Which Classifier to apply?: Bayesian Active Learning on ImageNet

Master Thesis
June 7, 2016

by Siddhartha, 11-940-202

Supervisors:
Prof. Andreas Krause
ETH, Zurich
Prof. Dr. Renato Pajarola
University of Zurich

Visualization and MultiMedia Lab
Department of Informatics
University of Zürich

Abstract

In this work, we study the sequential decision making problem where we seek to classify images from ImageNet by querying from a pool of binary classifiers. ImageNet is a widely used benchmark image dataset with a hierarchical structure. We will show how the hierarchical structure of the ImageNet dataset can be used in our framework for classification. Our framework operates in greedy fashion in the following way. Given a test image, the greedy framework sequentially and adaptively selects a subset of classifiers, from a given pool, that are most informative about an image. The classifiers are chosen based on some selected criterion. In our work, we extensively experiment and compare various different criteria for the classifier selection in two broad scenarios, namely, the Naïve Bayes Setting and the Equivalence Class setting. In the Naïve Bayes setting, we try to identify the true label of an image. However, in the equivalence class setting we try to identify the hypothesis group where the true label of the image lies in. In both the scenarios, we consider the tests to be noisy. We show that the results are promising and the framework has the potential to compete with other known techniques in active learning such as MIAL, SVM-MIAL etc. As a baseline, we use the convolutional neural network to compare the performance of our algorithm. We show that in certain cases for the ECD and DRD problem, our framework performs nearly as good as a convolutional neural network.

Contents

1. Introduction

Automating image classification has been a major pre-occupation of machine learners and computer vision scientists for decades. The goal of image classification is to classify an image into a known class, such as people, cars, buildings etc. Various algorithms [Rou13] have been proposed for this purpose and each performs well in a certain setting. Some of the most successful image classification methods in machine learning include support vector machines [AGT07], random forests [BZM07] etc. However, the underlying assumption when applying these techniques is that the number of total categories (classes) an image can be classified into is low. Usually, trying to classify an image becomes practically infeasible when the total number of possible classes is very large. Hence, an inevitable question comes to mind - *how can one classify an image if the total number of classes is very large?*

This question can be explored beyond the image setting. For example, choosing the correct scientific theory among many competing candidates or choosing among many different and expensive medical tests to correctly diagnose some patient's condition. Therefore, a technique is required which would choose the correct hypothesis among many competing ones at lowest cost. Applying a multi-class classifier (e.g. DNN) at test time is expensive when the number of classes is of the order of tens of thousands. Hence, the motivation is to find a cost-effective policy that picks the most informative set of binary classifiers so that we can identify the target class with minimum cost. In machine learning, the area of *Active Learning* is a natural fit for approaching the problem at hand. In active learning [Set10] (also known as "query learning"), the learning algorithm is allowed to query about the data points that it can learn from. Hence, the algorithm performs better with less training. The query (in the form of unlabelled instances) is answered by an *oracle* (e.g. a human annotator). Chapter 2 will give more details about the set-up for general active learning problems.

In this work, we take ideas in active learning to sequentially and adaptively obtain a probability distribution over the hypothesis (labels in our case) space. The notions are general in nature and can be used in many different fields where the hypothesis space is very large. We apply this technique on an image dataset called *ImageNet* [DDS+09]. ImageNet is a dataset organized according to the WordNet hierarchy and each node in the hierarchy is represented by thousands of images. Fig. 1.1 shows an example of ImageNet hierarchical dataset where the hierarchy is organized in a semantic fashion. We use a subset of the full ImageNet dataset used at the ILSVRC 2012 [RDS+15]. The full details are presented in Chapter 6.

The research question explored in this report can be simply stated as follows: *Given an unlabelled data point, a large set of possible hypothesis of the data point and a large set of tests to identify the true hypothesis (or a subset of hypotheses), can we intelligently pick as small number of tests as possible (one at a time) to identify the true hypothesis.* Broadly, there are two types of problems we deal with, in this work. In the first type, we try to find the true hypothesis of a test image i.e. we try to classify the test image into one of many different classes (see Chapter 4). And in the second type of problem, we try to identify a "hypothesis group" that the image might belong to (see Chapter 5). By hypothesis group, we mean a more abstract grouping of a subset of hypotheses together. For example, instead of classifying an image in one of the categories of a breed of dog, we would simply group all the breeds together to make the group "dog". The aim, then, would be to simply find out if the image can be classified as a "dog" or not. For the problem of the former type we use three different criteria namely *mutual information, version space reduction* and *expected error* to sequentially pick the most informative tests. For the latter, we use EC^2 algorithm and a "Noisy-OR construction" to select the appropriate tests. Hence, the final framework used to tackle the stated problem can be summarized as a greedy solution where we initially assume that all possible hypothesis are equally likely for a given data point and then pick tests to con-

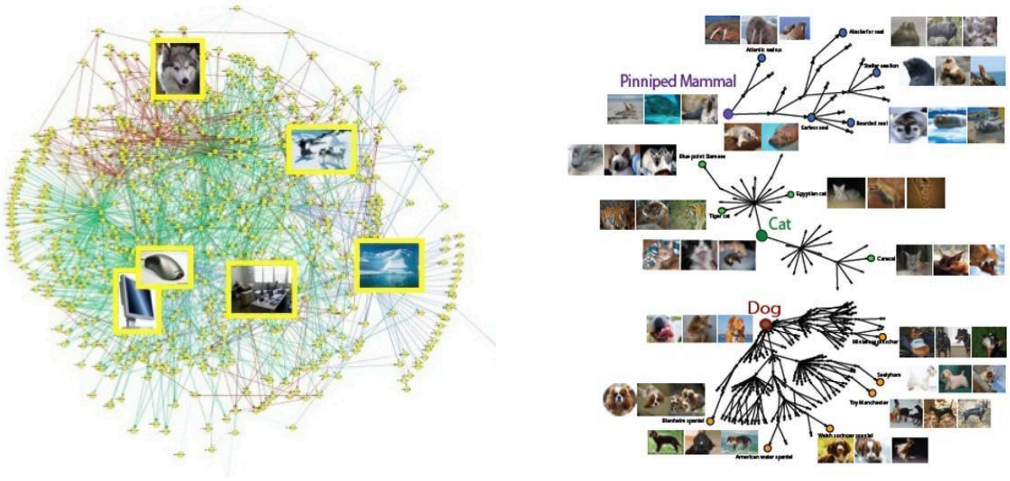


Figure 1.1.: Visualization of ImageNet dataset hierarchy

verge to a solution where the maximum-a-posteriori estimation is ideally the true distribution. We will also go into details about the motivation behind using a greedy algorithm. We will introduce the ideas of *submodularity* and *adaptive submodularity* [KG12], which have been proven to have certain guarantees over functions resulting in near-optimal solutions using simple greedy maximization technique.
It will become evident to the reader that the ideas in the paper are very general in nature and can be applied to wide areas of scientific applications. We will approach the report in a way so as to first define the general framework in a more abstract fashion and then point out how it applies in a dataset like ImageNet.
The report will proceed as follows. Chapter 2 will present the background ideas and an extensive review of related work. We will often mix the two together for better understanding. Chapter 3 will then introduce the mathematical notations, formulate the problem statement and present a greedy framework as a solution. Chapter 4 will then introduce three different criteria when the framework is used in the Naïve Bayes setting. In Chapter 5, we extend the Naïve Bayes assumption to handle groups of hypotheses, known as equivalence classes. Here, we introduce the Equivalence Class Determination Problem (ECD) and show how it fits into our framework. We will further generalize the ECD to Decision Region Determination Problem (DRD) where overlaps between hypothesis groups are allowed. Chapter 6 presents details about the dataset, binary classifier training used as "tests" and the results of the experiments using the above mentioned criteria on ImageNet dataset. Chapter 7 will analyse and point out important future steps for this work. Finally, Chapter 8 will summarize and conclude our work.

Contributions The following are the major contributions we make in our work.

- We use a well-known active learning framework based on sequential information maximization and show how ideas of mutual information, version space reduction and expected error minimization can be applied on ImageNet using the framework.
- We use the ImageNet hierarchy to introduce the notion of equivalence classes on ImageNet, and formulate the active image classification problem as an *Equivalence Class Determination Problem* and *Decision Region Determination Problem.*
- We do extensive empirical studies on the ImageNet dataset in an active learning framework and show that our framework has a potential to compete with some well-known algorithms. In certain cases under an equivalence class setting, our framework performs almost as well as convolutional neural networks.

2. Background & Related work

In this chapter, we will introduce the necessary background ideas essential for reading this work further. We will simultaneously also introduce variety of literature that have attempted to do similar or related work as ours.

2.1. Sequential Information Gathering

The main focus of our work is to explore a cost-effective yet robust approach for fast image class identification on ImageNet. Of course, the exact same framework proposed here can he applied to any type of problem/dataset that fits into this form. Before moving further, we would first like to explain some basic background about *active learning*.
Active learning (also known as "query learning") [Set10] is a sub-field of machine learning which attempts to choose the data from which it "learns". The consequence of this way of learning is that the amount of data needed by the algorithm to learn about a certain class is generally low. This property is highly desirable because, often times, in supervised learning, it becomes very difficult to obtain labelled datasets or labelling is very costly. Some of the examples where this is true includes speech recognition [RHT05], image classification [JPP12] etc. This hurdle of querying vast amounts of labels is reduced in active learning where it queries only limited (often fixed) number of labels from an *oracle*. An oracle is someone or something that we assume knows the correct answer to the queries made by the learning algorithm. An example of an oracle is a *human annotator*. It is usually assumed that the oracle is an infallible body. However, in reality this is almost never true, even if humans act as the oracles [Set11]. We use models in our technique that is capable of integrating the noise of the oracle in our framework.
Broadly, there two categories of active learning: *pool-based* and *stream-based*. In pool-based active learning, we select queries from a large pool of unlabelled instances. This is usually done using uncertainty sampling query strategy i.e. the queries about which the model is least certain is selected to be labelled by the *oracle*. On the other hand, a stream-based active learning first samples the label of an instance from a distribution before deciding if it wants to query the true label of the data or not. Our work falls in the pool-based category.

2.2. Active Learning on hierarchical datasets

There have been few attempts to apply active learning on a hierarchical dataset. Paper by [CZX+12] provides a general framework to classify different classes in a hierarchical dataset. The authors propose *variance* as an uncertainty measure which they apply on a dataset where all the labels in the leaf nodes and non-leaf nodes are embedded into a latent semantic space. The authors try to broaden the idea of "active labels" where the labels are not independent from each other, any more. They apply this method on both real and synthetic data and show that the variance based uncertainty performs better than entropy-based methods and random sampling. However, there are two basic assumptions in this work. The hierarchy is a complete binary tree and the number of labels are very small. Quite often in a real-world dataset, the hierarchy is not a tree but a graph and the number of labels are very large e.g. ImageNet.
The paper by [LKL12] for hierarchical text classification is closer to our research framework but with crucial differences. The paper proposes training binary classifiers for hierarchical nodes for active learning during training whereas our framework performs active learning at test time. The authors propose the

following idea. At each iteration of active learning, classifiers on different categories independently and simultaneously select the most informative examples from the unlabelled pool for themselves, and ask the oracles on the corresponding categories for the labels.
The authors propose the following steps for their active learning framework. They train a binary classifier on each category to distinguish each category from its siblings. Then a local pool is constructed for each classifier, the most informative examples from the unlabelled pool is selected and the corresponding oracles for the labels are queried. Then for each query, the oracle returns a "yes" or "no" answer to confirm (or deny) the category of the unlabelled data. The classification model then gets updated. At each iteration, the process is executed simultaneously on each category, until termination. The hierarchy of the dataset is leveraged in this framework. However, this technique becomes very expensive if the number of classifiers are very large. This, then, puts a computational constraint on the number of labels and the number of queries that can be made. Also, the number of queries required (to obtain an average F-score of 0.60 for four different datasets) is on average 50 per category in the dataset, which is very large.

2.3. Active Learning on ImageNet

In papers by [LHK13, HLYG13, LHK16], the authors introduce active learning on the ImageNet dataset. However, in these papers the number of ImageNet categories used are very small compared to the full dataset, limiting the scalability of their studies.
In [LHK13, LHK16], the authors propose a bayesian framework based on Gaussian Processes for classification. The oracle in their work is assumed to be noisy. Crowd is used as labellers in their work. Specifically, seven different labellers are used per image. They report that active selection of higher quality labellers does not improve the recognition accuracy very much. Note that the number of ImageNet classes used for the experiments in this paper was only three.
In [HLYG13], the authors propose a collaborative active learning framework wherein they assume many different oracles and that all of them are noisy. The experiments, in this paper, are done on 10 ImageNet categories with real crowd-sourced noisy labels. The authors claim to "readily" detect irresponsible labellers online.

2.4. ECD in Active Learning

[BBS10] studied the Equivalence Class Determination Problem (ECD) simultaneously with [GKR10]. An *equivalence class* is a subset of some larger set, where all the element in a subset are grouped together through some relationship known as an *equivalence relation*. The equivalence relations are a special type of relation which are reflexiv, symmetric and transitive [BB99]. In Chapter 6, we will show the way to construct an equivalence class in ImageNet. Essentially, a group of leaf nodes in the hierarchy can be grouped together into a single high-level abstract class to get an equivalence class.
In [BBS10], the authors present a new interpretation of generalized binary search (GBS) from a coding-theoretic perspective by viewing the problem of object identification as constrained source coding. They use this interpretation to consider the case where the objects are partitioned into groups. They show that GBS greedily minimizes the expected number of queries and propose a new algorithm known as Group-Generalized Binary Search (GGBS). The authors also show that the GGBS is equivalent to the decision-tree splitting algorithm based on the entropy impurity measures.
We will now introduce the main idea that is used for the equivalence class setting (Chapter 5), known as *submodularity* and *adaptive submodularity*.

2.5. Submodularity

Submodularity has been a widely studied property in the field of mathematics. Because of the nice mathematical properties of submodular functions, they have been widely accepted in the machine learning community in last few years. Specifically, maximization of submodular functions has been shown to have variety of applications in machine learning. In particular, [KKT03, KGGK06] have done a lot of work with submodular functions in the field of machine learning. We refer the interested readers to [BJ01, JB11] for more details on applications of submodularity in image classification.
A *submodular set function* (or submodular function) is a set function having the following property: *the difference in the gain of the function that a single element makes when added to an input set decreases as the size of the input set increases* (a formal definition can be seen in Appendix A). This diminishing returns property of submodular functions make them highly suitable for applications in machine learning. A classic example to demonstrate the idea is the sensor placement problem [KGGK06]. Consider the example of deploying sensors in a drinking water distribution network in order to detect contamination. In this domain, we may have a model of how contaminants, accidentally or maliciously introduced into the network, spread over time. Such a model, then, can give us the benefit of deploying a sensor over a particular set of locations - typically at the pipe junctions. We usually have a cardinality constraint on the sensors i.e. the number of sensors that can be placed is fixed. The notion of maximizing the marginal benefits in submodular functions can now be used to select a subset of locations in the water distribution network to maximize the utility (e.g. detection area) with fixed number of sensors.
The notion of submodularity is closely related to convexity and convex functions. Submodular functions have some very useful properties [KG12]. For example, a linear combination of submodular functions is submodular. Other properties like submodularity of the residual and submodularity preservation under truncation also make submodular functions very attractive to work with. However, the most important property of submodular functions that make them very effective is that these functions can be greedily maximized, with theoretical guarantees, using a very simple algorithm (see Chapter 3). An extension of submodularity known as *adaptive submodularity* was proposed by [GK10]. This property generalizes the notion of submodularity to a setting using adaptive policies i.e. we select the next action based on the information we already have. The authors prove that the guarantees provided by submodular functions can be generalized to this setting and we can still obtain near-optimal solutions in this setting (see Appendix A for formal definitions).

2.6. Miscellaneous Notions

There are a few other ideas important to our work. All the formal definitions can be found in A.

Entropy and Mutual Information *Entropy* (in information theory) is defined as the uncertainty in the system. In other words, the more certain we become about the information content of the system (e.g. a probability distribution), the lower the entropy becomes. Entropy is symmetric in nature i.e. the value remains unchanged if the outcomes are re-ordered.
A related concept to entropy is the notion of *mutual information*. It is defined as the information-theoretic measurement of mutual dependence between two random variables. It quantifies the amount of information about one random variable through the other. Mutual information is not adaptive submodular but in a noisy case, it has been shown to perform very well [CHKK15].

Version Space If we define the *hypothesis space* as the set of all hypotheses under consideration then *version space* can be defined as a subset of the hypotheses space that is consistent with the training data i.e. they make the correct predictions for all labelled training instances [Set12]. In other words, the version space represents the candidate hypotheses that can explain the observed training data equally well.

Version Space reduction has been widely used as a query selection criterion for active learning. The idea is to minimize the version space every time an active learning algorithm acquires a new training data. When test outcomes are noise-free and binary, version space reduction and maximizing mutual information are equivalent i.e. both the techniques choose the same set of tests.

3. Active Classification Framework

The idea behind the framework (see Chapter 3) is to use a greedy approach to *sequentially* update a prior on the hypothesis space to converge to a probability distribution that reflects the true hypothesis of the test data point (image in our case) [CHKK15]. The framework is *adaptive* in nature meaning that the current update step takes into account the previous observations and chooses the best possible test for the current update. Note that we typically start with a uniform distribution as the prior i.e. we assume all hypotheses are equally likely to be the true hypothesis for the test data point.
In this chapter, we will start by introducing some mathematical notations. The section will then formally define the problem statement and the framework to tackle the problem.

3.1. Mathematical Notation

In this section, we will introduce the mathematical notation that will be used in this report. But first we introduce a dummy hierarchical data (similar to our dataset) to relate the notations to the dataset to enable better understanding.

Fig. 3.1 shows a hierarchy of a toy dataset (note that the nodes in the hierarchy correspond to the ImageNet dataset but are not necessarily a part of the dataset that we use for our experiments). A dataset defined in such a hierarchy has non-leaf nodes that defines an abstraction level for the categories of the leaf nodes. The higher we go in the hierarchy (towards the root), the abstraction of the classes increases proportionally. For example, the leaf nodes are usually very specific categories/classes like different breeds of dogs. The higher we traverse up in the hierarchy, the categories of the leaf nodes get grouped into more general classes like dogs, mammals, animals, living creatures and so on (in the stated progression). Please note that, it is not a requirement that the hierarchy be a tree or be complete. In other words, the nodes in the hierarchy could have multiple parents and that a branch could arbitrarily end at any level of the given structure. In this work, we try to exploit this hierarchy to sequentially and adaptively learn the true label of a given test image. Some attempts have been made to use a hierarchical model for active learning (see Chapter 2).

Notation Table 3.1 lists the symbols used in this report to refer to different aspects of the framework. We also give the details about how the symbols relate to the data for better intuition and understanding of the reader. We, sometimes, use two different symbols for conveying one idea depending on the context. We will also introduce some new notations within the report for communicating an unfamiliar

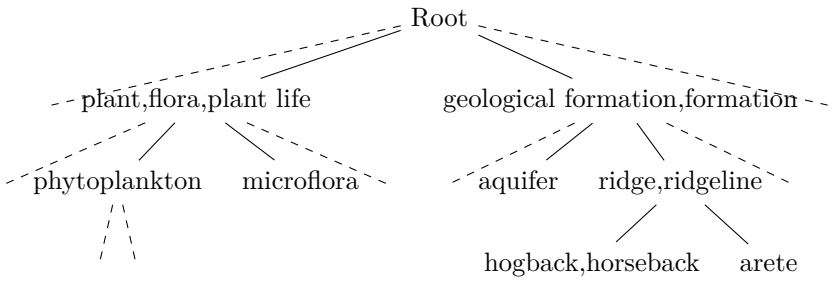


Figure 3.1.: Example of a dataset with a hierarchical structure.

idea. However, these notation will be used solely for that context and should not be confused with any other symbols from the table.

Table 3.1.: Mathematical Notations

Symbol List		
Symbol	Description	Relation to Data
$\mathcal{H}$	set of all hypothesis	*all the leaf nodes in the hierarchy*
H	random variable associated with the hypothesis space	*unknown class of an image*
h	realization of the random variable H	*one leaf node in the hierarchy*
$\mathcal{H}_i$	equivalence class	*a class group containing similar types of image categories*
$\mathcal{T}$	set of all the tests	*binary classifiers for each node in the hierarchy*
e	an index in set $\mathcal{A}$	*index of a binary classifier selected by the greedy framework*
t, T_e	a test from $\mathcal{T}$	*a binary classifier to check if an image belongs to that particular class or not*
$\mathcal{A}$	set of indices of selected tests	*indices of a subset of binary classifiers used to compute the true hypothesis of an image*
d, I	test data point	*test image for which the true hypothesis (or hypothesis group) must be determined*
C	confusion matrix	*probability that a binary classifier produces true, given an image*
$\psi, \mathbf{x}_\mathcal{A}$	observation history	*set of outcomes after selecting and applying classifiers on a test image*
$\mathcal{X}$	domain of observations for all tests	*set of all outcomes when binary classifiers are applied on an image*
x_e	observation after applying T_e	*observation when a binary classifier is applied on image*
π	policy	*criteria for selecting a binary classifier to test an image*
$\mathcal{E}$	edge set	*edges between different image groups in ImageNet*

3.2. Problem Formulation: Pool-based Bayesian Active Problem

Formally, the problem statement can be expressed in the following way:

Given:

Set $\mathcal{H} : \{h_1, h_2, ..., h_n\}$ of hypothesis
Set $\mathcal{T} : \{t_1, t_2, ..., t_m\}$ of tests
Known Prior $\mathcal{P}(\mathcal{H})$: probability distribution over $\mathcal{H}$

Goal:

Given a test data point d, find $\mathcal{A} \subset \mathcal{M}$ (where $\mathcal{M} = \{1, ..., m\}$ & ideally $|\mathcal{A}| \ll m$) such that $\mathcal{P}(\mathcal{H}|\mathbf{x}_\mathcal{A})$ reflects the true distribution. Here, $\mathbf{x}_\mathcal{A}$ are the set of observations after performing

the tests $t \in \mathcal{T}_{\mathcal{A}}$

Defining *hypothesis space* to be the set of all possible hypotheses that constitutes the outcome of a test when applied on a test data point, the main motivation and the subsequent goal of the work is to deal with a hypothesis space that is potentially very large. Therefore, in an ideal situation the cardinality of $\mathcal{A}$ should be much smaller than the cardinality of the full set of possible tests $\mathcal{T}$. In other words, the number of tests we perform should be as small as possible to reach the true probability distribution over the hypothesis space. The notion of pool based active learning fits our requirements and we will now proceed to explain an active learning greedy framework that can potentially be used to achieve this goal.

3.3. The Greedy Selection Framework

3.3.1. The Greedy Update Rule

As the title suggests, the framework proceeds in a greedy fashion i.e. it iteratively picks the best possible test to achieve an acceptable solution. Also, this framework is *sequential* and *adaptive* in nature, meaning that only one (new) test is picked at a given update step and the update step takes into account all the previous observations.
We can formalize the greedy update rule as follows:

Given:

Test $t \in \mathcal{T}_{\mathcal{M}\setminus\mathcal{A}}$
Observation x_e after applying test T_e
Current Posterior $\mathcal{P}(\mathcal{H}|\mathbf{x}_{\mathcal{A}})$ over the hypothesis space

We obtain the posterior $\mathcal{P}(\mathcal{H}|\mathbf{x}_{\mathcal{A}}, \mathrm{x}_e)$ as:

$$\begin{aligned} \mathcal{P}(\mathcal{H}|\mathbf{x}_{\mathcal{A}}, \mathrm{x}_e) \quad = \quad & \mathcal{P}'(\mathrm{x}_e = 1) * \mathcal{P}(\mathcal{H}|\mathbf{x}_{\mathcal{A}}, \mathrm{x}_e = 1) \quad + \\ & \mathcal{P}'(\mathrm{x}_e = 0) * \mathcal{P}(\mathcal{H}|\mathbf{x}_{\mathcal{A}}, \mathrm{x}_e = 0) \end{aligned} \tag{3.1}$$

where $\mathcal{P}'(\mathrm{x}_e)$ is the probability of the observation after applying the test T_e.

3.3.2. Stopping Criteria

In the above described greedy framework, we sequentially obtain a new test at every step and update the probability distribution over the hypothesis space. One question, however, still remains - *when should we stop performing more tests?*
There is no correct answer to this problem. It is a very difficult task to know the step at which the greedy algorithm has updated the prior closest to the true distribution. Another issue with the stopping criteria is that applying too many tests, that are not required, still updates the posterior and we could diverge away from the true distribution. Therefore, after few update steps, one can only hope to have found the true distribution.
In this work, we propose three different criteria for stopping the greedy update:

- *Cardinality constraint*: We fix the number of iterations k per test data point d i.e. $|\mathcal{A}| = k$. This is the most typical approach since we, usually, have a budget on the cost of tests to find an image label. However, the number of updates that should be performed on different test data point d is different. Using this criterion could potentially lead to a lower accuracy.

- *Entropy reduction*: We fix a reduction $r_e \in [0, 1]$ in entropy of the updated probability distribution over hypothesis space as more tests are performed over d i.e $\mathbb{H}_{final} = (1 - r_e)\mathbb{H}_{start}$. This criterion is more flexible and allows for different number of updates over of each d. However, there is no guarantee that a reduction in entropy of the prior distribution by r_e, as a stopping criteria, will lead to the true distribution over the hypothesis space for every test data point d.
- *Change in entropy*: We fix a threshold r_c over the rate of change in entropy for z number of consecutive updates. We also assert that there must be at least i number of updates performed and $z \leq i$. This criteria not only gives more flexibility over the number of updates performed but also makes the reduction in entropy r_e dynamic. This criterion assumes that if the change is entropy after certain number of updates is not significant, then, it indicates convergence towards the true distribution.

Algorithm We now give the algorithm that we use for the greedy update rule.

Result: Posterior $\mathcal{P}(\mathcal{H}|\mathbf{x}_\mathcal{A})$
Input: Test Image I, Prior $\mathcal{P}(\mathcal{H})$, C

$\mathcal{A} \leftarrow \{\}$, $\mathbf{x}_\mathcal{A} \leftarrow \{\}$
while !stop($\mathcal{A}$) **do**
  $e^* \leftarrow \text{argmax}_e\ \Delta(\mathcal{T}, \mathcal{P}(\mathcal{H}), C)$
  $\mathrm{x}_e, \mathcal{P}'(\mathrm{x}_e) \leftarrow T_{e^*}(I)$
  $\mathcal{P}(\mathcal{H}|\mathbf{x}_\mathcal{A}, \mathrm{x}_e) \leftarrow \mathcal{P}'(\mathrm{x}_e = 1) * \mathcal{P}(\mathcal{H}|\mathbf{x}_\mathcal{A}, \mathrm{x}_e = 1) + \mathcal{P}'(\mathrm{x}_e = 0) * \mathcal{P}(\mathcal{H}|\mathbf{x}_\mathcal{A}, \mathrm{x}_e = 0)$
  $\mathcal{A} \leftarrow \{\mathcal{A} \cup e^*\}$
  $\mathbf{x}_\mathcal{A} \leftarrow \{\mathbf{x}_\mathcal{A} \cup \mathrm{x}_e\}$
  $\mathcal{P}(\mathcal{H}) \leftarrow \mathcal{P}(\mathcal{H}|\mathbf{x}_\mathcal{A}, \mathrm{x}_e)$
end

Algorithm 1: The Greedy Update Rule. C is the confusion matrix that is used to encode the noise in the tests $\mathcal{T}$ and $\Delta(\cdot)$ is the score function used to select a test. (see Chapter 6).

In this section, we presented the greedy framework for determining the true hypothesis of a given test data point d. This was done by sequentially applying tests t on d and updating the probability distribution over hypothesis space after each test. However, it was assumed that the tests $t \in \mathcal{T}$ were already provided to us.
In the coming chapters 4 & 5, we will describe different methods by which we obtain these tests for our greedy approach. In chapter 4, we describe a Naïve Bayes setting and try to obtain the true hypothesis for d in this setting. Three different criteria, namely, *mutual information maximization, version space reduction* and *expected error minimization* is used for test selection in this setting. We will go into details about the notions behind each criterion and why it works (or doesn't work). In chapter 5, a different setting based on the equivalence class model will be described. Here we try to determine a group of hypotheses among which the true hypothesis might lie in. We formulate the problem as an *Equivalence Class Determination Problem* and *Decision Region Determination Problem* and show how a hierarchical dataset, as described in Fig. 3.1, can fit into this scenario.

4. Bayesian Active Learning for Classification

In the previous chapter, we presented the general framework for the greedy based sequential update of the distribution over the hypothesis space. To achieve it, we applied tests $\mathcal{T}_{\mathcal{A}}$ over our test data d and observed $\mathbf{x}_{\mathcal{A}}$. The update was done using the *greedy update rule* as shown in Eq. 3.1. In Eq. 3.1, however, the sequence of tests were assumed to be given to us. We will now see how one could select the tests that are most informative about d. In this chapter, we will start by describing a well-known model in active learning to obtain the tests applied on d at each update step.
We are given a pool of tests $\mathcal{T} = \{t_1, t_2, ..., t_m\}$. Typically, m is very large. While the greedy framework runs, we would like to select tests that are most informative about d and we would like to select the fewest number of tests before we stop. The Naïve Bayes model used in our work is described next [CHKK15].

4.1. Naïve Bayes Model

We assume a Naïve Bayes setting i.e. given the hypothesis H, all the tests in $\mathcal{T}$ are independent of each other. Fig. 4.1a depicts the idea. Essentially, it means that we can infer something about H every time we apply a test T_e and observe x_e. We can equivalently represent the Naïve Bayes model as shown in Fig. 4.1b. Here we explicitely model the noise as a random variable N_e (in a noise-free setting, $X_e \equiv D_e$). Each test outcome x_e, now, depends on the hidden variable H and a noise term N_e. Each hypothesis H now goes through a transformation $D_e(H)$ followed by a perturbation by the noise N_e to produce the outcome of a test T_e. In ImageNet dataset, test T_e (see Table 3.1) is a binary classifier confirming (or denying) a hypothesis about an image, D_e is the ground truth outcome of classification and x_e is the observation after applying test T_e. Keeping the Naïve Bayes setting in mind, we will now proceed to formulate the problem statement for this setting.

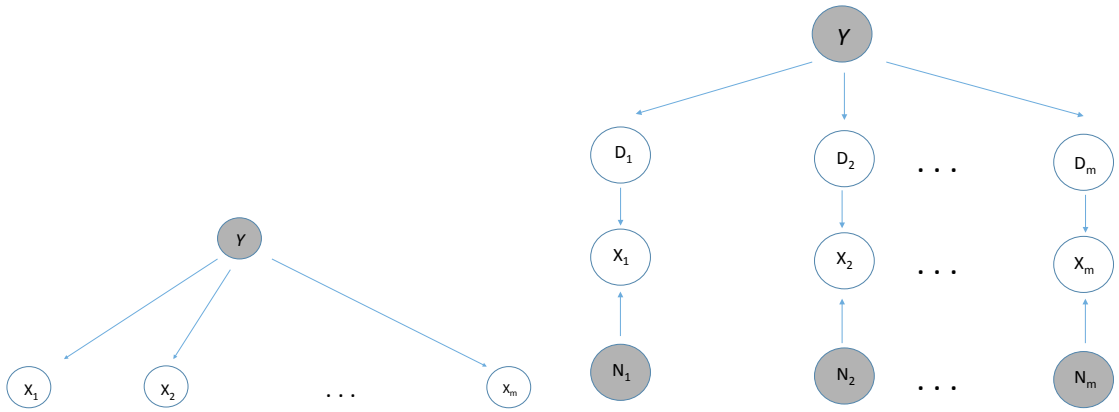
Given:

A hidden random variable H belonging to the set $\mathcal{H} = \{h_1, h_2, ..., h_n\}$.
The distribution of $H \sim \mathcal{P}(H = h)$.

Goal:

To learn the value of H from a subset of observable discrete random variables $T_1, T_2, ..., T_m$.

The main idea is that each $T_e \in \mathcal{T}$ is a test we can perform, which reveals some information about H. The goal is to sequentially and adaptively choose a set of k' tests that is most informative about H. Since our model is based on the Naïve Bayes paradigm, the underlying assumption is that T_e's are conditionally independent given H. Equivalently, we assume that the outcome of each test T_e depends on the hidden variable H and another independent latent variable called that noise N_e (see Fig. 4.1b for details).



(a) The Naive Bayes Model *(b) An equivalent representation*

Figure 4.1.: (a) X_e is conditionally independent give Y, (b) Each D_e is a function of Y. The output of D_e then gets perturbed by the noise N_e and produces a test outcome X_e. In a noiseless case, $X_e \equiv D_e$. [CHKK15]

4.2. Policies

In the above described setting, we consider three different adaptive strategies for picking the tests. We call the strategy that specifies the test to be picked based on the tests picked up previously and their corresponding outcomes as a *policy*. We assign the symbol π to refer to a policy. Suppose we have a sequence of observations (i.e. test pair outcomes or a partial realization) denoted by $\psi \in 2^{\{\mathcal{M},\mathcal{X}\}}$. Then a policy $\pi : 2^{\{\mathcal{M},\mathcal{X}\}} \nrightarrow \mathcal{M}$ is defined as a partial mapping that maps a partial realization to a test to be performed next. Hence, the policy π results in a sequence of k test outcomes which we denote by ψ_π, resulting in, $\psi_\pi \triangleq \{(e_{\pi,1}, x_{\pi,1}), ((e_{\pi,2}, x_{\pi,2}), ..., ((e_{\pi,k}, x_{\pi,k})\}$ [CHKK15]. Please note that we use three different stopping criteria. Therefore, k is not a fixed value for all test data d and depends on the stopping criteria we choose (see Chapter 3.3.2 for full details). Once we observe ψ_π, we obtain a new posterior over H, and consequently the associated entropy $\mathbb{H}(H \mid \psi_\pi)$. The conditional entropy of H given policy π can be defined as follows:

$$\mathbb{H}(H \mid \pi) \triangleq \mathbb{E}_{\psi_\pi}\big[\mathbb{H}(H \mid \psi_\pi)\big] \tag{4.1}$$

In other words, $\mathbb{H}(H \mid \pi)$ is the expected entropy of the posterior of H given the final outcome of policy π.
As already mentioned above, we use three different strategies to choose a test T_e. Next, we will explain each of the three strategies and give theoretical details on why these strategies work.

Proposition 1. *The expected entropy of the distribution over the hypothesis space $\mathcal{H}$ monotonically decreases as we pick and apply new tests.*

The proof of Prop. 1 is easy and there are multiple ways to approach it e.g. using Kullback-Leibler Divergence.

4.2.1. Mutual Information

Mutual information between π and H can be defined as

$$\mathbb{I}(\pi; H) = \mathbb{H}(H) - \mathbb{H}(H \mid \pi) \tag{4.2}$$

Given a sequence of observation ψ, we define the a score function $\Delta(X_e|\psi)$ for performing a test e (where X_e is a random variable associated with the outcome of test T_e). In the case of mutual information, the score function $\Delta_{\text{MI}}(X_e|\psi) \triangleq \mathbb{I}(X_e; H|\psi)$, for selecting a policy π. It is the conditional mutual information between X_e and H, given a posterior distribution over H i.e. $\mathcal{P}(H|\psi)$. Hence, the optimal

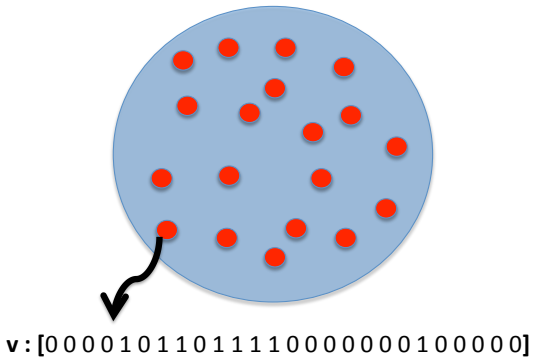


Figure 4.2.: Geometrical Interpretation of a version space $\mathcal{V}$.

policy $\pi^*_{\text{MI}[k]}$ based on mutual information criterion is the policy that achieves the maximal expected mutual information, where

$$\pi^*_{\text{MI}[k]} = \underset{\pi \in \Pi_{[k]}}{\text{argmax}}\ \mathbb{I}(\pi; H) \tag{4.3}$$

where $\Pi_{[k]}$ is the set of policies of length k. The work by [CHKK15] considers k to be fixed. However, in our work, k depends on the stopping criteria chosen.
Obtaining the optimal policy from Eq. 4.2 is, in general, an intractable problem. We, therefore, use a greedy algorithm to compute the optimal policy. The policy $\pi^G_{\text{MI}[k]}$ is then greedily obtained as follows: at iteration $i+1$, $0 \leq i \leq k-1$, a test $T_{e_{i+1}}$ will be picked taking into what was already observed in previous iterations i.e. $\psi_i \triangleq \{(e_1, x_{e_1}), (e_2, x_{e_2}), ..., (e_{i-1}, x_{e_{i-1}})\}$. Hence, we have

$$e_{i+1} = \underset{e \in [m]}{\text{argmax}}\ \mathbb{I}(X_e; H \mid \psi_i) \tag{4.4}$$

[CHKK15] provided the first approximation bound on the performance of the most informative selection policy under persistent noise i.e. repeating the experiment does not lead to a different result. The approximation bound can be stated as the follows: Consider the sequential information maximization problem, where we run the most informative selection policy π^G_{MI} till the length k'. For any $\delta > 0$ and $k \in \mathbb{N}$, we have

$$\mathbb{I}\left(\pi^G_{\text{MI}[k']}; H\right) \geq \left(\mathbb{I}\left(\pi^*_{\text{MI}[k]} H\right) \ - \ \delta\right)\left(1 \ - \ \exp\left(-\frac{k'}{k\gamma \max\{\log n, \log\frac{1}{\delta}\}}\right)\right) \tag{4.5}$$

where $n = |\mathcal{H}|$ is the number of possible values of H, and γ is a constant that only depends on the noise N, concretely: $\gamma = \frac{7}{S_{min}}$, $S_{min} = \min_{e \in [m]} S(W_e)$ where $S(W_e)$ is the the separability of channel W (see Appendix A). Further details and proofs can be found in the same paper by [CHKK15].

4.2.2. Version Space Reduction

Fig. 4.2 shows a depiction of a (dummy) version space. Each $v \in \mathcal{V}$ is a vector in $\{0,1\}^m$, where m is number of tests that could be applied on a hypothesis.

$$v_i = \begin{cases} 1 & \text{if test } t_i \text{ is consistent with the hypothesis,} \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

The space is cut every time a new test is performed. The aim is to reduce the space till we become certain about some hypothesis $h \in \mathcal{H}$. Hence, we choose a policy that maximally cuts the version space.

$$\pi^*_{\text{VS}[k]} = \underset{\pi \in \Pi_{[k]}}{\text{argmax}}\ \Delta_{\text{VS}}(\pi|\phi) \tag{4.7}$$

where $\Delta_{\text{VS}}(\pi|\phi)$ is a score function indicating the reduction in version space after applying a policy π, starting from an empty set.
Just like the mutual information case (see Section 4.2.1), choosing the correct policy by maximally reducing the version space is also intractable. The optimal policy $\pi^*_{\text{VS}[k]}$ is, then, greedily computed as follows: At iteration $i+1$, we define a reduction $\Delta_{\text{VS}}(X_{e_{i+1}})$ by a test $T_{e_{i+1}}$ (we observe $X_{e_{i+1}}$ after applying test $T_{e_{i+1}}$) as

$$\delta_{\text{VS}}(X_e|\psi_i) = \sum_{v \in \mathcal{V}} P(v|\psi_i) - \sum_{v} P(v, X_e|\psi_i) \tag{4.8}$$

where $\psi_i \triangleq \{(e_1, x_{e_1}), (e_2, x_{e_2}), ..., (e_{i-1}, x_{e_{i-1}})\}$. The score of a test $T_{e_{i+1}}$ then is defined in expectation as

$$\Delta_{\text{VS}}(X_{e_{i+1}}|\psi_i) = \mathbb{E}\left[\delta(X_e|\psi_i)\right] \tag{4.9}$$

The selection policy for the test then picks a test T_e that maximally cuts the version space. In other words, a test $T_{e_{i+1}}$ is picked if it maximizes $\Delta_{\text{VS}}(T_{e_{i+1}}|\psi_i)$.

$$e_{i+1} = \underset{e \in [m]}{\text{argmax}}\ \Delta_{\text{VS}}(X_{e_{i+1}}|\psi_i) \tag{4.10}$$

4.2.3. Expected Error

So far, we introduced two different notions of selecting the "best" instance of a test. The selection of a test was based on their uncertainty or their ability to reduce the hypothesis space. Often times, we do not care about the model uncertainty or correctness of its hypothesis but only care about how well the model makes predictions. We would, then, like to select test instances that reduces the future error most (once the answer is known). However, the problem evident in this approach is that the model does not know the answer before asking the question. Consequently, it also does not know the error value. Hence, in this approach, we try to minimize the *expected error* value. To compute the expected error, we require two probability distributions [Set12]: (1) The probability of the oracle's label y in answer to query label x; and (2) the probability that the learner will make an error on some other instance x' once the answer is known. The aim is, then, to reduce the expected error after a test is picked. Hence, a policy π is chosen if it minimizes the expected error of the model.

$$\pi^*_{\text{EE}[k]} = \underset{\pi \in \Pi_{[k]}}{\text{argmin}}\ \Delta_{\text{EE}}(\pi|\phi) \tag{4.11}$$

where $\Delta_{\text{EE}}(\pi|\phi)$ is a score function indicating the expected error of the model for a policy π, starting from an empty set. Similar to mutual information criteria and version space reduction problem, computing the optimal policy is intractable. Hence, we employ a similar greedy policy π^G_{EE} as for the mutual information and version space reduction criteria. The policy works as follows: Let the *expected error* of the model be indicated by $\Delta_{\text{EE}}(X_{e_{i+1}})$ after we apply a test $T_{e_{i+1}}$ at iteration $i+1$ (we observe $X_{e_{i+1}}$ after applying test $T_{e_{i+1}}$). The score of a test $T_{e_{i+1}}$ can then be defined as

$$\Delta_{\text{EE}}(X_{e_{i+1}}|\psi_i) = \mathbb{E}\left[\delta_{\text{EE}}(X_e|\psi_i)\right] \tag{4.12}$$

where $\psi_i \triangleq \{(e_1, x_{e_1}), (e_2, x_{e_2}), ..., (e_{i-1}, x_{e_{i-1}})\}$ is the history of all the observation made till iteration i and

$$\delta_{\text{EE}}(X_e|\psi_i) = \sum_{e' \in \mathcal{M}} P\left(\mathrm{x}_{e'} \neq \underset{\mathrm{x}_{e'}}{\text{argmax}} P(X_{e'}|\mathrm{x}_e, \psi_i)\right) \tag{4.13}$$

where $\mathcal{M}$ is the indices of pool of all possible tests that can be applied. So, a test $T_{e_{i+1}}$ is greedily picked only if it minimizes the expected error.

$$e_{i+1} = \underset{e \in [m]}{\text{argmin}}\ \Delta_{\text{EE}}(X_{e_{i+1}}|\psi_i) \tag{4.14}$$

In this chapter, we introduced three different criteria that can be greedily applied in a Naïve Bayes setting to select the "best" test for computing the optimal policy, to identify the true hypothesis. In the next chapter, we introduce the a new class of problem known as the *Equivalence Class Determination Problem* (ECD). Here, we try to locate the group where the true hypothesis lies instead of identifying the true hypothesis itself. The Naïve Bayes setting of this chapter can be seen as a special case of this class of problems, wherein each group consists of a single hypothesis. We will also introduce another set of problem called the *Decision Region Determination Problem* (DRD) where we further generalize the ECD to look at cases when the hypothesis groups overlap.

5. Bayesian Active Learning for Group Identification

In the previous chapter, all the strategies were aimed at finding the true hypothesis of a test data point d (image in our case). Often when classifying an image, instead of identifying the true hypothesis of an image, we would like to determine the group where the true hypothesis of d lies. For example, in case of ImageNet, instead of classifying the object in the image as a breed of dog, sometimes it is enough to determine that the object in the image is a "dog".
In this chapter, we introduce the class of problem that precisely aims at trying to identify the group (this group of hypothesis is known as an *equivalence class* [BB99]) rather than the class of the object in an image. Typically, we assume that all the equivalence classes are already provided to us. In case of ImageNet, the hierarchy of the dataset is exploited to obtain these equivalence classes (see Chapter 6.1.1).
Now we introduce two methods for determining the hypothesis group of a test data point. The first problem known as the *Equivalence Class Determination Problem* or *ECD* assumes that there are no overlaps between equivalence classes. However, this is not always the case. We, then, extend the idea of group identification to a more general problem known as *Decision Region Determination Problem* or *DRD*. DRD can be seen as a generalized version of ECD, where overlaps between equivalence classes are allowed. In other words, the former problem can be seen as a special case of the latter.
In the following sections, we introduce some new notations to explain the ideas properly but also keep some old ones from previous chapters.

5.1. Equivalence Class Determination Problem

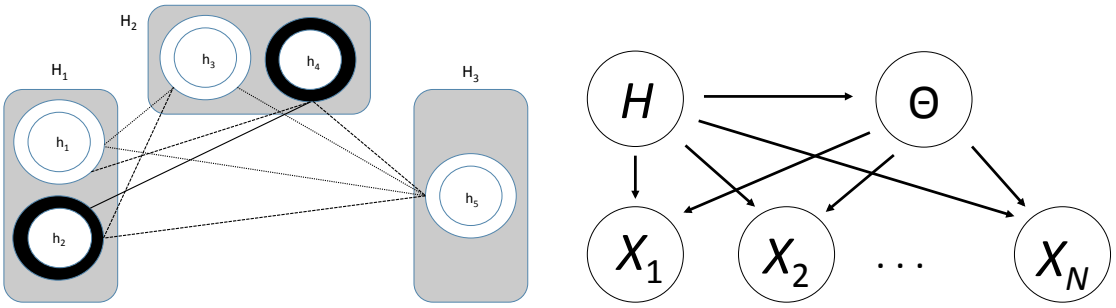
The notion of an equivalence class problem in Bayesian active learning termed as *Equivalence Class Determination Problem* (ECD) was introduced by [GKR10]. The authors also introduced an algorithm for a noisy setting, where the tests applied were not always perfect, termed as EC^2. As stated by the authors, ECD can be formulated as follows:
Given:

Set $\mathcal{H}$ of hypothesis partitioned into equivalence classes $\{\mathcal{H}_1, ..., \mathcal{H}_m\}$ such that $\mathcal{H} = \biguplus_{i=1}^{m} \mathcal{H}_i$

Goal:

Equivalence class $\mathcal{H}_i$, where the true hypothesis $h \in \mathcal{H}$ of d lies in.

If we denote a set of hypotheses consistent with event Λ by $\mathcal{H}(\Lambda)$, then $\mathcal{H}(\Lambda)$ is known as the *version space* associated with Λ (see Chapter 2). Upon termination of the ECD, we require that $\mathcal{H}(\mathbf{x}_{\mathcal{A}}) \subset \mathcal{H}_i$.
We now present the EC^2 algorithm, proposed by [GKR10], to tackle the ECD problem. The basic idea is as follows: edges are introduced between hypothesis in different classes and different tests outcomes allow us to cut inconsistent edges. The aim is to remove all the inconsistent edges while minimizing the expected cost incurred. Formally, it can be stated in the following manner.
A set of edges $\mathcal{E} = \cup_{1 \leq i < j \leq m} \{\{h, h'\} : h \in \mathcal{H}_i, h' \in \mathcal{H}_j\}$, consisting of unordered pair of hypotheses belonging to distinct equivalent classes. A test t under the true hypothesis h cuts an edge $\mathcal{E}_t(h) = \{\{h', h''\} : h' \neq h \text{ or } '' \neq h\}$. In other words, for an edge $\{h, h'\} \in \mathcal{E}$ to be cut i.e. eliminated from the version space, at least one of the hypothesis must be ruled out by a test t under the true hypothesis h. Then a weight function $\omega : \mathcal{E} \to \mathbb{R}_{\geq 0}$ is defined as $\omega(\{h, h'\}) := P(h) \cdot P(h')$. The objective function f_{EC}



(a) Equivalence Class Determination Problem *(b) Error Model*

Figure 5.1.: (a) ECD with 3 equivalence classes and edges between class hypotheses (b) the underlying noise model.[GKR10]

is defined as

$$f_{EC}(\mathcal{A}, h) := \omega\left(\bigcup_{t\in\mathcal{A}} \mathcal{E}_t(h)\right) \tag{5.1}$$

where, $\omega(\mathcal{E}) := \omega_{e\in\mathcal{E}'}(e)$. Then the optimal policy greedily maximizes the objective f_{EC}, to obtain the class where the true hypothesis lies.
Fig. 5.1a shows an example with three different equivalence classes and edges between the hypotheses of each equivalence class. The aim of the ECD is to cut all the inconsistent edges every time we apply a test. In the case of Fig. 5.1a, the edges $\{h_1,h_4\}$ & $\{h_2,h_5\}$ is always cut, the edges $\{h_1,h_3\}$ & $\{h_1,h_5\}$ are cut when the test outcome is 1, otherwise edge $\{h_2,h_4\}$ is cut (given that, black hypothesis are consistent with test outcome 1 and white hypothesis is consistent with test outcomes 0).
The authors in [GKR10] prove approximation guarantees and should be read for full details. We will only state the most important findings here.

Proposition 2. *The objective function f_{EC} is strongly adaptively monotone (Appendix A.5) and adaptively submodular (Appendix A.7).*

This means that we can exploit the diminishing returns property of submodular functions [KG12] to provide strong performance guarantees for the greedy algorithm w.r.t. f_{EC}.
Suppose $P(h)$ is rational $\forall h \in \mathcal{H}$. For the adaptive greedy policy π^G_{EC} implemented by EC^2 it holds that

$$\text{cost}(\pi^G_{EC}) \leq (2\ln(1/p_{min}) + 1)\,\text{cost}(\pi^*_{EC}) \tag{5.2}$$

where $p_{\min} := \min_{h\in\mathcal{H}} P(h)$ is the minimum prior probability of any hypothesis, and π^*_{EC} is the optimal policy for the Equivalence Class Determination Problem. Here, $\text{cost} : \mathcal{T} \to \mathbb{R}_{\geq 0}$ and is defined as $\text{cost}(\pi) = \mathbb{E}_{\psi_\pi}\left[\sum_{(e_\pi,\mathrm{x}_\pi)\in\psi_\pi} \text{cost}(e)\right]$. The paper also proposes approximation algorithms for better runtime. However, the accuracy suffers in that case.
Please note that, since our tests are noisy in nature, ruling out an edge completely is not possible. Instead, we interpret the EC^2 as only updating the edge weights after a test is picked and applied. Finally, when the stopping criterion is reached, MAP estimation is used for predicting the correct hypothesis group. In other words, we simply choose the hypothesis group to be the one having the maximum probability i.e. the sum of probability of all the hypothesis in this group is maximum.
Unlike the version space reduction technique, also known as the generalized binary search or GBS, the EC^2 provides theoretical guarantees over the solution. GBS does not take into account any information about a hypothesis group when selecting a test. This could lead to an arbitrarily poor performance on part of the generalized binary search. However, in the boundary case when the number of equivalent classes is equal to the number of hypothesis, GBS and EC^2 are equivalent.

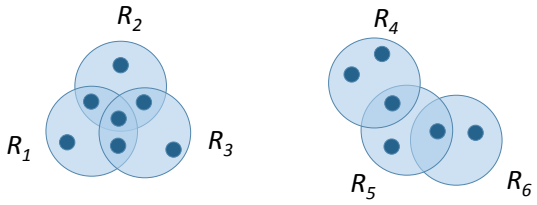


Figure 5.2.: The Decision Region Determination Problem

5.2. Decision Region Determination Problem

The previous section introduced the *Equivalence Class Determination Problem* and how it applies to ImageNet. However, the underlying assumption in ECD was that all the equivalence classes were independent i.e. there were no overlaps among different classes. In real-world datasets, this is usually not true. For example, ImageNet has a hierarchical structure in which many nodes have multiple parents. Translating this hierarchy into equivalence classes eventually leads to a situation where multiple equivalence classes have overlaps.
In this section we describe a problem known as *Decision Region Determination Problem* (DRD), which is designed to deal with a situation when equivalence classes have overlapping hypotheses. Fig. 5.2 shows an example with six equivalence classes, or regions R_i. The figure depicts the situation when the regions have overlapping hypotheses. To run the EC^2 algorithm we require edges between hypotheses of different regions to be cut in order to select the correct region. It becomes a problem in situations as shows in Fig. 5.2. This is because to preserve the guarantees on the performance of the algorithm, there cannot be an edge between a hypothesis with itself which could later be cut after we perform a test. This notion of DRD has been extensively dealt in [JCK$^+$14] & [CJK$^+$15]. In our approach of the DRD, the central idea is to separate the overlapping classes in a way so as to decompose the overlapping problem into multiple instances of non-overlapping ECD. Then EC^2 is applied on each instance and the results are combined with an "OR" logic [CJK$^+$15]. We now state the problem more formally.

Given:

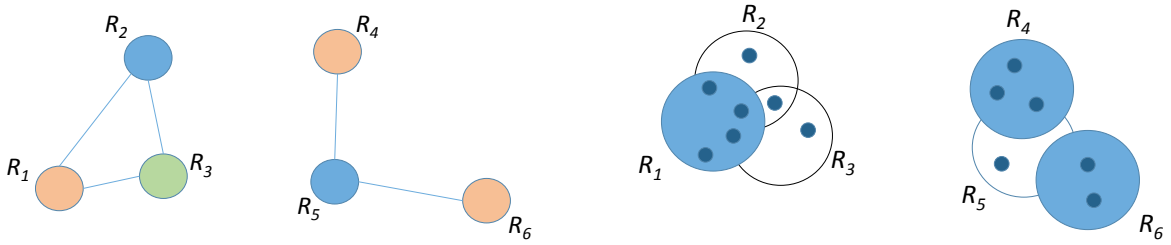
A set of hypothesis $\mathcal{H} = \{h_1, ..., h_n\}$
A random variable H with a distribution $\mathcal{P}$ over $\mathcal{H}$
A set of tests $\mathcal{T}$
A collection of subsets $R_1, ..., R_q \subseteq \mathcal{H}$ called *regions*, where $\mathcal{R} = \{R_1, ..., R_q\}$

Goal:

If variables $X_1, ..., X_n$ would result in outcomes $\mathbf{x}_\mathcal{T}$, we obtain a set of observations, denoted as $\mathcal{S}(\pi, \mathbf{x}_\mathcal{T}) \subseteq \mathcal{T} \times \mathcal{X}$ by running policy π till termination. We would like to obtain

$$\pi^*_{DRD} \in \underset{\pi}{\operatorname{argmin}}\ \text{cost}(\pi), s.t.\ \forall h \exists d\ :\ \mathcal{H}(\mathcal{S}(\pi, h)) \subseteq R_d \tag{5.3}$$

Here, $h \in \mathcal{H}$ and $\mathcal{H}(\mathbf{x}_\mathcal{A}) = \{h' \in \mathcal{H} : (i, \boldsymbol{x}) \in \mathbf{x}_\mathcal{A} \Rightarrow T_i(h') = \boldsymbol{x}\}$ is the set of hypothesis consistent with $\mathbf{x}_\mathcal{A}$. In other words, we try to compute the policy π^*_{DRD} of minimum cost, which adaptively picks tests , observes their outcomes $X_i = T_i(H)$, where $H \in \mathcal{H}$ is the unknown hypothesis, such that upon termination, there exists at least one region that contains all the hypothesis consistent with all the observations made by the policy and $T : \mathcal{H} \to \mathcal{X}$. Please note that in this work, we consider the cost to be equal for all tests.
We use the Noisy-OR construction as proposed by [CJK$^+$15] to solve the DRD problem. It works in the



(a) Applying graph colouring to Fig. 5.2 *(b) Single instance of ECD*

Figure 5.3.: Using Fig. 5.2 as the reference DRD problem : (a) is the result of applying the graph colouring algorithm to the resulting graphs from the DRD, (b) single instance of ECD with 7 disjoint equivalent classes.

following manner:
Suppose there are q possible regions: $|\mathcal{R}| = q$. For each of the q regions, one graph is constructed. The role of graph i is to determine whether the unknown hypothesis h^* is contained in the region R_i. Each model graph i is modelled as an ECD problem, with one of the regions being R_i. The remaining set of hypothesis $\mathcal{H} \setminus R_i$ is divided into a collection of *subregions*, such that within each subregion all hypotheses are contained in exactly the same collection of regions from the original DRD problem. By this process, we get that all subregions are disjoint and find ourselves with a well-defined ECD problem. The EC^2 algorithm can then be used to solve each instance of the ECD. The q optimization problems can then be joined together using the logical "OR" operation as follows. Denote the EC^2 objective function of graph i as f^i_{EC}, and assert $f^i_{\text{EC}}(\phi) = 0$ when there is no observation and $f^i_{\text{EC}}(\phi) = 1$ when all edges are cut. Then, combining the objective function $f^1_{\text{EC}}, ..., f^m_{\text{EC}}$ using the *Noisy-OR formulation* results in

$$f_{\text{DRD}}(\mathbf{x}_{\mathcal{A}}) = 1 \; - \; \prod_{i}^{m} \left(1 \; - \; f^i_{EC}(\mathbf{x}_{\mathcal{A}})\right) \tag{5.4}$$

The advantage of this formulation is stated in Prop. 3. We defer to [CJK+15] for full details on the matter.

Proposition 3. *f_{DRD} is strongly adaptive monotone (Appendix A.5), and adaptively submodular (Appendix A.7) w.r.t. $\mathcal{P}$.*

Hence, f_{DRD} can efficiently greedily optimized. The authors also prove a performance guarantee on f_{DRD}. Let q be the number of regions, and π_{DRD} be the adaptive greedy policy w.r.t. the objective function in Eq. 5.4. The it holds that

$$\text{cost}(\pi^G_{DRD}) \leq (2q\ln(1/p_{\min}) \; + \; 1)\,\text{cost}(\pi^*_{DRD}) \tag{5.5}$$

where $p_{\min} = \min_{h \in \mathcal{H}} \mathcal{P}[h]$ is the minimum prior probabilities of any set of observations, and π^*_{DRD} is the optimal policy for DRD problems.

Graph Colouring to improve the bounds The main motivation behind this method is that as long as we can guarantee at least one ECD instance contains each region, Eq. 5.4 is still remains valid for the problem. Graph colouring is used to select the set of regions for all ECD instance. More formally, an undirected $\mathcal{G} := \{\mathcal{R}, \Xi\}$ over all regions, an edge is established between any pair of overlapping regions. In other words, two regions R_i and R_j are adjacent in $\mathcal{G}$ if and only if $h \in R_i \cap R_j$. Now, finding the minimal set of non-overlapping region sets that covers all regions is equivalent to solving a graph colouring problem. In other words, the graph colouring algorithm transforms the scenario where regions overlap into a new set of regions that are non-overlapping. At the same time, it guarantees that all the regions (and sub-regions) in the overlapping case are included. The goal is, hence, to colour the graph $\mathcal{G}$, such that no two adjacent vertices share the same colour, using as few colours as possible. Denote the number of colours used as r. Then, all the regions of the same colour are used to construct one

ECD problem eventually resulting in r different instances, and then we use the Noisy-OR formulation to combine the r objective functions.
Fig. 5.3 shows how the graph and one EC^2 instance is found for the DRD example in Fig. 5.2. The single instance of EC^2 shown in this figure is for the orange nodes in the two graphs from the figure. This instance has seven disjoint regions, namely $R_1, R_4, R_6, R_2 \setminus (R_1 \cup R_3), R_3 \setminus (R_1 \cup R_2), (R_2 \cap R_3) \setminus R_1$, and $R_5 \setminus (R_4 \cup R_6)$. Similarly, the other EC^2 instances are created for the remaining colours in the graph. It can be shown that combining such EC^2 instances via a "Noisy-OR" construction, one can obtain better theoretical guarantees than provided by Theorem 5.2. We state the bound below and refer the interested readers to [CJK+15]. Let π_{DRD} be the adaptive greedy policy w.r.t. the objective function Eq. 5.4, which is computed over ECD problem instances obtained via graph colouring. Let r be the number of colours used. Then it holds that

$$\text{cost}(\pi^G_{DRD}) \leq (2r \ \ln(1/p_{\min}) \ + \ 1)\, \text{cost}(\pi^*_{DRD}) \tag{5.6}$$

where $p_{\min}$ is the minimum prior probability of any set of observations, and π^*_{DRD} is the optimal policy. Finding the minimum number of colours in the graph colouring problem is NP-hard. However, there are efficient greedy algorithms to colour the graphs, given that every node can be efficiently coloured with at most one colour more than the maximum vertex degree using a greedy colouring algorithm [WP67].

6. Experiments

6.1. ImageNet Dataset

For our work, we use a dataset known as ImageNet[1] [DDS+09]. ImageNet is a database of images organized according to the WordNet[2] hierarchy. Each node is depicted, on an average, by five hundred images. As of today, ImageNet has over 14 million images. In our work, we select a subset of these images. This subset was first used for the *ImageNet Large Scale Visual Recognition Competition* or ILSVRC 2012 (and every year since) [RDS+15]. ImageNet is a benchmark dataset and has been largely used in deep learning research [ARK10, Ben09] on images. In particular, ImageNet is the most popular dataset when working with a specific kind of deep neural networks known as *convolutional neural network* [KSH12, AK15]. In section 6.2, we will explain how we use these ideas on ImageNet during our experiments.
Henceforth, "dataset" will refer to the ILSVRC 2012 subset of the full ImageNet, which we use for our experiments. This dataset has a hierarchical structure with 1000 leaf nodes and 860 internal nodes i.e. 1860 nodes in total. The total number of images in the dataset is over 1.2 million. The number of images per node ranges between 700 & 1300 and is organized in 19 different levels, root node being the first. The hierarchy is not a *tree* but a *graph*, in the sense that some nodes in the structure have multiple parents. The structure is also not *complete* i.e. branches end arbitrarily at any level between 3 and 19.

6.1.1. Obtaining Equivalence Classes

Equivalence Classes (Chapter 5) in case of ImageNet is obtained by exploiting the hierarchical nature of the dataset. The dataset has 19 levels of hierarchy, the first level being the root node and the nineteenth being the leaf nodes. We will consider the index 0 for the root node and 18 for the last level of the tree. We define an equivalence class as the set of all leaf nodes l_i that could be reached starting from some internal node n_i. Since the structure of ImageNet is a graph, often, the equivalence classes are overlapping. To formulate the problem as an *Equivalence Class Determination Problem*, we remove the overlaps arbitrarily i.e. randomly assign the overlapping hypothesis to one of the equivalence classes to satisfy the independence assumption. In one boundary case, there could be 1000 equivalence classes, each class having one unique leaf node l_i. In another boundary case, there could be a single equivalence class with all the leaf nodes in it. This happens when we choose the root node to be the equivalence relation. This boundary condition is irrelevant to our problem.
We construct the equivalence class in the following way: the user inputs a hierarchy level (between 1 and 18). All the nodes at this level are taken as the starting point and we traverse down till we reach the leaf nodes. All the possible leaf nodes that could then be reached from one of the nodes at the chosen level together form one equivalence class. The process is repeated till all the nodes at the chosen level are covered. At this point, there could be some leaf nodes not covered by any of the equivalence classes - since the dataset hierarchy is not *complete*. We then assign a unique equivalence class for each leaf node not yet covered. Table 6.1 shows the number of equivalence classes that get constructed by choosing different levels of the hierarchy.
The above method is equivalent to the following technique. We first grow each incomplete branch in the dataset by adding a single child each time, till we reach the last level of the hierarchy. We repeat this process for each incomplete branch till we make the hierarchy *complete* i.e. all the branches end at last level of the hierarchy. We can then select a hierarchy level (between 1 and 18) and all the nodes at this

[1] http://image-net.org/
[2] https://wordnet.princeton.edu/

level become our equivalent classes. For each node at the selected level, we traverse till we reach the last level of the hierarchy and all the leaf-nodes that we reached constitutes the set that corresponds to that equivalence class.

Table 6.1.: Number of Equivalence classes per level of hierarchy

Equivalence Class List			
Level	Number of Equivalence Classes	Level	Number of Equivalence Classes
1	2	10	664
2	6	11	768
3	13	12	796
4	23	13	828
5	47	14	878
6	103	15	918
7	260	16	977
8	423	17	999
9	594	18	1000

6.2. Tests for ImageNet

We discussed Chapter 3 assuming the tests were given to us. Then in Chapters 4 & 5 we described different policies that could be used to select the "right" tests from a given pool of tests $t \in \mathcal{T}$. In this section, we show how we create the "test pool" and also encode the noise of the test pool in our framework.

Typically, when working with ImageNet, the hierarchy of the dataset is not exploited to improve the detection/classification. In our work, we try to use the hierarchical structure of the dataset to improve the performance of the greedy framework. Hence for the experiment, we make 1860 tests in total i.e. one per node in the dataset. This allows us to get to the true hypothesis faster using smaller number of tests. Each test is essentially a binary classifier that is trained as one-vs-all and informs us about a hypothesis (or a set of hypotheses in case of non-leaf nodes). More specifically, we train a classifier for each node in the dataset hierarchy resulting in 1860 different classifiers. We bias each binary classifier to minimize the false negatives. This is done by balancing the data for both positive and negative classes (positive class being the hypothesis or a set of hypotheses that the classifier is trained for, negative being all other hypotheses). We select all the images from the "positive" hypothesis set and a subset from the "negative" hypothesis set to balance the positive set (since negative set is usually larger) and train our binary classifier for this subset of images. However, this might result in higher false positive for each test. In case of leaf nodes, the classifier (in probability) confirms (or denies) whether a test image I belongs to that hypothesis that the classifier is designed for. And in case of parent (internal) nodes, a classifier would determine if the test image I lies within a set of hypothesis (since internal nodes, usually, represents a set of hypotheses).

To train the binary classifiers, we use *Logistic Regression* and *Random Forest* on top of Caffe[3] [JSD+14], a Convolutional Neural Network (CNN) based framework. We chose these methods rather than SVM[4] because the probabilistic interpretation of the output is more straightforward. We will not go into details about convolutional neural networks in this report but just give the motivation of its usage in our work. The readers can refer to [Ben09] for full details on CNN.

Convolutional Neural Networks have the ability to represent an image as a feature vector automatically.

[3] http://caffe.berkeleyvision.org/

[4] platt scaling can be used for interpreting SVM output as probability

Since we use very large and extremely diverse set of images in our work, traditional feature engineering is not a viable technique. Using a deep learning framework like Caffe helps us in the feature extraction part by automating this process. One could choose to use other frameworks like Tensorflow[5], Theano[6] etc. Once the features have been extracted, we can simply use standard machine learning algorithms (e.g support vector machines, random forests, logistic regression) to train classifiers over them [RASC14, YCBL14, AK15].
We choose to use Caffe since many different CNN models, pre-trained on ImageNet, are available on the internet. We use the VGG[7] models [CSVZ14] for extracting the features and as baseline. Specifically, we use the VGG_CNN_S for feature extraction, making tests over them and as the state-of-the-art baseline to compare the performance of our framework. One is free to use larger VGG models for this purpose or another deep model like GoogleNet.

Table 6.2.: Classification rate for the pool of binary classifiers (2000 randomly selected validation images)

Classifier	Classification rate
Random Forest	78.40%
Logistic Regression	93.85%

6.2.1. Encoding Noise

The models that we proposed for our experiments considered the noise associated with our tests as a part of their formulation i.e. the noise is assumed to be embedded when we run our framework. The way we encode it into our framework is via *confusion matrix*. The confusion matrix is a likelihood table where each row is a probability distribution $\mathcal{P}(X_t = 1|\mathcal{H})$ where $t \in \mathcal{T}$. This results in a likelehood table $C \in \mathbb{R}_{[0,1]}^{1860\times1000}$. This implies that each cell in the table indicates a conditional probability $P(X_t = 1|h)$ where X_t is the observation after applying a test t and $h \in \mathcal{H}$ is a hypothesis.
To obtain the values in table C, we use the data that classifier i was trained on and find the mean probability p_m over each training image being classified as a true positive. For the leaf nodes ($1 \leq i \leq 1000$), $C[i, i] = p_m$ and the rest of the values in the row are $1 - p_m$. But for the internal nodes ($1001 \leq i \leq 1860$), if p_i indicates the set of nodes associated with i (that classifier i is consistent with) then $C[i, p_i] = p_m$ and the rest of the row then is $1 - p_m$. Table 6.3 shows a snapshot of the confusion matrix. The likelihood values from the table can then be used to easily obtain posterior distribution over the hypothesis space $\mathcal{H}$ (in expectation).

6.2.2. Evaluation Criteria

We use three different ways to evaluate and compare our algorithms, namely, *precision-at-k plots*, *entropy reduction plots* and *precision-recall curves*.

Precision-at-k

Given a posterior distribution, *precision-at-k* looks at "top k peaks" of the distribution to check if the true label of a test image lies within these top k peaks or not.

Entropy reduction

This plot is used to confirm Proposition 1. It also gives us an insight into the relevance of the selected tests by looking at the nature of the curve.

[5]https://www.tensorflow.org/
[6]http://deeplearning.net/software/theano/
[7]https://github.com/BVLC/caffe/wiki/Model-Zoo

Table 6.3.: Snapshot of Confusion Matrix $\mathcal{P}(\mathbf{x} = 1|\mathcal{H})$

	$\mathcal{H}_1$	$\mathcal{H}_2$	$\mathcal{H}_3$	...	...	$\mathcal{H}_{1000}$
$\mathbf{x}_1$	0.92	0.08	0.08	...	...	...
$\mathbf{x}_2$	0.068	0.932	0.068	...	...	...
⋮		⋮		⋮		
$\mathbf{x}_{1002}$	0.95	0.95	0.05	...	...	...
⋮		⋮		⋮		
⋮		⋮		⋮		
$\mathbf{x}_{1860}$	...		...		...	

Precision-Recall

Precision is the fraction of retrieved instances that are relevant, while *Recall* is the fraction of relevant instances that are retrieved. Once a set of information is retrieved based on some pre-defined criterion, we can define precision and recall in the following manner:

$$\text{precision} = \frac{\text{true positive}}{\text{true positive} + \text{false positive}}, \text{recall} = \frac{\text{true positive}}{\text{true positive} + \text{false negative}} \tag{6.1}$$

We plot the Precision-Recall curve in the following way: We set a threshold value on the posterior distribution of each test image. Then, True positive = number of true labels retrieved given the threshold i.e. all the true labels that have a peak in the distribution greater than the threshold, true positive + false positive = total number of labels retrieved for all images, true positive + false negative = number of images. This notion can be extended to the equivalence class setting by replacing the "true labels" with the "true label group".

6.3. Results

In this section, we will show some results of the experiments that we performed on ImageNet. The algorithmic details can be seen in Chapter 3. In Section 6.3.1, we show results for the Naïve Bayes based models and in Section 6.3.2, results for the equivalence class is presented.

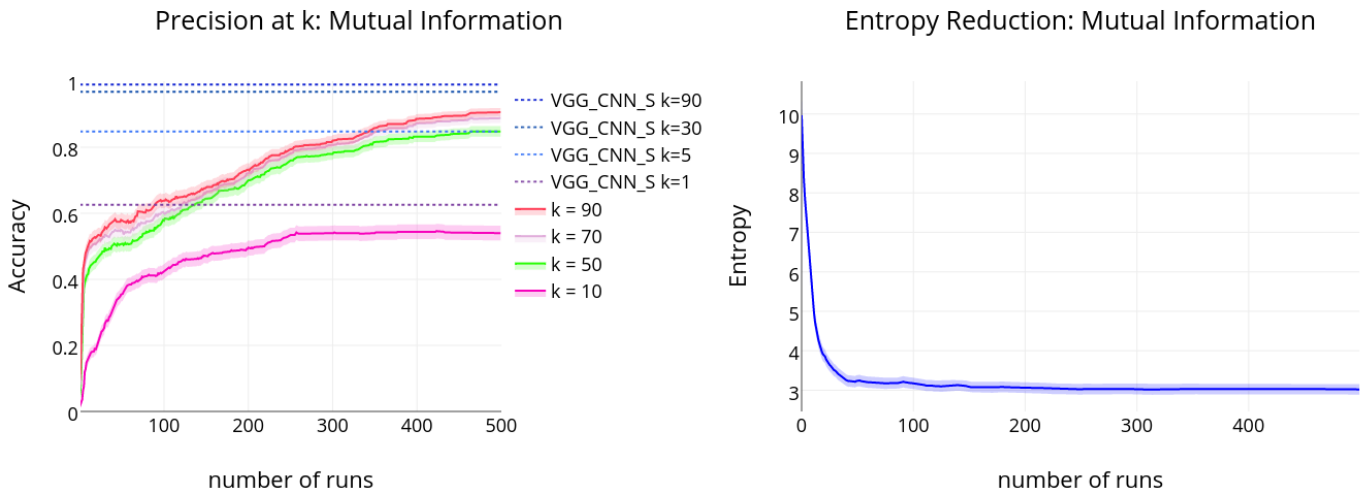
6.3.1. ImageNet Classification

Experiment Setup We use the ILSVRC 2012 validation dataset consisting of 50000 images. We randomly pick images from the validation set to conduct experiments on ImageNet. The binary classifiers trained using logistic regression are used as tests.
Since deep learning based techniques give the state-of-the-art performance, we use a convolutional neural network VGG_CNN_S as our hard baseline. As a soft baseline, we compare our algorithms to the (naive) "random sampling" algorithm where the tests are picked at random.
Here, we first start by showing results for each criteria separately and then compare them together at the end.

Stopping criteria: Cardinality constraint

Mutual Information *Number of test images*: 500, *Number of tests:* 500

We see from Fig. 6.1a that the performance of mutual information criteria increases rapidly with



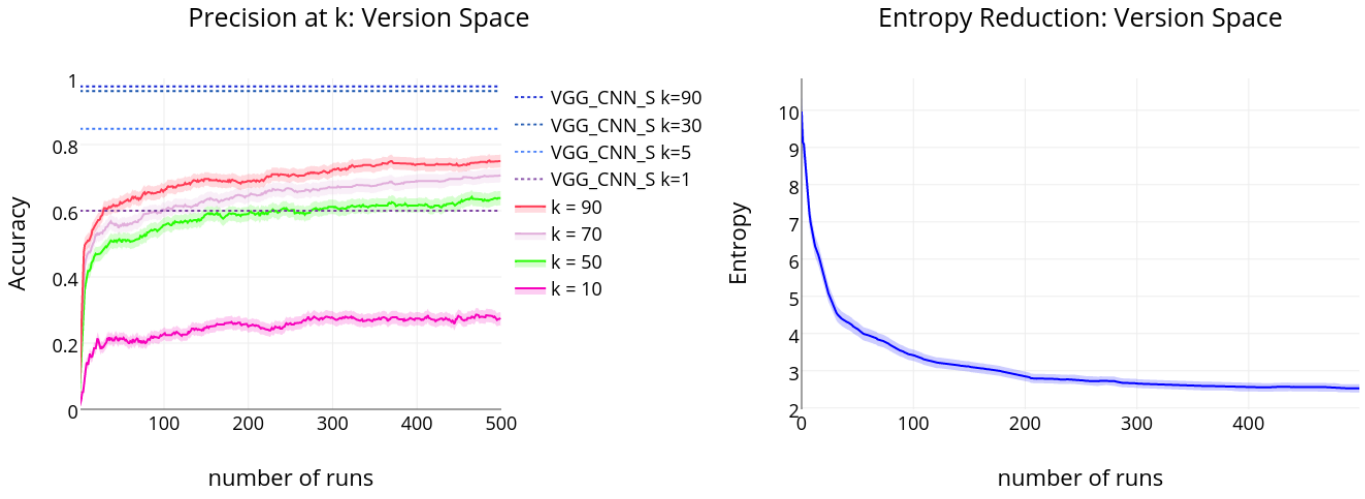
(a) Precision-at-k plot for accuracy *(b) Entropy Reduction*

Figure 6.1.: Plots for mutual information as the selection criteria.

increase in k. However, the overall accuracy of deep learning based technique is still much higher. Another interesting observation from Fig. 6.1b is the steep drop in entropy within the first 50 tests and almost a flat curve for rest of the tests. This seems to imply that the first 50 tests (approximately) give us majority of the tests that are most informative about our test image set.

Version Space Reduction *Number of test images*: 500, *Number of tests:* 500

The version space reduction method has a lower accuracy compared to the mutual information based



(a) Precision-at-k plot for accuracy *(b) Entropy Reduction*

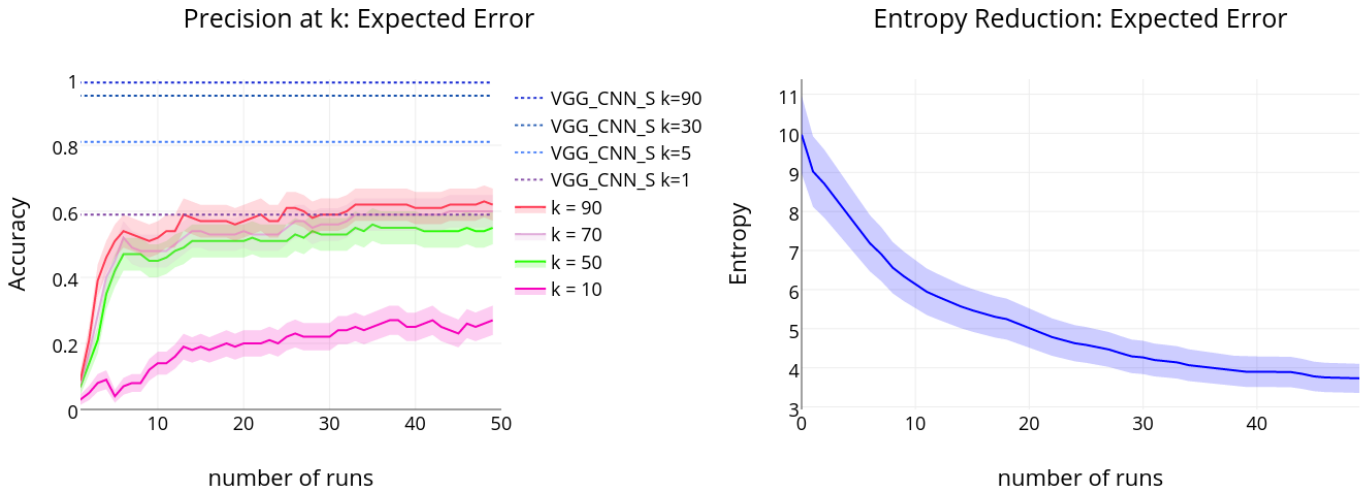
Figure 6.2.: Plots for version space as the selection criteria.

criterion. However, in terms of computation time, version space based criteria runs the fastest. Also, the entropy reduction is more gradual indicating that each new test selected based on this criterion is

less informative (compared to mutual information) and, hence, requires more number of tests to reduce to the same entropy level as mutual information.

Expected Error *Number of test images*: 100, *Number of tests:* 50

This criterion is drastically more expensive compared to the version space and mutual information



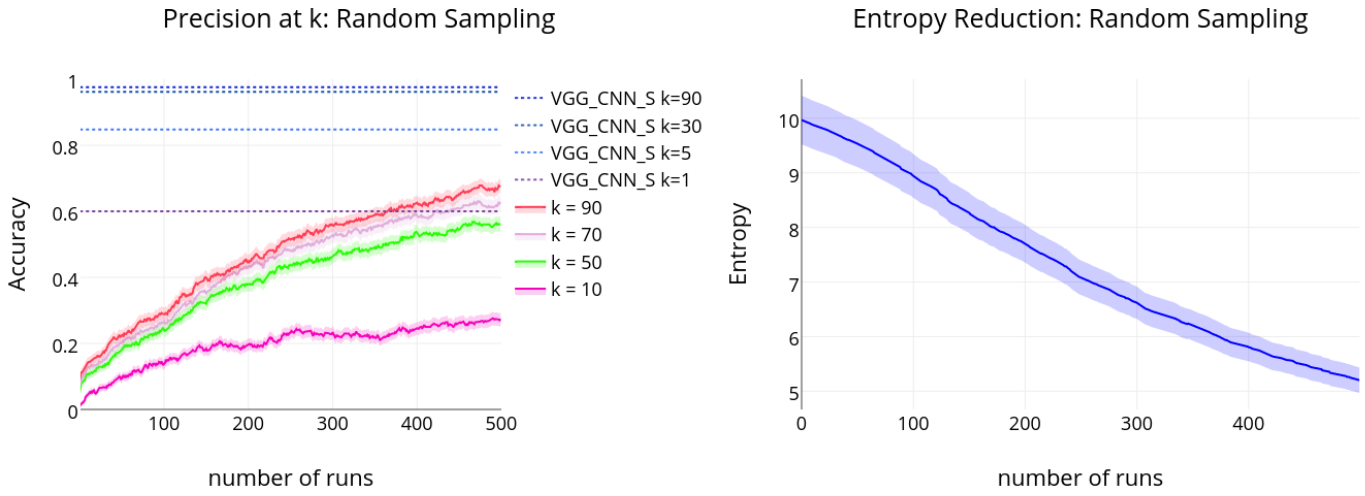
(a) Precision-at-k plot for accuracy *(b) Entropy Reduction*

Figure 6.3.: Plots for expected error as the selection criteria.

criteria. Therefore, we use less number of images and fewer tests to plot the performance curves. We observe the same trend as for the other criteria. The entropy reduction is more gradual.

Random Sampling *Number of test images*: 500, *Number of tests:* 500

This is the naive algorithm where the tests are picked randomly. As one can notice, the entropy reduction

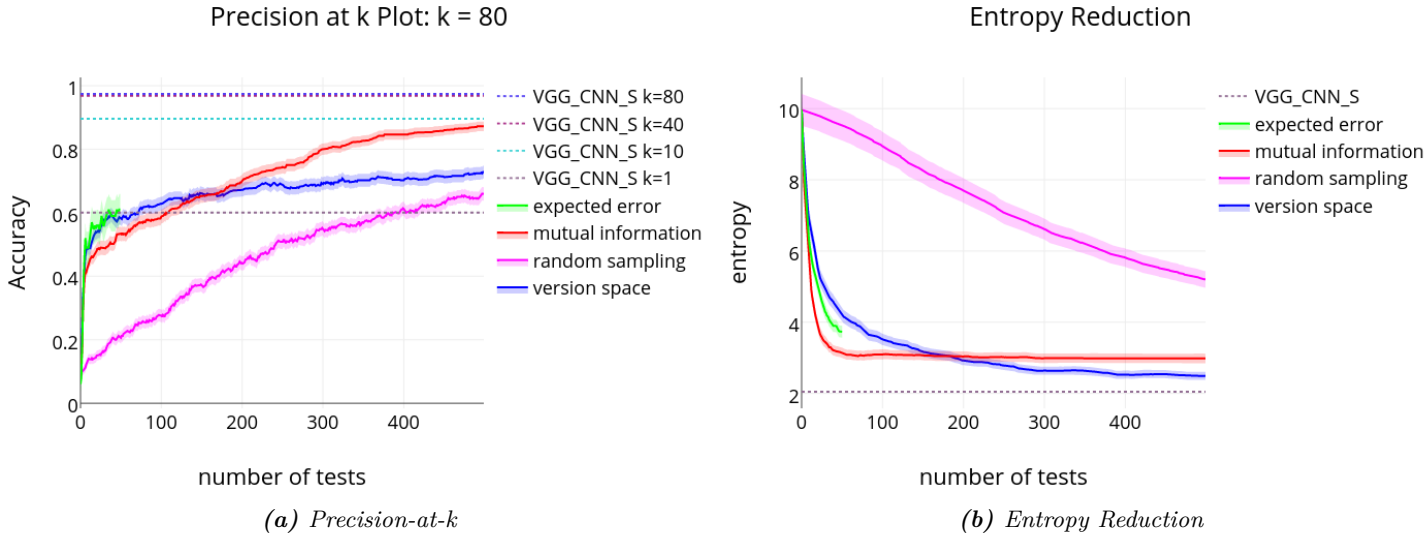


(a) Precision-at-k plot for accuracy *(b) Entropy Reduction*

Figure 6.4.: Plots for naive random sampling as the selection criteria.

is much slower and will require many more tests to reduce to a level achieved by more intelligent algorithms. The accuracy is also very low compared to other, more intelligent criteria.

Comparison From Fig. 6.5, we can clearly see that our learning algorithms perform much better than the naive idea of a random sampling.. Mutual Information seems to perform best as we increase the number of tests.



(a) Precision-at-k *(b) Entropy Reduction*

Figure 6.5.: Comparison plots for accuracy and entropy for the Naïve Bayes setting

Precision-Recall We now present the precision-recall plot for the 4 different test selection criteria we use in our greedy framework. Note that the *precision-recall* plot for the expected error curve will perform (at least) as good as the random sampling if we pick same number of tests as for the random sampling method.

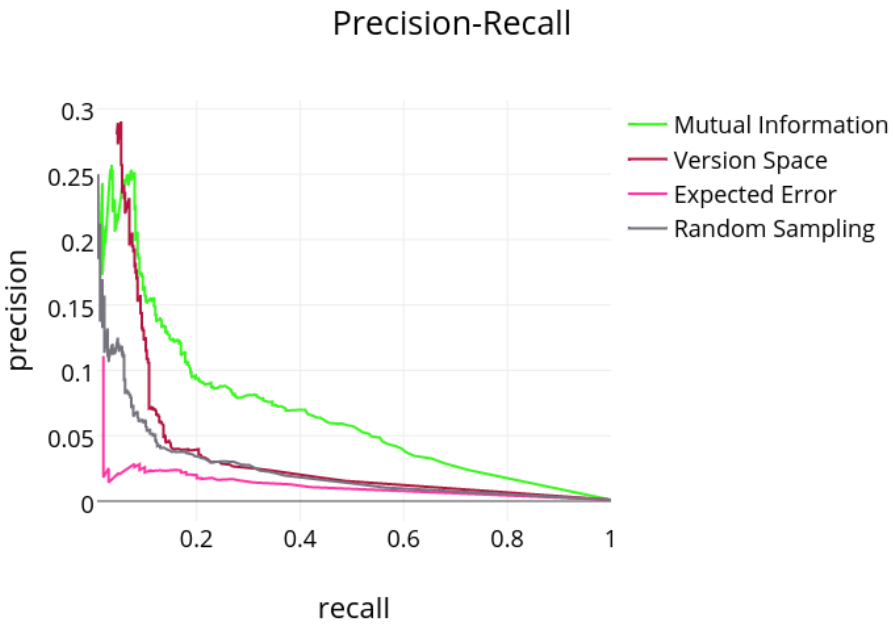
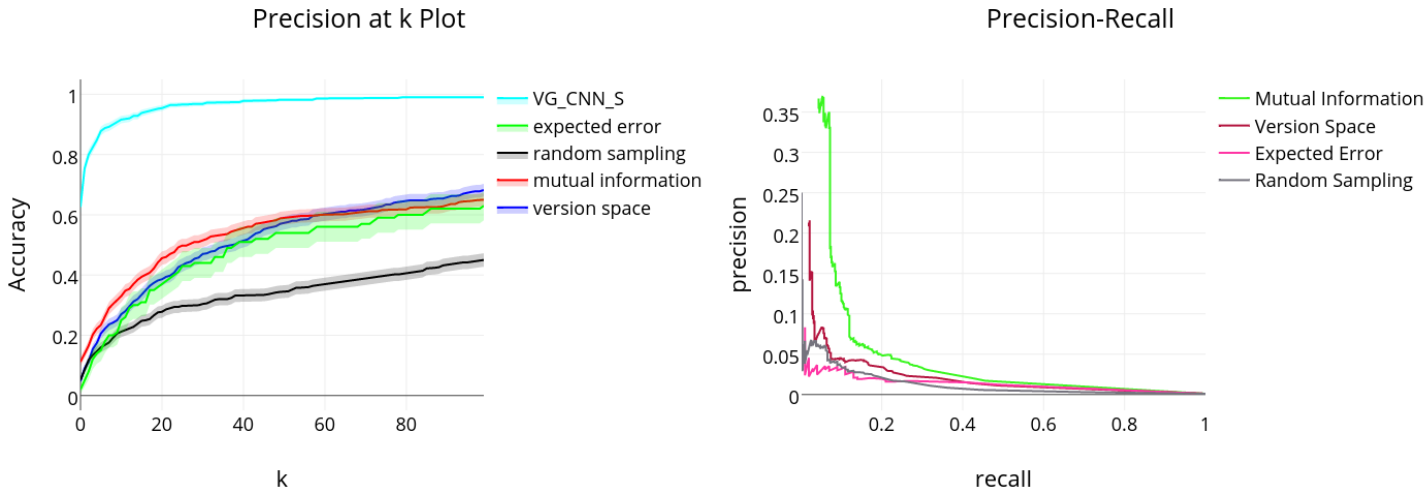


Figure 6.6.: Precision-Recall Plot for the Naïve Bayes setting.

Stopping criteria: Entropy reduction

In this section, we do not show any entropy plots since entropy reduction itself is the stopping criteria and plotting it will simply be redundant. Also, different number of tests are applied for every image and *precision-at-k* curve has been adapted accordingly.

Number of images: 500 (100 for expected error), *Entropy reduction:* 75% (50% for expected error)



(a) *Precision-at-k* ***(b)*** *Precision-Recall*

Figure 6.7.: Plots for a *entropy reduction* as a stopping criteria for the Naïve Bayes setting.

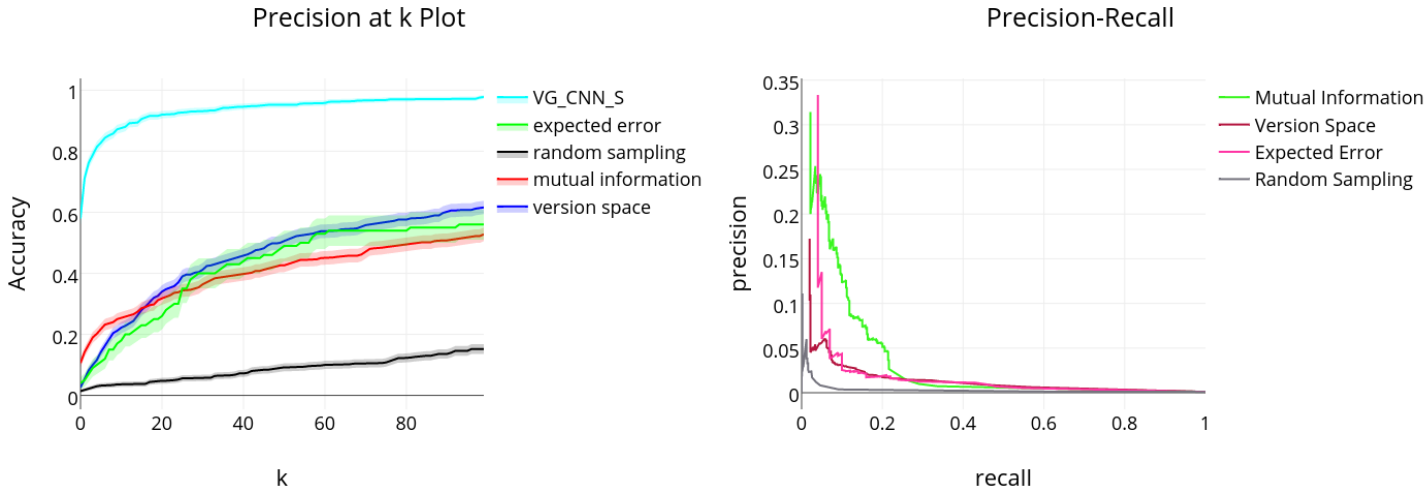
In Fig. 6.7, we see the precision-at-k and precision-recall plot for a different stopping criteria. It is interesting to note that an entropy of 75% leads to a worse performance compared to the cardinality constraint, on average. If we set a lower threshold for reduction in entropy, then, it cannot be guaranteed that the algorithm will necessarily decrease the entropy to reach the threshold value before all the tests run out[8]. However, setting a larger threshold seems to lower the performance of our framework.

Stopping criteria: Entropy change

Number of images: 500 (100 for expected error), *Entropy threshold:* 1%, *Number of consecutive iterations:* 10

Fig. 6.8 shows the accuracy plots for a different stopping criterion. Here, we stop if the change is entropy is very low. In other words, if the entropy reduction curve is almost flat then it is assumed that we have reached the correct probability distribution over the hypothesis space.
We observe that version space reduction technique performs better than the mutual information criterion, in this case. This contradicts our previous observations for the other two stopping criteria, wherein, we observed mutual information to perform much better than the version space reduction. It shows that version space reduction does not necessarily performs worse than entropy-based methods in all situations.

[8] we usually set a maximum number of tests that can be picked



(a) Precision-at-k *(b) Precision-Recall*

Figure 6.8.: Plots for a *change in entropy* as a stopping criteria for the Naïve Bayes setting.

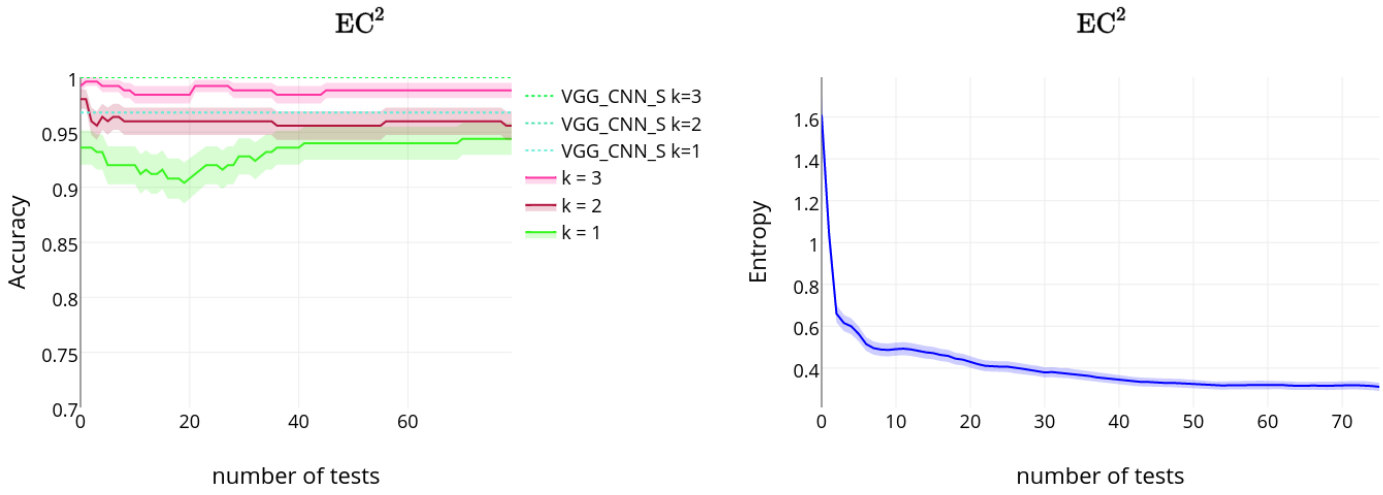
6.3.2. ImageNet Group Identification

Experiment Setup The setup is almost the same as that for the image classification in Section 6.3.1 with minor modifications. Since, we are now dealing with an equivalence class setting, the measure of accuracy, entropy and precision-recall is not for a single hypothesis but the group of hypotheses. We choose random images from the ILSVRC 2012 validation set for our experiments.

Stopping criteria: Cardinality constraint

Number of images: 250, *Number of tests:* 75, *level:* 2

From Fig. 6.9, we see that the performance of the EC^2 algorithm is very high. For $k = 1$, approximately



(a) Precision-at-k plot *(b) Entropy Reduction*

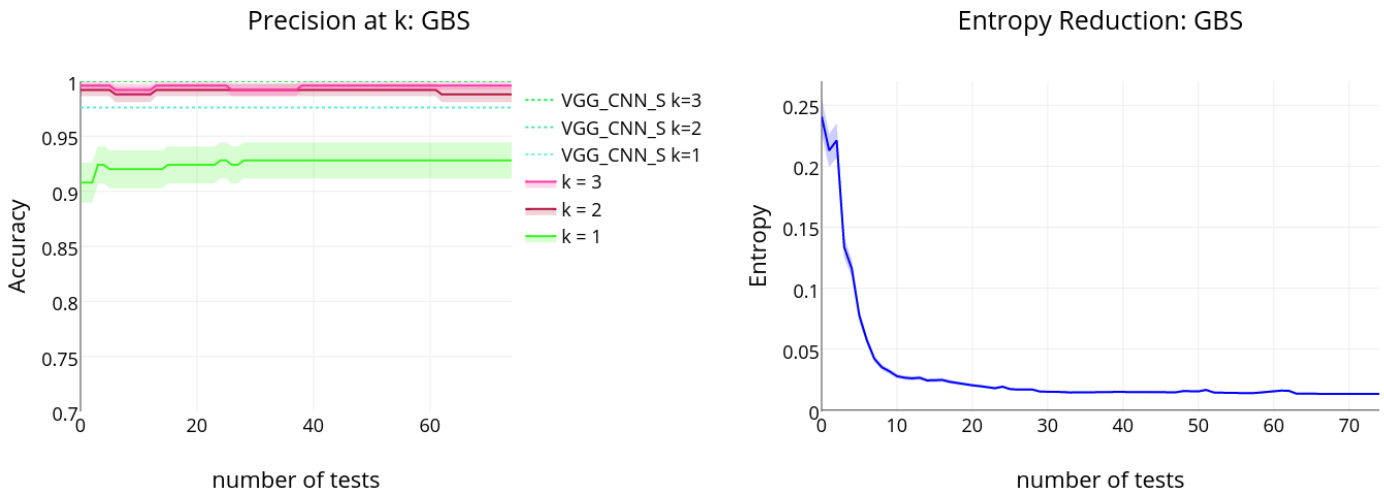
Figure 6.9.: Plots for selecting tests using the EC^2 algorithm.

95% of the images are classified correctly after 75 tests. This is highly desirable since the number of tests we picked is a fraction of the complete pool. Another interesting observation that we made during this experiment was that almost all the tests picked by EC^2 were for the parent nodes. It is not surprising

since these tests were designed to inform about the hypotheses groups themselves. However, it does show the robustness of the algorithm that we use.

GBS *Number of images:* 250, *Number of tests:* 75, *level:* 2

This is equivalent to the version space reduction in the Naïve Bayes setting. The performance of the



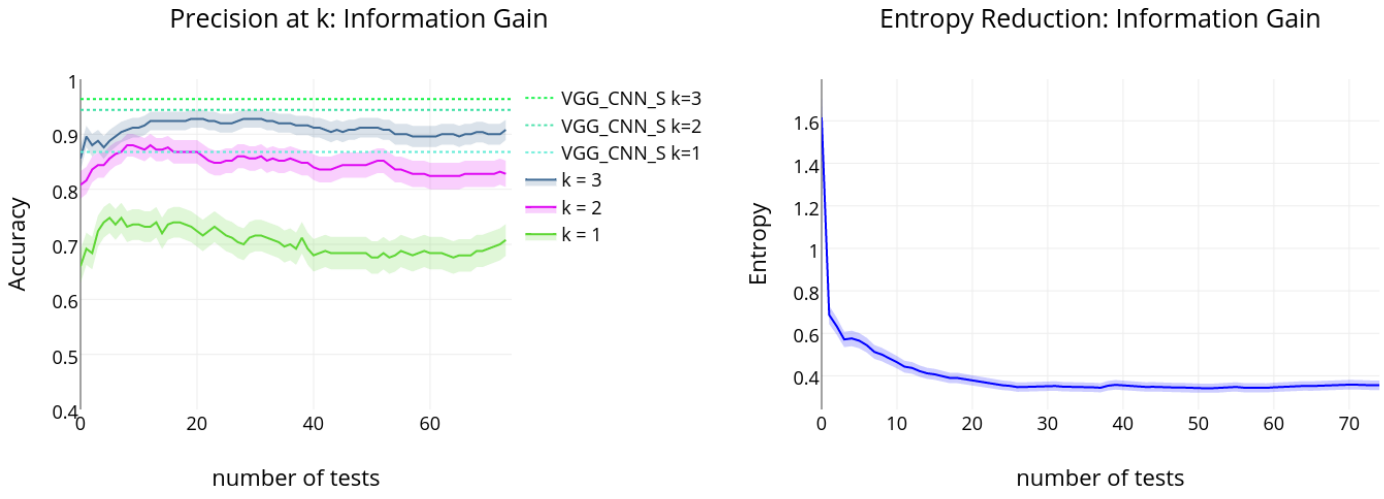
(a) Precision-at-k plot for accuracy *(b) Entropy Reduction with increasing number of selected tests*

Figure 6.10.: Plots for selecting tests using the generalized binary search.

generalized binary search is also high. However, the accuracy is not as high as that for the EC^2 algorithm.

Information Gain *Number of images:* 250, *Number of tests:* 75, *level:* 2

If we use the mutual information criteria in the equivalence class setting, it is known as *information gain*. The information gain criteria performs the worst among all algorithms that we used in the equivalence



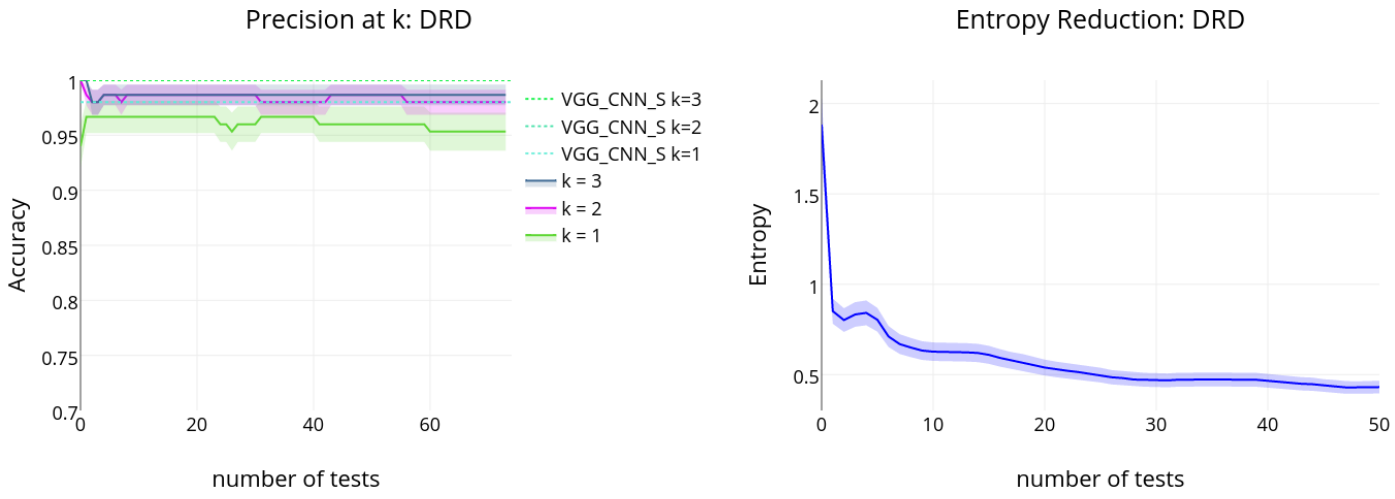
(a) Precision-at-k plot *(b) Entropy Reduction*

Figure 6.11.: Plots for selecting tests using the entropy based information gain criterion.

class setting.

Noisy-OR *Number of images:* 150, *Number of tests:* 75, *level:* 2

The Noisy-OR algorithm performs the best among all algorithm used in the equivalence class setting.

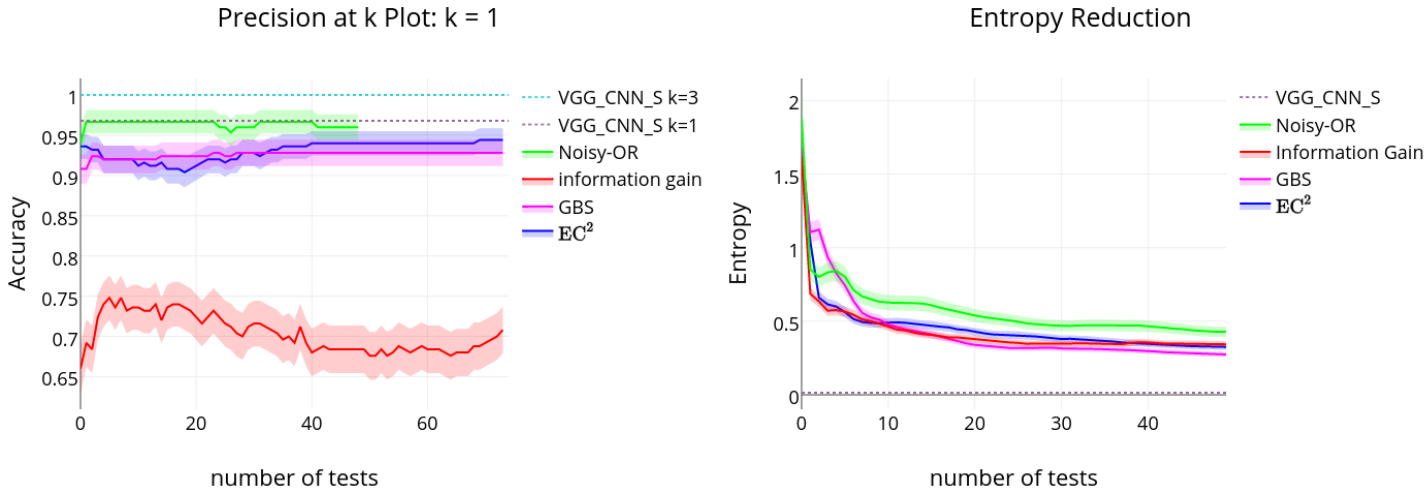


(a) Precision-at-k *(b) Entropy Reduction*

Figure 6.12.: Plots for selecting tests using the Noisy-OR construction for overlapping classes.

The accuracy is above 95% for $k = 1$ and this could be because we do not artificially construct non-overlapping equivalence classes in this scenario but allow the overlaps among the classes. This allows for more a robust algorithm leading to a slightly higher accuracy.

Comparison Fig. 6.13 shows the comparison of different algorithm that we used for identifying groups. We observe that GBS and EC^2 perform similarly. However, the performance of EC^2 is slightly better.



(a) Precision-at-k *(b) Entropy Reduction*

Figure 6.13.: Comparison plots for accuracy and entropy for the Equivalence Class setting.

Also, the accuracy from GBS seems to remain constant as we increase the number of tests. Since, we have a pool of tests that also includes tests for hypothesis groups, it could possibly lead to a higher performance for the GBS. GBS might not consider any information about the classes but the tests themselves do. If a certain test for a parent node is selected by GBS, this might increase the overall prediction power for a group.

Precision-Recall Precision-Recall for the equivalence class setting.

Fig. 6.14 shows the precision-recall curve for the 4 algorithms we used in the equivalence class setting.

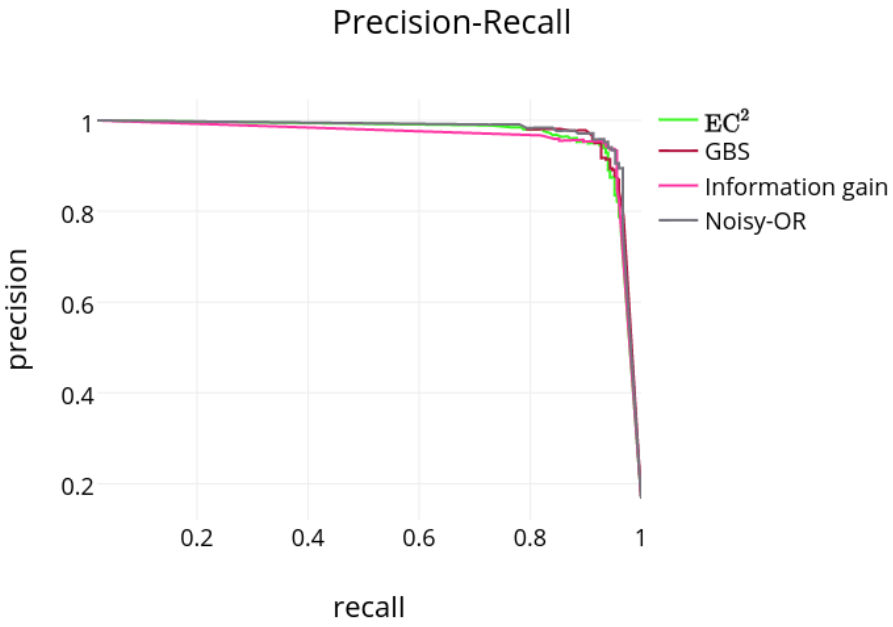


Figure 6.14.: Precision-Recall Plot for the Equivalence Class setting.

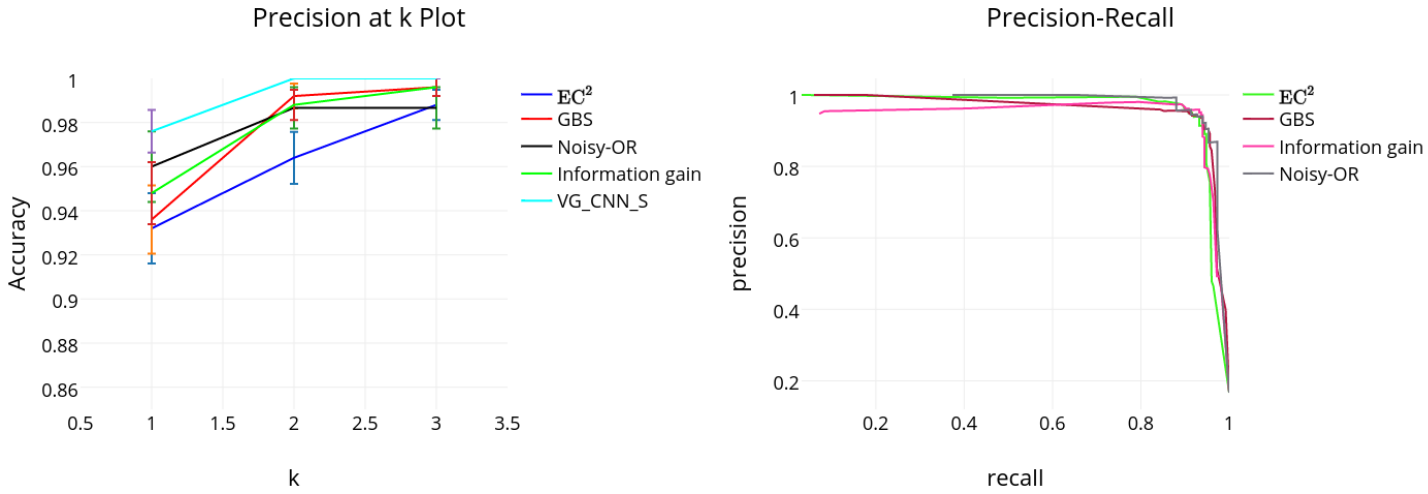
We can see that all the algorithms perform similarly with minor differences in performance. The Noisy-OR algorithm performs slightly better than the rest. This result is consistent from our findings from the accuracy plots. Note that the number tests selected via the Noisy-OR method is smaller than for all the other algorithms yet the performance is slightly better.

Stopping criteria: Entropy reduction

The same arguments from Section 6.3.1 is used to extend the idea of eliminating entropy reduction plots for this section.

Number of images: 250 (150 for Noisy-OR), *Entropy reduction:* 75%

Changing the stopping criteria significantly alters the performance of different algorithms. This is evident from Fig. 6.15a. We can clearly notice that the GBS outperforms the EC^2 in this case. Also, information gain as a criterion works well with this stopping criterion. However, for $k = 1$ i.e. the top equivalence class identified by each individual algorithm, we see that the Noisy-OR works best. This is consistent with our results that we obtained from the *cardinality constraint*, as a stopping criteria. Also, the difference in performances of all four algorithms are small.



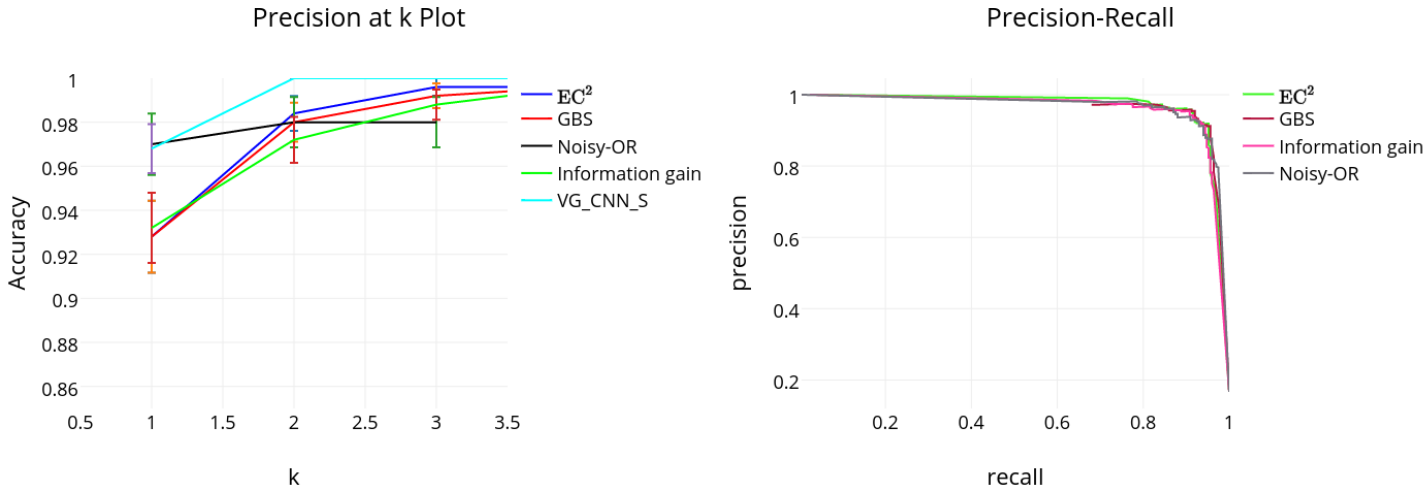
(a) Precision-at-k *(b) Precision-Recall*

Figure 6.15.: Plots for a *entropy reduction* as a stopping criteria for the Equivalence Class setting.

Stopping criteria: Entropy change

Number of images: 250 (150 for Noisy-OR), *Entropy threshold:* 1%, *Number of consecutive iterations:* 10

Unlike the Naïve bayes setting, *change in entropy* as a stopping criteria has a significant effect on performance of all the algorithms. In this case, EC^2 gives the best performance closely followed by the generalized binary search (GBS). However, for $k = 1$, the Noisy-OR still works best and outperforms the deep learning technique by a small margin. This result should be further investigated.



(a) Precision-at-k *(b) Precision-Recall*

Figure 6.16.: Plots for a *change in entropy* as a stopping criteria for the Equivalence Class setting.

7. Discussion & Future Work

Discussion The main task in our work was to select a set of binary classifiers that together perform as well as a multi-class classifier, convolutional neural network in our case.
In our work on Bayesian active learning on ImageNet, we tried to apply a greedy framework on the ImageNet dataset to select the right set of tests and compared it to state-of-the-art deep learning methods. We noticed that the greedy approach has a high performance in many cases but is almost always sigh of the hard baseline. However, the method performs significantly better than the naive way of randomly selecting tests. Hence, there is some value to these methods.

Pros

Firstly, the major advantage of this framework is the lower cost of determining a label of an image at test time while maintaining a good accuracy. Since a fraction of binary classifiers are required by our framework, the cost is lower as compared to a multi-class classifier outcome which essentially requires a cost equivalent to applying all the binary classifiers in our pool.
This leads us to the second advantage of our framework. We observe that the entropy of the posterior distribution decrease very rapidly indicating a fast convergence towards the true distribution. In other words, our framework selects the majority of significant classifiers in the first few iterations of our greedy framework.
Finally, as a side advantage, our greedy framework is very simple and improvements are much easier as it is a much better studied technique with a strong mathematical foundation.

Cons

There are, however, several disadvantages in the framework as well. The main idea of the framework is based on the notion of adaptive learning, more specifically, adaptive submodularity. Since, we deal with noisy cases, often, the nice properties associated with submodular functions are not valid any more and the performance suffers significantly. Hence, objective functions that retain such properties in a real-world noisy case have to be designed, which is usually a highly non-trivial task.
Another issue is the problem of halting the algorithm. Different images need different number of tests to be applied on them to find their true class. We usually fix a budget on the number of tests or the reduction in entropy to specify the stopping criterion. This, usually, results in either applying too many tests to too few tests leading to a lower *peak accuracy* ($k = 1$ in precision-at-k curve).

Computational Considerations

In this work, we considered many different criteria that could be used to select tests for identifying the true class of a given test image. Considering computational complexity is an important aspect when choosing a certain criteria. In terms of runtime, we observed that version space reduction method was fastest and expected error took the maximum time. This, in fact, is not surprising since the expected error method requires estimating the expected future error over the query space $\mathcal{M}$ for each query. This leads to a drastic increase in computational cost. Expected error is quadratic over the query space $\mathcal{O}(|\mathcal{M}|^2)$). Mutual information and version space reduction, on the other hand, are linear over the query space $\mathcal{O}(|\mathcal{M}|)$). Version space reduction runs faster than mutual information because the number of elementary operations required to calculate the score functions is lower.
In the equivalence class setting, the generalized binary search runs fastest in runtime units and DRD

runs the slowest. This again is not unexpected. The generalized binary search is equivalent to version space reduction in the Naive Bayes setting and is linear in query space with low number of elementary operations required to compute the corresponding score function. However, other than a linear dependency on the number of queries ($|\mathcal{M}|$) and the number of hypotheses ($|\mathcal{H}|$), the per-iteration computational complexity of DRD also depends on the number of ECD instances (which are upper bounded by the number of equivalence classes).

Stopping Criteria

During our experiments, the stopping criteria turned out be of high significance, particularly in the case of the equivalence class setting. We observed that the chosen stopping criterion altered the performance of each algorithm noticeably. However, if we only look at the top equivalence class chosen after the algorithms stop ($k = 1$ in precision-at-k plot), the Noisy-OR consistently gives a better performance. This is because the Noisy-OR is more robust, in the sense that the idea behind *Decision Region Determination Problem* has the least number of assumptions by design. It, in fact, outperformed the deep learning model by a small margin in one case. This gives us a hint that stopping criterion is an essential part to the framework and needs to be further investigated to improve the performance of the framework further.

Future work An important addition could be using the full ImageNet dataset instead of a small subset. Deep learning on full ImageNet dataset is too expensive and the methods proposed above can be very useful to bypass the expensive test time of deep neural networks.
Secondly, via a pre-processing step, we can try to get a better prior distribution over the hypothesis space. A uniform prior usually results in a very large number of tests being picked before certainty can be reached about the true hypothesis. A non-uniform and more informed prior will help in this regard by making sure that (on an average) fewer number of tests will be needed to find the true hypothesis.
Allowing test repetition i.e. allowing tests to be picked up multiple times could potentially improve the results further.
Another important extension could be to assign a non-uniform cost to the tests. As of now all the tests are assumed to have equal cost. A benefit to cost ratio might help to find the true hypothesis with fewer number of tests than the experiments required.
Finally, finding a good criterion to stop the framework from picking more tests should be actively investigated to increase the performance further.

8. Conclusion

In this work, we extensively studied the application of Bayesian active learning on ImageNet. The *Equivalence Class Determination Problem* and *Decision Region Determination Problem* turned to have a natural mapping onto the ImageNet dataset due to its hierarchical structure. There were some interesting observations and conclusions from this work. Firstly, mutual information based criteria can be effective in choosing a correct test to determine the true label of a test image. It also turned out that mutual information seemed to select the most informative tests very quickly, which was evident from the nature of its entropy plot. The version space reduction was slower in this regard.
Secondly, in our experiment setting the EC^2 and GBS (on average) had similar performances. This can be attributed to the fact that the test pool had tests not just for one label but also for a group of labels, making GBS effective even though it carries no information about an equivalence classes. This could be because of the fact that we trained binary classifiers (i.e. tests) for each of the class/group of classes in the hierarchy. Hence, by design we can always select tests that are informative about a group, making GBS effective. Also, the "Noisy-OR" consistently performed better than all the other algorithms used in the equivalence class setting.
Finally, the stopping criteria was of higher significance than expected. The performance of each algorithm changed (more so in the equivalence class setting) according to the chosen stopping criteria. This leads us to believe that a better stopping criterion could result in a higher performance for each algorithm.

A. Definitions

Definition A.1 (Submodularity [KG12]). *A function $f : 2^V \to \mathbb{R}$, $S \subseteq V$, and $e \in V$, let $\Delta_f(e|S) := f(S \cup \{e\}) - f(S)$ be the discrete derivative of S w.r.t e, then f is submodular if for every $A \subseteq B \subseteq V$ and $e \in V \setminus B$ it hold that*

$$\Delta_f(e|A) \geq \Delta_f(e|B) \tag{A.1}$$

Corollary A.1.1. *A function $f : 2^V \to \mathbb{R}$ is said to be submodular if for every $A, B \subseteq V$,*

$$f(A \cap B) + f(A \cup B) \leq f(A) + f(B) \tag{A.2}$$

Algorithm A.1 (Submodular Maximization). *Given a submodular function f, the optimization problem $\max_{S \subseteq V} f(S)$ can be solved using the following greedy algorithm*

$$S_i \;=\; S_{i-1} \cup \{\operatorname*{argmax}_e \; \Delta(e|S_{i-1})\} \tag{A.3}$$

Cardinality constraint on S has been assumed here but the constraint could also be in any other form.

Theorm A.1 (Bound of Greedy optimal solution [NWF78]). *For a non-negative monotone submodular function $f : 2^V \to \mathbb{R}_+$ and let $\{S_i\}_{i \geq 0}$ be the greedily selected set defined in Eq. A.3. Then for all positive integers k and l,*

$$f(S_l) \geq \left(1 \;-\; e^{(-l/k)}\right) \max_{S:|S| \leq k} f(S) \tag{A.4}$$

Definition A.2 (Strong Adaptive Monotonicity [GK10]). *A function $f : 2^E \times O^E \to \mathbb{R}$ is strongly adaptive monotone w.r.t. $p(\phi)$ if, informally "selecting more items never hurts" with respect to the expected reward. Formally, for all ψ, all $e \in \mathrm{dom}(\psi)$, and all the possible outcomes of $o \in O$ such that $\mathbb{P}\left[\Phi(e) = 0, \Phi \sim \psi\right] > 0$, we require*

$$\mathbb{E}\left[f(\mathrm{dom}(\psi), \Phi) | \Phi \sim \psi\right] \leq \mathbb{E}\left[f(\mathrm{dom}(\psi) \cup \{e\}, \Phi) | \Phi \sim \psi, \Phi(e) = o\right] \tag{A.5}$$

Strong adaptive monotonicity implies adaptive monotonicity, as the latter means that "selecting more items never hurts in expectation," i.e.

$$\mathbb{E}\left[f(\mathrm{dom}(\psi), \Phi) | \Phi \sim \psi\right] \leq \mathbb{E}\left[f(\mathrm{dom}(\psi) \cup \{e\}, \Phi) | \Phi \sim \psi\right] \tag{A.6}$$

Definition A.3 (Adaptive Submodularity [GK10]). *A function $f : 2^E \times O^E \to \mathbb{R}_{\geq 0}$ is adaptive submodular* w.r.t. *$p(\phi)$ if for all ψ and ψ' such that $\psi \subseteq \psi'$, and for all $e \in E \setminus \mathrm{dom}(\psi')$, we have*

$$\Delta_f(e|\psi) \geq \Delta_f(e|\psi') \tag{A.7}$$

Definition A.4 (Mutual Information). *Let X, Y be two random variables jointly distributed according to $p(x, y)$ then the mutual information between X and Y is defined as*

$$I(X;Y) = \sum_{x \in X, y \in Y} p(x,y) \log\left(\frac{p(x,y)}{p(x)p(y)}\right) \tag{A.8}$$

Corollary A.4.1. *If $H(X)$ is defined as the entropy of the random variable X and $H(X|Y)$ as the conditional entropy of random variable X given Y, then*

$$I(X;Y) = H(X) \;-\; H(X|Y) \tag{A.9}$$

Definition A.5. *Consider a channel W with associated conditional probability distribution $\{p(x \mid d)\}_{d \in \mathcal{D}, x \in \mathbf{x}}$. Note that given each $d \in \mathcal{D}, p(.|\, d)$ is a probability distribution over $\mathbf{x}$- The separability of channel W, denoted by $S(W)$, is then defined by*

$$S(W) = \left(\min_{d, d' \in \mathcal{D}: d \neq d'} |p(.|\, d) \;-\; p(.|d')|_{TV}\right) \tag{A.10}$$

where $|.|_{TV}$ is the total variation distance.

B. Derivations

Derivation B.1 (Score function for Version Space Reduction)**.** *From 4.8, we get the following,*

$$\begin{aligned}\delta(X_e|\psi_i) &= \sum_{v\in\mathcal{V}} P(v|\mathrm{x}_{\mathcal{A}}) - \sum_{v\in\mathcal{V}} P(v,\mathrm{x}_e|\mathbf{x}_{\mathcal{A}}) \\ &= 1 \;-\; P(X_e|\mathbf{x}_{\mathcal{A}})\end{aligned} \tag{B.1}$$

Given $\Delta(X_{e_{i+1}}|\psi_i)$, the score for test T_e at iteration $i+1$ is given as,

$$\begin{aligned}\Delta(X_{e_{i+1}}|\psi_i) &= \mathbb{E}\left[\delta(X_e|\psi_i)\right] \\ &= \sum_{X_e=\{0,1\}} P(X_e|\mathbf{x}_{\mathcal{A}})\delta(Xt_e|\psi_i) \\ &= \sum_{X_e=\{0,1\}} P(X_e|\mathbf{x}_{\mathcal{A}})\left(1 \;-\; P(X_e|\mathbf{x}_{\mathcal{A}})\right) \\ &= 2P(X_e=1|\mathbf{x}_{\mathcal{A}})P(X_e=0|\mathbf{x}_{\mathcal{A}})\end{aligned} \tag{B.2}$$

Derivation B.2 (Score function for Expected Error Minimization)**.** *We proceed from Eq. 4.13, We begin from Eq. 4.13 for this derivation,*

$$\Delta_{EE}(X_{e_{i+1}}|\psi_i) = \sum_{e'\in\mathcal{M}} P\left(\mathrm{x}_{e'} \neq \underset{\mathrm{x}_{e'}}{\operatorname{argmax}} P(X_{e'}|\{x_e \cup \mathbf{x}_A\})\right) \tag{B.3}$$

The score for test T_e at $(i+1)^{th}$ iteration is,

$$\begin{aligned}\Delta_{EE}(\mathrm{x}_{e_{i+1}}|\psi_i) &= \mathop{\mathbb{E}}_{X_e\in\{0,1\}}\left[\sum_{e'\in\mathcal{M}} P\left(\mathrm{x}_{e'} \neq \underset{\mathrm{x}_{e'}}{\operatorname{argmax}} P(X_{e'}|\{\mathrm{x}_e \cup \mathbf{x}_{\mathcal{A}}\})\right)\right] \\ &= \sum_{\mathrm{x}_e=\{0,1\}} P(X_e|\mathbf{x}_{\mathcal{A}})\left[\sum_{e'}\sum_{X_{e'}\in\{0,1\}} P(X_{e'})\mathbb{1}(\mathrm{x}_{e'} \neq \underset{\mathrm{x}_{e'}}{\operatorname{argmax}} P(X_{e'}|\{\mathrm{x}_e \cup \mathbf{x}_{\mathcal{A}}\}))\right] \\ &= \sum_{\mathrm{x}_e=\{0,1\}} P(X_e|\mathbf{x}_{\mathcal{A}})\left[\sum_{e'}\min\left(\; P(X_{e'}=1|\{\mathbf{x}_{\mathcal{A}}\cup\mathrm{x}_e\}) \;,\; P(X_{e'}=0|\{\mathbf{x}_{\mathcal{A}}\cup\mathrm{x}_e\}) \;\right)\right]\end{aligned} \tag{B.4}$$

Let us consider and solve for a part of the equation for simplicity. We consider $P(X_{e'}=1|\{\mathbf{x}_{\mathcal{A}}\cup\mathrm{x}_e\})$ as the starting point. Note that given h, $\{X_i, X_j\}$ are independent (Naive Bayes setting).

$$\begin{aligned}P(X_{e'}=1|\{\mathbf{x}_{\mathcal{A}}\cup\mathrm{x}_e\}) &= \sum_{h\in\mathcal{H}} P\left(X_{e'}=1|h\right) P\left(h|\{\mathbf{x}_{\mathcal{A}}\cup\mathrm{x}_e\}\right) \\ &= \sum_{h\in\mathcal{H}} P\left(X_{e'}=1|h\right) \frac{P(\{\mathbf{x}_{\mathcal{A}}\cup\mathrm{x}_e\}|h)P(h)}{P(\{\mathbf{x}_{\mathcal{A}}\cup\mathrm{x}_e\})} \\ &= \sum_{h\in\mathcal{H}} P\left(X_{e'}=1|h\right) \frac{P(\mathbf{x}_{\mathcal{A}}|h)P(\mathrm{x}_e|h)P(h)}{P(\{\mathbf{x}_{\mathcal{A}}\cup\mathrm{x}_e\})} \\ &= \sum_{h\in\mathcal{H}} P\left(X_{e'}=1|h\right) \frac{P(\mathrm{x}_e|h)P(h|\mathbf{x}_{\mathcal{A}})}{\sum\limits_h P(\mathrm{x}_e|h)P(h|\mathbf{x}_{\mathcal{A}})}\end{aligned} \tag{B.5}$$

Similarly, we can derive for the part of the equation with $X_{e'} = 0$. Since, we are working in expectation, the final result comes out to be the following,

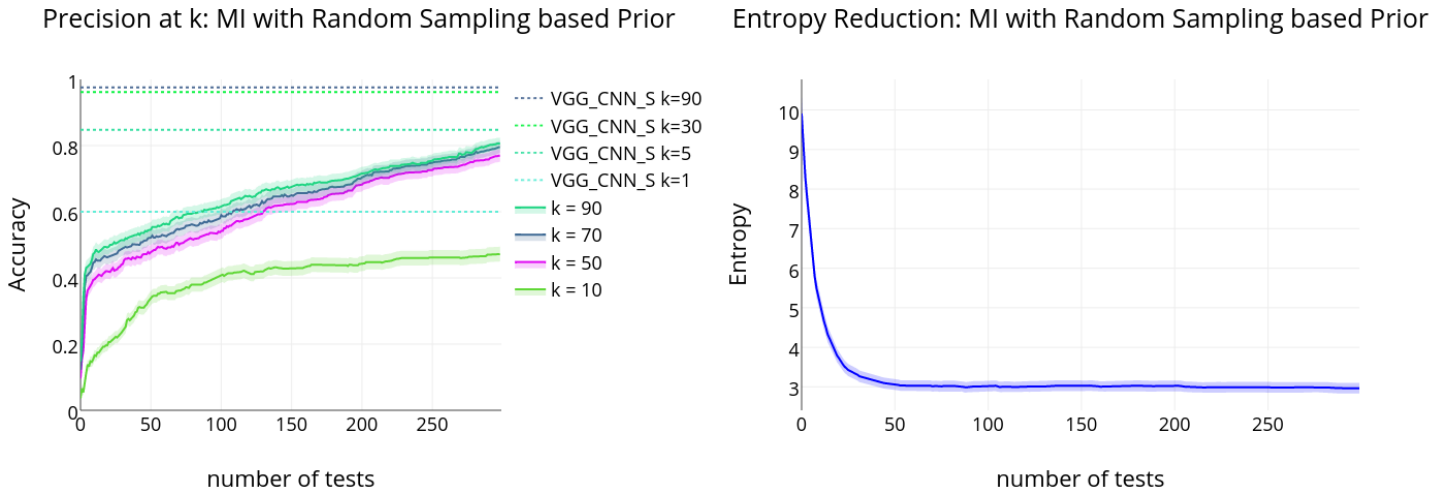
$$\Delta_{EE}(X_{e_{i+1}}|\psi_i) = \sum_{\mathrm{x}_e=\{0,1\}} P(\mathrm{x}_e|\mathbf{x}_{\mathcal{A}}) \left[\sum_{e'} \min \left(\sum_h P\left(X_{e'}=1|h\right) \frac{P(\mathrm{x}_e|h)P(h|\mathbf{x}_{\mathcal{A}})}{\sum_h P(\mathrm{x}_e|h)P(h|\mathbf{x}_{\mathcal{A}})}, \right. \right. \\ \left. \left. \sum_h P\left(X_{e'}=0|h\right) \frac{P(\mathrm{x}_e|h)P(h|\mathbf{x}_{\mathcal{A}})}{\sum_h P(\mathrm{x}_e|h)P(h|\mathbf{x}_{\mathcal{A}})} \right) \right] \tag{B.6}$$

C. Supplementary Plots

Here, we will provide a few supplementary plots that could be potentially useful for better understanding the framework and for further investigating our work.

Non-uniform Prior *Number of test images*: 500, *Number of tests:* 300
In this experiment for the Naïve Bayes setting, we try to obtain a more informed prior over the hypothesis space before running a criteria based test selection. The prior was obtain by simply applying random tests over a uniform distribution. We, then, used this "modified prior" as the input prior to the mutual information based greedy update.

It is noticeable that the performance of the mutual information criteria degrades in this case. This could be



(a) Precision-at-k plot *(b) Entropy Reduction*

Figure C.1.: Plots for mutual information as the selection criteria with Random Sampling-based prior.

because the new prior, that we begin with, has peaks at unwanted hypotheses (because of random application of tests over the uniform distribution) and the mutual information criteria cannot sufficiently converge to the true distribution within the budget of 300 tests.

Classification Example with EC^2 Now, we will show a classification example using the EC^2 algorithm to have a better intuition of the framework.
We select *level 5* of the tree to construct the equivalence classes. Then, we arbitrarily remove the overlaps between the classes and obtain the following equivalence classes:

'foodstuff, food product' ; 'fare' ; 'nutriment, nourishment, nutrition, sustenance, aliment, alimentation, victuals' ; 'feed, provender' ; 'beverage, drink, drinkable, potable' ; 'baked goods' ; 'produce, green goods, green groceries, garden truck' ; 'medicine, medication, medicament, medicinal drug' ; 'athlete, jock' ; 'diver, frogman, underwater diver' ; 'associate' ; 'valley, vale' ; 'promontory, headland, head, foreland' ; 'mountain, mount' ; 'ridge' ; 'lakeside, lakeshore' ; 'seashore, coast, seacoast, sea-coast' ; 'geyser' ; 'geographic point, geographical point' ; 'mechanism' ; 'plant part, plant structure' ; 'organism, being' ; 'instrumentality, instrumentation' ; 'structure, construction' ; 'creation' ; 'sheet, flat solid' ; 'fabric, cloth, material, textile' ; 'covering' ; 'padding, cushioning' ; 'commodity, trade good, good' ; 'excavation' ; 'decoration, ornament, ornamentation' ; 'plaything, toy' ; 'article' ; 'surface' ; 'piece of cloth, piece of material' ; 'sphere' ; 'light' ; 'material, stuff' ; 'fluid' ; 'street sign' ; 'cliff, drop, drop-off'

We select the image in Fig. C.2 belonging to the category *'African Grey, African Gray, Psittacus erithacus'* as our test image. For the above set of equivalence classes, this image belongs in the group *'organism, being'* (ground truth label).

Figure C.2.: Test Image

We start with a uniform distribution and run our greedy framework with the EC^2 algorithm to select 100 tests. The steps associated with the distribution update are shown in Fig. C.4.
The distribution over the image groups can be seen in Fig. C.3. At the end of 100 tests, MAP estimation tells us that equivalence class *'organism, being'* has the maximum probability of 0.411739414461. Hence, the algorithm identifies the correct image group after the updates have been performed.

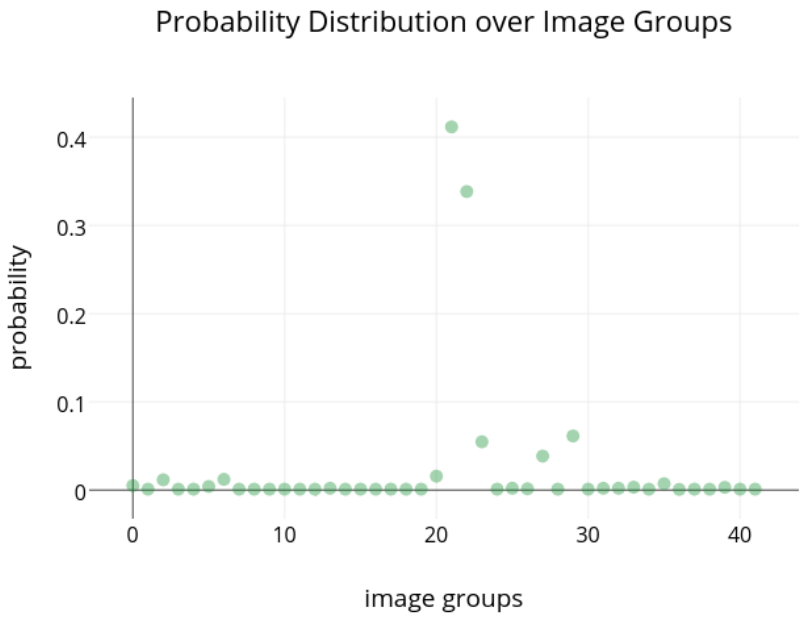


Figure C.3.: Distribution over Image Groups

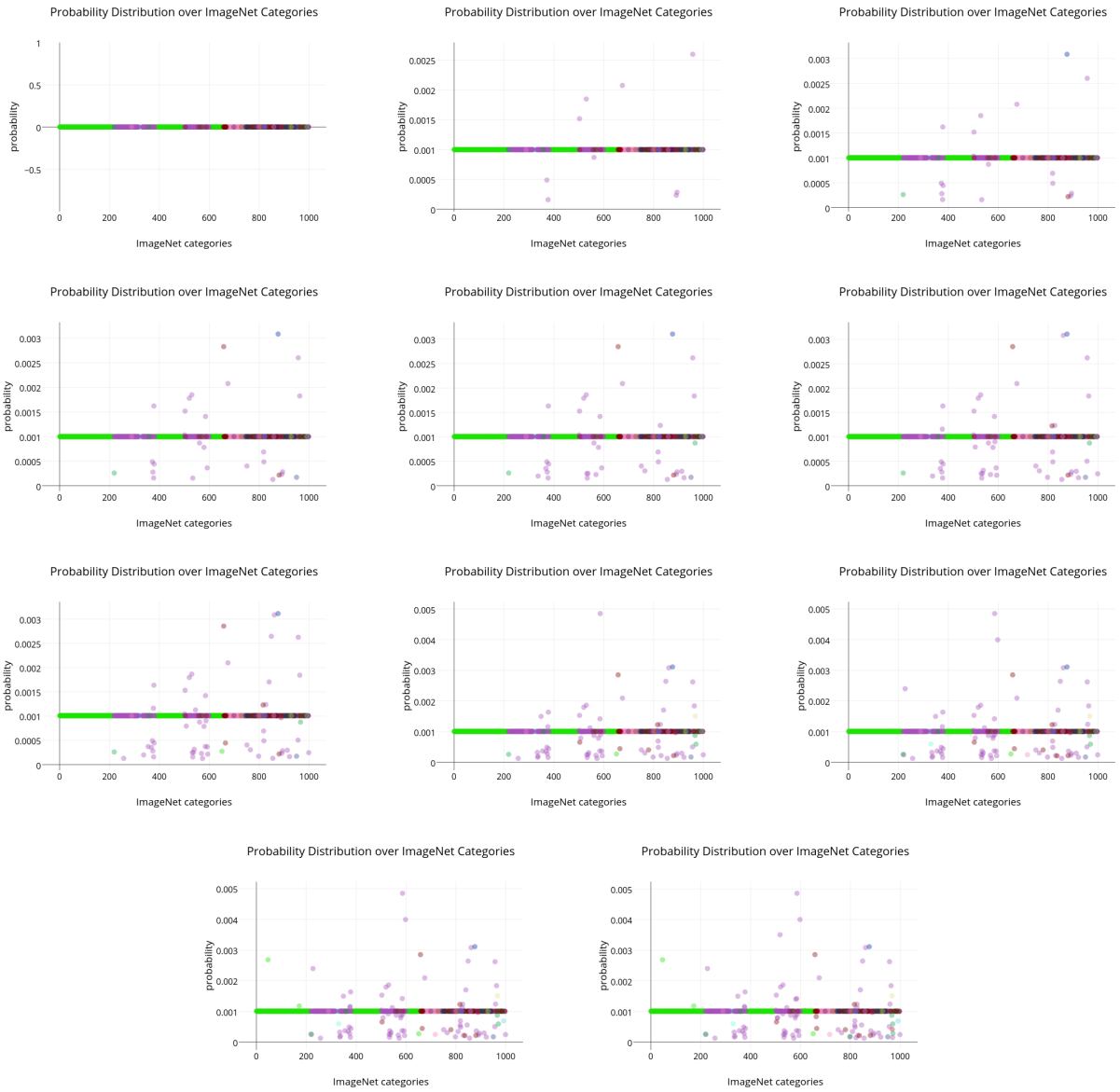


Figure C.4.: Posterior updates on the prior. The plots show the update steps starting from a uniform prior at an interval of every 10 test. Each color in the distribution represents one equivalence class.

Bibliography

[AGT07] Gidudu Anthony, Hulley Greg, and Marwala Tshilidzi. Classification of images using support vector machines, 2007.

[AK15] Ben Athiwaratkun and Keegan Kang. Feature representation in convolutional neural networks, 2015.

[ARK10] Itamar Arel, Derek C. Rose, and Thomas P. Karnowski. Research frontier: Deep machine learning–a new frontier in artificial intelligence research. *Comp. Intell. Mag.*, 5(4):13–18, November 2010.

[BB99] Bond and R.J. Bond. *Introduction to Abstract Mathematics*. Mathematics Series. Brooks/Cole, 1999.

[BBS10] Gowtham Bellala, Suresh K. Bhavnani, and Clayton Scott. Extensions of generalized binary search to group identification and exponential costs. In *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada.*, pages 154–162, 2010.

[Ben09] Yoshua Bengio. Learning deep architectures for ai. *Found. Trends Mach. Learn.*, 2(1):1–127, January 2009.

[BJ01] Y. Y. Boykov and M. P. Jolly. Interactive graph cuts for optimal boundary amp; region segmentation of objects in n-d images. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 105–112 vol.1, 2001.

[BZM07] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, Oct 2007.

[CHKK15] Yuxin Chen, S. Hamed Hassani, Amin Karbasi, and Andreas Krause. Sequential information maximization: When is greedy near-optimal? In *Proc. International Conference on Learning Theory (COLT)*, July 2015.

[CJK+15] Yuxin Chen, Shervin Javdani, Amin Karbasi, James Andrew Bagnell, Siddhartha Srinivasa, and Andreas Krause. Submodular surrogates for value of information. In *Proc. Conference on Artificial Intelligence (AAAI)*, January 2015.

[CSVZ14] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *British Machine Vision Conference*, 2014.

[CZX+12] Yu Cheng, Kunpeng Zhang, Yusheng Xie, Ankit Agrawal, and Alok Choudhary. *On active learning in hierarchical classification*, pages 2468–2471. 2012.

[DDS+09] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[GK10] Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization, 2010.

[GKR10] Daniel Golovin, Andreas Krause, and Debajyoti Ray. Near-optimal bayesian active learning with noisy observations, 2010.

[HLYG13] Gang Hua, Chengjiang Long, Ming Yang, and Yan Gao. Collaborative active learning of a kernel machine ensemble for recognition. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[JB11] S. Jegelka and J. Bilmes. Submodularity beyond submodular energies: Coupling edges in graph cuts. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1897–1904, June 2011.

[JCK+14] Shervin Javdani, Yuxin Chen, Amin Karbasi, Andreas Krause, J. Andrew Bagnell, and Siddhartha Srinivasa. Near optimal bayesian active learning for decision making, 2014.

[JPP12] A. J. Joshi, F. Porikli, and N. P. Papanikolopoulos. Scalable active learning for multiclass image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2259–2273, Nov 2012.

[JSD+14] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22Nd ACM International Conference on Multimedia*, MM '14, pages 675–678, New York, NY, USA, 2014. ACM.

[KG12] Andreas Krause and Daniel Golovin. Submodular function maximization, 2012.

[KGGK06] Andreas Krause, Carlos Guestrin, Anupam Gupta, and Jon Kleinberg. Near-optimal sensor placements: Maximizing information while minimizing communication cost. In *Proceedings of the 5th International Conference on Information Processing in Sensor Networks*, IPSN '06, pages 2–10, New York, NY, USA, 2006. ACM.

[KKT03] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pages 137–146, New York, NY, USA, 2003. ACM.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[LHK13] Chengjiang Long, Gang Hua, and Ashish Kapoor. Active visual recognition with expertise estimation in crowdsourcing. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[LHK16] Chengjiang Long, Gang Hua, and Ashish Kapoor. A joint gaussian process model for active visual recognition with expertise estimation in crowdsourcing. *International Journal of Computer Vision*, 116(2):136–160, 2016.

[LKL12] Xiao Li, Da Kuang, and Charles X. Ling. *Advances in Knowledge Discovery and Data Mining: 16th Pacific-Asia Conference, PAKDD 2012, Kuala Lumpur, Malaysia, May 29-June 1, 2012, Proceedings, Part I*, chapter Active Learning for Hierarchical Text Classification, pages 14–25. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[NWF78] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14(1):265–294, 1978.

[RASC14] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: An astounding baseline for recognition. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, CVPRW '14, pages 512–519, Washington, DC, USA, 2014. IEEE Computer Society.

[RDS+15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[RHT05] G. Riccardi and D. Hakkani-Tur. Active learning: theory and applications to automatic speech recognition. *IEEE Transactions on Speech and Audio Processing*, 13(4):504–511, July 2005.

[Rou13] Rupesh Kumar Rout. A survey on object detection and tracking algorithms, 2013.

[Set10] Burr Settles. Active learning literature survey. Technical report, 2010.

[Set11] Burr Settles. From theories to queries: Active learning in practice. *Active Learning and Experimental Design W*, pages 1–18, 2011.

[Set12] Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

[WP67] D. J. A. Welsh and M. B. Powell. An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, 10(1):85–86, January 1967.

[YCBL14] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In Z. Ghahramani, M. Welling, C. Cortes, N.d. Lawrence, and K.q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3320–3328. Curran Associates, Inc., 2014.