we had only two levels, with root node NIN at level-1 and multiple MLP units in level-2. Each MLP was specialized to address a particular cluster consisting of the most confusing classes.

5.3.2 CIFAR-10

5.3.2.1 Experimental Setup

The CIFAR-10 dataset [63] consists of 10 classes of natural images with a total of 50K training images and a total of 10K testing images. Each image is of size 32x32 and we follow the same pre-processing of global contrast normalization and ZCA whitening as in [38, 79]. For the validation dataset, we used the last 10K samples of the training to determine the confidence level threshold and data splits based on the confusion matrix. After determining the data-splits and the confidence level threshold, we combined the training and validation dataset to re-train the network before splitting.

5.3.2.2 Quantitative Results

The error rates for our proposed method in comparison to the existing methods is provided in Table 5.1. We obtain a test error of 9.68% without any data-augmentation and this sets a new state-of-the-art result on the CIFAR-10 dataset. The accuracy was imporved by nearly 1% when compared to our strong baseline NIN (same model complexity).

5.3.2.3 Further Analysis

Figure 5.5a provides the confusion matrix of the root node at level-1. Figure 5.5b shows the clusters of confusion classes obtained by applying a spectral co-clustering algorithm. We observe three clusters - Cluster-1: {0-airplane, 8-ship}, Cluster-2: {1-automobile, 9-truck}, Cluster-3: {2-bird, 3-cat, 4-deer, 5-dog, 6-frog, 7-horse}. This clustering can be interpreted as a data hierarchy automatically generated from the data.

As described in Section 5.2.3, we use a joint-loss optimization to fine-tune the network which helps in block diagonalizing the confusion matrix. The impact of using the joint loss can be observed in Figure 5.4; notice that the use of joint loss brings the samples of the same cluster closer while the samples in different clusters are moved farther apart in the feature space. We try to minimize the joint-loss function without compromising on the classification performance and in fact, this is shown to slightly improve the performance. Figure 5.5c shows the effect of varying the cluster

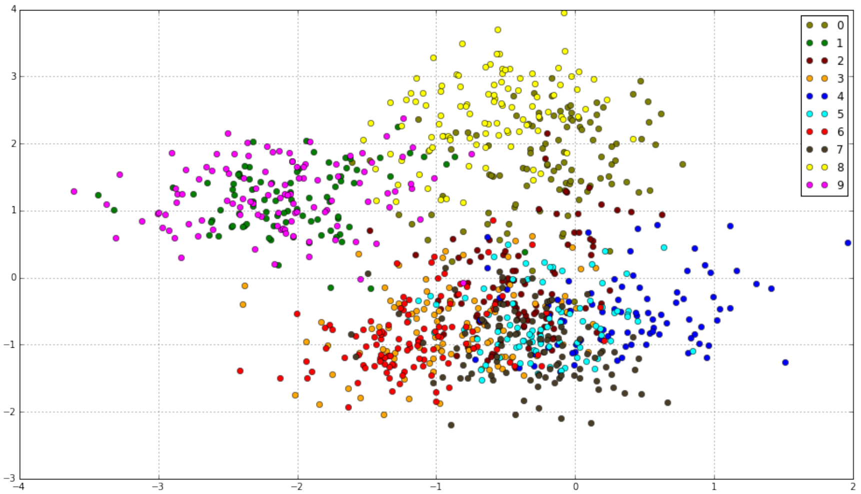


Figure 5.4: Visualization of the learnt feature space on CIFAR-10 dataset. Each point corresponds to an image in CIFAR-10 dataset, and it's color correspond to its image class. Observe that each cluster has only certain group of classes. For instance, Class-1 (magenta) and Class-9 (green) belong to the same cluster, hence they are close to each other but away from the remaining classes.

size K on the error rate (due to misclassification); notice the error is least when K is 3.

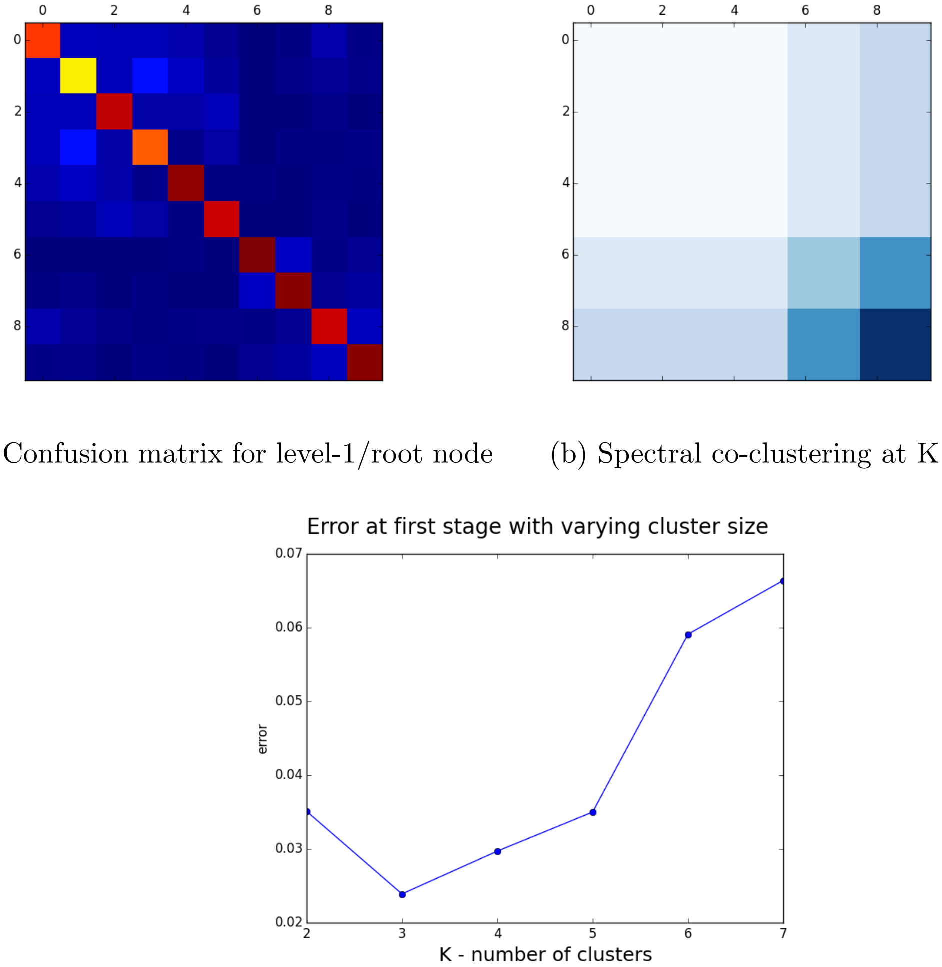
Table 5.1: Results on CIFAR-10 Dataset without data augmentation

Method	Test Error
Stochastic Pooling[148]	15.13
CNN + Spearmint[118]	14.98
Conv. maxout +Dropout[38]	11.68
NIN+Dropout[79]	10.41
DSN[72]	9.78
DDN (ours)	**9.68**

5.3.3 CIFAR-100

5.3.3.1 Experimental Setup

The CIFAR-100 dataset [63] consists of 100 classes of natural images, making it more challenging compared to the CIFAR-10 dataset. It consists of 50K training and 10K testing images. The number of training samples per class is only 100 as



(a) Confusion matrix for level-1/root node

(b) Spectral co-clustering at K = 3

(c) Effect of cluster size K on error rate

Figure 5.5: CIFAR-10 results. The optimal clustering obtained using the fitness measure is at K = 3, which incidentally also corresponds to the lowest misclassification error.

compared to 1000 in CIFAR-10. The dataset is pre-processed using global contrast normalization and ZCA whitening as described in [38, 79]. Similar to NIN [79], the last 10K samples of the training set were used as the validation dataset.

5.3.3.2 Quantitative Results

Our proposed method yields a test error of 31.65%, surpassing Deeply Supervised Nets (DSN [72]) by nearly 3%. The error rate comparison is shown in Table 5.2. Note that HD-CNN [146] uses data augmentation and 10 crop testing, so the performance is not directly comparable to other methods, since it is difficult to isolate the impact of the data augmentation from the methodology. However notice that DDN still performs better than HD-CNN even without any data augmentation.

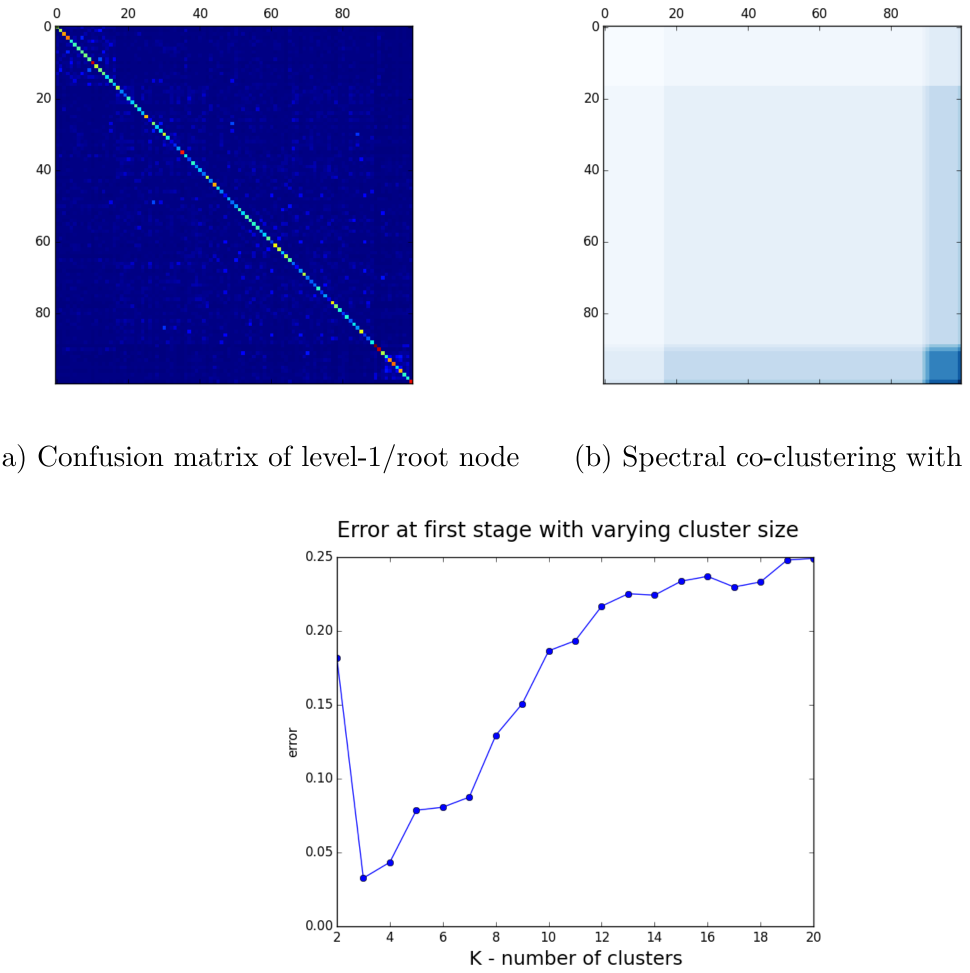
We cannot directly compare with [119] because they have reported numbers with data augmentation only. Since the data augmentation process for CIFAR-100 is not standardized, we reported numbers without augmentation to enable a fair comparison with the existing literature and showed a significant improvement in accuracy.

Table 5.2: Results on CIFAR-100 Dataset without data augmentation. *-with data augmentation and 10 view testing [64]

Method	Test Error
Learned Pooling[85]	43.71
stochastic Polling[148]	42.51
Conv. maxout +Dropout[38]	38.57
Tree based priors[123]	36.85
NIN+Dropout[79]	35.68
DSN[72]	34.57
NIN+LA units[4]	34.40
HD-CNN* [146]	32.62
DDN (ours)	**31.65**

5.3.3.3 Further Analysis

Figure 5.6a provides the confusion matrix of the root node at level-1 and the resulting clusters obtained after applying spectral co-clustering are shown in Fig-



(a) Confusion matrix of level-1/root node

(b) Spectral co-clustering with K=6

(c) Effect of cluster size K on error rate

Figure 5.6: CIFAR-100 results. Notice that even though the misclassification error is lowest at K = 3, the optimal clustering obtained using the fitness measure is at K = 6. This is consistent with our intuition that the fitness measure encourages partitioning into more clusters while keep the error low.

ure 5.6b. The effect of varying the cluster size K on the error rate is shown in the Figure 5.6c. Notice that even though the error rate is lowest at $K = 3$, the algorithm chose $K = 6$ for optimal clustering based on the overall fitness measure that tries to strike a balance between number of clusters (requires expert networks) and the error rate (introduced due to data splitting). Please note that the error rate at the stage of data splitting (branching) will be minimum if there was only one cluster, in which case there would not be any expert networks to follow and this leaves no room improvement for further improvement in the classification accuracy. Out of these six clusters, three of them didn't require expert network as they were composed of a single class. The other three clusters have 72, 17 and 8 classes. The expert networks for each of these clusters reduce the classification error by more than 10%.

Table 5.3: Detailed Quantitative Performance on CIFAR-10 and CIFAR-100 Dataset. NIN+JL: NIN network with joint-loss optimization.

	CIFAR-10					CIFAR-100				
	NIN	NIN+JL	DDN			NIN	NIN+JL	DDN		
	Error(%)	Error(%)	Level-0	Level-1	Error (%)	Error (%)	Error(%)	Level-0	Level-1	Error (%)
Cluster-1	-	7.15	1148	767	7.0	-	37.97	1802	5093	34.52
Cluster-2	-	5.50	668	1280	4.8	-	14.0	102	0	14.0
Cluster-3	-	12.43	1704	4199	12.2	-	24.0	88	0	24.0
Cluster-4	-	-	-	-	-	-	26.35	548	983	23.35
Cluster-5	-	-	-	-	-	-	28.62	213	483	26.25
Cluster-6	-	-	-	-	-	-	24.0	89	0	24.0
Overall	10.41	9.99			**9.68**	35.68	34.73			**31.55**

5.3.4 Detailed analysis of DDN performance on CIFAR dataset

In Table 5.3, we provide a detailed performance analysis of DDN at each node in the network in comparison to the baseline NIN. We have also reported the results using NIN with joint loss optimization (NIN + JL). Though the main objective of joint-loss optimization was to reduce the confusion cases across the cluster samples, we get some improvement over the baseline on both the datasets. This is probably because the joint-loss optimization helped in regularization of the network.

DDN helps in providing some insight into the data regarding the classes that are hard to distinguish from others. For instance, if we consider CIFAR-10, the root node (NIN) produced three clusters of confusion classes. We can see that cluster-3 performance is low when compared to other clusters -The reason being that cluster-3 has 6 classes: cat, dog, deer, dog, frog and horse, and all of them belong to the animal category and it is relatively hard to distinguish among them when compared to the automobile/truck in cluster-2. It is also important to note that the DDN helped to improve the performance in each of the clusters that led to the overall improvement. This also verifies the fact that expert network nodes were in fact helpful as compared to training one large network end-to-end.

For the CIFAR-100 dataset, the performance improvement of DDN is significant when compared to our baseline NIN. Part of the reason is that, DDN seems to benefit more from having large number of classes and there remains room for improvement on this particular dataset. The expert network nodes were introduced only to clusters with at least 2 classes and hence clusters with single class do not get any performance improvement. This indicates that DDN by design will not bring down the performance of any of the existing network used at the root but, it only tries to improve the performance by addressing the most confusing cases. Clusters with at least two classes benefit from the expert network node which results in an overall improved performance.

5.4 Experimental results on ImageNet

5.4.0.1 Experimental Setup

The ILSVRC 2012 dataset [106] consists of 1.2M training set images and 50K validation set images. Here we use ResNet-50 [48] as a building block (root node) and show that adding just a few expert layers for each cluster can yield performance that is comparable to the ResNet-101 [48]. For training and testing, we follow [48]'s protocol

Table 5.4: Detailed Quantitative Performance on ILSVRC12 Dataset. ResNet-50-JL: 50 layered ResNet with joint-loss optimization. Top-5 error rates are reported. Clusters with only one class label are ignored. Here the reported error for ResNet-50 is based on my implementation using PyTorch [98] with 1-crop during testing.

	ILSVRC12					
	ResNet-50	ResNet-50-JL	DDN (ResNet-50 expert)			ResNet-101
Top-5	Error (%)	Error(%)	Level-0	Level-1	Error (%)	Error (%)
Cluster-1	-	10.1	2608	1093	9.3	-
Cluster-2	-	7.9	13864	3289	7.6	-
Cluster-5	-	6.0	1808	899	5.8	-
Cluster-6	-	8.35	4248	883	7.75	-
Cluster-7	-	6.62	3137	1483	6.35	-
Cluster-9	-	7.0	7454	4245	6.1	-
Cluster-10	-	14.0	1289	612	12.0	-
Overall	7.26	7.11			**6.53**	**6.44**

and parameter settings. In order to make DDN feasible on this large dataset, we introduced some changes during the training phase without affecting the underlying principle. In the original proposed DDN, we used a Siamese setup with the weighted contrastive loss to minimize the errors during the split. But, since the network is deep and computationally intensive (150 layer), we replaced the weighted contrastive loss layer by weighted cross entropy loss and show that we could achieve similar results. To validate this setup, we repeated the experiments on CIFAR-10 and were able to reproduce similar results.

From the confusion matrix of root node (ResNet-50), the optimal number of clusters was chosen to be 10. The number of classes in each cluster is represented as a histogram in figure 5.7. The complete setup was implemented using Pytorch.

5.4.0.2 Network Details

The network details are provided in table 5.5. DDN (ResNet50-expert) is very similar to ResNet50, but varies only in conv5_x with additional K blocks representing experts for K clusters. During the expert network training stage, all the previous layers are frozen and only the expert network block conv5_x gets trained. And during

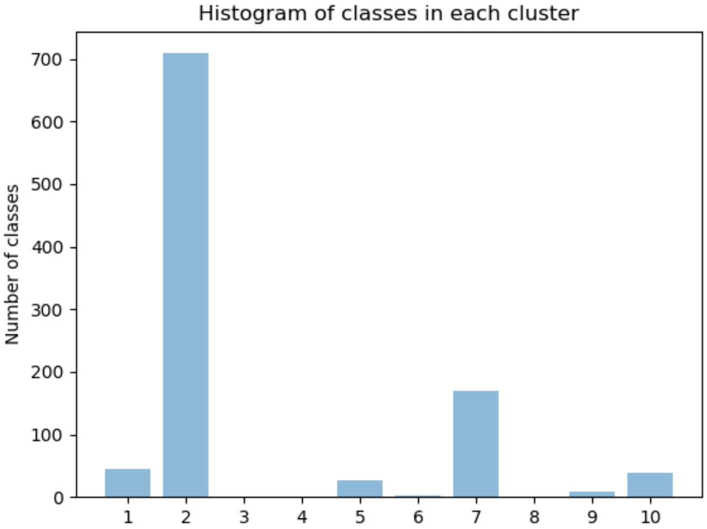


Figure 5.7: Histogram of classes in each cluster. Clusters 3, 4, 6 and 8 end up having only one class label. Hence they are not clearly visible.

Table 5.5: DDN network details for ImageNet. K is the number of clusters/experts.

layer_name	output_size	ResNet50	ResNet50-expert	ResNet101
conv1	112x112	7x7, 64, stride 2		
		3x3 maxpool, stride2		
conv2_x	56x56	$\begin{bmatrix}\text{1x1}, 64\\ \text{3x3}, 64\\ \text{1x1}, 256\end{bmatrix}\times 3$	$\begin{bmatrix}\text{1x1}, 64\\ \text{3x3}, 64\\ \text{1x1}, 256\end{bmatrix}\times 3$	$\begin{bmatrix}\text{1x1}, 64\\ \text{3x3}, 64\\ \text{1x1}, 256\end{bmatrix}\times 3$
conv3_x	28x28	$\begin{bmatrix}\text{1x1}, 128\\ \text{3x3}, 128\\ \text{1x1}, 512\end{bmatrix}\times 4$	$\begin{bmatrix}\text{1x1}, 128\\ \text{3x3}, 128\\ \text{1x1}, 512\end{bmatrix}\times 4$	$\begin{bmatrix}\text{1x1}, 128\\ \text{3x3}, 128\\ \text{1x1}, 512\end{bmatrix}\times 4$
conv4_x	14x14	$\begin{bmatrix}\text{1x1}, 256\\ \text{3x3}, 256\\ \text{1x1}, 1024\end{bmatrix}\times 6$	$\begin{bmatrix}\text{1x1}, 256\\ \text{3x3}, 256\\ \text{1x1}, 1024\end{bmatrix}\times 6$	$\begin{bmatrix}\text{1x1}, 256\\ \text{3x3}, 256\\ \text{1x1}, 1024\end{bmatrix}\times 23$
conv5_x	7x7	$\begin{bmatrix}\text{1x1}, 512\\ \text{3x3}, 512\\ \text{1x1}, 2048\end{bmatrix}\times 3$	$\begin{bmatrix}\text{1x1}, 512\\ \text{3x3}, 512\\ \text{1x1}, 2048\end{bmatrix}\times 3 \times K$	$\begin{bmatrix}\text{1x1}, 512\\ \text{3x3}, 512\\ \text{1x1}, 2048\end{bmatrix}\times 3$
	1x1	average pool, 1000-d fc, softmax		

the test time, the sample gets routed through only one of the K conv5_x blocks to get the final class label.

5.4.0.3 Quantitative Results and Analysis

Detailed DDN results on ILSVRC12 are provided in Table 5.4. Error rates are provided for each cluster and it is clear that DDN lowers the error rate and the overall performance is comparable to ResNet-101. ResNet-50 with joint loss (ResNet-50-JL) optimization setup (introduced for reducing the confusion cases across the cluster samples, for details please refer to Section 5.2.3) does improve over the baseline network (ResNet-50), but it is not always guaranteed. The DDN shows further improvement in results when compared to joint-loss indicating that expert networks indeed help in resolving some of the confusion classes. We can observe that error rates are significantly reduced for clusters with large number of classes when compared to clusters with fewer classes. This indicates that the expert network gets benefited if there are more classes and potentially the corresponding cluster can be further split into subclusters adding in more expert networks.

5.5 DDN principle for multi-label classification problem

We showed the applications of DDN principle to address the binary (CDNN) and multi-class (DDN) problem. In this section, we look into extending the underlying DDN principle to address multi-label (also know as the image annotation/tagging problem) classification problem and show its effectiveness on IAPRTC-12 and NUS-WIDE datasets [18].

5.5.1 Extension of DDN

The underlying principle of DDN-annot for addressing the multi-label classification problem remains similar to DDN, except for a few changes. Here the root node is realized by a VGG16 (details in section 2.1.4) network with sigmoid cross entropy loss

Table 5.6: DDN-annot performance on NUS-WIDE dataset with type-1 evaluation in Comparison to other methods.

	NUS-WIDE						**IAPRTC-12**					
Method	K=3			K=5			K=3			K=5		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
WARP [36]	27	45	34	20	57	30	50	27	35	43	38	40
Fast0Tag (net.) [151]	**31**	**52**	**39**	**23**	**65**	**34**	**58**	**31**	**41**	**50**	**44**	**47**
VGG16-annot	27	50	35	20	61	30	46	27	34	40	39	39
DDN-annot	**29**	**51**	**38**	**22**	**63**	**33**	**56**	**31**	**40**	**47**	**43**	**45**

instead of a softmax loss. For the weight initialization, we use the network trained on ImageNet and finetune it with the cross-entropy loss for our problem. Once the root network is trained, we subsequently perform K-means clustering in the feature space (can be any one of the network layers, preferably the last layer), and based on that, we partition the input samples. In our experiment, we choose K to be 4 and it seems to capture the co-existence of labels and makes it relatively easy for validating the idea. Please note that some clusters might end up with only subset of labels with the expectation of capturing coexistence of labels. Similar to the earlier DDN, we build expert networks for each cluster to resolve ambiguities. The expert network consists of the last three convolutional layers along with two fully connected layers built on top of the root node. The expert network gets trained by freezing all the previous layers of the root node. During inference for a given test sample, feature representations are obtained using the root node and then it gets routed to one of the expert networks based on the distance to the cluster centroids. Finally, the expert network outputs a probability distribution over the label set. Here the intuition is that, the probability distribution of the labels predicted by the root node will get refined or in other words reranked by the subsequent expert networks to yield better performance.

Table 5.7: DDN-annot performance on NUS-WIDE dataset with type-2 evaluation in Comparison to other methods.

			NUS-WIDE			**IAPRTC-12**		
Method	Feature		K=5			K=5		
	Visual	Text	P	R	F1	P	R	F1
TagProp(σ)	VGG-16	-	18	41	25	38	30	34
2PKNN	VGG-16	-	18	44	26	38	23	29
SVM-DMBRM	VGG-16	-	23	58	33	58	27	37
CCA	VGG-16	W2V	22	55	31	33	32	33
CCA-KNN	VGG-16	W2V	**26**	**64**	**37**	**45**	**38**	**41**
Below are end-to-end deep learning based models								
Fast0Tag (net.)	-	-	**23**	**65**	**34**	**23**	**65**	**34**
VGG16-annot	-	-	22	55	31	39	33	36
DDN-annot	-	-	**25**	**62**	**36**	**47**	**34**	**39**

5.5.2 Experimental setup

5.5.2.1 NUS-WIDE dataset

NUS-WIDE [18] is one of the largest publicly available multilabel dataset. It originally contains 269,648 images but we were able to retrieve only 223,821 because some of the images were either corrupted or deleted from Flickr. It consists of images with multiple tags that are manually annotated by students (high school or college). The annotation vocabulary consisted of carefully chosen 81 vocabulary tags. We follow the same split as recommended in [18], which ends up with 134, 281 training samples and 89,603 testing samples.

5.5.2.2 IAPRTC-12 dataset

It is a collection of 19,627 images of natural scenes which are split into a training set consisting of 17665 images and a testing set consisting of 1962 images [40]. The label vocabulary consists of 291 labels with an average of 5.7 labels used for annotating each image.

5.5.2.3 Evaluation

In order to have a fair comparison, we follow the standard evaluation metrics as reported in most of the previous work [91, 90, 132, 84, 27, 67] for type-2 evaluation and [151] for type-1 evaluation. In type-1 evaluation, the overall **R**, **P** and **F1** s are computed per image. But, in type-2 evaluation, the **P, R and F1** are computed per tag and this is similar to what is being reported in Chapter-3 experimental results.

5.5.2.4 Quantitative Results

The experimental results of DDN-annot in comparison with other methods on IAPRTC-12 and NUS-WIDE datasets are provided in table 5.6 (with type-1 evaluation) and 5.7 (with type-2 evaluation). Among the end-to-end deep learning based models, our proposed approach DDN-annot does improve the performance in comparison to our baseline VGG16-annot. This indicates that the proposed idea seems to capture the coexistence of labels and expert networks are contributing to the improvement. Among deep learning based models, our proposed method DDN-annot yields competitive results in comparison to (Fast0Tag (net.)). Note that, Fast0Tag (net.) method uses both image and text embeddings to determine the labels, but our method uses only image embeddings and achieves similar results. In comparison to our earlier proposed models (which is a two-stage combining engineered features and statistical model), DDN-annot performance is slightly worse than CCA-KNN method, but it's still competitive and a promising technique. Among all the proposed models, CCA-KNN appears to be the best performing model even though it is a two-stage approach. We believe that the results of DDN-annot can be further improved by selecting an appropriate number of clusters/experts and going deeper than two level. Also during inference, if we could combine predictions with appropriate weights from the root node and the expert network, then we expect the performance to improve further.

In this chapter, we saw DDN principle and it's potential to improve the performance accuracy on both multi-class and multi-label classification problems. In the following chapter, we provide a summary of our work along with the future work that could potentially further enhance the DDN applicability and performance.

CHAPTER 6
CONCLUSIONS AND FUTURE WORK

In this dissertation, we showed the effectiveness of deep learning representation with the following proposed models for addressing the image annotation task.

6.1 SVM-DMBRM

it's a hybrid approach that combined generative DMBRM with discriminative SVM approach to get the best out of both. Initially, we showed SVM-DMBRM using conventional features (over a dozen local and global features) outperforms most of the existing techniques on all four standard image annotation datasets: Corel-5k, ESP-GAME, IAPRTC-12 and NUS-WIDE. Most importantly, this method had the best N+ score (indicates how many labels/tags were used for annotating test images) when compared to all the other models. This indicated the proposed approach was not solving the dataset problem but rather it was more generalizable. SVM-DMBRM even with a single deep learning feature outperformed all the other existing models using the same feature.

6.2 HDD

we took a slightly different approach to solving the image annotation task using the multi-scale hypergraph heat diffusion framework. This enabled us to capture the higher order similarity among multiple images in the feature space and subsequently exploited the topology of the underlying hypergraph. Such topological analysis enabled us to perform simultaneous diffusion of the training labels at multiple scales in the transductive setup thereby addressing the key problem of class imbalance. The method was evaluated on all four datasets, the results were better than most of the existing methods but it was slightly worse when compared to the state-of-the-art 2PKNN method. One of the limiting factor of this approach was eigen decomposition, which made it difficult to scale for larger datasets.

6.3 CCA based models

showed the efficient way to combine deep learning representation for both image and its associated labels. Images were represented using pre-trained (ImageNet data) VGG16 network features and labels were represented by Word2vec (pre-trained neural network on millions of documents learning the relationship between words). Among CCA based models, the two step approach (CCA-KNN) of first finding the semantic neighbors (using K-means over visual features) per test image followed by CCA outperformed all the other existing methods. Overall, we showed that a single deep learning feature performed almost similar to using over a dozen engineered features. This helped us in avoiding the use of computing multiple engineered features and also the computationally expensive process of metric learning (a trend in most recent papers) to determine optimal weights to combine them, thus making it more suitable for the real world applications. The success of deep learning features paved the way for designing an end-to-end deep learning network.

We proposed a generic data driven framework for designing a neural network that could solve all three binary, multi-class and multi-label classification problems more efficiently.

6.4 CDDN

We first validated our idea on a binary classification task and called it cascaded deep decision network (CDDN). The main underlying principle was to separate the hard samples from the easier ones and pay more attention to solving hard cases by training expert networks. CDDN was evaluated on two endoscopic medical image datasets, one dealt with identifying the key frames (containing polyp) from the rest (not containing polyp) of the image frames (continuous video stream of endoscopy) and the other one was brain tumor classification (glioblastoma vs meningioma). Our approach outperformed the classical approaches (engineered features with SVM classifier). Also we provided a detailed study of comparing our approach with the conventional approach of building a neural network. The conventional approaches for building cascaded networks encounters vanishing gradient or data scarcity (while training expert nodes) problem, but in our approach to an extent we mitigate this problem by choosing to do piece-wise training and build the expert network based on the parent node's feature space.

6.5 DDN

We extended the CDDN idea to solve the multi-class classification problem and called it Deep Decision Network (DDN). The underlying principle remained same, but we incorporated some changes that involved identifying clusters (based on root network performance) of classes using spectral clustering. For each cluster, we built an expert network on top of the existing network to resolve the ambiguities. The width (number of clusters/experts) and depth (residual error on validation data) of the network is still completely data driven. DDN yielded state-of-the-art results on CIFAR-10 and CIFAR-100 dataset (at the time of publication of this work [92]). DDN was also shown to scale to larger datasets by testing it on one of the largest publicly available ImageNet dataset. Most importantly, DDN also helps in providing some insights into the data by identifying the most confusing classes and their performances.

6.6 DDN-annot

We further extended the data-driven approach to solve more complex multi-label classification task. The main difference when compared to DDN was that, the clustering was done in the feature space as opposed to DDN (was based on softmax predictions of the root network). We validated the approach on IAPRTC-12 and relatively large NUS-WIDE dataset. We showed that the proposed approach yields competitive performance even though we restricted to only $K = 4$ clusters/experts (we hope it can be further improved by choosing large number of clusters that requires more time and resources).

Overall, the data-driven framework (CDDN, DDN and DDN-annot) is built on the underlying principle of partitioning the data and building expert networks for hard samples that require more attention. Though the main goal here was to improve the classification accuracy, in addition it also provides some insights into the data. Some of the recent work [126, 48, 54] suggests building deeper networks helps improve the classification accuracy, but our method provides an alternative approach for data-driven network design that grows both wide (number of experts at each stage/level) and deep (different stages/levels) depending on the complexity of the problem. In terms of interpretability, there has been some recent work [149, 115, 121] exploring ways to visualize the convolutional filters of the network and try to be more reasonable about network's prediction. Alternatively in our approach we try to provide data interpretability in terms of identifying/grouping the type of error cases that requires more attention.

Most of the work and the papers published based on this dissertation reflect the work carried out until 2017. Since this a rapidly moving field, a lot of progress has been made and thus some of the results and the conclusions might be slightly outdated. For example, the state of the art performance on CIFAR-100 is currently [145, 21, 54] yielding a error rate of 10-17% (which is down by 20% when compared to our published results) using a variety of techniques mostly orthogonal to those presented here. Nevertheless, this dissertation still provides techniques that are applicable even to this date and has some unique abilities. For instance, the data-driven DDN framework has the capability to boost the current performance of any state-of-the-art image classification network by using it as the root node and building a tree like structured network based on its performance/error. This provides a unique opportunity to build an efficient solution by leveraging other researchers efforts and be part of the growth (in designing an efficient network) in the vision community by pushing the boundaries. In addition, it also provides data interpretability (ability to better understand the error cases) especially when working on a problem with little or no domain knowledge (this happens more often in the medical field).

Several recent papers [77, 12, 114, 57] cite our DDN based networks, suggesting that our data-driven approach still remains applicable and are worth exploring further, especially in the field of medicine.

6.7 Future Work

Considering conventional approaches, in the case of SVM-DMBRM, it would be interesting to learn more efficient ways to combine the two models. For the hypergraph approach, one can explore the adaptive techniques for finding the optimal parameters of hypergraph construction like hyperedge weights as well as the parameters of the heat diffusion framework like the scales of diffusion. In terms of improvements to the CCA based models, instead of using pre-trained features if we could design a network to backpropagate the errors then that could certainly make it an end-to-end approach and possibly improve the performance.

Data-driven approaches (DDN based networks) have certain limitations. The training process is a bit involved due to clustering (which is not part of the end-to-end training) and it lacks a good choice for the optimal number of clusters. As future work, if we could incorporate a scheme to determine the number of clusters into our objective function while training the network then that could be of help. Since the network depth depends on the problem/data, as of now piece-wise training looks more appropriate. But, one could take advantage of training the expert networks in parallel. DDN performance can be further improved by going beyond just two stages and using the latest best performing network (as root node). In our setup, we just added one expert network to resolve the cluster, but if it doesn't help then it might be worth exploring stacking more than one expert network to better discriminate the confusion classes. In the current setup, clusters of classes are non-overlapping and that puts lot of pressure on the root network, so it will be interesting to explore the idea of overlapping clusters with a soft clustering technique instead of hard clustering. In the case of DDN-annot, certainly increasing the number of clusters/experts should help better capture the dependencies between labels. Also, it would be interesting to cresentations into this framework.

BIBLIOGRAPHY

[1] Cvpr statistics. `https://bit.ly/2FpNvtm`. Accessed: 2019-03-20.

[2] Facebook statistics. `http://bit.ly/1MsvKaE`. Accessed: 2016-10-22.

[3] Youtube growth statistics. `http://bit.ly/1I38Hxm`. Accessed: 2016-10-22.

[4] Agostinelli, Forest, Hoffman, Matthew, Sadowski, Peter, and Baldi, Pierre. Learning activation functions to improve deep neural networks. *arXiv preprint arXiv:1412.6830* (2014).

[5] Andre, B., Vercauteren, T., Buchner, A. M., Wallace, M. B., and Ayache, N. A smart atlas for endomicroscopy using automated video retrieval. *Med Image Anal 15*, 4 (Aug 2011), 460–476.

[6] Andr, Barbara, Vercauteren, Tom, Perchant, Aymeric, and Buchner, Anna M. Introducing space and time in local feature-based endomicroscopic image retrieval. In *Medical Content-Based Retrieval for Clinical Decision Support*, vol. 5853 of *Lecture Notes in Computer Science*. 2010, pp. 18–30.

[7] Ballan, Lamberto, Uricchio, Tiberio, Seidenari, Lorenzo, and Del Bimbo, Alberto. A cross-media model for automatic image annotation. In *Proceedings of International Conference on Multimedia Retrieval* (2014), ACM, p. 73.

[8] Bar, Yaniv, Diamant, Idit, Wolf, Lior, Lieberman, Sivan, Konen, Eli, and Greenspan, Hayit. Chest pathology detection using deep learning with non-medical training. In *12th IEEE International Symposium on Biomedical Imaging* (2015), pp. 294–297.

[9] Barnard, Kobus, Duygulu, Pinar, Forsyth, David, Freitas, Nando de, Blei, David M, and Jordan, Michael I. Matching words and pictures. *Journal of machine learning research 3*, Feb (2003), 1107–1135.

[10] Baxter, Nancy N., Goldwasser, Meredith A., Paszat, Lawrence F., Saskin, Refik, Urbach, David R., and Rabeneck, Linda. Association of colonoscopy and death from colorectal cancer. *Annals of Internal Medicine 150*, 1 (2009), 1–8.

[11] Bengio, Yoshua, Courville, Aaron, and Vincent, Pascal. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence 35*, 8 (2013), 1798–1828.

[12] Bilal, Alsallakh, Jourabloo, Amin, Ye, Mao, Liu, Xiaoming, and Ren, Liu. Do convolutional neural networks learn class hierarchy? *IEEE transactions on visualization and computer graphics 24*, 1 (2018), 152–162.

[13] Bishop, Christopher M. *Pattern recognition and machine learning.* springer, 2006.

[14] Carneiro, Gustavo, Chan, Antoni B., Moreno, Pedro J., and Vasconcelos, Nuno. Supervised learning of semantic classes for image annotation and retrieval. *IEEE Trans. Pattern Anal. Mach. Intell. 29*, 3 (Mar. 2007), 394–410.

[15] Carneiro, Gustavo, and Nascimento, Jacinto C. Multiple dynamic models for tracking the left ventricle of the heart from ultrasound data using particle filters and deep learning architectures. In *IEEE CVPR* (2010), pp. 2815–2822.

[16] Carneiro, Gustavo, Nascimento, Jacinto C., and Bradley, Andrew P. Unregistered multiview mammogram analysis with pre-trained deep learning models. In *MICCAI* (2015), pp. 652–660.

[17] Carneiro, Gustavo, Nascimento, Jacinto C., and Freitas, António. Robust left ventricle segmentation from ultrasound data using deep neural networks and efficient search methods. In *Proceedings of the 2010 IEEE International Symposium on Biomedical Imaging: From Nano to Macro, Rotterdam, The Netherlands, 14-17 April, 2010* (2010), pp. 1085–1088.

[18] Chua, Tat-Seng, Tang, Jinhui, Hong, Richang, Li, Haojie, Luo, Zhiping, and Zheng, Yan-Tao. Nus-wide: A real-world web image database from national university of singapore. In *Proc. of ACM Conf. on Image and Video Retrieval (CIVR'09)* (Santorini, Greece., July 8-10, 2009).

[19] Chua, Tat-Seng, Tang, Jinhui, Hong, Richang, Li, Haojie, Luo, Zhiping, and Zheng, Yantao. Nus-wide: a real-world web image database from national university of singapore. In *Proceedings of the ACM international conference on image and video retrieval* (2009), ACM, p. 48.

[20] Cristianini, Nello, and Shawe-Taylor, John. *An introduction to support vector machines and other kernel-based learning methods.* Cambridge university press, 2000.

[21] Cubuk, Ekin D, Zoph, Barret, Mane, Dandelion, Vasudevan, Vijay, and Le, Quoc V. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501* (2018).

[22] Cui, Zhen, Chang, Hong, Shan, Shiguang, Zhong, Bineng, and Chen, Xilin. Deep network cascade for image super-resolution. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V* (2014), pp. 49–64.

[23] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In *IEEE CVPR* (2009).

[24] Dhillon, Inderjit S. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining* (2001), ACM, pp. 269–274.

[25] Duygulu, P., Barnard, Kobus, Freitas, J. F. G. de, and Forsyth, David A. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proceedings of the 7th European Conference on Computer Vision-Part IV* (London, UK, UK, 2002), ECCV '02, Springer-Verlag, pp. 97–112.

[26] Duygulu, Pinar, Barnard, Kobus, de Freitas, Joao FG, and Forsyth, David A. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *European conference on computer vision* (2002), Springer, pp. 97–112.

[27] Feng, S. L., Manmatha, R., and Lavrenko, V. Multiple bernoulli relevance models for image and video annotation. In *CVPR'04* (2004), pp. 1002–1009.

[28] Florack, Luc, ter Haar Romeny, Bart M., Koenderink, Jan J., and Viergever, Max A. Scale and the differential structure of images. *Image Vision Computing 10*, 6 (1992), 376–388.

[29] Freund, Yoav, and Schapire, Robert E. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci. 55*, 1 (Aug. 1997), 119–139.

[30] Friedman, Jerome H. Greedy function approximation: A gradient boosting machine. *Ann. Statist. 29*, 5 (10 2001), 1189–1232.

[31] Fu, Hao, Zhang, Qian, and Qiu, Guoping. Random forest for image annotation. In *European Conference on Computer Vision* (2012), Springer, pp. 86–99.

[32] Gerig, Guido, Kuoni, Walter, Kikinis, Ron, and Kübler, Olaf. Medical imaging and computer vision: An integrated approach for diagnosis and planning. In *Mustererkennung 1989*. Springer, 1989, pp. 425–432.

[33] Ghesu, Florin C., Georgescu, Bogdan, Zheng, Yefeng, Hornegger, Joachim, and Comaniciu, Dorin. Marginal space deep learning: Efficient architecture for detection in volumetric image data. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5-9, 2015, Proceedings, Part I* (2015), pp. 710–718.

[34] Girshick, Ross. Fast r-cnn. In *International Conference on Computer Vision (ICCV)* (2015).

[35] Girshick, Ross, Donahue, Jeff, Darrell, Trevor, and Malik, Jitendra. Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv preprint arXiv:1311.2524* (2013).

[36] Gong, Yunchao, Jia, Yangqing, Leung, Thomas, Toshev, Alexander, and Ioffe, Sergey. Deep convolutional ranking for multilabel image annotation. *arXiv preprint arXiv:1312.4894* (2013).

[37] Gonzalez, Rafael C., and Woods, Richard E. *Digital Image Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2008.

[38] Goodfellow, Ian J, Warde-Farley, David, Mirza, Mehdi, Courville, Aaron, and Bengio, Yoshua. Maxout networks. *arXiv preprint arXiv:1302.4389* (2013).

[39] Grangier, David, and Bengio, Samy. A discriminative kernel-based approach to rank images from text queries. *IEEE transactions on pattern analysis and machine intelligence 30*, 8 (2008), 1371–1384.

[40] Grubinger, Michael. *Analysis and evaluation of visual information systems performance*. PhD thesis, Victoria University, 2007.

[41] Guillaumin, Matthieu, Mensink, Thomas, Verbeek, Jakob, and Schmid, Cordelia. Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *In ICCV* (2009).

[42] Guillaumin, Matthieu, Mensink, Thomas, Verbeek, Jakob, and Schmid, Cordelia. Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *International Conference on Computer Vision* (2009), pp. 309–316.

[43] Gupta, Vishal, and Lehal, Gurpreet Singh. A survey of text summarization extractive techniques. *Journal of emerging technologies in web intelligence 2*, 3 (2010), 258–268.

[44] Hanbury, Allan. A survey of methods for image annotation. *Journal of Visual Languages & Computing 19*, 5 (2008), 617–627.

[45] Haralick, Robert M, Shanmugam, Karthikeyan, et al. Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, 6 (1973), 610–621.

[46] Hardoon, David, Szedmak, Sandor, and Shawe-Taylor, John. Canonical correlation analysis: An overview with application to learning methods. *Neural computation 16*, 12 (2004), 2639–2664.

[47] He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *arXiv preprint arXiv:1502.01852* (2015).

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

[49] Heitz, Geremy, Gould, Stephen, Saxena, Ashutosh, and Koller, Daphne. Cascaded classification models: Combining models for holistic scene understanding. In *Advances in Neural Information Processing Systems* (2009), pp. 641–648.

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

[51] Hjelmås, Erik, and Low, Boon Kee. Face detection: A survey. *Computer vision and image understanding 83*, 3 (2001), 236–274.

[52] Hotelling, Harold. Relations between two sets of variates. *Biometrika 28*, 3/4 (1936), 321–377.

[53] Hsieh, William W. *Machine Learning Methods in the Environmental Sciences: Neural Networks and Kernels*. Cambridge University Press, 2009.

[54] Huang, Gao, Liu, Zhuang, Van Der Maaten, Laurens, and Weinberger, Kilian Q. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 4700–4708.

[55] Huang, Yanping, Cheng, Yonglong, Chen, Dehao, Lee, HyoukJoong, Ngiam, Jiquan, Le, Quoc V, and Chen, Zhifeng. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *arXiv preprint arXiv:1811.06965* (2018).

[56] Ioffe, Sergey, and Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* (2015).

[57] Izadyyazdanabadi, Mohammadhassan, Belykh, Evgenii, Mooney, Michael, Eschbacher, Jennifer, Nakaji, Peter, Yang, Yezhou, and Preul, Mark. Prospects for theranostics in neurosurgical imaging: empowering confocal laser endomicroscopy diagnostics via deep learning. *Frontiers in oncology 8* (2018), 240.

[58] Jeon, J., Lavrenko, V., and Manmatha, R. Automatic image annotation and retrieval using cross-media relevance models. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval* (2003), SIGIR '03, pp. 119–126.

[59] Jia, Yangqing, Shelhamer, Evan, Donahue, Jeff, Karayev, Sergey, Long, Jonathan, Girshick, Ross, Guadarrama, Sergio, and Darrell, Trevor. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093* (2014).

[60] Kalayeh, Mahdi M., Idrees, Haroon, and Shah, Mubarak. NMF-KNN: image annotation using weighted multi-view non-negative matrix factorization. In *Computer Vision and Pattern Recognition* (2014), pp. 184–191.

[61] Khalid, Samina, Khalil, Tehmina, and Nasreen, Shamila. A survey of feature selection and feature extraction techniques in machine learning. In *Science and Information Conference (SAI), 2014* (2014), IEEE, pp. 372–378.

[62] Kim, Yoon. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[63] Krizhevsky, Alex. Learning multiple layers of features from tiny images, 2009.

[64] Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, F. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[65] Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In *NIPS* (2012), pp. 1097–1105.

[66] Lacoste-Julien, Simon, Sha, Fei, and Jordan, Michael I. Disclda: Discriminative learning for dimensionality reduction and classification. In *Advances in neural information processing systems* (2008), pp. 897–904.

[67] Lavrenko, V., Manmatha, R., and Jeon, J. A model for learning the semantics of pictures. In *in NIPS* (2003), MIT Press.

[68] Lazebnik, Svetlana, Schmid, Cordelia, and Ponce, Jean. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *null* (2006), IEEE, pp. 2169–2178.

[69] Le, Quoc V., Han, Ju, Gray, Joe W., Spellman, Paul T., Borowsky, Alexander, and Parvin, Bahram. Learning invariant features of tumor signatures. In *9th IEEE International Symposium on Biomedical Imaging: From Nano to Macro, ISBI 2012, May 2-5, 2012, Barcelona, Spain, Proceedings* (2012), pp. 302–305.

[70] LeCun, Yann, Bottou, Léon, Bengio, Yoshua, and Haffner, Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*, 11 (1998), 2278–2324.

[71] LeCun, Yann, and Cortes, Corinna. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/, 2010.

[72] Lee, Chen-Yu, Xie, Saining, Gallagher, Patrick, Zhang, Zhengyou, and Tu, Zhuowen. Deeply-supervised nets. *arXiv preprint arXiv:1409.5185* (2014).

[73] Lee, Joung H., and Sade, Burak. Meningiomas of the central neuraxis: Unique tumors. In *Meningiomas*. 2009, pp. 157–162.

[74] Li, B., and Meng, M. Q.-H. Automatic polyp detection for wireless capsule endoscopy images. *Expert Systems with Applications 39*, 12 (2012), 10952–10958.

[75] Li, B., Meng, M.Q.-H., and Xu, L. A comparative study of shape features for polyp detection in wireless capsule endoscopy images. In *Proc. of IEEE Eng Med Biol Soc* (2009), pp. 3731–3734.

[76] Li, Li-Jia, Wang, Chong, Lim, Yongwhan, Blei, David M, and Fei-Fei, Li. Building and using a semantivisual image hierarchy. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on* (2010), IEEE, pp. 3336–3343.

[77] Li, Xiaoxiao, Liu, Ziwei, Luo, Ping, Change Loy, Chen, and Tang, Xiaoou. Not all pixels are equal: Difficulty-aware semantic segmentation via deep layer cascade. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 3193–3202.

[78] Li, Xirong, Uricchio, Tiberio, Ballan, Lamberto, Bertini, Marco, Snoek, Cees GM, and Bimbo, Alberto Del. Socializing the semantic gap: A comparative survey on image tag assignment, refinement, and retrieval. *ACM Computing Surveys (CSUR) 49*, 1 (2016), 14.

[79] Lin, Min, Chen, Qiang, and Yan, Shuicheng. Network in network. *arXiv preprint arXiv:1312.4400* (2013).

[80] Litjens, Geert, Kooi, Thijs, Bejnordi, Babak Ehteshami, Setio, Arnaud Arindra Adiyoso, Ciompi, Francesco, Ghafoorian, Mohsen, Van Der Laak, Jeroen Awm, Van Ginneken, Bram, and Sánchez, Clara I. A survey on deep learning in medical image analysis. *Medical image analysis 42* (2017), 60–88.

[81] Liu, Wei, and Chang, Shih-Fu. Robust multi-class transductive learning with graphs. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA* (2009), pp. 381–388.

[82] Long, Jonathan, Shelhamer, Evan, and Darrell, Trevor. Fully convolutional networks for semantic segmentation. *arXiv preprint arXiv:1411.4038* (2014).

[83] Makadia, Ameesh, Pavlovic, Vladimir, and Kumar, Sanjiv. A new baseline for image annotation. In *ECCV '08* (2008), pp. 316–329.

[84] Makadia, Ameesh, Pavlovic, Vladimir, and Kumar, Sanjiv. A new baseline for image annotation. In *European Conference on Computer VisionI* (2008), pp. 316–329.

[85] Malinowski, Mateusz, and Fritz, Mario. Learning smooth pooling regions for visual recognition. In *24th British Machine Vision Conference* (2013), BMVA Press, pp. 1–11.

[86] Masoudnia, Saeed, and Ebrahimpour, Reza. Mixture of experts: a literature survey. *Artificial Intelligence Review 42*, 2 (2014), 275–293.

[87] Mikolov, Tomas, Chen, Kai, Corrado, Greg, and Dean, Jeffrey. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[88] Mishkin, Daniel S., Chuttani, Ram, Croffie, Joseph, DiSario, James, Liu, Julia, Shah, Raj, Somogyi, Lehel, Tierney, William, Wong Kee Song, Louis M., and Petersen, Bret T. Asge technology status evaluation report: wireless capsule endoscopy. *Gastrointestinal Endoscopy 63*, 1 (2006), 539–545.

[89] Monay, Florent, and Gatica-Perez, Daniel. Plsa-based image auto-annotation: constraining the latent space. In *Proceedings of the 12th annual ACM international conference on Multimedia* (2004), ACM, pp. 348–351.

[90] Moran, Sean, and Lavrenko, Victor. A sparse kernel relevance model for automatic image annotation. *International Journal of Multimedia Information Retrieval 3*, 4 (2014), 209–229.

[91] Murthy, Venkatesh N., Can, Ethem F., and Manmatha, R. A hybrid model for automatic image annotation. In *Proceedings of International Conference on Multimedia Retrieval* (2014), pp. 369:369–369:376.

[92] Murthy, Venkatesh N., Singh, Vivek, Chen, Terrence, Manmatha, R., and Comaniciu, Dorin. Deep decision network for multi-class image classification. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), 2240–2248.

[93] Nair, Vinod, and Hinton, Geoffrey E. 3d object recognition with deep belief nets. In *Advances in Neural Information Processing Systems* (2009), pp. 1339–1347.

[94] Nasrabadi, Nasser M. Pattern recognition and machine learning. *Journal of electronic imaging 16*, 4 (2007), 049901.

[95] Ngo, Tuan Anh, and Carneiro, Gustavo. Left ventricle segmentation from cardiac MRI combining level set methods with deep belief networks. In *IEEE International Conference on Image Processing, ICIP 2013, Melbourne, Australia, September 15-18, 2013* (2013), pp. 695–699.

[96] Nister, David, and Stewenius, Henrik. Scalable recognition with a vocabulary tree. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on* (2006), vol. 2, IEEE, pp. 2161–2168.

[97] Park, Jin Hyeong, Zhou, Shaohua Kevin, Simopoulos, Costas, Otsuki, Joanne, and Comaniciu, Dorin. Automatic cardiac view classification of echocardiogram. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on* (2007), IEEE, pp. 1–8.

[98] Paszke, Adam, Gross, Sam, Chintala, Soumith, Chanan, Gregory, Yang, Edward, DeVito, Zachary, Lin, Zeming, Desmaison, Alban, Antiga, Luca, and Lerer, Adam. Automatic differentiation in pytorch. In *NIPS-W* (2017).

[99] Paull, P. E., Hyatt, B. J., Wassef, W., and Fischer, A. H. Confocal laser endomicroscopy: a primer for pathologists. *Arch. Pathol. Lab. Med. 135*, 10 (Oct 2011), 1343–1348.

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

[101] Pinheiro, Pedro HO, and Collobert, Ronan. Recurrent convolutional neural networks for scene parsing. *arXiv preprint arXiv:1306.2795* (2013).

[102] Qi, Xiaojun, and Han, Yutao. Incorporating multiple svms for automatic image annotation. *Pattern Recognition 40*, 2 (2007), 728–741.

[103] Quinlan, J. Ross. Simplifying decision trees. *International journal of man-machine studies 27*, 3 (1987), 221–234.

[104] Razavian, Ali S, Azizpour, Hossein, Sullivan, Josephine, and Carlsson, Stefan. Cnn features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on* (2014), IEEE, pp. 512–519.

[105] Razavian, Ali Sharif, Azizpour, Hossein, Sullivan, Josephine, and Carlsson, Stefan. Cnn features off-the-shelf: an astounding baseline for recognition. *arXiv preprint arXiv:1403.6382* (2014).

[106] Russakovsky, Olga, Deng, Jia, Su, Hao, Krause, Jonathan, Satheesh, Sanjeev, Ma, Sean, Huang, Zhiheng, Karpathy, Andrej, Khosla, Aditya, Bernstein, Michael, Berg, Alexander C., and Fei-Fei, Li. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV) 115*, 3 (2015), 211–252.

[107] Schapire, Robert E. The boosting approach to machine learning: An overview. In *Nonlinear estimation and classification*. Springer, 2003, pp. 149–171.

[108] Schmidhuber, Jürgen. Deep learning in neural networks: An overview. *Neural Networks 61* (2015), 85–117.

[109] Schroff, Florian, Kalenichenko, Dmitry, and Philbin, James. Facenet: A unified embedding for face recognition and clustering. *arXiv preprint arXiv:1503.03832* (2015).

[110] Sharif Razavian, Ali, Azizpour, Hossein, Sullivan, Josephine, and Carlsson, Stefan. Cnn features off-the-shelf : An astounding baseline for recognition. In *Computer Vision and Patter Recognition* (2014), pp. –.

[111] Sharif Razavian, Ali, Azizpour, Hossein, Sullivan, Josephine, and Carlsson, Stefan. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (2014), pp. 806–813.

[112] Sharma, Avinash. *Representation, Segmentation and Matching of 3D Visual Shapes using Graph Laplacian and Heat-Kernel*. PhD thesis, 2012.

[113] Shi, Jianbo, and Malik, Jitendra. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence 22*, 8 (2000), 888–905.

[114] Shvets, Alexey A, Iglovikov, Vladimir I, Rakhlin, Alexander, and Kalinin, Alexandr A. Angiodysplasia detection and localization using deep convolutional neural networks. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)* (2018), IEEE, pp. 612–617.

[115] Simonyan, Karen, Vedaldi, Andrea, and Zisserman, Andrew. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* (2013).

[116] Simonyan, Karen, and Zisserman, Andrew. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[117] Sivic, Josef, Russell, Bryan C, Zisserman, Andrew, Freeman, William T, and Efros, Alexei A. Unsupervised discovery of visual object class hierarchies. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* (2008), IEEE, pp. 1–8.

[118] Snoek, Jasper, Larochelle, Hugo, and Adams, Ryan P. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems* (2012), pp. 2951–2959.

[119] Snoek, Jasper, Rippel, Oren, Swersky, Kevin, Kiros, Ryan, Satish, Nadathur, Sundaram, Narayanan, Patwary, Mostofa, Prabhat, Mr, and Adams, Ryan. Scalable bayesian optimization using deep neural networks. In *International conference on machine learning* (2015), pp. 2171–2180.

[120] Socher, Richard, Ganjoo, Milind, Manning, Christopher D, and Ng, Andrew. Zero-shot learning through cross-modal transfer. In *Advances in neural information processing systems* (2013), pp. 935–943.

[121] Springenberg, Jost Tobias, Dosovitskiy, Alexey, Brox, Thomas, and Riedmiller, Martin. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806* (2014).

[122] Srivastava, Nitish, Hinton, Geoffrey, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research 15*, 1 (2014), 1929–1958.

[123] Srivastava, Nitish, and Salakhutdinov, Ruslan R. Discriminative transfer learning with tree-based priors. In *Advances in Neural Information Processing Systems* (2013), pp. 2094–2102.

[124] Sun, Yi, Wang, Xiaogang, and Tang, Xiaoou. Deep convolutional network cascade for facial point detection. In *IEEE CVPR* (2013), pp. 3476–3483.

[125] Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, and Rabinovich, Andrew. Going deeper with convolutions. *CoRR abs/1409.4842* (2014).

[126] Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, and Rabinovich, Andrew. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), pp. 1–9.

[127] Szlam, Arthur D., Maggioni, Mauro, and Coifman, Ronald R. Regularization on graphs with function-adapted diffusion processes. *Journal of Machine Learning Research 9* (2008), 1711–1739.

[128] Szummer, Martin, and Picard, Rosalind W. Indoor-outdoor image classification. In *caivd* (1998), IEEE, p. 42.

[129] Tajbakhsh, Nima, Liang, Jianming, del Noza, Jorge Bernal, and Gurudu, Suryakanth R. Automatic polyp detection challenge in colonoscopy video. international symposium on biomedical imaging, 2015.

[130] Toshev, Alexander, and Szegedy, Christian. Deeppose: Human pose estimation via deep neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on* (2014), IEEE, pp. 1653–1660.

[131] Vailaya, Aditya, Jain, Anil, and Zhang, Hong Jiang. On image classification: City images vs. landscapes. *Pattern Recognition 31*, 12 (1998), 1921–1935.

[132] Verma, Yashaswi, and Jawahar, C. V. Image annotation using metric learning in semantic neighborhoods. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part III* (Berlin, Heidelberg, 2012), ECCV'12, Springer-Verlag, pp. 836–849.

[133] Verma, Yashaswi, and Jawahar, CV. Exploring svm for image annotation in presence of confusing labels. In *BMVC* (2013).

[134] Viola, Paul, and Jones, Michael. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on* (2001), vol. 1, IEEE, pp. I–511.

[135] Viola, Paul, Jones, Michael J, and Snow, Daniel. Detecting pedestrians using patterns of motion and appearance. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on* (2003), IEEE, pp. 734–741.

[136] Von Ahn, Luis, and Dabbish, Laura. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (2004), ACM, pp. 319–326.

[137] Vyborny, Carl J, and Giger, Maryellen L. Computer vision and artificial intelligence in mammography. *AJR. American journal of roentgenology 162*, 3 (1994), 699–708.

[138] Vyborny, Carl J, Giger, Maryellen L, and Nishikawa, Robert M. Computer-aided detection and diagnosis of breast cancer. *Radiologic Clinics of North America 38*, 4 (2000), 725–740.

[139] Wan, Li, Zeiler, Matthew, Zhang, Sixin, Cun, Yann L, and Fergus, Rob. Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)* (2013), pp. 1058–1066.

[140] Wang, Jiang, Yang, Yi, Mao, Junhua, Huang, Zhiheng, Huang, Chang, and Xu, Wei. Cnn-rnn: A unified framework for multi-label image classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016).

[141] Witten, Ian H, Frank, Eibe, Hall, Mark A, and Pal, Christopher J. *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann, 2016.

[142] Xiang, Y., Zhou, X., Chua, T.S., and Ngo, C. A revisit of generative models for automatic image annotation using markov random fields. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2009).

[143] Xiang, Yu, Zhou, Xiangdong, Chua, Tat-Seng, and Ngo, Chong-Wah. A revisit of generative model for automatic image annotation using markov random fields. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on* (2009), IEEE, pp. 1153–1160.

[144] Yakhnenko, Oksana, and Honavar, Vasant. Annotating images and image objects using a hierarchical dirichlet process model. In *Proceedings of the 9th International Workshop on Multimedia Data Mining: held in conjunction with the ACM SIGKDD 2008* (2008), ACM, pp. 1–7.

[145] Yamada, Yoshihiro, Iwamura, Masakazu, Akiba, Takuya, and Kise, Koichi. Shakedrop regularization for deep residual learning. *arXiv preprint arXiv:1802.02375* (2018).

[146] Yan, Zhicheng, Zhang, Hao, Piramuthu, Robinson, Jagadeesh, Vignesh, DeCoste, Dennis, Di, Wei, and Yu, Yizhou. Hd-cnn: Hierarchical deep convolutional neural network for large scale visual recognition.

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

[148] Zeiler, Matthew D, and Fergus, Rob. Stochastic pooling for regularization of deep convolutional neural networks. *arXiv preprint arXiv:1301.3557* (2013).

[149] Zeiler, Matthew D, and Fergus, Rob. Visualizing and understanding convolutional networks. In *European conference on computer vision* (2014), Springer, pp. 818–833.

[150] Zhang, Dengsheng, Islam, Md Monirul, and Lu, Guojun. A review on automatic image annotation techniques. *Pattern Recognition 45*, 1 (2012), 346–362.

[151] Zhang, Yang, Gong, Boqing, and Shah, Mubarak. Fast zero-shot image tagging. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on* (2016), IEEE, pp. 5985–5994.

[152] Zhao, Q., and Meng, M.Q.-H. Polyp detection in wireless capsule endoscopy images using novel color texture features. In *Intelligent Control and Automation (WCICA), 2011 9th World Congress on* (2011), pp. 948–952.

[153] Zheng, Yefeng, Liu, David, Georgescu, Bogdan, Nguyen, Hien, and Comaniciu, Dorin. 3d deep learning for efficient and robust landmark detection in volumetric data. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5-9, 2015, Proceedings, Part I* (2015), pp. 565–572.

[154] Zhou, Dengyong, Huang, Jiayuan, and Schölkopf, Bernhard. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in Neural Information Processing Systems* (2006), pp. –.

[155] Zhou, Shaohua Kevin, Park, JH, Georgescu, Bogdan, Comaniciu, Dorin, Simopoulos, Costas, and Otsuki, Joanne. Image-based multiclass boosting and echocardiographic view classification. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on* (2006), vol. 2, IEEE, pp. 1559–1565.