AN ABSTRACT OF THE DISSERTATION OF

Mandana Hamidi-Haines for the degree of Doctor of Philosophy in Computer Science presented on September 20, 2019.

Title: Active Learning from Examples, Queries and Explanations

Abstract approved: _____

Prasad Tadepalli

Humans are remarkably efficient in learning by interacting with other people and observing their behavior. Children learn by watching their parents' actions and mimic their behavior. When they are not sure about their parents demonstration, they communicate with them, ask questions, and learn from their feedback. On the other hand, parents and teachers ask children to explain their behavior. This explanation helps the parents know whether the children learned their task correctly or not. So, why not have intelligent systems that learn from examples and interaction with humans, and explain their decisions to humans? This dissertation makes three contributions toward this goal.

The first contribution is towards designing an intelligent system that incorporates human's knowledge in discovering of hierarchical structure in sequential decision problems. Given a set of expert demonstrations. We proposed a new approach that learns a hierarchical policy by actively selecting demonstrations and using queries to explicate their intentional structure at selected points.

The second contribution is a generalization of the framework of adaptive submodularity. Adaptive submodular optimization, where a sequence of items is selected adaptively to optimize a submodular function, has been found to have many applications from sensor placement to active learning. We extend this work to the setting of multiple queries at each time step, where the set of available queries is randomly constrained. A primary

contribution of this paper is to prove the first near optimal approximation bound for a greedy policy in this setting. A natural application of this framework is to crowd-sourced active learning problem where the set of available experts and examples might vary randomly. We instantiate the new framework for multi-label learning and evaluate it in multiple benchmark domains with promising results.

The third contribution of this dissertation is the introduction of a framework for explaining the decisions of deep neural networks using human-recognizable visual concepts. Our approach, called interactive naming, is based on enabling human annotators to interactively group the excitation patterns of the neurons in the critical layer of the network into groups called ”visual concepts". We performed two user studies of visual concepts produced by human annotators. We found that a large fraction of the activation maps have recognizable visual concepts, and that there is significant agreement between the different annotators about their denotations. Many of the visual concepts created by human annotators can be generalized reliably from a modest number of examples.

Active Learning from Examples, Queries and Explanations

by

Mandana Hamidi-Haines

A DISSERTATION

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Presented September 20, 2019
Commencement June 2020

Doctor of Philosophy dissertation of Mandana Hamidi-Haines presented on September 20, 2019.

APPROVED:

Major Professor, representing Computer Science

Head of the School of Electrical Engineering and Computer Science

---

Dean of the Graduate School

I understand that my dissertation will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my dissertation to any reader upon request.

Mandana Hamidi-Haines, Author

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Professor Prasad Tadepalli, who has been an amazing mentor during my time at Oregon State University. His encouragement and support kept me energetic while working to complete this thesis. I am also grateful for the opportunities that he gave me to travel, and attend the conferences.

Professor Alan Fern has been an excellent co-advisor and collaborator, he consistently provided helpful guidance and important correction. I would like to thank him for the innumerable discussions about different aspects of the thesis. I feel fortunate to have had the experience of working with him.

I would like to thank my minor advisor Professor Sarah Emerson from the Statistics Department at OSU. The knowledge I gained from Professor Emerson's courses directly contributed to my research.

I also wish to thank all past and present collaborators in my research. Robby Goetschalckx has been an excellent collaborator, he consistently provided valuable guidance and feedback on my research. I am thankful Dr. Fuxin Li for his vision and unique perspective. It has been a pleasure and a great learning experience working with him. I would like to thank Zhongang Qi for all his help and suggestions in the last chapter of my Ph.D. thesis.

I also thank my committee members, Professor Alan Fern and Professor Weng-Keen Wong for the innumerable discussions about different aspects of the thesis. I am grateful for Professor John Dill's willingness to be my Graduate Council Representative, for the time and fexibility he provided.

I would like to give my very special thank to Mr. Jed Irvine, who coached me to improve my communication skills. His guidance and advice was immensely helpful.

I would like to thank Dr. Sriraam Natarajan, who helped me a lot to come to Oregon State University and start working under the supervision of Professor Prasad Tadepalli.

I have had the pleasure of interacting with my excellent colleagues at Oregon State. Among them, I would especially like to thank Sogol Balalli, Sahar Alizadeh, Prisa Ataee, Abtin Khodadadi, Mahtab Aboufazeli, Megha Sinha, Si Liu, and Michael Slater. I appreciate the good conversations, encouragement, and friendship of Tadesse Zemicheal and Beatrice

Moissinac. Tadesse Zemicheal is a unique and spatial friend who has been always available for unconditional help. I learned a lot form him.

I am truly grateful to my parents, Simin and Khosro Hamidi. I owe everything to them. Without the support of them, this thesis would not have been possible. I thank them so much for their continuous and unconditional support. They always give me a courage and strong determination to overcome any difficult challenges in my life and keep pushing me forward to pursue my dream. I also want to thank the rest my family for their support. Katayoon, Azarakhsh and Azaran are the best sibling I could ask for.

I would like to express my immeasurable gratitude for the unquestioning love and support of my parents-in-law Harold Haines and Mary Ann Haines. They have always encouraged me to pursue my dreams.

Completing this work simply would not have been possible without the love and support of my husband, Ralph Haines. I consider myself extremely lucky to have found such a supportive, caring, handsome and fun partner to share the life with. I would not have made it this far without him. Through both success and failure he was constant in his love and encouragement, I could not ask for a better partner in life.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

Figure Page

Figure Page

LIST OF TABLES

LIST OF TABLES (Continued)

LIST OF ALGORITHMS

Chapter 1: Introduction

Humans are remarkably efficient in learning by interacting with other people and observing their behavior. Children learn by watching their parents' actions and mimic their behavior. When they are not sure about their parents' behavior, they communicate with them, ask questions, and learn from their feedback. On the other hand, parents and teachers ask children to explain their behavior. Their explanations help the parents know whether the children learned their task correctly or not.

Learning by imitation and interaction with other humans are the primary ways children can understand and reproduce human behavior. Asking questions and receiving feedback is one way that can reduce their confusion and uncertainty. Explaining the decisions is another way that children can build trust for their parents. So, why not have intelligent systems learn from examples and interaction with humans, and explain their decisions to humans?

As Artificial Intelligence (AI) is increasingly being applied to enhance and improve our lives, the challenge of supporting interaction with humans is becoming more apparent. One of the advantages of the interaction between humans and machines is when humans and intelligent systems cooperate in problem-solving. Machines are good at computation on data, whereas humans are better at abstracting knowledge from their experience, and transferring the knowledge across domains. Humans can directly be involved in training, tuning, and testing data. They may provide feedback while observing the machine's outputs or answer the questions and queries generated by the machines. This feedback can be in the form of labels, demonstrations, corrections, rankings, or evaluations. The internal states of machines and systems are often updated upon the interactions. This cooperation, can be beneficial in solving computationally hard problems and helps create more efficient machine learning algorithms that can discover useful information from complex datasets.

The other advantages of the interaction between humans and machines is when machines can provide explanations based on their decisions, which can provide trust and transparency as humans shift greater responsibility to such systems.

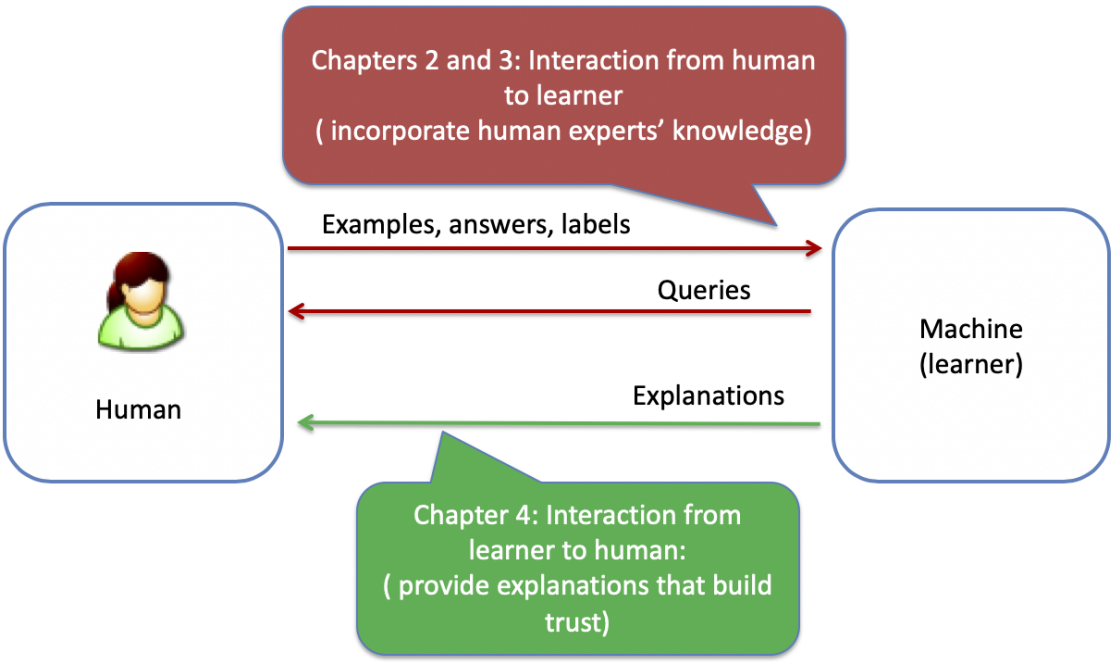


Figure 1.1: Outline of this thesis.

This thesis explores several steps towards designing an intelligent system that learns from interaction and examples in three machine learning problems: 1) sequential decision making, 2) adaptive submodular optimization, and 3) explaining the decisions of deep neural networks. In the first two problems, we focused on incorporating human experts' knowledge in machine learning algorithm, where the learner iteratively asks queries, the expert provides feedback to the machine, and the machine updates its internal states and models given the feedback. In the third problem, we focused on interaction between machines and humans that builds trust and transparency for the humans. In this setting the machine not only interacts with humans to incorporate human knowledge, but also provides explanations in terms of human-recognizable visual concepts to humans. The machine in turn generalizes these visual concepts to produce meaningful explanations of test examples. The outline of this thesis is represented in Figure 1.1.

We start by studying active imitation learning of hierarchical policies from demonstrations in sequential decision domains. The main difficulty in learning hierarchical policies

by imitation is that the high level intention structure of the policy, which is often critical for understanding the demonstration, is unobserved. The second learning framework we study is that of adaptive submodular optimization and its application to active learning. We extend the active learning framework to the problem of learning from multiple queries at each time step, where the set of available queries is randomly constrained. A natural application of this framework is to crowd-sourced active learning problem where the set of available human experts and examples might vary randomly. Finally, We describe an interactive machine learning framework, called interactive naming, where the system explains the decision of the deep neural networks in terms of human-recognizable visual concepts using an interactive user interface.

We end with a summary of our contributions, which will be elaborated in the remaining chapters of the thesis.

Active Imitation Learning of Hierarchical Policies: The first contribution in this thesis is the introduction of a framework for imitation learning of hierarchical policies from demonstrations. The main difficulty in learning hierarchical policies by imitation is that the high level intention structure of the policy, which is often critical for understanding the demonstration, is unobserved. We formulate this problem as active learning of Probabilistic State-Dependent Grammars (PSDGs) from demonstrations [Pynadath and Wellman, 2000]. Given a set of expert demonstrations, our approach learns a hierarchical policy by actively selecting demonstrations and using queries to explicate their intentional structure at selected points.

Our contributions include a new algorithm for imitation learning of hierarchical policies and principled heuristics for the selection of demonstrations and queries. We developed a novel two-level active learning framework, where the top level selects a trajectory and the lower level actively queries the teacher about the intention structure at selected points along the trajectory. We developed a new information-theoretically justified heuristic, called "cost-normalized information," for selecting trajectories, and employed Bayesian active learning for the lower-level query selection. Experimental results in five different domains exhibit successful learning using fewer queries than a variety of alternatives.

Adaptive Submodularity with Varying Query Sets: Our second contribution is a new framework for adaptive submodular optimization, where a sequence of items is selected

adaptively to optimize a submodular function. Adaptive submodular optimization has been found to have many applications from sensor placement to active learning. We extend this work to the setting of learning from multiple queries at each time step, where the set of available queries is randomly constrained. Our main theoretical result is the first near optimal approximation bound for a greedy policy in this setting. We generalize the framework of adaptive submodular optimization to the setting of varying queries, and provide a greedy algorithm with a near-optimal approximation guarantee.

An interesting application that motivates our research is active multi-label learning where an example has multiple labels, where each expert can only label a subset of them. A common scenario is crowd sourcing, where different workers are experts at labeling different classes, e.g., identifying different species of plants or animals in pictures. Moreover, only some experts are available at any time. This motivates the following problem. Given a set of unlabeled instances, and a set of experts who randomly become available at a given time, how best to choose the next example to be labeled and by which expert? We instantiate the new framework for multi-label learning and evaluate it in multiple benchmark domains with promising results.

Explaining the Decisions of Deep Learning via Interactive Naming: The third contribution of this thesis is a new framework for explaining the decision of deep neural networks in terms of human-recognizable visual concepts. In particular, given a set of test images, we aim to explain each classification in terms of a small number of image regions, or activation maps, which have been associated with semantic concepts by a human annotator. This allows for generating summary views of the typical reasons for classification, which can help build trust in a classifier or identify example types for which the classifier may not be trusted. For this purpose, We developed a user interface for "interactive naming," which allows a human annotator to manually cluster significant activation maps in a test set into meaningful groups called "visual concepts." We did two user studies and evaluated two different visualization methods on disentanglement of features, human recognizability of activation maps, inter-annotator agreement, and the generalizability of visual concepts. Our work is an exploratory study of the interplay between machine learning and human recognition mediated by the visual explanations of the results of learning.

The rest of the thesis is organized as follows: In the second chapter, we present a new

model and an algorithm for imitation learning of hierarchical policies, propose active learning heuristics for the selection of demonstrations and queries, and describe experimental results in multiple domains. In the third chapter, we describe the adaptive submodularity with varying queries setting and present a proof of a near optimal approximation bound for a greedy policy in this setting. In the fourth chapter, we present the interactive naming framework for explaining the decisions of neural networks. We conclude the thesis in the final chapter by outlining some possible directions for future work.

Chapter 2: Active Imitation Learning of Hierarchical Policies

In this chapter, we study the problem of imitation learning of hierarchical policies from demonstrations. The main difficulty in learning hierarchical policies by imitation is that the high level intention structure of the policy, which is often critical for understanding the demonstration, is unobserved. We formulate this problem as active learning of Probabilistic State-Dependent Grammars (PSDGs) from demonstrations. Given a set of expert demonstrations, our approach learns a hierarchical policy by actively selecting demonstrations and using queries to explicate their intentional structure at selected points. Our contributions include a new algorithm for imitation learning of hierarchical policies and principled heuristics for the selection of demonstrations and queries. Experimental results in five different domains exhibit successful learning using fewer queries than a variety of alternatives.

2.1 Introduction

Hierarchical structuring of policies and procedures is a common way to cope with the intractability of sequential decision making for both people and machines. Almost any complex task from cooking to arithmetic is taught by hierarchically decomposing it to simpler tasks. Indeed, hierarchical task structures are known to improve the efficiency of automated planning and [Nau et al., 2003] reinforcement learning [Dietterich, 2000] in many complex domains.

In hierarchical imitation learning, we seek to imitate an agent who has an effective hierarchical policy for a sequential decision making domain. One reason to do this is the ubiquity and naturalness of hierarchical policies relative to the non-hierarchical (flat) policies. Another virtue of hierarchical policies is their transferability to other related domains. A third, perhaps more practical reason, is their utility in learning from humans and tutoring other humans. The tutoring application suggests that the policy needs to be constrained by the human vocabulary of task names for machines to communicate naturally with people [Byrne and Russon, 1998, Whiten et al., 2006, Koechlin and Jubault, 2006].

Recent research in reinforcement learning and hierarchical planning has considered the problem of automatically discovering hierarchical structure by analyzing the dynamics and interactions of state variables during execution [Hengst, 2002, Jonsson and Barto, 2006], by finding bottlenecks in the state space [Mcgovern and Barto, 2001], and by doing a causal analysis based on known action models [Mehta et al., 2008, Hogg et al., 2009, Nejati et al., 2006]. However, by being autonomous, these approaches have the problem of discovering unnatural hierarchies, which may be difficult to interpret and communicate to people.

In this paper, we study the problem of learning policies with hierarchical structure from demonstrations of a teacher whose policy is structured hierarchically, with natural applications to problems such as arithmetic tutoring, cooking, and furniture assembly. A key challenge in this problem is that the demonstrations do not reveal the hierarchical task structure of the teacher. Rather, only ground states and teacher actions are directly observable. This can lead to significant ambiguity in the demonstration data from the perspective of learning the teacher policy. For example, it might only be clear in hindsight why the teacher opened a drawer in a given state, and might require substantial reasoning.

Indeed, theory suggests that imitation learning of hierarchical policies in the form of decision lists is NP-Hard [Khardon, 1999]. One approach for learning hierarchical policies is to apply algorithms such as Expectation-Maximization (EM) which work by iteratively guessing the intentional structure and learning the policy. Our extensive experiments in this direction showed that the hierarchies produced by these methods are often poor local optima and are typically inconsistent with the true intentional structure of the teacher. Further, even if the correct intention structure was learned, in order to communicate about that structure, a mapping to human-understandable terms would need to be learned. Thus, in this paper we follow a different learning approach where we directly work with a human to efficiently uncover the intention structure of demonstrations.

Our approach is motivated by the work of Khardon (1999) which showed that if demonstrations are annotated with the teacher's intention structure, then learning hierarchical policies becomes tractable. An analogous lesson from the tutoring literature is that learning can be made efficient if the examples illustrate one simple new concept at a time [VanLehn, 1987]. However, carefully ordering the examples to satisfy this constraint or fully annotating every trajectory with its intention structure are too prohibitive. We address both of these

problems by making the example selection and the task annotation *active* and *incremental*.

Our contributions are as follows. We first show that Probabilistic State-Dependent Grammars (PSDGs) [Pynadath and Wellman, 2000] strictly generalize common prior representations such as MAXQ hierarchies [Dietterich, 2000] to represent hierarchical policies. Second, we develop a new algorithm for efficiently learning PSDGs from sets of demonstrations and answers to intention queries, leveraging the relationship of PSDGs to probabilistic context-free grammars. Third, we introduce a novel two-level active learning approach for selecting demonstrations to annotate and then acquiring the intention annotations. For acquiring annotations we draw on ideas from Bayesian active learning. However, for trajectory selection, the natural heuristic of selecting maximum entropy trajectory can perform poorly. We analyze this observation and develop a principled heuristic for selecting trajectories that is significantly more effective. Our final contribution is to demonstrate the effectiveness of the learning algorithm and our active learning heuristics compared to competitors in five domains, including a domain relevant to arithmetic tutoring.

2.2 Problem Setup and Background

We consider a rewardless Markov Decision Process (MDP) defined by a a set of states S, a set of primitive actions A, and a transition function $\mathcal{P}(s'|s,a)$ which represents the distribution of next states s' given the state s and action a. A set of task names T, and their associated termination conditions are known to the learner. Following the MAXQ hierarchical reinforcement learning framework [Dietterich, 2000], we define a task hierarchy as a directed acyclic graph over $T \bigcup A$, where there is an edge between two tasks if the first task can call the second task as a subroutine. The primitive actions A are the leafs of the hierarchy and can be executed directly in 1 step and change the state. The child nodes of a task are called its subtasks. We assume that all hierarchies have a special root task labeled Root. Some tasks have a small number of parameters which allows more efficient generalization. Some example task hierarchies (those used in the experiments in this paper) are shown in Figures 2.5, 2.6, 2.7, 2.8, and 2.9.

A task is executed by repeatedly calling one of its subtasks based on the current state until it terminates. When a subtask terminates, the control returns to its parent task. We

define a deterministic hierarchical policy π as a partial function from $T \times S$ to $T \bigcup A$ which is defined only for states that do not satisfy their termination conditions for any task and maps to one of its subtasks. A trajectory or a demonstration d of a policy π starting in state s is a sequence of alternating states and primitive actions generated by executing that policy from s starting with the Root task. We also assume that every task t_i has a special symbol t_i' that denotes its termination, which is included in the trajectory at the point the task terminates.[1] More formally, a demonstration d of a policy π in a state s can be defined as follows, where ; is used for concatenation, $t_i\gamma$ represents a task stack with t_i as the topmost task and γ is the rest of the stack to be executed.

$d(s, \pi) = d(s, \pi, \mathsf{Root})$
$d(s, \pi, t_i\gamma) = t_i' d(s, \pi, \gamma)$ if $t_i \in T$ and t_i terminates in s
$d(s, \pi, t_i\gamma) = d(s, \pi, \pi(s, t_i)t_i\gamma)$ if $t_i \in T$ and t_i does not terminate in s
$d(s, \pi, t_i\gamma) = t_i; s'; d(s', \pi, \gamma)$ where $t_i \in A$ and $s' \sim \mathcal{P}(.|s, a)$

In this section we consider the problem of learning a deterministic hierarchical policy from a set of demonstrations. We assume that the different task definitions are known but the task hierarchy is not. The key problem in learning is that the demonstrations only include the primitive actions and the termination symbols of the subtasks when they are completed. In particular, they do not indicate the starting points of various subtasks, which leaves significant ambiguity in understanding the demonstration. Indeed, it was shown that the problem of inferring hierarchical policies is NP-hard when such annotations of subtasks are not given [Khardon, 1999]. This is true even when there is a single subtask whose policy is fully specified and it is known where the subtask ends in each trajectory. All the complexity comes from not knowing where the subtasks begins. In this paper, we consider the problem of efficiently acquiring such annotations by asking a minimum number of queries to the expert.

[1] When these termination symbols are not given, they can be inferred from the task definitions (which are known) and by asking additional queries to the user to rule out accidental task fulfillment. We ignore this complication in this thesis.

2.3 Probabilistic State-Dependent Grammars

The above formulation suggests that a hierarchical policy can be viewed as a form of Push Down Automata (PDA), which can be automatically translated into a Context Free Grammar (CFG). Unfortunately such a CFG will have a set of variables that corresponds to $S \times T \times S$, which makes it prohibitively complex. Instead, we adapt an elegant alternative formalism called Probabilistic State-Dependent Grammar (PSDG) to represent the task hierarchies [Pynadath and Wellman, 2000]. A PSDG is a 4-tuple $\langle V, \Sigma, P, Z \rangle$, where V is a set of variables (represented by capital letters), Σ is the terminal alphabet, P is a set of production rules, and Z is the start symbol. PSDGs generalize CFGs in that each production rule is of the form $s, t_i \rightarrow \gamma$, where s is a state, t_i is a variable, and γ is a string over variables and terminal symbols. The above production rule is only applicable in state s, and reduces the variable t_i to the sequence of variables and terminals described by γ.

It is desirable for a PSDG to be in Chomsky Normal Form (CNF) to facilitate parsing. We can represent the hierarchical policies in the MAXQ hierarchy as a PSDG in CNF as follows. For each task $t \in T$, we introduce a variable $V_t \in V$, and a terminal symbol t' that represents the termination of t. For each primitive action $a \in A$ and state $s \in S$, we introduce a variable $V_a \in V$ and a terminal string $a; s$. Root is the start symbol. Further, it suffices to restrict the rules to the following 3 types:

1. $s, t_i \rightarrow t_j \; t_i$, where t_j is a non-primitive subtask of t_i
2. $s, t_i \rightarrow a_j; \delta(s, a_j)$, where a_j is a primitive action, and δ is the transition function
3. $s, t_i \rightarrow t'_i$ where t'_i is a symbol representing the termination of task t_i.

The first rule represents the case where t_i calls $t_j \in T$ when in state s and returns the control back to t_i after it is done. The second rule allows a primitive action in A to be executed, changing the state as dictated by the transition function. The third rule is applicable if s satisfies the termination test of t_i. In a deterministic hierarchical policy, there is a single rule of the form $s, t_i \rightarrow \gamma$ for each state-task pair s, t_i. A PSDG, on the other hand, allows multiple rules of the above form with right hand sides $r_1, \ldots, r_m$, and associates a probability distribution $p(.|s, t_i)$ over the set $r_1, \ldots, r_m$. As is normally the case, we assume that a state is represented as a feature vector. Since only a small set of features is usually

relevant to the choice of a subtask, we specify the above distributions more compactly in the form of $p(r_i|s_1, \ldots, s_k, t_i)$, where $s_1, \ldots, s_k$ are the only features which are relevant for choosing the subtask.

Table 2.1 represents the skeleton PSDG equivalent to the task hierarchy of the Taxi domain (see Figure 1(e)) without the probability distributions that correspond to a specific policy. It is easy to see that for any deterministic hierarchical policy, initial state and action dynamics, the corresponding deterministic PSDG, when started from the Root symbol and the same initial state, will derive the same distribution of demonstrations.

However, the PSDG formalism is more general in that it allows us to represent stochastic hierarchical policies. In this work, we exploit this property and view the PSDG as representing a distribution over deterministic hierarchical policies. This allows us to adapt the efficient algorithms developed for probabilistic CFGs such as the inside-outside algorithm to learn PSDGs [Lari and Young, 1990].[2] It also allows us to efficiently compute some information-theoretic heuristics which are needed to guide active learning.

Table 2.1: An example of PSDG skeleton of Taxi Domain without the state information. Primitive actions start with small letters and the task names start with capitals.

$\mathsf{Root} \rightarrow \mathsf{Get}, \mathsf{Root}$	$\mathsf{Up} \rightarrow \mathsf{up}$	$\mathsf{Goto(L)} \rightarrow \mathsf{Left}, \mathsf{Goto(L)}$
$\mathsf{Root} \rightarrow \mathsf{Put}, \mathsf{Root}$	$\mathsf{Root} \rightarrow \mathsf{root}'$	$\mathsf{Goto(L)} \rightarrow \mathsf{Right}, \mathsf{Goto(L)}$
$\mathsf{Root} \rightarrow \mathsf{Refuel}, \mathsf{Root}$	$\mathsf{Put} \rightarrow \mathsf{put}'$	$\mathsf{Goto(L)} \rightarrow \mathsf{Up}, \mathsf{Goto(L)}$
$\mathsf{Dropoff} \rightarrow \mathsf{dropoff}$	$\mathsf{Left} \rightarrow \mathsf{left}$	$\mathsf{Goto(L)} \rightarrow \mathsf{Down}, \mathsf{Goto(L)}$
$\mathsf{Get} \rightarrow \mathsf{Goto(L)}, \mathsf{Get}$	$\mathsf{Goto(L)} \rightarrow \mathsf{goto}'$	$\mathsf{Refuel} \rightarrow \mathsf{refuel}'$
$\mathsf{Get} \rightarrow \mathsf{Pickup}, \mathsf{Get}$	$\mathsf{Get} \rightarrow \mathsf{get}'$	$\mathsf{Put} \rightarrow \mathsf{Goto(L)}, \mathsf{Put}$
$\mathsf{Put} \rightarrow \mathsf{Dropoff}, \mathsf{Put}$	$\mathsf{Down} \rightarrow \mathsf{down}$	$\mathsf{Right} \rightarrow \mathsf{right}$
$\mathsf{Refuel} \rightarrow \mathsf{Goto(L)}, \mathsf{Refuel}$		$\mathsf{Pickup} \rightarrow \mathsf{pickup}$
$\mathsf{Refuel} \rightarrow \mathsf{Fillup}, \mathsf{Refuel}$		$\mathsf{Fillup} \rightarrow \mathsf{fillup}$

[2]This view is not exactly correct in that a distribution over deterministic grammars requires us to first choose a deterministic grammar and use it to parse all parts of the trajectory, as opposed to making a different choice at each point. However, in practice this is perfectly adequate as we rarely encounter the same production distribution twice while parsing the same trajectory.

2.4 Active Learning of Hierarchical Policies

We now present our main algorithm for active learning of PSDGs from sets of demonstrations (see Algorithm 1), followed by a more detailed description of each algorithm component. The algorithm takes a set of task names, primitive actions, and trajectories as input (lines 1 & 2). It then constructs an initial uninformed distribution of PSDGs by including every possible production in some MAXQ hierarchy with all production distributions initialized to be uniform (line 3). It then iterates over the following steps. First, the learner heuristically selects a trajectory, trj, to annotate (line 5, details in Section 2.4.4). It then parses the trajectory using a version of the inside-outside algorithm and constructs a CKY table data structure, which compactly represents all possible parse trees of the trajectory and their probabilities (line 6, details in Section 2.4.1) . Finally in lines 7 through 21 (details in Section 2.4.2), it uses Bayesian active learning to ask queries about the intention structure of the trajectory. The information gathered from the queries is used to update the CKY-table (line 16). Whenever an unambiguous parse is found for a sub-trajectory, it is used to generate concrete examples to learn a production rule (lines 17-20, details in Section 2.4.3). The rest of the section describes the different steps of the algorithm in more detail.

An overview of our approach is represented in Figure 2.1

2.4.1 Parsing the Trajectories

We apply the inside-outside algorithm to parse a given trajectory using the current PSDG, which is assumed to be in CNF. The inside-outside algorithm is a way of re-estimating production probabilities in a probabilistic context-free grammar. The algorithm is based on dynamic programming and incrementally builds a parsing table called CKY table to compactly represent all possible parse trees of all segments of the trajectory.

For each segment of the trajectory trj and indices i, j, and each variable A, it recursively computes two different probabilities (shown in Figure 2.4):
1) *The inside probability*, $\alpha_{i,j}(A)$, which is the probability that A derives the sub-sequence of trj from i to j. This probability is computed by the following equation:

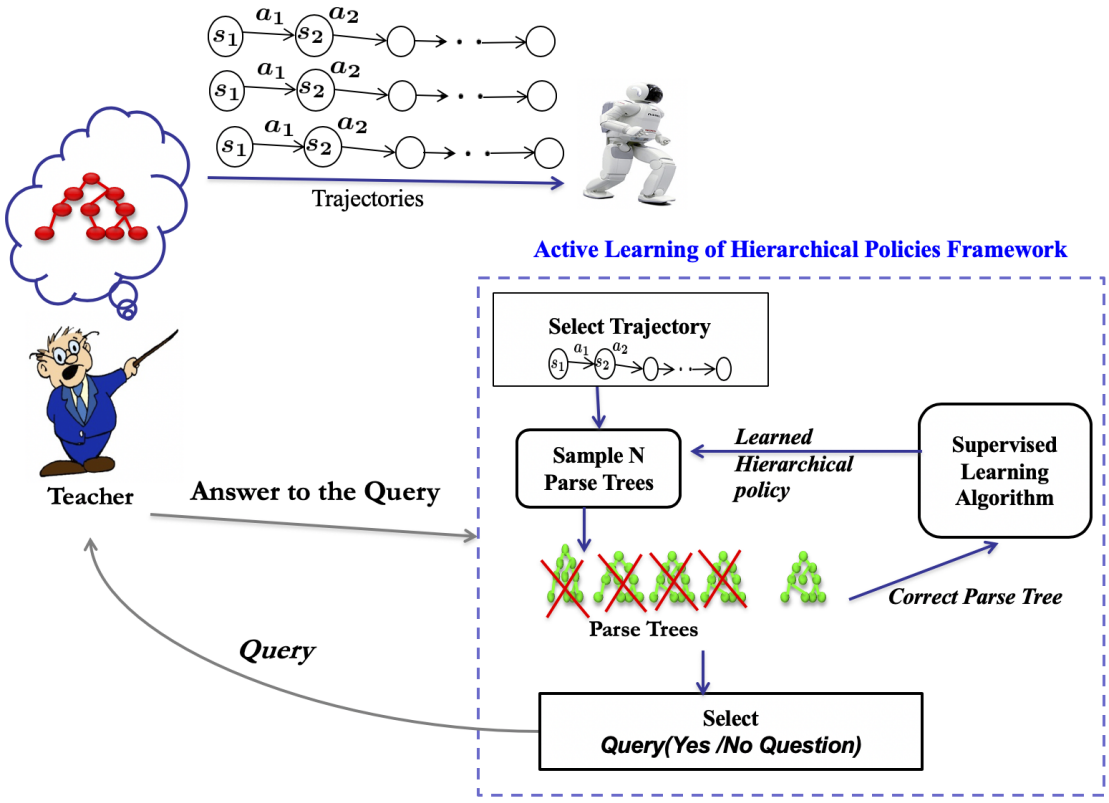


Figure 2.1: Active learning of hierarchical policy framework

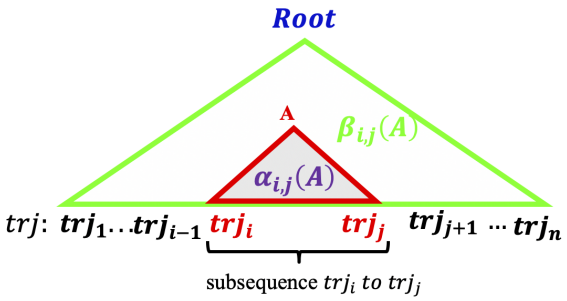


Figure 2.2: The inside, $\alpha_{i,j}(A)$ and outside, $\beta_{i,j}(A)$ probabilities of a sub-sequence, i to j of a trajectory trj

$$\alpha_{i,j}(A) = \sum_{B,C} \sum_{i \leq k \leq j} P(A \rightarrow B, C) \alpha_{i,k}(B) \alpha_{k+1,j}(C) \tag{2.1}$$

Algorithm 1 Learning Hierarchical Policies

Input: A set of trajectories Trjs, tasks T, actions A.
Output: Hierarchical policies
Initialize the PSDG
loop
trj = **SelectTrajectory**(Trjs)
Generate CKY table for trj
Sample N parse trees for trj from CKY table
$L \leftarrow \emptyset$ // a set of answered queries
while there is not a unique parse tree **do**
  for each query $q \in Q = \{(s_i, task_j)_{i=1}^{|\text{trj}|}\}_{j=1}^{|T|}$ **do**
    Compute $H(\text{Answer})$
  end for
  newQuery $= argmax_{q \in Q}\{H(\text{Answer})\}$
  answer = **AskQuery**(newQuery)
  $L \leftarrow (L \bigcup \text{answer})$
  UpdateCKYTable(L)
  if there is a unique parse (L) for a sub-trajectory **then**
    GenerateExamples(CKY, L)
    run TILDE // supervised learning algorithm for updating the hierarchical policy probabilities
  end if
end while
end loop

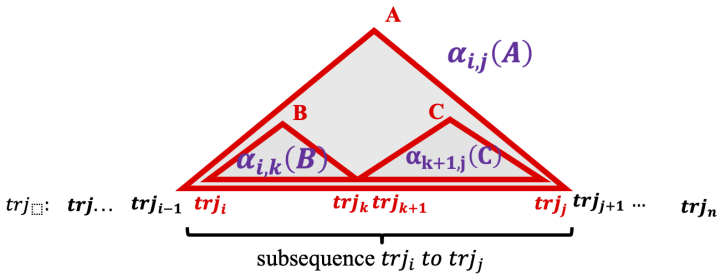


Figure 2.3: The inside probability $\alpha_{i,j}(A)$ of a sub-sequence i to j of a trajectory trj

2) *The outside probability*, $\beta_{i,j}(A)$, which is the probability that trajectory $trj_1, ..., trj_{i-1}, A$,

$trj_{j+1}, ..., trj_n$ can be derived from the start symbol $Root$. This probability is computed by the following equation:

$$\beta_{i,j}(A) = \sum_{B,C} \sum_{1 \le k \le i} P(B \to C, A) \alpha_{k,i-1}(C) \beta_{k,j}(B) + \sum_{B,C} \sum_{j < k \le n} P(B \to A, C) \alpha_{j+1,k}(C) \beta_{i,k}(B)$$

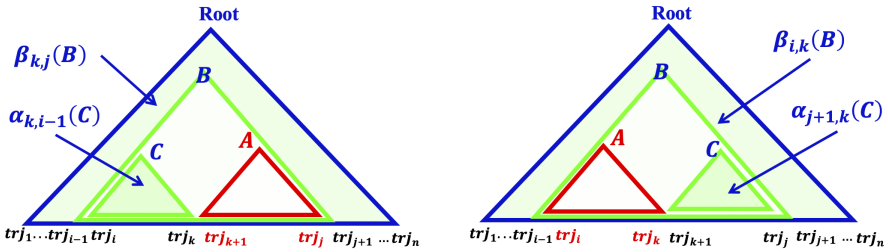


Figure 2.4: The outside probability $\beta_{i,j}(A)$ of a sub-sequence i to j of a trajectory trj

The current parameters of the distributions of the PDSG (which are uniform in the beginning) are used to compute these probabilities. Since all the intermediate states are observed, the state-dependence of the grammar does not pose any additional problems. Thanks to the context-free nature of PSDG, and the full observation of the intermediate states, the α's and β's allow us to efficiently compute various other probabilities. For example, the probability that a trajectory (of length n) is generated starting from the Root task is $\alpha_{1:n}(\mathsf{Root})$. See [Lari and Young, 1990] for details.

2.4.2 Query Selection via Bayesian Active Learning

We now consider selecting the best queries to ask in order to uncover the true intention structure (parse tree) of a selected trajectory. In particular, each query highlights a certain trajectory state s and asks if a particular task B is part of the intention of the user in that state (i.e. part of the task stack). The answer is "yes" if B is part of the true task stack at that point, and "no" otherwise.

The question now is how to select such queries in order to efficiently identify the parse tree of the trajectory. For this we follow the framework of Bayesian Active Learning (BAL) for query selection [Golovin et al., 2010] , which considers the efficient identification of a target hypothesis among candidates via queries. We begin by generating a large set of N hypothesis parse trees for the trajectory by sampling from the parse tree distribution induced by our current PSDG estimate (line 7). BAL then attempts to heuristically select the most informative query for identifying the correct parse tree. After receiving the answer to each query the learner removes all parse trees or hypotheses that are not consistent with the answer and updates the CKY table appropriately. It also updates the CKY table to exclude all entries that are not consistent with the answer to the query. Querying continues until a single parse tree remains, which is taken to be the target. If all parse trees happen to be eliminated then we move on to select another trajectory, noting that information gathered during the querying process is still useful for learning the PSDG.

It remains to specify the query selection heuristic. A standard heuristic inspired by generalized binary search is to select the query that maximizes the expected information gain (IG) of the answer. Prior work has shown that this heuristic achieves near optimal worst case performance [Dasgupta, 2004]. It is straightforward to show that this heuristic is equal to the entropy of the answer distribution, denoted by H(Answer).

$$\begin{aligned} IG(Parse; Answer) &= H(Parse) - H(Parse|Answer) \\ &= H(Answer) - H(Answer|Parse) \\ &= H(Answer) - 0. \end{aligned}$$

The entropy of the answer distribution is computed as follows:

$$\begin{aligned} H(Answer) &= -p(Answer_q = yes|N) \times log(p(Answer_q = yes|N)) \\ &\quad -p(Answer_q = no|N) \times log(p(Answer_q = no|N)) \end{aligned}$$

where N is the number of parse tree samples, and

$$p(Answer_q = yes|N) = \frac{(\#Samples\ with\ Answer_q{=}yes)}{(\#Samples\ of\ Answer_q)}.$$

Thus, we select the question that has maximum entropy over its answer distribution (line 13). The Bayesian active learning proceeds until there is at most one parse tree left in the set. If exactly one is left, then the CKY table is updated to remove all other possible parses from consideration. If none is left, we move on to select another trajectory.

2.4.3 Example Generation and Generalization

Recall that the answers to the queries are simultaneously used to update the CKY table as well as to remove the inconsistent parse trees from the sample. We consider a parse of a trajectory segment between the indices i and j to be unambiguous if the inside and outside probabilities of some variable (task) B for that segment are both 1. When that happens for some task B, and its subtasks, say C and B, we create positive and negative training examples of the production rule for the pair (s, B), where s is the state at the trajectory index i where B was initiated. The positive examples are those that cover the correct children of B, namely C and B, and the negative examples are those that were removed earlier through queries.

To generalize these examples and ignore irrelevant features in the state, we employ a relational decision tree learning algorithm called TILDE [Blockeel and De Raedt, 1998]. TILDE uses a form of information gain heuristic over relational features and learns the probabilities for different right hands sides of production rules of PSDG. Ideally only the features of the state relevant for the correct choice of the subtask are tested by the tree. These probabilities are used to compute the αs and βs during the future parsing steps.

2.4.4 Trajectory Selection Heuristics

In this section we focus on the problem of trajectory selection. On the face of it, this appears to be an instance of the standard active learning problem for structured prediction problems such as parsing and sequence labeling [Baldridge and Osborne, 2003, Settles and Craven, 2008]. A popular heuristic for these problems is based on "uncertainty sampling," which can be interpreted in our context as picking the trajectory whose parse is most ambiguous.

A natural measure of the ambiguity of the parse is the entropy of the parse tree distribution, which is called the "tree entropy" given by:

$$TE(trj) = -\sum_{v \in V} p(v|trj) \times log(p(v|trj)) \tag{2.2}$$

where V is a set of all possible trees that our current model generates to parse trj.

The probability of each trj is $p(trj) = \sum_{v \in V} p(v, trj)$, which is the sum over the probability of all possible trees, where $p(v, trj)$ denotes the probability of generating trj via the parse tree v. The distribution of parsing likelihoods for trj is $\sum_{v \in V} p(v, trj|M) = 1$. Both $p(trj)$ and $TE(trj|M)$ can be efficiently computed using the CKY table without enumerating all parse trees. In [Hwa, 2004], the author introduced the bottom-up, dynamic programming technique of computing inside probabilities, which can be used to efficiently compute the probability of a trajectory.

Tree entropy (TE) is one of the first heuristics we considered and evaluated. Somewhat surprisingly we found that it does not work very well and is in fact worse than random sampling. One of the problems with tree entropy is that it tends to select long trajectories, as their parse trees are bigger and hence are more likely to be ambiguous. As a result they require more effort by the user to disambiguate. To compensate for this researchers have tried length-normalized tree entropy (LNTE) as another heuristic [Hwa, 2004]. We also evaluated this heuristic in our experiments.

$$LNTE = \frac{TE(trj)}{length(trj)} \tag{2.3}$$

Our third heuristic, *cost-normalized information (CNI)*, is based on the observation that the tree entropy represents the amount of information in the correct parse tree, and hence can be viewed as a proxy for the number of binary queries we need to ask to identify the parse. Indeed, we have empirically verified that the number of queries needed grows linearly with the tree entropy. However it is not a good proxy for the information we gain from a labeled trajectory. Indeed, it would be best to use a trajectory with zero tree entropy, i.e., a trajectory with an unambiguous parse to learn, as it needs no queries at all. However, such trajectories might also be useless from the learning point of view, if they are

already predicted by the current model. An ideal trajectory would be something the learner has only a low probability of generating by itself, but would require the least number of queries to learn from, *given* that trajectory. This suggests that given the same cost, we should select the trajectories that are most informative or most surprising. The amount of surprise or novelty in an event E is nicely formalized by Shannon's information function $I(E) = -\log(p(E))$. Thus we are led to the heuristic of maximizing cost-normalized information of the trajectory, which is approximated by:

$$CNI = -\log(p(trj))/TE(trj) \tag{2.4}$$

Fortunately the probability of the trajectory is something we can easily compute from the CKY table.

2.5 Empirical Results

2.5.1 Domains

We consider five domains for evaluating our active learning approach.

Subtraction Domain: This domain is motivated by applications to automated tutoring systems, which raises the problem of allowing human teachers to easily specify hierarchical tasks without necessarily understanding the semantics and syntax of a formal hierarchy description language. There are two multi-digit numbers $A = A_n, ..., A_1$ and $B = B_m, ..., B_1$, where A $>$ B, and the goal is to induce the standard "pencil and paper" hierarchical procedure for subtracting B from A (similar to [VanLehn, 1987]). This procedure is shown in Figure 2.5 and involves primitive actions such as crossing numbers out, overwriting numbers, and higher-level tasks such as borrowing. The state consists of the two digits being subtracted annotated by the result of all prior primitive actions. The trajectories used for training involve a number of digits ranging from 1 to 15 as in [VanLehn, 1987].

Kitchen Domain: This stochastic version of the kitchen domain originally described in [Natarajan et al., 2011], models the process of preparing a meal involving a main dish and one or more side dishes. The primitive actions involve steps such as pouring, fetching ingredients, and baking. The agent has two high-level goals: (1) preparing dishes from a

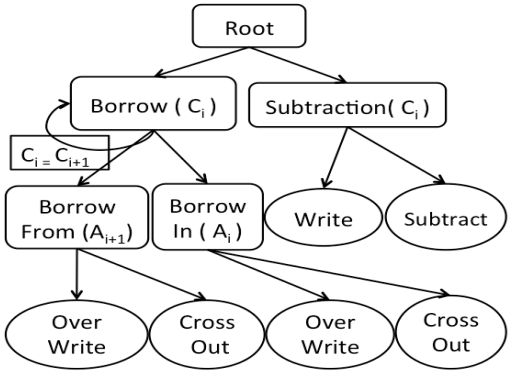


Figure 2.5: The task hierarchy of Subtraction Domain.

recipe, which contains a main dish and a side dish, and (2) preparing the main dish and side dish using some instant food. There are 2 shelves with 3 ingredients each. Each shelf has a door that must be opened before fetching ingredients. The state consists of the contents of the bowl, the ingredient on the table, the mixing state, the temperature state of the ingredient, and the door state. Each training trajectory specifies the main meal and side dishes and ends when they have been successfully prepared. The user's actions are: open the doors, fetch the ingredients, pour them into the bowl, mix, heat and bake the contents of the bowl, or replace an ingredient back to the shelf. The episode begins with all the ingredients in the shelves and the doors closed. The episode ends when the user achieves the goal of preparing a main dish and a side dish [Natarajan et al., 2011].

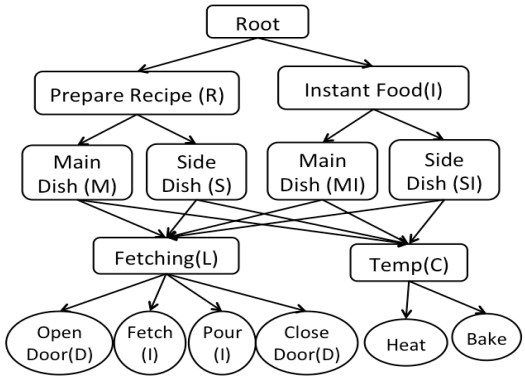


Figure 2.6: The task hierarchy of Kitchen Domain.

Assembly Domain: This domain models a furniture assembly problem, which is another domain where a computer assistant would need knowledge of a hierarchical assembly procedure. The agent's objective is to assemble cabinets, which contain one or more shelves. There are two types of shelves: shelves with a door and shelves with drawers. The agent starts by assembling shelves and connecting them together. For assembling each shelf, it first assembles a frame and a door or a drawer, then it connects the door or drawer to the frame. Each door, drawer and frame have different pieces, such as a handle, left side, right side, etc. Each frame has a connector for connecting doors or drawers. The actions are to add each side of the frame, add the door connector, add the door handle, connect shelves, etc. The task hierarchy of this domain is represented in Figure 2.7.

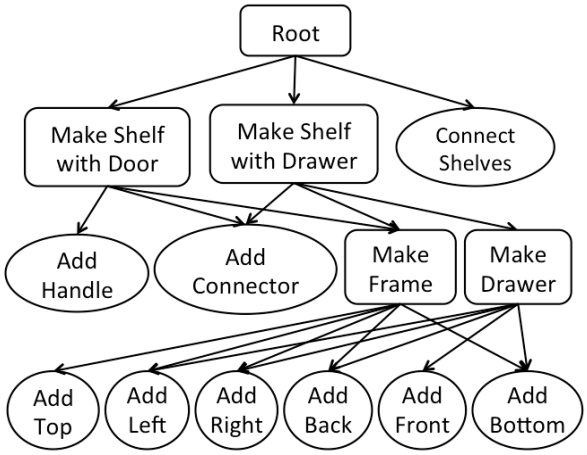
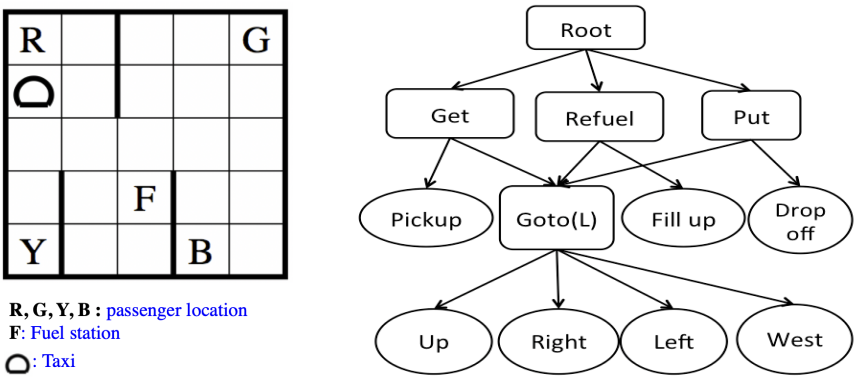


Figure 2.7: The task hierarchy of Assembly Domain.

Taxi Domain: This is a stochastic version of a domain that is a commonly used benchmark from hierarchical reinforcement learning, described in [Dietterich, 2000]. We use the more complex version that involves refueling the taxi. In each episode, the taxi starts in a randomly chosen location and with a randomly chosen amount of fuel (ranging from 5 to 12 units). The episode ends when the passenger is successfully transported. The stochastic dynamics dictates that with certain probability the taxi fails to move. The Taxi domain and task hierarchy of this domain is represented in Figure 2.8 (left) and Figure 2.8 (right), respectively.

RTS Domain: This stochastic version of the RTS domain originally described in



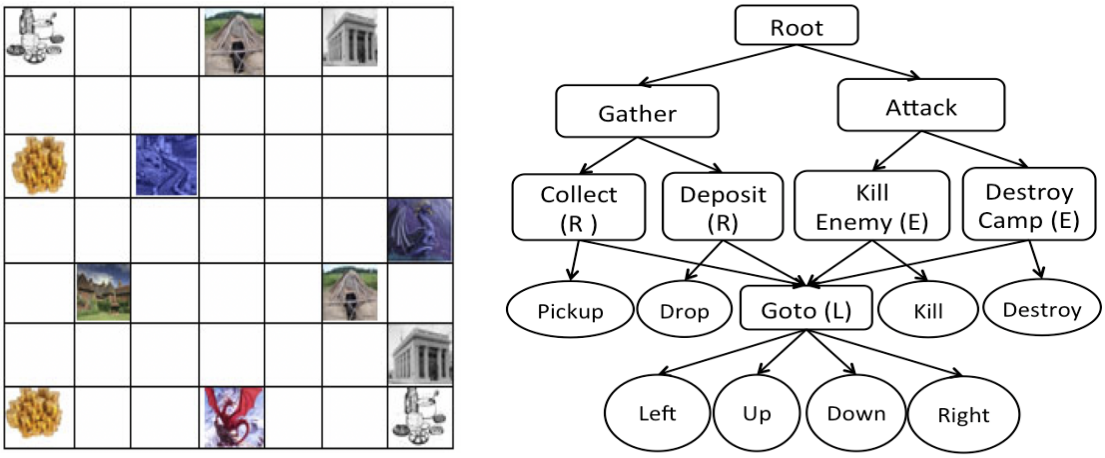
Taxi Domain The task hierarchy of Taxi Domain

Figure 2.8: Taxi Domain and the corresponding task hierarchy.

[Natarajan et al., 2011], models a real-time strategy (RTS) game where an agent must move around a map collecting resources and attacking enemies. The agent's objective is to gather resources and kill enemies in a 7×7 grid. To gather a resource(i.e gold, food) it has to be collected and deposited to a designated location (e.g. gold in a bank, food in a granary). There are two locations for each resource and its storage and also two kinds of enemies, red and blue dragons. The agent has to kill the enemy dragon and destroy an enemy castle of the same color as the dragon. The actions that the user can perform are to navigate in 4 directions, open the 4 doors, pick up, put down and attack [Natarajan et al., 2011]. The goals and sub- goals are specified using a relational hierarchy. The episode ends when the user achieves the highest level goal. The RTS domain and the task hierarchy of this domain are represented in Figure 2.9.

2.5.2 Experiments

In each domain, we collected a set of training and test trajectories from a hand-coded hierarchical policy that follows the hierarchical structure of each domain. The trajectories are stored with the corresponding hierarchical intention structures which are used to answer queries. We use 10 training trajectories and 30 test trajectories in total. The results are averaged over 10 runs of the system with the same hierarchical policy but with a different



RTS Domain The task hierarchy of RTS Domain

Figure 2.9: RTS Domain and the corresponding task hierarchy.

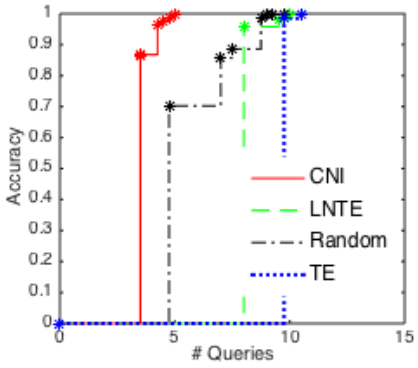
set of randomly generated training and test trajectories. We used 70 sampled parse trees to select subgoal queries in each iteration.

2.5.2.1 Accuracy of Hierarchy Learning

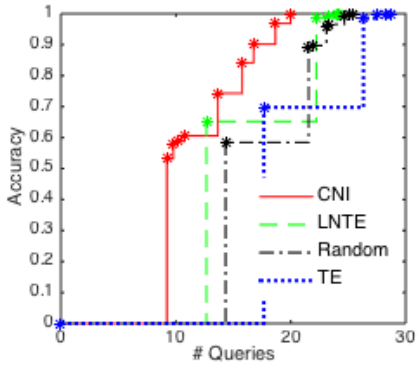
Since one of the main motivations of this work is to learn the same hierarchical structure as used by the teacher, we first evaluate the accuracy of the hierarchies learned by our approach. In each domain, we compared the performance of our preferred heuristic, namely, maximizing *cost-normalized information (CNI)*, with three other trajectory selection strategies for learning hierarchical policies: 1) *Random* , 2) Maximizing *tree entropy (TE)*, and 3) Maximizing *length-normalized tree entropy (LNTE)*

Each heuristic was trained on the same hierarchical policy and the same set of training examples. Our performance metric is *accuracy*, which measures the fraction of times along the trajectory that the learned hierarchical policy agrees with the target policy about both primitive actions and the intention stack on test set.

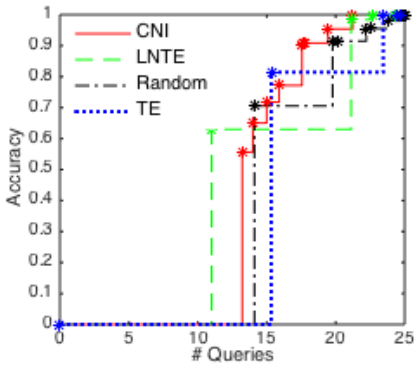
Figures 2.10 (a)-(d), and (f) show accuracy vs. the number of queries for each domain. Each point on the plot shows the performance after learning from one more trajectory than the previous point. The curve is flat between the trajectories because the performance



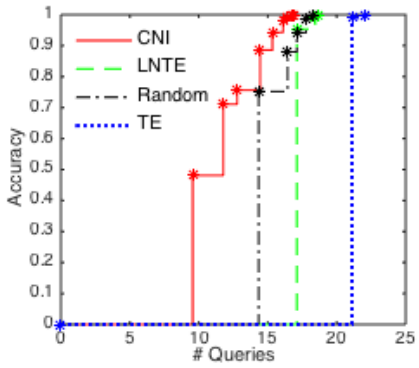
(a) Subtraction Accuracy



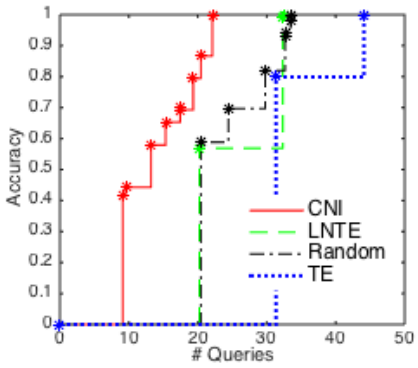
(b) Assembly Accuracy



(d) RTS Accuracy



(f) Taxi Accuracy



(c) Kitchen Accuracy

Figure 2.10: Comparison between different heuristics on trajectory selection strategies in five domains. The X-axis shows the number of queries and the Y-axis shows the accuracies of the task stacks at different points in the test trajectory.

does not change until a new trajectory is correctly parsed and generalized. Unlike typical learning curves, the different points are not uniformly spaced because the number of queries used to learn each trajectory is usually different. One clear trend is that fewer queries are needed to disambiguate each successive trajectory. This is expected because it gets easier to infer the intentions behind actions, as more is learned about the target policy. Indeed, towards the end of the trial, many trajectories are disambiguated using only 1 or 2 queries.

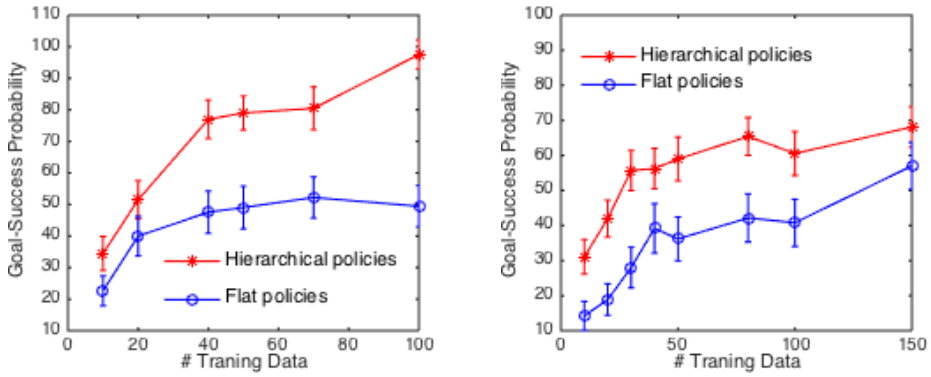
We first observe that selecting trajectories using TE performs poorly, even worse than random in all domains. The reason is that maximizing tree entropy encourages the selection of the most ambiguous trajectory, which maximizes the number of queries asked. Normalizing by length mitigates this effect somewhat and makes it perform closer to or better than the random strategy in the RTS and the Kitchen domains. However, the random strategy still performs better in the Taxi domain. This is because length does not necessarily reflect the number of queries needed to disambiguate a trajectory.

Our preferred heuristic, CNI, consistently performs better than all other heuristics in all domains with a possible exception of LNTE in RTS. These results indicate that CNI's approach of taking both the information gain and the cost of querying into account is effective. The RTS domain is an exception in that the trajectories in that domain lack diversity, which makes the different heuristics perform similarly. The results also show that it is important for trajectories to be diverse in terms of number and set of subgoals. The shapes of the different curves suggests that CNI allows the learner to learn from more trajectories by spending fewer queries on each trajectory, achieving better performance after the same number queries as other heuristics.

2.5.2.2 Comparing to Flat Policy Learning

Here we compare the performance of actively learned hierarchical policies to traditional imitation learning of flat (non-hierarchical) policies. We compared the performance of our algorithm using CNI trajectory selection with the flat policy learning algorithm, whose policies are learned using TILDE [Blockeel and De Raedt, 1998]. Figures 2.11 (a) and (b) show goal-success percentage on a test set of problems vs. the number of the training trajectories for the Taxi and RTS domains. The performance metric, *goal-success accuracy*,

is the percentage of the learned trajectories that reaches to the goal state using the learned policies and measured on the test set.



(a) Taxi goal-success (b) RTS goal-success

Figure 2.11: Figures (a) and (b): Comparison between flat and hierarchical policy learning in Taxi and RTS domains. The X-axis shows the number of training trajectories and the Y-axis shows the goal-success percentage.

The results show that our active hierarchy learning approach is able to learn the tasks much more quickly in terms of number of training trajectories compared to the flat learning algorithm. Note, however, that this evaluation does not take into account the cost to the expert of answering the intention queries on the generated trajectories, which is zero for the flat learner. Thus, an overall comparison of the two approaches would need to weigh the relative cost of generating trajectories and answering queries. However, we see that the learning curve for the flat method is increasing very slowly for both problems, suggesting that for reasonable query costs the hierarchical learner would likely make the most efficient use of the expert's time.

2.6 Summary

We studied active learning of hierarchical policies in the form of PSDGs from trajectories generated by a hierarchical policy. We showed that a previously introduced plan representation language, namely PSDG, nicely captures hierarchical policies in the MAXQ framework and is strictly more general. We developed a novel two-level active learning

framework, where the top level selects a trajectory and the lower level actively queries the teacher about the intention structure at selected points along the trajectory. We developed a new information-theoretically justified heuristic, cost-normalized information, for selecting trajectories, and employed Bayesian active learning for the lower-level query selection.

Experimental results on five benchmark problems indicate that our approach compares better to a number of baselines in learning hierarchical policies in a query-efficient manner.

Our cost-normalized information heuristic outperforms the other heuristics and is justified from an information-theoretic perspective.

Chapter 3: Adaptive Submodularity with Varying Query Sets: An Application to Active Multi-label Learning

Adaptive submodular optimization, where a sequence of items is selected adaptively to optimize a submodular function, has been found to have many applications from sensor placement to active learning. In this chapter of the thesis, we extend this work to the setting of multiple queries at each time step, where the set of available queries is randomly constrained. A primary contribution is to prove the first near optimal approximation bound for a greedy policy in this setting. A natural application of this framework is to crowd-sourced active learning problem where the set of available experts and examples might vary randomly. We instantiate the new framework for multi-label learning and evaluate it in multiple benchmark domains with promising results.

3.1 Introduction

Adaptive submodularity has found many applications ranging from sensor placement to active learning. The goal is to optimize a submodular function, such as the expected information gathered by a set of sensors or a set of training examples, with a limited budget. Unlike the static setting of submodular optimization, the agent is allowed to be *adaptive* in the sense that each new selection could depend on the stochastic outcomes of previous selections.

The current approaches to adaptive submodular optimization [Golovin and Krause, 2011] assume that the set of possible choices is fixed for all of time. They provide a greedy algorithm with a near optimal approximation guarantee that exploits the adaptive submodularity of the objective function. However, in real-world applications such as crowd sourcing, all selections may not always be available. We formalize this problem through a set of constraints, where the constraints restrict the set of choices available to the agent. The constraints themselves can change randomly at each step. We address this problem of adaptive submodularity with varying queries by generalizing the framework of adaptive

submodular optimization and provide a greedy algorithm with a near-optimal approximation guarantee.

An interesting application that motivates our research is active multi-label learning where an example has multiple labels, and each expert can only label a subset of them. A common scenario is crowd sourcing, where different workers are experts at labeling different classes, e.g., identifying different species of plants or animals in pictures. Moreover, only some experts are available at any time. This motivates the following problem. Given a set of unlabeled instances, and a set of experts who randomly become available at a given time, how best to choose the next example to be labeled and by which expert?

We formalize this problem by generalizing the framework of adaptive submodular optimization in two orthogonal directions. First, following [Chen and Krause, 2013], we allow a *set* of queries to be asked at each time step, where the sets themselves are constrained. Second, we allow the constraints to vary at each step according to a stationary distribution. In multi-label learning, the query sets correspond to the set of labels that can be labeled by a single expert. At every time step, the system chooses an available query set and gets the answers back. The system refines its model based on the answers, and the cycle repeats. Importantly, the set of query sets available varies randomly according to a stationary probability distribution and is avaialble to the system only just in time for the query. The goal is to maximize the information obtained with a fixed query budget.

Our main contribution is to reduce this new setting to the standard adaptive submodularity setting, implying that a natural greedy adaptive policy is near-optimal. The key idea behind the proof is to view the greedy adaptive policy as an efficient implementation of a permutation policy, which orders all possible queries by a permutation, and then selects the first query in the ordering available at the current time step. The permutation policies are without loss of generality when the query set distribution is stationary and independent of the label distribution.

We empirically evaluate this framework on the active multi-label learning scenario descried above. There are two main lines of prior work on active multi-label learning. In the first line of work, all labels of a selected example are queried at once. Several heuristics for query selection have been explored, including uncertainty sampling [Brinker, 2005, Singh et al., 2009], minimizing the smallest margin of several one-vs-all classifiers [Tong

and Koller, 2002], Max Loss and Mean Max Loss (MML) [Li et al., 2004], Maximum Loss Reduction with Maximum confidence (MMC) [Yang et al., 2009], and others. In the second line of work, each query consists of an example-label pair. Query selection heuristics include those based on uncertainty and diversity [Huang and Zhou, 2013, Wu et al., 2014], MML based on relationship between labels and samples [Li et al., 2004], and chi-square statistics [Ye et al., 2015]. Our framework generalizes both of these lines of work by allowing queries over arbitrary subsets of labels, where the allowed subsets may vary randomly across time.

While most of the previous approaches are driven by heuristics with no performance guarantees, [Vasisht et al., 2014] describes an approach with near-optimality guarantees on selecting batches of examples, for which all labels of those examples are queried. Importantly, this near-optimality guarantee is non-adaptive and is closely tied to a gaussian process regression model. It also does not apply to our more general scenario where queries can be over arbitrary subsets of labels. To the best of our knowledge, our work is the first to provide such adaptive near-optimality guarantees in any multi-label setting.

3.2 Adaptive Submodular Optimization with Varying Query Sets

In this section we introduce a general framework for adaptive submodular optimization with varying queries, where we have a set of items in unknown states. The states can be queried subject to some exogenously chosen constraints that only appear at the time of the query. The goal is to find an adaptive query policy that optimizes a submodular objective function with a budget on the total number of queries. Our framework naturally captures active multi-label learning via crowd sourcing, where each worker is an expert on a subset of the labels and is randomly available. The set of items corresponds to the cross product of instances and labels. Each item can be in one of two states, *yes* and *no*. The query constraints C specify which sets of items can be queried at a given time. For example, we can allow different workers to be experts on different label (or instance) subsets, where the set of available experts varies randomly. Our analysis assumes that the subsets C at each step are drawn i.i.d. from an unknown distribution P_c, although the approach is applicable in other cases.

In this work, we borrow the framework of adaptive submodular optimization [Golovin and Krause, 2011] and extend it to randomly varying queries and query sets. We then develop a greedy algorithm which has a near-optimal approximation guarantee by reducing it to the standard adaptive submodular optimization setting. The rest of this section reviews the previous work on adaptive submodularity and extends it to varying queries and query sets.

3.2.1 Adaptive Submodularity

We are given a finite set of items E and a finite set of observable states O. Each item $e \in E$ is associated with a state $o \in O$ through a function $\phi_o : E \rightarrow O$, which is called a *realization*. It is assumed that the realization ϕ_o is a random variable with known distribution $\mathbb{P}_o[\phi_o]$. Initially we are unaware of realization of the items, but they will be revealed to us via queries. Let ψ_o denote a *partial realization*, i.e., a subset of items and their observations, where $\mathrm{dom}(\psi_o)$ represents the items in ψ_o that have been observed. Moreover, we are given a utility function $f : 2^E \times O^E \rightarrow \Re \geq 0$, which maps a set of items and their observations to a real value.

The goal is to come up with a *query policy* π, or more simply a *policy*, that maximizes the expected utility under a total query budget. A query policy is a mapping from ψ_o to items in E, which specifies the element to query next given the history of observations. The policy π is executed by iteratively selecting the element suggested by π given the history of observations, and obtaining the observation for the selected element from the true realization.

Definition 1. *The* expected utility *of a policy in a partial realization ψ_o for a horizon (query budget) of l is $f_{avg}(\pi, \psi_o, l) \stackrel{\mathrm{def}}{=} \mathbb{E}[f(E(\pi, \psi_o, l, \phi_o), \phi_o)]$, where $E(\pi, \psi_o, l, \phi_o)$ denotes the set of elements selected by π for l steps starting with the partial realization ψ_o when the true realization is ϕ_o. Let $f_{avg}(\pi, l) \stackrel{\mathrm{def}}{=} f_{avg}(\pi, \{\}, l)$, where $\{\}$ is an empty partial realization.*

An l-horizon policy π^* is optimal if $f_{ave}(\pi^*, \psi_o, l)$ is at least as high as that of any other policy for all partial realizations $\psi_o \in \Psi$. The theory of Markov Decision Processes implies

that optimal l-horizon policies always exist, although they depend on l and the distribution of ϕ [Puterman, 1994].

In general, the problem of computing an optimal policy is NP-hard. However, we can efficiently compute a near-optimal greedy policy when the utility function f satisfies a diminishing returns property. To make these notions more precise, we now summarize the adaptive submodular optimization framework of [Golovin and Krause, 2011].

Definition 2 (Conditional expected marginal benefit[Golovin and Krause, 2011]). *Given a partial realization ψ_o, the conditional expected marginal benefit of e denoted $\Delta_f(e|\psi_o)$, is:*

$$\Delta_f(e|\psi_o) \stackrel{\text{def}}{=} \mathbb{E}_{\phi_o \sim \psi_o}[f(\text{dom}(\psi_o) \cup \{e\}, \phi_o) - f(\text{dom}(\psi_o), \phi_o)] \tag{3.1}$$

where ϕ_o is a random variable, $\phi_o \sim P[\phi_o|\psi_o]$, and the expectation is taken with respect to $\mathbb{P}[\phi_o|\psi_o]$.

The expected marginal benefit $\Delta_f(e|\psi_o)$ gives the additional expected utility of an item e given the current partial realization ψ_o. Adaptive monotonicity requires that it is always non-negative.

Definition 3 (Adaptive Monotonic [Golovin and Krause, 2011]). *A function f is adaptive monotonic with respect to distribution $P_o[\phi_o]$ if the conditional expected marginal benefit of any query is nonnegative, i.e., for all ψ_o with $P_o[\psi_o]$ and all $e \in E$ we have $\Delta_f(e|\psi_o) \geq 0$.*

A partial realization ψ_o is said to be a *subrealization* of ψ'_o, written as $\psi_o \leq \psi'_o$, if its observed elements are a subset of those of ψ'_o, i.e., $dom(\psi_o) \subseteq dom(\psi'_o)$. Adaptive submodularity captures the diminishing returns property that the expected marginal benefit of a query never increases with more prior observations.

Definition 4 ([Adaptive Submodularit [Golovin and Krause, 2011]). *A function f is adaptive submodular w.r.t. $P_o[\phi_o]$ if for all ψ_o and ψ'_o such that $\psi_o \leq \psi'_o$ and for all $e \in E$, we have $\Delta_f(e|\psi'_o) \leq \Delta_f(e|\psi_o)$.*

An approximate greedy policy picks the element which maximizes the expected marginal benefit of an item modulo a multiplicative approximation factor α.

Definition 5. *[Golovin and Krause, 2011] An* α-approximate greedy policy *for* $\alpha \geq 1$ *w.r.t. to the utility function* f *is a policy* π *which, for any partial realization* ψ_o*, picks an item* $\pi(\psi_o)$ *whose marginal expected benefit is* $\geq \Delta_f(e|\psi_o)/\alpha$ *for any other item* e*.*

The following theorem from [Golovin and Krause, 2011] shows that when the utility function is adaptive monotonic and adaptive submodular, an approximate greedy policy with a query budget l has a multiplicative approximation bound relative to an optimal policy with a query budget k. When $l = k$ and $\alpha = 1$, the greedy policy is $(1 - e^{-1})$-optimal.

Theorem 1. *([Golovin and Krause, 2011] Theorem 5.2.) If* f *is adaptive monotonic and adaptive submodular, then for any* $\alpha-$*approximate greedy policy* π *for* $\alpha \geq 1$*, optimal policies* π^**, and positive integers* l, k*,* $f_{avg}(\pi, l) \geq (1 - e^{\frac{-l}{\alpha k}}) f_{avg}(\pi^*, k)$*.*

The next two sections generalize this model in two orthogonal directions. Section 3.2.2 extends it by randomly constraining the set of items to be selected. Section 3.2.3 allows multiple items to be queried at the same time. Finally, Section 3.2.4 combines the two extensions and allows querying sets of items, where the query sets are constrained randomly.

3.2.2 Extension to Varying Item Sets

We now introduce an extension of adaptive submodular optimization to a setting, where at each step the learner is constrained to select an item from an exogenously chosen random subset of items $C \subseteq E$. This naturally models problems such as crowd-sourcing where different workers have different expertise and not all of them are available all the time. We assume that the availability of items does not depend on their state distribution.

Assumption 1 The item set $C \in 2^E$ is i.i.d. according to a fixed but unknown distribution P_c, which is independent of P_ϕ.

Definition 6. *A valid policy* π *maps a partial realization* ψ *and item set* C *to an item in* C*.*

Definition 7. *The value of a valid policy in partial realization* ψ *for horizon* l*,* $f_{c,ave}(\pi, \psi, l) \stackrel{def}{=} E_{\vec{C}} E_{\phi_o \sim \psi}[f(M(\pi, \phi_o, \vec{C}), \phi_o)]$ *where* $\vec{C} = C_1, \ldots, C_l$ *is the vector of item sets available at steps* $1, \ldots, l$*, and* $C_i \sim P_c$ *i.i.d. and* $M(\pi, \phi_o, \vec{C})$ *represents the set of all items chosen until step* l*.*

An optimal l-horizon policy maximizes $f_{c,\psi,avg}$ for a query budget of l for all partial realizations ψ.

We seek to reduce this setting to the standard adaptive submodularity setting. The major difference in our setting is that the policy cannot choose an item until the available set of items C is known; so we cannot directly appeal to the results of the standard setting. To get around this limitation, instead of items we let the policies choose permutations over items. Given a set of allowed items, the first item in the set according to the permutation will be chosen. While there are an exponentially large number of permutations to consider, we will later show that such permutation policies can be implemented efficiently.

Let $U = \mathrm{Perm}(E) = \{\mu_1, \ldots, \mu_P\}$ represent all permutations over E. We implement the constraints via a random variable ϕ_c, which represents a mapping from permutations to subsets of E. $\phi_c : U \rightarrow 2^E$. The reason to make the constraints a function of the permutation is merely for book keeping and avoiding time indices. In fact we assume that they are independent of the permutation.

Assumption 2 The constraints are independent of μ, i.e., $Pr(\phi_c(\mu) = C) = P_c(C)$.

Definition 8. *A permutation policy is a mapping from the set of partial realizations Ψ to permutation set U. Given a permutation policy $\pi_{\mathrm{Perm}} : \Psi \rightarrow U$, the query policy it implements $\pi : \Psi \times 2^E \rightarrow E$ is such that $\pi(\psi, C) = \sigma(\pi_{\mathrm{Perm}}(\psi), C)$, where $\sigma(\mu, C)$ is the first element from μ to be in C.*

The following lemma shows that permutation policies do not lose optimality.

Lemma 1. *For every query policy there is a permutation policy that implements a query policy that is at least as good.*

Proof. Let π^* be an optimal policy, which is not implementable by a permutation policy. This implies that there is a partial realization ψ, and two constraint sets C, C' with corresponding distinct optimal items e, e'. If $e \notin C'$, ordering e before e' gives a permutation policy that implements π^*. Similarly if $e' \notin C$, we can order it before e. If they are both in $C \bigcap C'$, since e is preferred by π^* when e' is also available in C, it results in at least as good a value as e'. Importantly, C or C' does not influence what occurs after e is chosen. Hence e is also optimal for C' and we can order e before e' for ψ. □

The following defines an adaptive submodular and monotonic function V_f over the permutation sets, which represents the utility of applying all permutation policies in the set to select items. Lemma 2 then shows V_f is adaptive monotonic and adaptive submodular.

Definition 9. *For all* $S_u \subseteq U$, $\sigma(S_u, \phi_c) \stackrel{\text{def}}{=} \bigcup_{\mu \in S_u} \sigma(\mu, \phi_c(\mu))$;
$V_f(S_u, \phi_o, \phi_c) \stackrel{\text{def}}{=} f(\sigma(S_u, \phi_c), \phi_o)$; *and* $V_{f,ave}(\pi, l) \stackrel{\text{def}}{=} f_{c,ave}(\pi, \{\}, l)$.

Lemma 2. *If function* f *is adaptive monotonic and adaptive submodular with respect to* P_o, *then so is the function* V_f *with respect to* P_o, P_c.

Proof. V_f is adaptive monotonic if $\Delta_{V_f}(\mu|\psi) \geq 0$. Let $\mathrm{dom}_U(\psi)$ and $\mathrm{dom}(\psi)$ represent respectively the set of permutations and the set of items queried in ψ. Let ψ be the composition of ψ_c and ψ_o, which represent the partial realizations over item sets and labels, respectively. We write $\psi = (\psi_c, \psi_o)$.

$$
\begin{aligned}
\Delta_{V_f}(\mu|\psi_c, \psi_o) &= E_{\phi_c \sim \psi_c} E_{\phi_o \sim \psi_o}[V_f(\mathrm{dom}_U(\psi) \cup \{\mu\}, \phi_o, \phi_c) - V_f(\mathrm{dom}_U(\psi), \phi_o, \phi_c)] \\
&= E_{\phi_c \sim \psi_c} E_{\phi_o \sim \psi_o}[f(\mathrm{dom}(\psi) + \sigma(\mu, \phi_c(\mu)), \phi_o) - f(\mathrm{dom}(\psi), \phi_o)] && (3.2) \\
&= E_{\phi_c \sim \psi_c}[\Delta_f(\sigma(\mu, \phi_c(\mu))|\psi_o)] && (3.3) \\
&= E_{\phi_c \sim \psi_c}[\Delta_f(\sigma(\mu, C)|\psi_o)|\phi_c(\mu) = C] && (3.4) \\
&= \sum_C Pr(\phi_c(\mu) = C|\psi_c)\Delta_f(\sigma(\mu, C)|\psi_o) && (3.5) \\
&= \sum_C P_c(C)\Delta_f(\sigma(\mu, C)|\psi_o) && (3.6) \\
&\geq 0 && (3.7)
\end{aligned}
$$

Equation 3.2 employs the definition of V_f. Equation 3.3 follows from the definition of the marginal utility Δ_f and Assumption 1. Equation 3.6 follows from Assumption 2 and Equation 3.7 from adaptive monotonicity of f. Let $\psi = (\psi_c, \psi_o) \leq \psi' = (\psi'_c, \psi'_o)$. To

show V_f is adaptive submodular we argue $\Delta_{V_f}(\mu|\psi) - \Delta_{V_f}(\mu|\psi') \geq 0$.

$$\begin{aligned}
\Delta_{V_f}(\mu|\psi) - \Delta_{V_f}(\mu|\psi') &= \sum_C P_c(C)\Delta_f(\sigma(\mu,c)|\psi_o) - \sum_C P_c(C)\Delta_f(\sigma(\mu,c)|\psi'_o) && (3.8)\\
&= \sum_C P_c(C)[(\Delta_f(\sigma(\mu,c)|\psi_o) - \Delta_f(\sigma(\mu,c)|\psi'_o)] && (3.9)\\
&\geq 0 && (3.10)
\end{aligned}$$

Equation 3.8 follows from Equation 3.6. Equation 3.10 follows from the adaptive submodularity of f since $\psi_o \leq \psi'_o$. □

A direct adaptation of Theorem 1 to the varying items setting requires us to find a greedy permutation policy with respect to Δ_{v_f}. However, the space of all permutations is too big to search even for a greedy policy. Fortunately that is not necessary. Rather than first finding the greedy permutation μ for ψ and then selecting an item from C using $\sigma(\mu, C)$, we can greedily select the item in C with the most marginal utility. Since, by definition, no item in C has more marginal utility than the greedy choice, we can assert the following.

Lemma 3. *The marginal utility of the item selected by the greedy permutation policy from C is never more than that of any greedy query policy that selects an item with the most marginal utility among the items in C.*

Proof. Suppose that the best permutation μ^* selected an item $e_i \in C$, where $\Delta_f(e_i|\psi_o) < \Delta_f(e_j|\psi_o)$ for some $e_j \in C$. From Equation 3.7, we have:

$$\max_\mu(\Delta_{V_f}(\mu|\psi)) = \max_\mu \sum_C P_c(C)\Delta_f(e_i|\psi_o). \tag{3.11}$$

Since $\Delta_f(e_i|\psi_o) < \Delta_f(e_j|\psi_o)$, swapping e_i with e_j would have yielded a higher objective than that of μ^*. This is a contradiction since μ^* is optimal. □

We now state and prove an approximation result for submodular optimization with varying item sets.

Theorem 2. *If f is adaptive monotonic and adaptive submodular, then for any $\alpha-$approximate greedy query policy π for $\alpha \geq 1$, optimal query policy π^*, and positive integers l, k, $V_{f,avg}(\pi, l) \geq (1 - e^{\frac{-l}{\alpha k}})V_{f,avg}(\pi^*, k)$.*

Proof. From Lemma 2, if f is adaptive monotonic and adaptive submodular, so is V_f. From Lemma 3 the marginal utility of π is at least as high as that of any query policy implemented by a greedy permulation policy. From Theorem 1, the result follows when π^* is the query policy implemented by the best permutation policy. From Lemma 1, permutation policies can implement optimal policies without any loss. Hence the result follows. □

3.2.3 Extension to Query Sets

We now extend the standard adaptive submodular optimization problem of Section 3.2.1 to a setting where multiple items are queried in each step. This is a natural setting for many sequential decision problems, where it would be preferred to adaptively choose a set of items in parallel rather than an individual item. This is a straightforward adaptation of batch mode active learning setting of [Chen and Krause, 2013], where a small batch of k unlabeled examples are selected at each step. All labels of the selected batch are received in parallel, followed by the next batch of examples. In the current work, we let the queries be chosen from an arbitrary subset Q of 2^E rather than batches of size k. We seek a near optimal policy for picking those sets. We introduce a new function f^* that extends f to subsets of Q. $f^* : 2^Q \times O^E \rightarrow \Re \geq 0$ is defined as

$$\forall S \subseteq Q, f^*(S, \phi_o) = f(\bigcup_{q \in S} q, \phi_o). \tag{3.12}$$

Now, we define the expected marginal of a query as the expected improvement in the value of f^*.

$$\Delta_{f^*}(q|\psi_o) = \mathbb{E}_{\phi_o \sim \psi_o}[f^*(S \cup \{q\}, \phi_o) - f^*(S, \phi_o)]$$

where $S = dom(\psi_o)$. We similarly extend the definition of f_{avg} of a policy to define f^*_{avg} of a l-horizon policy π as the expected value of f^* when choosing the queries according to

π with a query budget of l. The following lemma is a straightforward extension of Theorem 1 of [Chen and Krause, 2013] which was restricted to batch mode active learning.

Lemma 4. *If the function f is adaptive monotonic and adaptive submodular with respect to P_o, then so is the extended function f^*.*

Proof. Let $S = \mathrm{dom}(\psi_o)$, $q_i = \{e_1, \ldots, e_n\}$, $R = \bigcup_{q \in S} q$, and $\psi_o \leq \psi'_o$. Then, adaptive monotonicity follows from:

$$
\begin{aligned}
\Delta_{f^*}(q|\psi_o) &= \mathbb{E}_{\phi_o \sim \psi_o}[f^*(S \cup \{q\}, \phi_o) - f^*(S, \phi_o)] && (3.13)\\
&= \mathbb{E}_{\phi_o \sim \psi_o}[f(\bigcup_{q \in S} q \cup \{q_i\}, \phi_o) - f(\bigcup_{q \in S} q, \phi_o)] && (3.14)\\
&= \mathbb{E}_{\phi_o \sim \psi_o}[f(R \cup \{e_1, \ldots, e_n\}, \phi_o) - f(R, \phi_o)] && (3.15)\\
&= \mathbb{E}_{\phi_o \sim \psi_o}[f(R \cup \{e_1, \ldots, e_n\}, \phi_o) - f(R \cup \{e_2, \ldots, e_n\}, \phi_o)\\
&\quad + f(R \cup \{e_2, \ldots, e_n\}, \phi_o) - f(R \cup \{e_3, \ldots, e_n\}, \phi_o)\\
&\quad + \ldots + f(R \cup \{e_n\}, \phi_o) - f(R, \phi_o)] && (3.16)\\
&= \mathbb{E}_{\phi_o \sim \psi_o}[f(R \cup \{e_1, \ldots, e_n\}, \phi_o) - f(R \cup \{e_2, \ldots, e_n\}, \phi_o)]\\
&\quad + \mathbb{E}_{\phi_o \sim \psi_o}[f(R \cup \{e_2, \ldots e_n\}, \phi_o) - f(R \cup \{e_3, \ldots, e_n\}, \phi_o)] && (3.17)\\
&\quad + \ldots + \mathbb{E}_{\phi_o \sim \psi_o}[f(R \cup \{e_n\}, \psi_o) - f(R.\phi_o)] && (3.18)\\
&= \Delta_f(e_1|\psi_o \cup R \cup \{e_2, \ldots, e_n\}) + \Delta_f(e_2|\psi_o \cup R \cup \{e_3, \ldots, e_n\})\\
&\quad + \ldots + \Delta_f(e_n|\psi_o)\\
&\geq 0 && (3.19)
\end{aligned}
$$

Equations 3.16, 3.18 and 3.19 follow from telescoping, the additivity of expectations, and the adaptive monotonicity of f respectively.

Adaptive submodularity is shown below, where S= dom (ψ), S' = dom(ψ'), $R = \bigcup_{q \in S} q$, and $R' = \bigcup_{q \in S'} q$.

$$
\begin{aligned}
\Delta_{f^*}(q_i|\psi) - \Delta_{f^*}(q_i|\psi') &= \mathbb{E}_{\phi\sim\psi}[f^*(S\cup\{q_i\},\phi) - f^*(S,\phi)] - \\
&\quad \mathbb{E}_{\phi'\sim\psi'}[f^*(S'\cup\{q_i\},\phi') - f^*(S',\phi')] && (3.20)\\
&= \mathbb{E}_{\phi\sim\psi}[f(\bigcup_{q\in S} q\cup\{e_1,\ldots,e_n\},\phi) - f(\bigcup_{q\in S} q,\phi)] \\
&\quad -\mathbb{E}_{\phi'\sim\psi'}[f(\bigcup_{q\in S'} q\cup\{e_1,\ldots,e_n\},\phi') - f(\bigcup_{q\in S'} q,\phi')] && (3.21)\\
&= \mathbb{E}_{\phi\sim\psi}[f(R\cup\{e_1,\ldots,e_n\},\phi) - f(R,\phi)] \\
&\quad -\mathbb{E}_{\phi'\sim\psi'}[f(R'\cup\{e_1,\ldots,e_n\},\phi') - f(R',\phi')] && (3.22)\\
&= \mathbb{E}_{\phi\sim\psi}[f(R\cup\{e_1,\ldots,e_n\},\phi) - f(R\cup\{e_2,\ldots e_n\},\phi) \\
&\quad +f(R\cup\{e_2,\ldots e_n\},\phi) - f(R\cup\{e_3,\ldots e_n\},\phi) + \ldots + \\
&\quad +f(R\cup\{e_n\},\phi) - f(R,\phi)] \\
&\quad -\mathbb{E}_{\phi'\sim\psi'}[f(R'\cup\{e_1,\ldots,e_n\},\phi') - f(R'\cup\{e_2,\ldots e_n\},\phi') \\
&\quad +f(R'\cup\{e_2,\ldots e_n\},\phi') - f(R'\cup\{e_3,\ldots e_n\},\phi') + \ldots + \\
&\quad +f(R'\cup\{e_n\},\phi') - f(R',\phi')] && (3.23)\\
&= \mathbb{E}_{\phi\sim\psi}[f(R\cup\{e_1,\ldots,e_n\},\phi) - f(R\cup\{e_2,\ldots e_n\},\phi)] \\
&\quad -\mathbb{E}_{\phi'\sim\psi'}[f(R'\cup\{e_1,\ldots,e_n\},\phi') - f(R'\cup\{e_2,\ldots e_n\},\phi')] \\
&\quad +\mathbb{E}_{\phi\sim\psi}[f(R\cup\{e_2,\ldots e_n\},\phi) - f(R\cup\{e_3,\ldots e_n\},\phi)] \\
&\quad -\mathbb{E}_{\phi'\sim\psi'}[f(R'\cup\{e_2,\ldots,e_n\},\phi') - f(R'\cup\{e_3,\ldots e_n\},\phi')] \\
&\quad +\ldots+ \\
&\quad +\mathbb{E}_{\phi\sim\psi}[f(R\cup\{e_n\},\phi) - f(R,\phi)] \\
&\quad -\mathbb{E}_{\phi'\sim\psi'}[f(R'\cup\{e_n\},\phi') - f(R',\phi')] && (3.24)\\
&= \Delta_f(e_1|R\cup\{e_2,\ldots,e_n\}|\psi) - \Delta_f(e_1|R'\cup\{e_2,\ldots,e_n\}|\psi') \\
&\quad +\Delta_f(e_2|R\cup\{e_3,\ldots,e_n\}|\psi) - \Delta_f(e_2|R'\cup\{e_3,\ldots,e_n\}|\psi') \\
&\quad +\ldots+ \\
&\quad +\Delta_f(e_n|\psi) - \Delta_f(e_n|\psi') && (3.25)\\
&\geq 0 && (3.26)
\end{aligned}
$$

Equation 3.21 expands the definition of f^*, Equation 3.22 uses the definitions of R and R', and Equations 3.23 follows from telescoping the terms. Equation 3.24 rearranges the terms on the right hand side. Equations 3.25 follows from the definition of Δ_f and Equation 3.26 from the submodularity of f. □

Lemma 4 and Theorem 1 imply the following bound.

Theorem 3. *If f is adaptive monotonic and adaptive submodular, then for any $\alpha-$approximate greedy query set policy π for $\alpha \geq 1$, optimal query set policy π^*, and positive integers l, k, $f^*_{avg}(\pi, l) \geq (1 - e^{\frac{-l}{\alpha k}}) f^*_{avg}(\pi^*, k)$.*

The implication of the above theorem is that the greedy algorithm with respect to the marginal utility of query sets is approximately optimal.

3.2.4 Extension to Varying Query Sets

We now combine the extensions of Sections 3.2.2 and 3.2.3 into a new setting of Adaptive Submodularity with Varying Query Sets. This combines the idea of randomly constraining the available queries with the idea of simultaneously querying multiple items. Thus, the learner is now constrained to choose from an exogenously picked random subset of query sets $C \subseteq Q \subseteq 2^E$.

Given the base utility function f, we define a new utility function V_{f^*} as the expected utility of a permutation policy over query sets, which are constrained randomly according to the distribution P_c.

Lemma 5. *If function f is adaptive monotonic and adaptive submodular with respect to P_o, then so is the function V_{f^*} with respect to P_o, P_c.*

Proof. From Lemma 4, since f is adaptive monotonic and adaptive submodular, so is f^*. From Lemma 2, since f^* is adaptive monotonic and adaptive submodular, so is V_{f^*}. □

We now state our main theorem which gives an approximation bound on the performance of a greedy policy over query sets.

Theorem 4. *If f is adaptive monotonic and adaptive submodular, then for any $\alpha-$approximate greedy query policy π for $\alpha \geq 1$, optimal query policy π^*, and positive integers l, k, $V_{f^*,avg}(\pi, l) \geq (1 - e^{\frac{-l}{\alpha k}})V_{f^*,avg}(\pi^*, k)$.*

Proof. The proof follows directly by composing Theorem 2 and Theorem 3. □

An important thing to note is that although the greedy heuristic does not assume the knowledge of the constraint distribution P_c, it is competitive with the optimal query policy π^* that *does have* this knowledge. While this might appear counter-intuitive, the reason that it works is that P_c is a stationary distribution where the constraints are i.i.d. Thus, the optimal policy cannot afford to sacrifice current gain in selecting a good example in the hope that it can plan for it in the future. The diminishing returns property of the utility function implies that there is a price to pay in delaying the reward, which plays a central role in the proof of Theorem 1 and is inherited by the other theorems.

We close this section by describing a generic greedy algorithm for adaptive sub-modular optimization with varying query sets. The pseudo-code is presented in Algorithm 2. The input of the algorithm is a set of all possible queries Q, adaptive submodular function f, and budget l. It initializes the probabilistic model with a prior. It then repeats for l iterations, where in each iteration, it first observes the set of available queries C_t at time t (Step 4). The algorithm uses an evaluation function that estimates the incremental value of the query defined by the function Δ_f (Step 5). The query results in a set of observations which are added to the partial realization (Step 7) and they are used to update the model in Step 8, and the cycle repeats.

3.3 Application: Active Multi-Label Learning with Varying Experts

Here we discuss the active multi-label learning application that motivates our research. We have a set of unlabeled instances X, where each instance can have multiple labels form the set of possible labels Y. We have a set of experts each of whom can only annotate a subset of labels for instances. Moreover, we have a set of all possible queries $Q \subseteq X \times 2^Y$ and a set of available queries $C_t \subseteq Q$ at a given time step. Each query corresponds to an

Algorithm 2 Adaptive Submodular Optimization with Varying Query Sets

1: **Input:** Query set Q ; function f; query budget l.
2: Initialize the model, and set $\psi \leftarrow \emptyset$
3: **for** $t = 1$ to l **do**
4: Let C_t be the set of available queries
5: $q^* = \underset{q \in C}{argmax} \left[\Delta_f(q|\psi)\right]$
6: $\Phi(q*)$ = Observe the states of the set of items of q^*
7: $\psi \leftarrow \psi \cup \{\langle q^*, \Phi(q*)\}$
8: Update the model given ψ
9: **end for**

example-expert pair, where an expert is identified with a subset of the labels, $\mathbf{y} \subseteq Y$. Each query $(x, \mathbf{y}) \in C_t$ results in determining which of the labels in $\mathbf{y}$ are present for x. The goal of multi-label learning algorithm is to adaptively select a sequence of l queries that optimizes some objective that measures the information obtained from the experts.

We now describe the greedy algorithm for active multi-label learning with varying queries. The inputs to the algorithm are a set of unlabeled examples and a set of experts, each of whom is indicated by a subset of the labels. In this algorithm we need a model to compute the joint posterior probability distribution of the labels which is intimately involved in computing the marginal utility function. In our experiments we assumed that the labels are independent and used logistic regression to model the conditional posterior distributions of each label given the instance.

The algorithm initializes m logistic regression weight vectors, i.e., one W_y for each label $y \in Y$ (line 2 of Algorithm 3). These weights are used for computing posterior probability of each classifier:

$$P_\psi(O_{x,y}) = 1/(1 + e^{-W_y^T f(x)}) \tag{3.27}$$

Where $f(x)$ is the feature vector of x, and $O_{x,y}$ is the observation of label y for instance x. Logistic regression algorithms, typically learn a weight vector w by optimizing the log-likelihood of the training data, which is a convex optimization problem that can typically be solved quite effectively via gradient methods.

At each time step, the available example-expert pairs C_t is given (line 4). The algorithm computes the marginal benefit of an example-expert pairs $(x, \mathbf{y}) \in C_t$ and picks a pair $(x^*, \mathbf{y}^*)$, whose marginal benefit is the highest (line 5). We applied maximum *Gibbs error* criterion, explained in Section 3.3.2, to implement the greedy policy. Next, the algorithm asks the expert to annotate the vector of the subset of labels $\mathbf{y}^*$ for x, which is denoted by $\mathbf{O}_{x^*,\mathbf{y}^*}$ (line 6). Finally, the algorithm updates its partial realization ψ (line 7) and the set of classifiers based on new data (line 8).

Algorithm 3 Active Multi-Label Learning with Varying Experts

1: **Input:** unlabeled data X; utility function f
2: Initialize the logistic regression weight vector W_y for each label y, and set $\psi \leftarrow \emptyset$
3: **for** $t = 1$ to l **do**
4: Let C_t be the set of available example-query set pairs
5: $\langle x^*, \mathbf{y}^* \rangle = \underset{\langle x,\mathbf{y}\rangle \in C_t}{argmax} \left[\Delta_f(\mathbf{O}_{x,\mathbf{y}}|\psi)\right]$
6: $\mathbf{O}_{x^*,\mathbf{y}^*}$ = The observations for $\langle x^*, \mathbf{y}^* \rangle$
7: $\psi \leftarrow \psi \cup \{\langle x^*, \mathbf{O}_{x^*,\mathbf{y}^*}\rangle\}$
8: Update the weights W_y for each label y given ψ
9: **end for**

The following two sections analyze two popular greedy heuristics for active learning, namely, the maximum entropy criterion and the maximum Gibbs error criterion. Although these two criteria are quite different in general, and only the maximum Gibbs error criterion is provably near-optimal, they are equivalent for binary label classification and behave identically [Cuong et al., 2013, 2014].

3.3.1 Maximum Entropy Criterion

The maximum entropy criterion selects the next example whose posterior label distribution has the maximum Shannon entropy [Dagan and Engelson, 1995]. Although it is a popular heuristic, as we show below, it does not satisfy adaptive submodularity. Our counterexample is based on Cuong's thesis [Cuong, 2015] [Cuong et al., 2014] where it is shown that the maximum entropy heuristic does not yield a multiplicative approximation bound.

Example. Consider that there are $n+3$ items including three special items one *id* and two *decoys*. The other items are labeled $X_1, \ldots, X_n$, one of which is an unknown *target*. All items $X_1, \ldots, X_n$ except the target have the same label 0. The target has a label of $\log m$ bits. The label of *id* has $\log n$ bits and represents the index of the target item. The labels of the two decoy items are $\log(n+1)$ bits each and are independent of each other and every other label. We denote partial realizations as sets of item-label pairs, with $\{\}$ representing the initial empty partial realization.

$$\begin{aligned}
\Delta_f(id|\{\}) &= H(id) = \log n && (3.28)\\
\Delta_f(decoy|\{\}) &= H(decoy) = \log(n+1) && (3.29)\\
\Delta_f(X_p|\{\}) &= H(X_p|\{\}) = \frac{1}{n}\log mn + \frac{n-1}{n}\log\frac{n}{n-1} && (3.30)\\
&= \frac{1}{n}\log m + \log n - \frac{n-1}{n}\log(n-1) && (3.31)\\
\Delta_f(X_p|\{(id,q)\}, p \neq q) &= 0 && (3.32)\\
\Delta_f(X_p|\{(id,p)\}) &= H(X_p|\{(id,p)\}) = \log m && (3.33)
\end{aligned}$$

Equation 3.31 is the entropy of the items X_p. Equation 3.32 follows because the non-target items do not have any information. Finally Equation 3.33 is true because there are $\log m$ bits revealed from the target. We note that our definitions of adaptive monotonicity is satisfied for all items. To satisfy adaptive submodularity, we need:

$$\begin{aligned}
\Delta_f(X_p|\{\}) \geq \Delta_f(X_p|\{(id,p)\}) &\Leftrightarrow \frac{1}{n}\log m + \log n - \frac{n-1}{n}\log(n-1) \geq \log m && (3.34)\\
&\Leftrightarrow \frac{n}{n-1}\log n - \log(n-1) \geq \log m && (3.35)
\end{aligned}$$

As n approaches ∞ the left hand side of this equation tends to 0, and fails to satisfy the above equation when $m \geq 2$.

3.3.2 Maximum Gibbs Error Criterion

In pool-based active learning we are given a set of unlabeled data points E and a set of hypotheses H, and the goal is to select points to query until we can output a hypothesis.

Version space is the set of hypotheses that are consistent with the observed labeled training data. In [Cuong et al., 2014] and [Golovin and Krause, 2011], it was proven that version space reduction function is adaptive submodular.

In [Cuong et al., 2014] it was shown that maximum Gibss error Criterion maxGEC is equivalent to maximizing the version space reduction objective, which is known to be adaptive monotone submodular. This criterion selects the next query q^* whose posterior label distribution has the maximum Gibbs error or minimal negative Gibbs error:

$$q^* = \operatorname*{argmax}_q [1 - \sum_{o \in O} p_\psi(y = o|q)^2] = \operatorname*{argmin}_q \sum_{o \in O} p_\psi(y = o|q)^2 \tag{3.36}$$

Gibbs error GE is the expected error of the Gibbs classifier, which samples a label from the posterior label distribution, p_ψ, and uses it for prediction.

$$GE(y) = \sum_{o \in O} p_\psi(y = o)(1 - p_\psi(y = o)) = 1 - \sum_{o \in O} p_\psi(y = o)^2 \tag{3.37}$$

Where $o \in O$ is the set of all possible values (states) for label y, and p_ψ is the current posterior obtained after observing the labeled examples ψ.

This greedily maximizing the Gibbs error of the selected examples corresponds to the algorithm that greedily maximizes the expected version space reduction.

Policy Gibbs error is the expected error rate of a Gibbs classifier on the set adaptively selected by the policy. In [Cuong et al., 2014], it is shown that the policy Gibbs error corresponds to the expected reduction in the volume of the version space . Since the expected version space reduction is adaptive monotonic and adaptive submodular [Golovin and Krause, 2011], choosing an item that maximzies the reduction in the expected policy Gibbs error would lead to a near-optimal policy.

In our active multi-label learning algorithm each query, q, consist of a set of k labels, $y_1, \ldots, y_k$. Thus, we compute the maximum Gibbs error with respect to the joint posterior

distribution of a vector of labels in the query. So we have:

$$q^* = \underset{q}{\operatorname{argmax}}[1 - \sum_{o_1,...,o_k \in O} p_\psi(y_1 = o_1, ..., y_k = o_k|q))^2] \tag{3.38}$$

While the above computation requires summing exponentially many terms in general,it simplifies to a polynomial-time computation when the different predictors are independent as in our multi-label learning experiments. The joint distribution $(p_\psi[y_1(q) = o_1, ..., y_k(q) = o_k; q])$ can be written as a product of each probability.

$$\sum_{o_{1,..,k} \in O} (p_\psi[y_1(q) = o_1, ..., y_k(q) = o_k; q])^2 = \prod_{i=1}^{k} \sum_{o_i \in O} (p_\psi[y_i(q) = o_i; q]$$

Where p_ψ is the current posterior obtained after observing the labeled examples ψ. Hence, in this case, the next example q^* can be computed efficiently as:

$$q^* = \underset{q}{\operatorname{argmax}}[1 - \prod_{i=1}^{k} \sum_{o_i \in O} (p_\psi[y_i(q) = o_i; q])^2]$$

3.4 Related Work

The research in this area can be divided into the following categories based on the query type used.

a) Example-based queries in which the whole label vector of a selected example is asked to be annotated. A drawback of this type of query selection is that annotating of all labels of an example is expensive and time consuming for experts.

A branch of research in this category focus on the strategies to train a binary SVM classifier for each label and combine the SVM margins using different heuristics to select the training set from a pool of available data for annotation. For examples, Singh et al. [2009] use another uncertainty sampling method in which the average of the uncertainty from all SVM binary classifiers as the instance selection measure. Li et al. [2004] selected examples using Max Loss (ML) and Mean Max Loss (MML) strategies to count prediction losses of all binary classifiers. MML selects the unlabeled instance which had the maximum mean loss

value over the predicted classes. Li and Guo [2013a] proposed two novel multi-label active learning strategies, a Max-Margin prediction Uncertainty (MMU) strategy and a Label Cardinality Inconsistency (LCI) strategy, in which the relative multi-label classification margin structure on each unlabeled instance and the statistical label cardinality information are exploited, respectively. Brinker [2005] selected the instance that minimizes the smallest SVM margin among all binary classifications. Yang et al. [2009] presented a method called Maximum loss reduction with Maximal Confidence (MMC), where sample selection is based on the sum of losses from SVM classifiers on all labels. The state-of-the-art method Li and Guo [2013b] uses a combination of deviation from mean label cardinality and an SVM-based loss function for active learning. To the best of our knowledge, all these works are myopic and do not provide any theoretical guarantees on the optimality of the selected set.

On the other hand, another branch of the research in this category focus on the optimality of the selected set. In these schemes, each point is selected from a pool of available unlabeled data, so as to maximize the information gain over the remaining unlabeled data [Vasisht et al., 2014].

Finally, there is a branch of work in this category that focus on selecting the unlabeled data which can lead to the largest reduction of the expected model loss. For example in [Tong and Koller, 2002] the model loss is approximated by the size of version space, and the reduction rate of the size of version space is optimized with Support Vector Machines (SVM).

b) Example-label pair Query, in which the partial labels of a selected example is queried to be labeled [Qi et al., 2009, Yang et al., 2009, Huang et al., 2014, Huang and Zhou, 2013, Ye et al., 2015]. In [Qi et al., 2009], a two-dimensional active learning (2DAL) based on MML [Li et al., 2004] was introduced. 2DAL considered both relationships between samples and between labels. Sample-label pairs were selected in order to minimize the multi-label Bayesian error bound. Zhang et al. [2012] proposed a multi-label batch active learning (MLBAL) approach that allowed the learning algorithm to actively select a batch of informative example-label pairs from which it learns in each learning iteration, so as to learn accurate classifiers with less annotation efforts. Huang and Zhou [2013] select instance-label pairs based on uncertainty and diversity in the instance space as well as the

label space, and actively query the instance-label pairs which can improve the classification model most. Huang et al. [2014] select instance-label pairs by simultaneously considering informativeness and representativeness. Wu et al. [2014] propose a method to select the most uncertain example-label pairs to decrease the labeling cost. They neglect the inherent label correlations across all labels of the example.

c) Relation between labels: Huang et al. [2015], propose a new query type that queries the relevance ordering of labels pairs, which on the one hand reduces the requirement of annotators' expertise, and on the other hand, provides more useful information to improve the classification model.

d) Example-Expert-label query: Recently, Li et al. [2015] have proposed a new multi-label active learning under the crowd-sourcing setting, where during active query process multiple low cost imperfect annotators with various expertise are available for labeling. This paper is the most relevant paper to our work. However, in our setting we assume that all the experts are not always available and that the experts' annotation is correct.

3.5 Empirical Evaluation

We now describe an experimental evaluation of our algorithm in the context of multi-label learning. We simulate the crowd-sourcing scenario using synthetic experts where each expert is assigned a subset of the labels of equal size. We compare 3 versions of our algorithm: selecting an example and an expert according to Gibbs error criterion, selecting the example using Gibbs error criterion and the expert randomly among those available, and selecting both the example and the expert randomly. We evaluate the methods on six benchmark datasets (see Table 3.1), selected based on the diversity of domains and their popularity within the multi-label learning community [1].

3.5.1 Experimental Setup

We assume that all examples are always available, but only a subset of experts are available at a time. Each expert is identified with a subset of labels. Thus each query, i.e., example-

[1] `http://mulan.sourceforge.net/datasets.html`

Table 3.1: Characteristics of the datasets.

Data set	Domain	Instance	Features	Labels
Emotions	music	593	72	6
Scene	image	2407	294	6
Flags	image	194	19	7
Yeast	biology	2417	103	14
Mediamill	video	43907	120	101
CAL500	music	502	68	174

expert pair, consists of an example and a set of labels that the expert can label. The number of labels L for each domain represents the importance of expert selection and is a relevant parameter. For each experiment the sizes of the label sets of all experts are equal. We call this the *query size* s, which is another important control parameter. To represent the expertise of different experts, we repeatedly generate random label sets of a given query size and add them to the pool until all labels are covered, rejecting a label set if it has been previously generated. We sample the labels with replacement, so the label sets for experts can overlap.

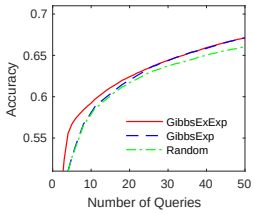
We compare the following 3 schemes for query selection.

1) *Random:* Randomly selects an available example-expert pair, such that there is at least one new label that has not been previously queried on that example.

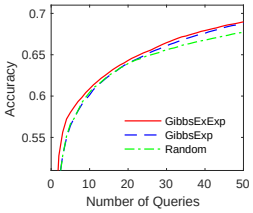
2) *GibbsEx:* Randomly selects an available expert and then chooses an example that has the maximum Gibbs error for that expert.

3) *GibbsExExp:* Selects an example-expert pair which has the maximum Gibbs error among all available pairs.

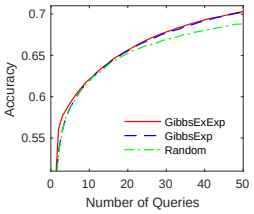
We evaluate the algorithms by their *accuracy* on the test data. We normalized the datasets to make the values of each feature have zero-mean and unit-variance. We used the binary-class logistic regression classifiers with $L2$ regularization and tuned the regularization parameter α via 5-fold cross validation. We computed the averaged result over 10 runs of 5-fold cross validation.



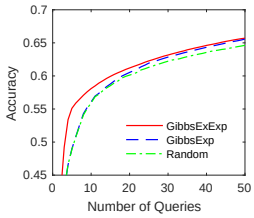
(a) Emotions, $s = 2$, $L = 6$



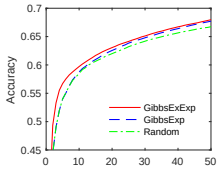
(b) Emotions, $s = 3$, $L = 6$



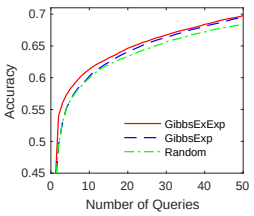
(c) Emotions, $s = 4$, $L = 6$



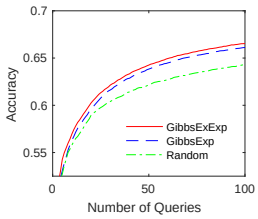
(d) Scene, $s = 2$, $L = 6$



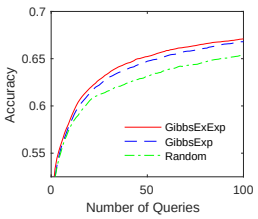
(e) Scene, $s = 3$, $L = 6$



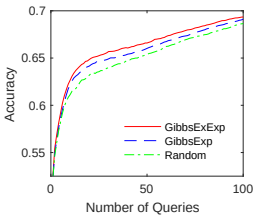
(f) Scene, $s = 4$, $L = 6$



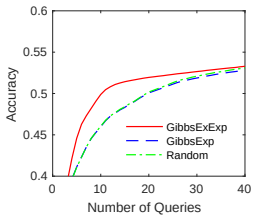
(g) Flags, $s = 2$, $L = 7$



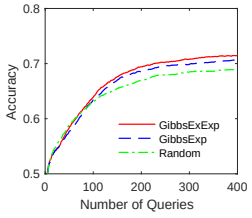
(h) Flags, $s = 3$, $L = 7$



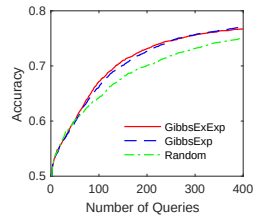
(i) Flags, $s = 6$, $L = 7$



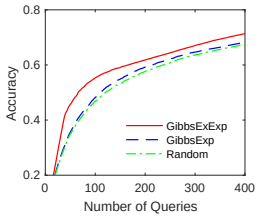
(j) Yeast, $s = 2$, $L = 14$



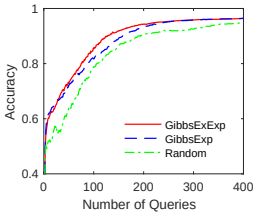
(k) Yeast, $s = 7$, $L = 14$



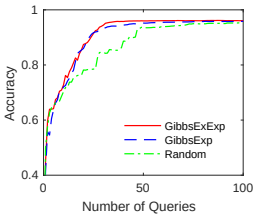
(l) Yeast, $s = 11$, $L = 14$



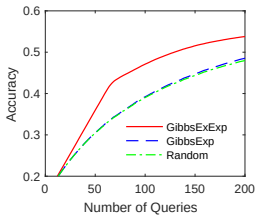
(m) Mediamill, $s = 2$, $L = 101$



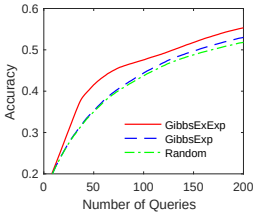
(n) Mediamill, $s = 50$, $L = 101$



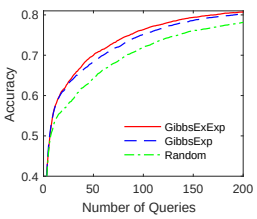
(o) Mediamill, $s = 80$, $L = 101$



(p) CAL500, $s = 2$, $L = 174$



(q) CAL500, $s = 3$, $L = 174$



(r) CAL500, $s = 50$, $L = 174$

Figure 3.1: The average results over 10 runs on six datasets with query size s and label set size L.

3.5.2 Experimental Results

Figure 3.1 shows the results for all the domains for different query sizes. Each row of plots represents a different domain. The number of labels increases as we go down from top to bottom. For each domain, the query size increases as we go from left to right. Each plot shows the accuracy on the test data as a function of the number of training queries.

We can make a number of observations from these results. First, as one would expect, increasing the query size (i.e. compare Figure 3.1(m) against Figure 3.1(o)), would increase the accuracies of all the algorithms as the learners get more information from each query. Second, the differences between *GibbsExExp* and the other methods are more prominent as the number of labels L is increased, i.e., as we go down from top to bottom in the plots. This is especially true when the query size s is small because many experts are typically needed to label each example in these cases and it becomes important to select experts smartly. Third, when query size is small, *GibbsExExp* performs comparably and often better than *GibbsEx* and *Random*. In experiments with small query sizes, more experts are needed to label the data. Hence, *GibbsExExp* which has a better heuristic in selecting experts, has a higher accuracy than *Random* and *GibbsEx*, which select experts randomly. By increasing the query size the gap between *GibbsExExp* and *GibbsEx* reduces, while the gap between *GibbsEx* and *Random* increases. With increased query size, expert selection becomes less important than example selection because each expert can now label many labels. This behavior can be observed in all datasets, especially in datasets with large label sizes (e.g. *Mediamil* and *CAL500*).

3.6 Conclusions

In this work we extended the framework of adpative submodular optimization to a setting where the available queries are randomly constrained and gave a simple greedy algorithm with a near-optimal performance bound. We applied the new framework to the problem of multi-label learning and showed promising results based on the Gibbs error heuristic.

It would be interesting to consider further generalizations of our framework and other potential applications. One new problem that arises in the crowd sourcing setting is to simultaneously learn the expertise of different workers. Another potential extension is to

explore the situation where the cost of labeling for each query is different, e.g., due to the differences in the sizes of their label sets. Another research direction is to study robustness against noisy labels. Finally, for the multi-label classification application, it would be important to relax the assumption of label independence.

Chapter 4: Interactive Naming for Explaining Deep Neural Networks

In this chapter, we consider the problem of explaining the decisions of deep neural networks for image recognition in terms of human-recognizable visual concepts. In particular, given a test set of images, we aim to explain each classification in terms of a small number of image regions, or activation maps, which have been associated with semantic concepts by a human annotator. Since it is often very difficult to directly interpret the highly distributed concepts in large networks, we focus on explanatory neural networks proposed in [Qi et al., 2017], which transform large distributed representations into a more explainable form without losing accuracy. This allows for generating summary views of the typical reasons for classifications, which can help build trust in a classifier and/or identify example types for which the classifier may not be trusted. The classification decision on each image is explained by a small number of image regions, or significant activation maps, which are minimally sufficient to make the decision, and are clustered into meaningful concepts by human annotators. For this purpose, we developed a user interface for "interactive naming," which allows a human annotator to manually cluster significant activation maps in a test set into meaningful groups called "visual concepts".

The main contribution of this work is a systematic study of the visual concepts created by the human annotators using the interactive user interface in terms of their adequacy for explaining the test images, correspondence between the visual concepts and the individual neurons, the inter-annotator agreement of visual concepts, and the learnability of visual concepts. We performed two user studies of visual concepts produced by human annotators. We found that a large fraction of the activation maps have recognizable visual concepts, and that there is significant agreement between the different annotators about their denotations. Many of the visual concepts created by the annotators could be generalized reliably from a modest number of examples.

4.1 Introduction

Deep Neural Networks (DNNs) are powerful learning models that achieve excellent performance on many problems ranging from object recognition to machine translation. However, the potential utility of DNNs is limited by the lack of human interpretability of their decisions, which can lead to a lack of trust. The goal of this paper is to study an approach, called *interactive naming*, for improving our understanding of the decision-making process of DNNs. In particular, this approach allows a human annotator to visualize and organize activation maps of critical neurons into meaningful visual concepts, which can then be used to explain decisions made over the test data (Figure 4.1).

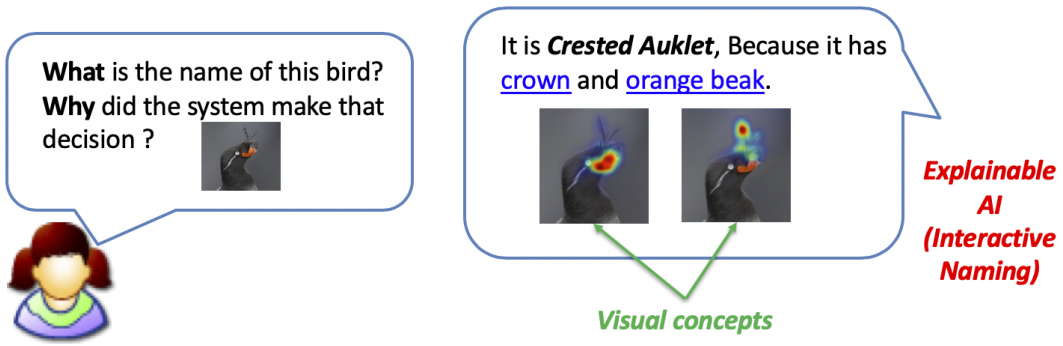


Figure 4.1: Overview of our approach.

Interpreting the roles and functions of individual and groups of neurons in DNNs in order to explain their decision making has been a long-standing problem in AI [Hinton et al., 1986]. Much recent work on interpretibility has focused on visualizing activation maps which highlight parts of the input that are most important to the final decision of the DNN or the output of an individual neuron. These include backpropagation-based methods, e.g., [Zeiler and Fergus, 2014, Simonyan et al., 2014, Springenberg et al., 2015, Shrikumar et al., 2016, Sundararajan et al., 2017, Bach et al., 2015, Shrikumar et al., 2016, Zhang et al., 2016, Selvaraju et al., 2017], which compute the activations for all input features in a single forward and backward pass through the network; and perturbation-based methods, e.g., [Zeiler and Fergus, 2014, Dabkowski and Gal, 2017, Fong and Vedaldi, 2017], which

perturb parts of the input to see which ones are most important to preserve the final decision. While activation maps can be a useful tool for analyzing DNNs, they do not necessarily identify the human recognizable concepts embedded in the network, nor do they provide an understanding for how those concepts combine to form the final decisions.

In this work, we make progress toward this goal by building an interface for interactive naming, and conducting a formative study on a set of non-trivial image classification tasks. In particular, our approach is based on the idea that the final decision of a DNN is dominated by the most highly-weighted neuron activations (the *significant activations*) in the penultimate network layer. Explanations of the decisions can thus be formed by 1) identifying the significant activations for each decision, and 2) attaching meaningful concepts to the significant activations. Since DNNs typically have thousands of units in the penultimate layer, (1) can result in an overwhelming number of activations. To address this issue we draw on recent work that augments the original DNN with a learned *explanation Neural Network* (*xNN*), which mimics the predictions of the DNN using a much smaller penultimate layer of "X-features". Since the xNN is effectively equivalent to the original DNN, we can use it to make predictions on test instances with no loss in accuracy, but with a dramatic reduction in the number of significant activations to be considered for explanations.

To deal with (2), our interface displays the (significant) activation maps of X-features for decisions made on a test set and allows a human annotator to cluster the activations into meaningful groups called "visual concepts." Even though there are a small number of significant activations that sufficiently explain the final decisions, there may not be a one-to-one correspondence between them and human-recognizable visual concepts. Indeed, unlike in the standard supervised learning setting, where the number of classes/concepts is typically fixed beforehand, the number of visual concepts covered by the set of all significant activations is unknown. To make matters more interesting, the set of visual concepts might be different for different annotators. Finally, the annotators may not be able to label a map in isolation, and might need to see multiple images and find similarities and differences before labeling them. Indeed, this last problem has been studied under the name of "structured labeling," in the context of active learning and provides an inspiration for our work [Kulesza et al., 2014].

Drawing from the lessons of the above work, our interface provides maximum flexibility to the human annotators by presenting them with the activation maps of all X-features of all test images that belong to each category. Unlike the previous work on supervised and active learning which seek labels from a fixed label set, the annotators are asked to cluster the maps in a way that makes most sense to them and give them meaningful names. The subjects can create new clusters, move images across clusters, and merge clusters. They are also allowed to leave some maps not clustered, and discard others that they consider noisy.

The result of interactive naming is a set of explanations of test set predictions in terms of visual concepts. This enables summarizing the types of predictions that are made to gain confidence in the predictor and/or identify potential flaws in the predictor. Importantly, this type of summary is dependent on the human annotator, which raises interesting questions about differences in explanations that might result from different annotators. Specifically, we seek answers to the following research questions (RQs) through our study:

RQ1 (Coverage of Interactive Naming): What fraction of the examples are explainable using human recognizable visual concepts? If a significant fraction of the examples are not explainable via visual concepts, it might mean that the X-features are not properly aligned with human concepts and will have to be retrained from human data.

RQ2 (Correspondence Between X-features and Visual Concepts): Is there strong correspondence between visual concepts and individual X-features? In the absence of strong correspondence, we might explain the final decision of the network directly in terms of visual concepts rather than in terms of the X-features which are in turn interpreted by visual concepts.

RQ3 (Inter-annotator Agreement): How much overlap exists between the annotated sets of activations between different subjects? How much do the clusters of different subjects overlap? Existence of significant overlaps might suggest that we can move toward building a standardized ontology of visual concepts for explanations. Lack of significant agreement might mean that we will have to personalize explanations to different annotators.

RQ4 (Visual Concept Generalization): Is it possible to reliably classify attention maps into visual concepts with a small number of labeled examples? If this is possible, we could signifcantly reduce the labeling burden on the annotators by interactively generalizing from examples and by using active learning methods to intelligently select examples to be

labeled.

We explore the above questions through empirical experiments and two user studies. In the first user study, we generated the significant activations using 90% impact in classification method, we created the visualization using Excitation Backprop (ExcitationBP) [Zhang et al., 2016], and we conducted our experimental result on 12 categories of Caltech-UCSD Birds dataset [Wah et al., 2011]. We had the activation maps of the different images annotated by 5 different subjects using the annotation interface. The activation maps were separated by the class, but not by the X-feature. The second user study is similar to the first user study with the following difference: we generated the significant activations using the Minimal Sufficient Explanation (MSX) method [Erwig et al., 2018][Khan et al., 2009], we created the visualization using Integrated Gradient in Quadratic (I-GOS) [Qi et al., 2019] algorithm, and we conducted our experimental result on both 12 categories of Birds dataset and 5 categories of CEL dataset [Eddy et al., 2018]. We had the activation maps of different images of 12 categories of Birds dataset, annotated by 10 different subjects for using the annotation interface. Also, we had the activation maps of different images of 5 categories of CEL dataset, annotated by 5 different subjects using the annotation interface

In these user studies, the annotators are asked to name the activation maps of different bird images with human-recognizable visual concepts with no attempt to recognize their species. The studies reveal that a significant fraction of the images are human recognizable with some individual differences among different annotators.

4.2 Related Work

A lot of the current work on explaining the decisions of deep neural networks is focused on visualizing the activation maps of the neurons computed from various methods [Zeiler and Fergus, 2014, Simonyan et al., 2014, Springenberg et al., 2015, Shrikumar et al., 2016, Sundararajan et al., 2017, Bach et al., 2015, Shrikumar et al., 2016, Zhang et al., 2016, Selvaraju et al., 2017]. Two recently proposed explanation techniques, PatternNet and PatternAttribution, produce better visualizations for linear and nonlinear models than the earlier Gradient, DeConvNet and GuidedBackprop methods [Kindermans et al., 2017]. In contrast, a new perturbation-based approach identifies relevant patches in the input image by

sampling the image through perturbations, identifies the important neurons, and then passes them through a deconvolutional neural network [Lengerich et al., 2017]. Unfortunately, most of these activation map-based methods only provide qualitative results as they do not try to associate activations with human-recognizable concepts.

Another class of approaches generates visual explanations using natural language justifications, e.g. see [Hendricks et al., 2016]. However, sentence-based explanation approaches require an extra corpus of sentence descriptions in the training process. A different line of work, e.g. [Ribeiro et al., 2016], attempts to extract rule-based explanations of DNN decisions. This is done via analyzing the impact of perturbations to the input and extracting a rule that "locally explains" the network decision for a given input. Lundberg and Lee [2017] identify the class of feature importance methods and shows that there is a unique solution in this class that satisfies some desirable properties. One challenge for using these approaches is that they require predefined concepts to be used for explanations.

There has also been some recent work which analyzes the alignment between individual hidden neurons and a set of semantic concepts [Bau et al., 2017]. While this provides additional insight into the semantics of neurons, it requires large sets of data labeled by the semantic concepts and is limited to the semantic concepts in that data. This approach also does not attempt to relate the concepts to final DNN predictions. A more recent approach disentangles the evidence encoded in the activation feature vector, and quantifies the contribution of each piece of evidence to the final prediction [Zhou et al., 2018]. However, similar to previous work, it is also based on supervised learning of predefined concepts and cannot identify concepts which are not learned beforehand.

xNN [Qi et al., 2017], a recent explanation approach, attempts to do away with predefined concepts. It performs a nonlinear dimensionality reduction from the deep learning feature space to a low-dimensional space and tries to mimic the deep learning prediction using this low-dimensional space. It is very faithful to the deep learning classifier it is attempting to explain, although sometimes the learned concepts cannot be recognized by human. In this paper we build on xNN and attempt to obtain human-understandable visual concepts via an interactive user interface.

Importantly, none of the current approaches support human interaction in recognizing, clustering, and naming the concepts implicitly employed by the neural network in making

its decisions. While some methods do employ human recognizable concepts, they are learned by the system offline from a large amount of labeled data that may or may not be relevant to the task at hand.

4.3 Interactive Naming for Test Set Explanations

We first give an overview of the overall approach and then describe each component of the system.

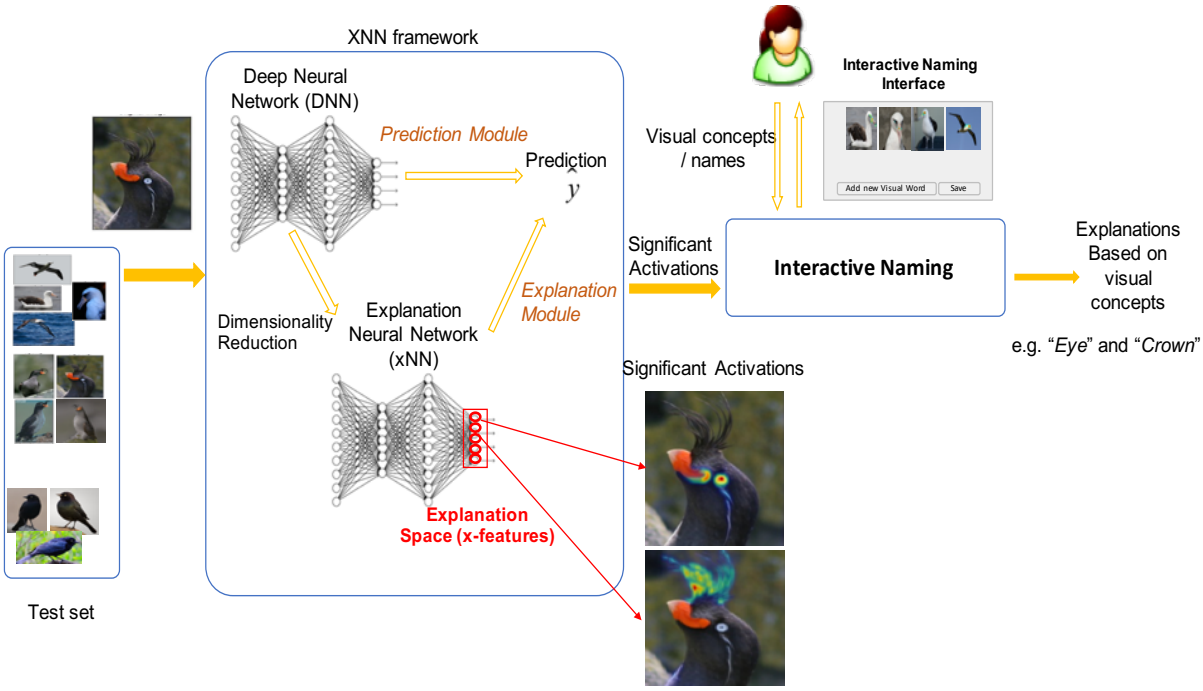


Figure 4.2: Interactive Naming Framework.

4.3.1 Overview

Our overall goal is to develop tools to help understand the decisions of deep neural networks (DNNs) that are trained for image recognition via supervised learning. In particular, we aim to generate meaningful explanations for decisions made over a representative set of test

images. This can provide insight into the strengths and weaknesses of the learned DNN that may not be apparent by just observing test set accuracy. For example, one might hope to discover situations where the DNN is making the right decision, but for the wrong reason, which would identify potential future failure modes.

Figure 4.2 give an overview of our *interactive naming* approach for producing test set explanations. At a high-level, each DNN decision for a test image is dominated by a set of the most *significant activations* of neurons in the penultimate layer. Thus, attaching meaningful concepts to those activations is one way to explain decisions. However, typical DNNs use very large penultimate layers, which makes training easier, but can result in less compact explanations due to the large numbers of significant activations. For this reason we attach an *explanation neural network (xNN)* to the penultimate layer of the DNN, which is trained to reproduce the decisions of the DNN, but dramatically reduces the number of significant activations. The xNN is thereafter used to make decisions and explanations can be formed in terms of its much smaller number of significant activations.

In order to attach meaning to the significant xNN activations we developed an interactive naming interface which displays visualizations of the significant activations to a human annotator. The annotator is then able to cluster the activations into meaningful groups, called visual concepts, and attach linguistic labels to the groups if desired. Given a test instance, we can then form an explanation by producing the significant xNN activations and displaying the group identities/names of those activations. Qualitatively different decisions will tend to have different explanations. A key functionality of the system is to allow for the investigation into the different qualitative decision types over the test set.

The rest of this section explains the above steps in more detail.

4.3.2 Explanation Neural Networks (xNNs)

An xNN [Qi et al., 2017] is an additional network module, which can be attached to any intermediate layer of an original DNN, which typically have thousands of neurons. The xNN learns a lower dimensional embedding for the DNN layer, resulting in a vector of *X-features*, and then linearly maps the X-features to the output $\hat{y}$ in order to mimic the output y of the original DNN model. In this work we focus on xNN modules that are

attached to the first fully-connected layer of the orignal DNN, which reduces thousands of features in the penultimate later to a handful of X-features.

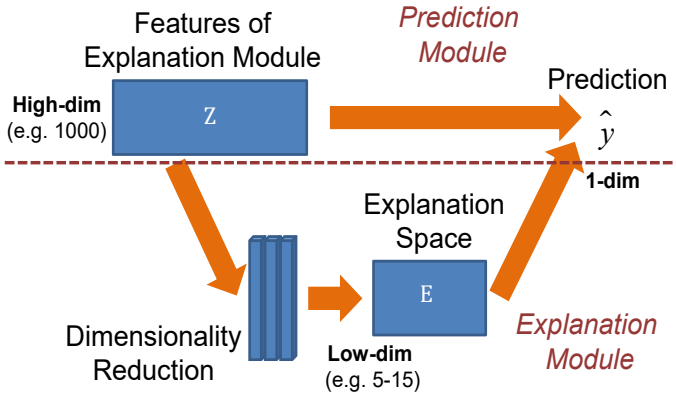


Figure 4.3: Conceptually, the explanation module is a dimensionality reduction mechanism so that the original deep learning prediction $\hat{y}$ can be reproduced from this low-dimensional space. An explanation module can be attached to any layer in the prediction deep network (DNN). The output of the DNN can be faithfully recovered from this low-dimensional explanation space, which represents high-level features that are interpretable to humans [Qi et al., 2017].

In our work, we will be applying xNNs to DNNs for multi-class classification. To do this, as proposed in [Qi et al., 2017] a distinct xNN is trained for each class in a one-against-all manner to predict the numeric score assigned to the class by the DNN. The xNNs can then be used for multi-class prediction by computing the scores produced by each xNN and returning the highest scoring class. This process is depicted in Figure 4.3.

It is desirable for X-features to have the following 3 properties:

1) *Faithfulness*, the DNN predictions can be faithfully approximated from a simple linear transform of the X-features.

2) *Sparsity*, a relatively small number of X-features are active per image

3) *Orthogonality*, the X-features are as independent from each other as possible. Training the xNN thus involves optimizing an objective function via backpropagation that includes terms for each of these properties. The reader is referred to [Qi et al., 2017] for more details.

4.3.3 Explanations via Interactive Naming

Given a test image and a class c, we can use the xNN for c to produce a class score. This score is a weighted linear combination $\sum_i w_i \cdot x_i$ of the X-features x_i.

The positive terms (i.e. X-features with positive weights) in the linear combination can be viewed as providing positive evidence for c. Typically only a subset of the positive terms are significant. Since the X-features with negative weights do not provide positive evidence to the class at hand, their activation maps are not used for annotation. We need to filter the activation maps and select the set of significant activations. We tried two different methods for selecting the *significant X-features* that we will further explain. The significant X-features can be viewed as a type of explanation of why the image might be assigned to class c. However the significant X-features do not have associated semantics, so the explanation is not very useful for human consumption.

To assign semantics to explanations, we can first produce an *activation map* for each significant X-feature in an image for the class under consideration, which identifies the "salient" image region that is responsible for the X-feature activation. In this work, we applied two methods for generating activation maps: the ExcitationBP [Zhang et al., 2016] and *Integrated Gradient in Quadratic (I-GOS)* [Qi et al., 2019]. We call these maps the *significant activation maps* or simply the *significant activations*. While one can gain insight into a prediction by simply viewing the significant activations for specific images, it is difficult to obtain a general understanding of the core semantic concepts and combinations of those concepts used for predictions across an entire test set, which is our goal.

Our interactive naming interface is designed to attach semantics to all of the significant activations across a test set. In particular, the interface allows a human annotator to cluster, or group, the significant activations in a test set of images, where each group is intended to represent a semantically meaningful *visual concept* to the annotator. Not all significant activations need to be assigned to a group, which allows for noisy or confusing activations to be ignored. Activations that are assigned to a visual concept are considered to the *named*, while other activations are considered to be *unnamed*. The complete set of named activations resulting from interactive naming is called a *naming* of the test set.

Given a naming of a test set, we can now generate an *explanation* for each test image by

generating the significant activations of the image and outputting the visual concept names for those activations. Thus, an explanation is just a set of names. If a significant activation is unnamed, then the explanation includes “other" for the name of the activation.

4.3.3.1 Significant Activation Selection Techniques:

There are two different techniques that we applied for filtering the activations and selecting the significant activations:

1) 90% impact on classification: This method keeps only those maps that contribute to 90% of the total positive weight for the final decision. We call these *significant activations*.

2) Minimal Sufficient Explanation (MSX): This method keeps the minimal subset of positive weights whose sum of activations exceeds the sum of all negative activations. [Erwig et al., 2018][Khan et al., 2009]. The concept of minimal sufficient explanation was also introduced in closely related work on explanations for optimal Markov decision process policies [Khan et al., 2009].

4.3.3.2 Different Visualization Techniques:

A lot of research has been focused on generating heat maps that highlight parts of the input image that are most important to the decision of the deep networks on a particular classification target [Zhang et al., 2016] [Qi et al., 2019]. In this thesis we applied two following visualization techniques on each X-feature respectively and generated the activations for this X-feature:

1) Excitation Backprop (ExcitationBP): Excitation Backprop [Zhang et al., 2016] is a gradient-based method to visualize a CNN’s top-down attention map. This method computes the attention map using a probabilistic Winner-Take-All formulation, which is inspired by the selective tuning attention model. We applied ExcitationBP on each X-feature and generated the activations for this X-feature.

2) Integrated Gradient in Quadratic (I-GOS): In [Qi et al., 2019] a new visualization method was proposed that optimizes for a heatmap so that the classification scores on the masked image would maximally decrease. The main novelty of the approach is to compute

descent directions based on the integrated gradients instead of the normal gradient, which avoids local optima and speeds up convergence. Compared with previous approaches, this method can flexibly compute heatmaps at any resolution for different user needs.

Fig 4.4 shows an example of a bird's original image followed by its 5 X-feature activations. The X-feature activations superimposed on the original image are presented as "activation maps" for visualization.

ExcitationBP visualization I-GOS visualization

Figure 4.4: Comparison between Excitation BP and I-GOS [Qi et al., 2019]

4.3.4 Interactive Naming Interface

One of the key aspects of interactive naming is that the set of visual concepts is not known beforehand and varies from person to person. Moreover, the visual concepts in an image are not immediately apparent until the annotator sees multiple images. In previous work, it has been shown that human labelers are more efficient when they are presented with multiple instances at once and are allowed to choose the ones they want to label [Kulesza et al., 2014]. In another real-world comparative study of pairwise and setwise comparison, the authors demonstrated that not only is setwise comparison more efficient, but also elicits more consistent labels [Sarkar et al., 2016].

Following the previous work, we designed a flexible user interface (Fig 4.5) to group the significant activations into different visual concepts and give them textual labels/names. The set of X-feature activations is shown to the annotator in the "Unlabeled Examples" section of the interface. The annotator can freely cluster X-feature activations into visual concepts

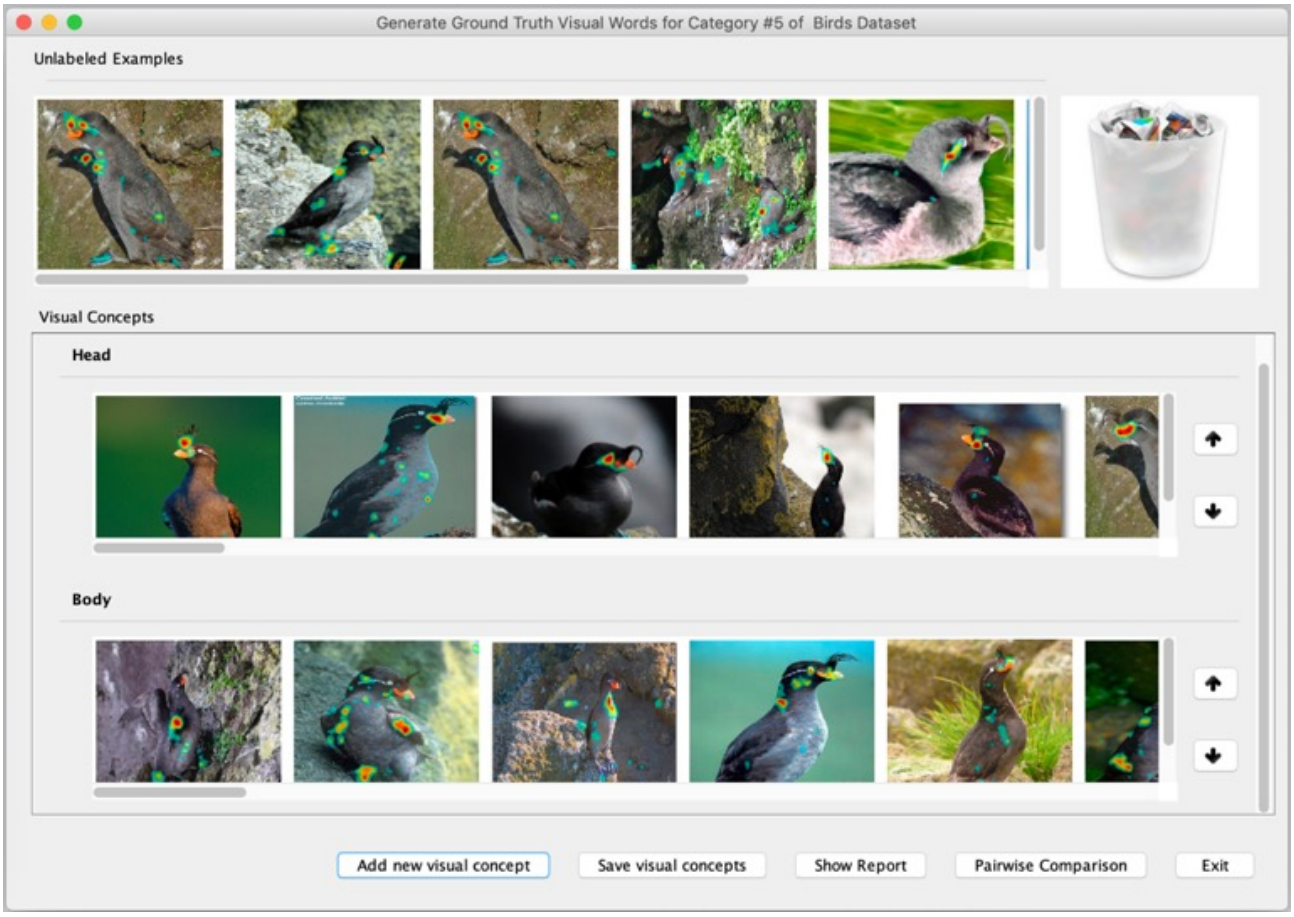


Figure 4.5: Annotation Interface: Our approach allows annotators to explore feature activations and group them into different meaningful textual / visual concepts.

and give them names. The interface allows the annotators to compare all instances, and create visual concepts as and when they are confident. If the annotator is not comfortable with grouping or labeling some activations, they can leave them in the unlabeled section.

4.3.5 Data Preparation

We begin with a convolutional deep neural network (DNN) trained on the available multi-class data. The DNN outputs $p(c_i|I)$ for each given image I and category $c_i \in 1, \ldots, C$.

The penultimate layer of the DNN can be considered as scoring functions for each category $s(c_i|I)$, where a softmax unit, Equation 4.1, serves as the final layer of the DNN that computes the class-conditional probability from the scores.

$$p(c_i|I) = \frac{e^{s(c_i|I)}}{\sum_{j=1}^{C} e^{s(c_j|I)}} \tag{4.1}$$

Next, xNN is trained starting from the first fully-connected layer in the DNN for each class, aiming at being faithful to the scoring functions $s(c_i|I)$ for each category.

All our experiments were conducted on two datasets:

Caltech-UCSD Birds-200-2011 dataset [Wah et al., 2011]. is an image dataset with photos of 200 bird species (mostly North American). As reported in [Qi et al., 2017], for Birds dataset, this approach reduces the dimensionality from 4,096 features in the DNN to 5 X-features in the xNN without significant loss of accuracy. To explain the decision of a DNN using interactive naming approach, we selected 12 categories of Bird Dataset, labeled a-l, shown in Table 4.1.

CEL dataset: This dataset [Eddy et al., 2018], contains 5 categories of breast cancer cells (*Actinedge*, *Filopodia*, *Hemispherebleb*, *Lamellipodia*, and *Smalbleb*) from microscopic images [Eddy et al., 2018]. As reported in [Qi and Li, 2017], for CEL dataset, this approach reduces the dimensionality from 4,096 features in the DNN to 9 X-features in the xNN without significant loss of accuracy. To explain the decision of a DNN using interactive naming approach, we selected 5 categories shown in Tabble 4.2.

The first rows of Table 4.1 and Table 4.2 show the number of images in each category of Birds and CEL datasets, respectively. Since the goal is to explain the decision of DNN, we focus on images that are classified as positive by the DNN, which includes both the false positive and the true positive images. The third, fourth, fifth and sixth rows of the tables, represent the number of the true positive, false positive, false negative and classified positive images for each category.

The seventh and eighth rows of the Tables respectively show the multi-class classification accuracies of the DNN and the xNN trained from the DNN on the original 200 categories. It can be seen that xNN has identical accuracies as the DNN on 11 of the 12

Table 4.1: The relevant X-feature activations of 12 bird categories: *(a) Laysan Albatross, (b)Crested Auklet, (c) Brewer Blackbird, (d) Red-winged Blackbird, (e) Northern Fulmar, (f) Green Jay, (g) Mallard, (h) Black Tern, (i) Common Tern, (j) Elegant Tern, (k) Green-tailed Towhee*, and *(l) Black-capped Vireo*. The table shows the number of images for which each feature makes a significant positive contribution.

	Index of category in Birds dataset											
	a	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*
# images	60	44	59	60	60	57	60	60	60	60	60	51
TP images	52	41	40	57	53	54	53	45	39	45	55	49
FP images	10	3	23	5	15	9	0	10	18	11	9	7
FN images	8	3	19	3	7	3	7	15	21	15	5	2
positive images	62	44	63	62	68	63	53	55	57	56	64	56
DNN accuracy (%)	86.7	93.18	67.8	95.0	88.33	94.74	88.33	75.0	65.0	75.0	91.67	96.08
xNN accuracy (%)	86.7	93.18	67.8	95.0	88.33	96.49	88.33	75.0	65.0	75.0	91.67	96.08
xNN RMSE	0.23	0.23	0.51	0.41	0.35	0.30	0.50	0.20	0.21	0.34	0.45	0.33
Significant activations (MSX)	62	44	63	62	68	63	53	55	57	56	64	56
Mean Significant activations (MSX)	1	1	1	1	1	1	1	1	1	1	1	1
Significant activations (90% impact)	86	88	184	123	102	122	54	74	73	101	101	78
Mean of significant activations (90% impact)	1.4	2	2.92	1.98	1.5	1.94	1.01	1.3	1.27	1.80	1.58	1.40

categories, and performs even better than DNN in one category. The next row shows the Root Mean Squared Error (RMSE) of xNN relative to DNN, which is between $0.2 - 0.5$ while the range of the scoring function is usually $0 - 50$. Since the X-features with negative weights do not provide positive evidence to the class at hand, their activation maps are not used for annotation. We further filter the activation maps to only those maps that are MSX,which we call these *significant activations*.

The ninth and tenth rows of Table 4.1 and Table 4.2 show the total number of generated

Table 4.2: The relevant X-feature activations of 5 breast cancer cells categories of CEL dataset. The table shows the number of images for which each feature makes a significant positive contribution.

Category	*Actinedge*	*Filopodia*	*Hemispherebleb*	*Lamellipodia*	*Smalbleb*
Number of images	275	268	231	324	250
True positive images	214	254	228	264	247
False positive images	4	4	62	34	37
False negative images	122	14	3	60	3
Positive images	218	258	290	298	284
DNN accuracy (%)	78	95	98.7	81.5	99
xNN accuracy (%)	78	95	98.7	81.5	99
Significant activations (MSX)	218	258	290	298	284
Mean of significant activations (MSX)	1	1	1	1	1
Significant activations (90% impact)	445	1175	807	827	845
Mean of significant activations (90% impact)	2.04	4.55	2.78	2.77	2.97

significant activations using MSX in each category and the average number of significant activations per image. The last two rows of Table 4.1 and Table 4.2 show the total number of generated significant activations using 90% impact method in the classification, in each category and the average number of significant activations per image. These results show that the number of selected *significant activations* using the MSX method is less than the number of selected *significant activations* using 90% impact method. On average one significant activation is selected for each positively classified image using MSX. However, using the 90% impact method the average of selected significant activations for each image is larger than one, which means that for each positively classified image more than one significant activation is selected.

Figure 4.6 represents the activation maps of 2 images in Birds and CEL datasets.

4.3.6 X-feature Coverage for Significant Activations

Now we consider the question of what fractions of significant activations are covered by different X-features using both significant activation selection methods. Figure 4.7 and

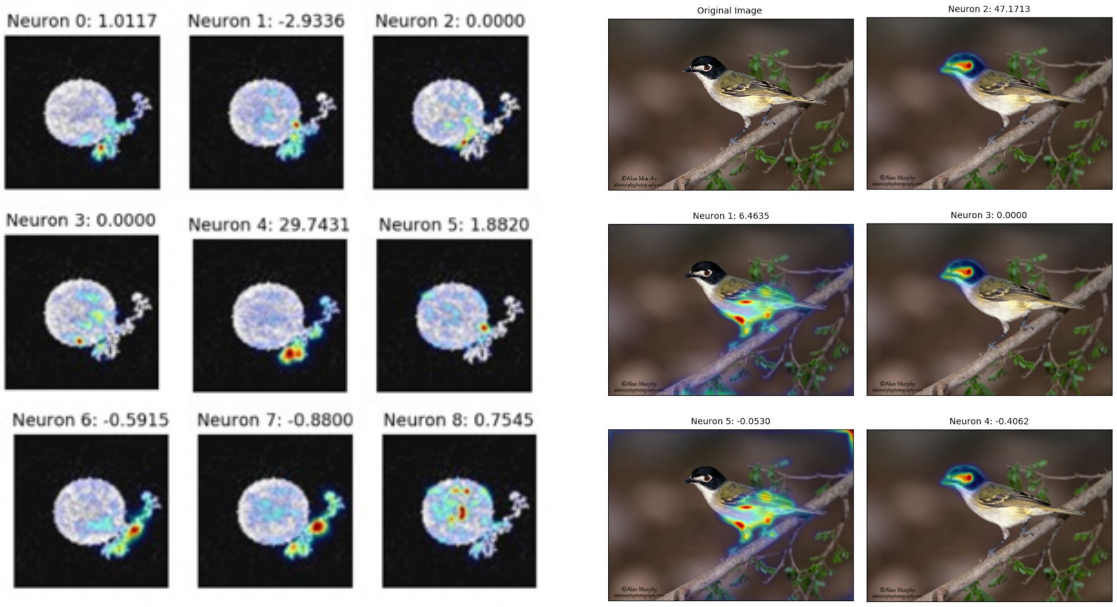
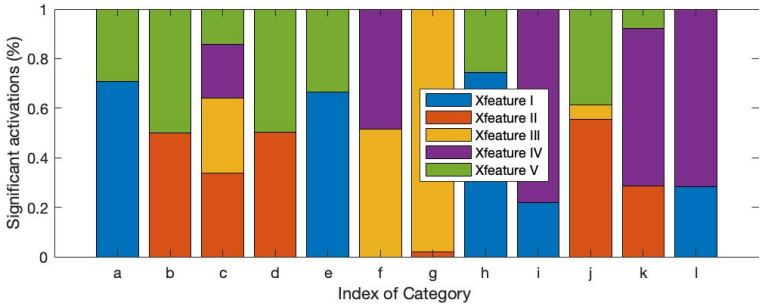


Figure 4.6: Examples of visualization of x-feature activations of an image from CEL dataset (left), and an image from Birds dataset (right). The number above each heatmap represents the contribution of this x-feature to the final prediction.

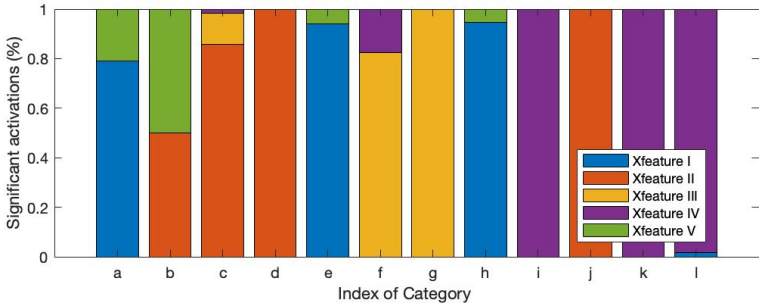
Figure 4.8 represent the comparison between the two methods of selecting significant activations in each category of Birds dataset and CEL dataset, respectively. It can be understood from these figures that using MSX the selected significant activations are covered by fewer X-features than using the 90% impcat in classification method. In some categories of Birds dataset using MSX method (e.g. Figure 4.7 and categories d, i, j and k) only one X-feature was involved in selecting significant activations).

4.4 Classifier Training for Visual Concepts

Part of the goal of our research is to automatically generate explanations in terms of visual concepts. Towards this end, we train classifiers to recognize visual concepts present in the feature activation maps. As input to the classifier, we develop a focused representation for each X-feature activation, called an *I-feature map*. The I-feature map is computed in the following steps: 1) we create a masked image of the X-feature activation, 2) give



Significant activations(90% impact)



Significant activations(MSX)

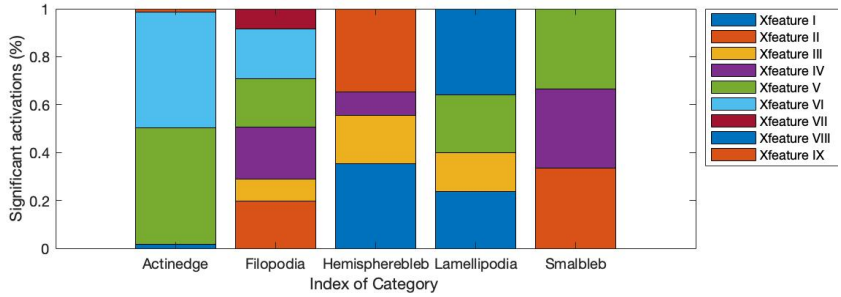
Figure 4.7: Comparison between X-feautre coverage for significant activations across 12 bird categories. Top) The significant activations are generated by 90% impact in classification method, Bottom) the significant activations are generated by minimal sufficient explanations method.

the masked image to the learned DNN and xNN, 3) save the values of some layer of the networks (chosen experimentally) as the I-feature map.

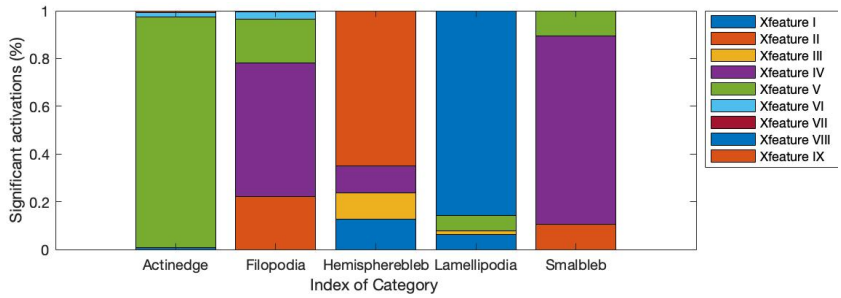
We applied the two following techniques for creating masked images:

1) In-painting technique: In the in-painting step, images are masked so that pixels that have $\leq$ a given threshold percentage of the total response in the heatmap are painted as black. Then an in-painting algorithm is applied to blur the masked image not to show artificial edge effects [Zhang et al., 2013]. We tried different thresholds for the in-painting process and found that a threshold of 30% is the best (Table 4.3). The I-feature map derived from the in-painting is expected to provide the appropriate features for the classifier to learn the visual concepts in the X-feature activations.

2) Blurring technique*:* In this method images are first blurred using a Gaussian function



Significant activations(90% impact)



Significant activations(MSX)

Figure 4.8: Comparison between X-feautre coverage for significant activations across 12 bird categories. Top) The significant activations are generated by 90% impact in classification method, Bottom) The significant activations are generated by minimal sufficient explanations method.

Table 4.3: Comparison of the effect of different in-painintg thresholds on accuracy of weighted SVM with RBF Kernel classifier over the ground-truth generated by Annotator #1 for 12 bird categories.

	Index of bird category											
Threshold	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*
15%	0.46	0.72	0.89	0.81	0.75	0.83	0.75	0.43	**0.49**	0.69	0.75	0.67
20%	0.57	**0.79**	0.91	0.42	0.68	0.89	0.68	**0.52**	0.42	0.64	0.77	0.73
25%	0.50	0.75	0.85	0.75	0.75	0.90	0.68	0.47	0.38	0.58	0.70	0.67
30%	0.69	0.75	**0.93**	**0.84**	0.75	**0.93**	**0.71**	0.507	0.462	0.68	**0.82**	**0.81**
35%	**0.71**	0.72	0.86	0.81	0.75	0.87	0.70	0.48	0.34	0.69	0.71	0.67
40%	0.64	0.78	0.9	0.86	0.77	0.86	0.73	0.57	0.35	0.61	0.75	0.77

and then the original regions of the image which have been highlighted by the X-feature activations are added to the blurred image.

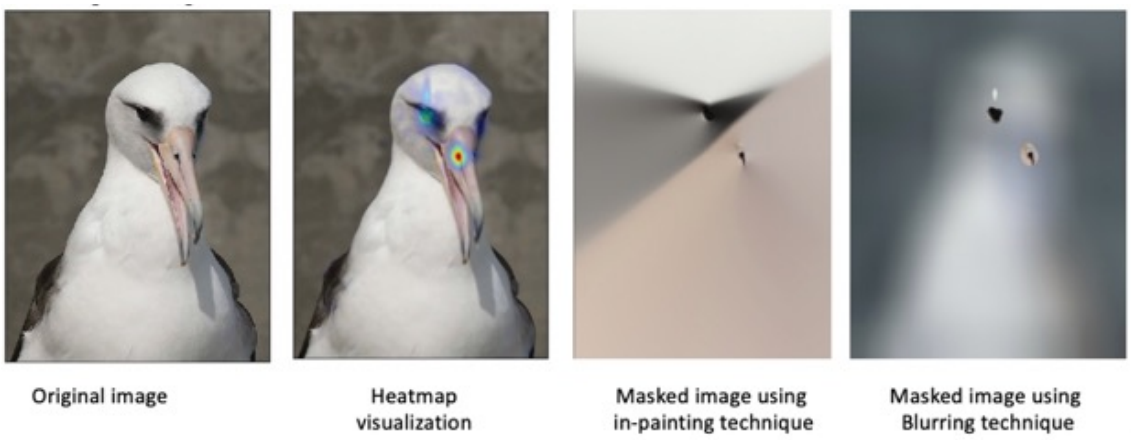


Figure 4.9: An example of masking techniques.

Training classifiers for visual concepts is a challenging multi-class problem, because some visual concepts contain very few instances and the others contain quite a few instances. To learn these imbalanced classes, weighted SVM with RBF Kernel, K- nearest neighbor, or Deep Neural Networks are good classification algorithm candidates [Chang and Lin, 2011].

The last question that remains to be answered is how we generate features for the classification algorithm. To answer this question we trained multi-class SVM classifiers from 4 different layers of DNN/xNNs to decide which layer of the network yields the best I-feature map. Table 4.4 represents the comparison between these four layers.

In the end, we chose the penultimate layer of the DNN, which has 4096 units. Moreover, as pre-processing step we applied PCA to reduce the dimensionality of the I-feature space.

4.5 Human Subject Study 1

In our first user study, we generated the significant activations using 90% impact in classification method, we created the visualization using Excitation BP, and we conducted our experiments using the 12 categories of Birds dataset. We had the activation maps of the different images annotated by 5 different subjects using the annotation interface. The activation maps were separated by the class, but not by the X-feature. The annotators were

Table 4.4: Comparison of the effect of selecting different layers of xNN or DNN as I-features on the accuracy of weighted SVM with RBF Kernel classifier over the ground-truth generated by Annotator #1 for 12 bird categories (in-painting threshold is 30%)

		Index of bird category											
I-Feature	size	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*
Penultimate layer of DNN	4096	**0.69**	**0.75**	**0.93**	0.84	0.75	0.93	**0.71**	**0.51**	**0.46**	**0.68**	**0.82**	**0.81**
1st layer of xNN	4096	0.67	0.75	0.90	**0.86**	0.73	0.93	0.71	0.49	0.41	0.67	0.79	0.81
2nd layer of xNN	800	0.48	0.66	0.76	0.84	**0.75**	0.75	0.53	0.37	0.35	0.61	0.68	0.55
3rd layer of xNN	100	0.41	0.65	0.71	0.86	0.75	0.67	0.54	0.37	0.36	0.42	0.64	0.71

instructed to not introduce visual concepts that only applied to one or two images, but were otherwise free to cluster and label as many images as it made sense to them. However, not all subjects followed the instructions and left some clusters with less than 3 images. In the following analysis, we first cleaned the data by removing a small number of clusters with less than 3 images.

4.5.1 RQ1: Coverage of Interactive Naming

Since the human annotators are not forced to assign visual concepts to, or name all significant activations, some of the activations in the data are unnamed and treated as noise/outliers. Here we are interested in how well the annotations cover the activations and how this coverage varies across annotators.

Figure 4.16 shows the fraction of significant activations that are named by each annotator for each bird category. In addition, the last bar for each category, labeled “Any Annotator", shows the fraction of significant activations that were assigned to a visual concept by at least one annotator. We see that within a particular class, there is relatively small variation among users and that the “Any Annotator" bar is not much higher than that of the typical individual annotator. This indicates that there is some consistency in the set of activations that users consider to be noise. We also see that for most categories there is a relatively significant amount of activations not labeled by users, approximately ranging from 20% to

40%.

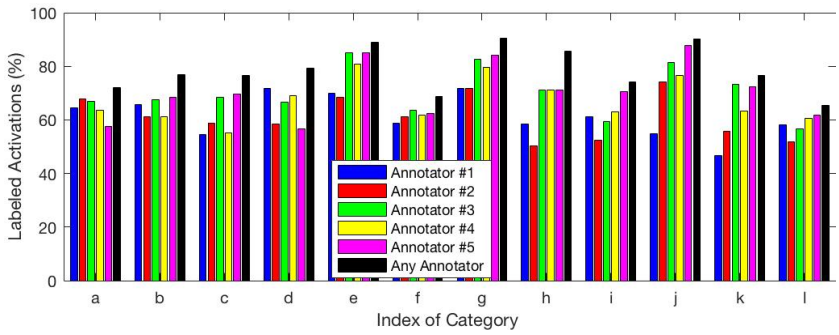


Figure 4.10: Fraction of labled significant activations for each category across annotators.

We now consider how well the annotations cover explanations, which gives a better sense of how useful they will be for analyzing explanations. In particular, we consider an explanation for an image to be *completely (partially) covered* by an annotation if all (at least one) of the significant activations for that image are named.

Figures 4.11 and 4.12 show the partial and complete coverage for each annotator and the “Any Annotator". We see that for most annotators the fraction of explanations that are at least partially covered is quite high. This means that at least partial semantics will be available for explanations on the vast majority of cases. We also see that the “Any Annotator" bar is similar to the individual annotators, which indicates that the sets of partially covered explanations across annotators is similar.

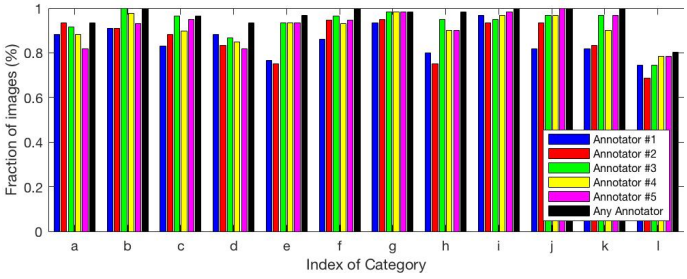


Figure 4.11: Partial explanation coverage for each category across annotators.

The complete coverage percentages drop substantially, which is not surprising given the results for activation coverage from Figure 4.10. Once again the “Any Annotator” bar is

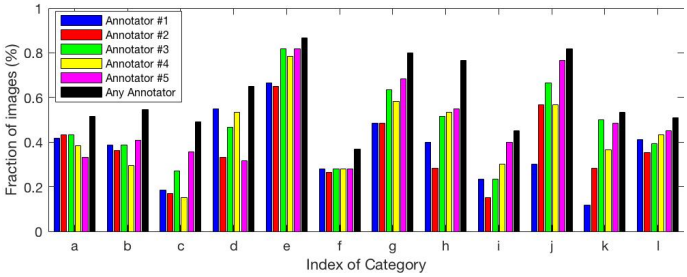


Figure 4.12: Complete explanation coverage for each category across annotators.

not significantly different from the rest.

Overall, we see that a non-trivial fraction of significant annotations are not named by users. This necessarily resulted in less than 50% complete coverage rates for most users. From further analysis, the low rates of complete coverage seems to be due to the fact that in most cases the final decision is dominated by a single significant activation. The other significant activations appear to be not strong enough to lead to easily recognizable concepts and hence are not named by the annotators. Still there are a non-trivial number of examples that are completely covered, which allows for analysis of full explanations for a fraction of the test data. Even this can help build trust and identify potential flaws. On the other hand, most explanations are at least partially covered, which allows for gaining some semantic insight into most decisions. Even partial explanations can be useful in identifying flaws and building trust. These results suggest future work on increasing the coverage by bootstrapping from partial coverage, e.g. by incorporating the annotations into retrained DNNs.

We performed a qualitative analysis to understand some of the reasons that annotators were not able to assign names to activations. One of the major reasons was when activations were difficult to interpret and appeared to be noise. For example, when activations highlight the edge of the image or fall on background with unclear semantics. Such activations are potential warning indicators about a classifier. Thus, uncovering these examples through interactive naming has value. In other cases, the activation map was interpretable to the annotator, but there were not enough similar activation maps to form a cluster. This case may be resolved by using a larger test set.

4.5.2 RQ2: X-feature-Visual Concept correspondence

We now consider the question of correspondence between the visual concepts and the X-features. Since one of the goals of xNNs is to disentangle the concepts into independent components, we wanted to see how well this goal has been achieved. In an ideal case, one might hope for a one-to-one correspondence between visual concepts and the X-features. However, this is unrealistic as our xNN did not have any supervision on the visual concepts.

We adopt the metric of *purity* to measure the correspondence between visual concepts and X-features [Manning et al., 2008]. The purity of a clustering is defined as the number of examples that belong to the plurality class of each cluster as a fraction of the total number of examples. Here, we employ purity to measure both the degree to which each visual concept maps to a single X-feature, i.e., visual concept $\rightarrow$ X-feature purity or *CX-purity*, and the degree to which each X-feature maps to a single visual concept, i.e., X-feature $\rightarrow$ concept or *XC-purity*.

We call the set of clusters of activations created by each annotator, *a naming*. To compute CX-purity of a naming, we first assign each visual concept to the X-feature to which a plurality of its significant activations belong. We call it *the majority X-feature* of that concept. CX-purity is the number of activations in the naming that belong to the majority X-feature of their concept as a fraction of all named activations. Similarly we define the majority concept of an X-feature to be the concept that is assigned to a plurality of activations of that X-feature. XC-purity is the number of activations in the naming that belong to the majority concept assigned to their X-feature as a fraction of all named activations.

As it may be clear, both CX-purity and XC-purity vary from annotator to annotator and from category to category. Table 4.5 shows the CX-purity values for each category and each annotator. The purity numbers varying from 0.52 to 0.90 for different categories and annotators. Different annotators appear to be consistent across categories. The same categories, e.g., j and f, score lower in purity for most annotators.

Table 4.6 shows the XC-purity values for different categories and annotators. The purity numbers are generally worse, going down to 0.30 in one case (Category a, Annotator #3) and a maximum of 0.83 in a rare case (Category d, Annotator #5). This suggests that

Table 4.5: CX-purity in Birds dataset: Shows the degree to which the activations of a visual concept map to a single X-feature.

Category	Annotator					
	#1	#2	#3	#4	#5	average
a	0.658	0.66	0.683	0.653	0.65	0.66
b	0.690	0.757	0.753	0.757	0.78	0.75
c	0.604	0.634	0.620	0.609	0.6	0.61
d	0.79	0.757	0.8	0.734	0.76	0.77
e	0.882	0.88	0.903	0.932	0.90	0.90
f	0.581	0.647	0.60	0.582	0.60	0.60
g	0.717	0.707	0.72	0.745	0.68	0.72
h	0.767	0.77	0.76	0.730	0.76	0.76
i	0.615	0.606	0.61	0.523	0.57	0.59
j	0.558	0.608	0.62	0.589	0.60	0.60
k	0.803	0.746	0.71	0.75	0.76	0.75
l	0.872	0.880	0.85	0.857	0.86	0.86
average	0.66	0.67	0.67	0.65	0.66	

the mapping from X-features to visual concepts is more one-to-many than the other way around. This is to be expected since the number of significant features is often only 2-3 and smaller than the number of visual concepts. Again, different annotators appear to be largely consistent across categories with the X-features in some categories, e.g., d and l, more easily disentangled than the X-features of other categories, e.g., a and g.

Figure 4.13 represents the fraction of visual concept labels that correspond to each X-feature for different bird categories. It can be seen from this figure that the activations of a single X-feature is often assigned to different visual concepts. The same visual concept is also attached to the activation maps of multiple X-features. Thus the mapping between visual concepts and X-features is many-to-many. This suggests that the X-features are not perfectly disentangled and we might do well to explain the final category decisions directly in terms of visual concepts rather than through the X-features.

4.5.3 RQ3: Inter-annotator Agreement

In general we can expect different annotators to produce different namings for a test set, where at least some of the visual concepts differ. Here we consider the extent to which these

Table 4.6: XC-purity in Birds dataset: Shows the degree to which the activations of an X-feature map to a single visual concept.

Category	Annotator					
	#1	#2	#3	#4	#5	average
a	0.394	0.512	0.30	0.453	0.382	0.41
b	0.62	0.65	0.57	0.59	0.648	0.62
c	0.709	0.548	0.49	0.79	0.509	0.61
d	0.709	0.8	0.75	0.77	0.84	0.77
e	0.71	0.7	0.66	0.73	0.645	0.69
f	0.724	0.44	0.75	0.75	0.61	0.65
g	0.44	0.373	0.4	0.645	0.491	0.46
h	0.46	0.460	0.5	0.494	0.382	0.46
i	0.33	0.348	0.31	0.55	0.308	0.37
j	0.47	0.56	0.58	0.54	0.550	0.54
k	0.428	0.38	0.36	0.526	0.574	0.46
l	0.702	0.81	0.61	0.75	0.86	0.75
average	0.52	0.51	0.48	0.58	0.52	

different namings agree and in turn whether explanations produced by different namings are semantically similar. Understanding this issue is important for understanding the extent to which explanations are fundamentally annotator-specific.

First, we consider annotator agreement about which significant activations should be named. Figure 4.17 shows, for each bird category, the fraction of significant activations that were named by different numbers of annotators - 0 thru 5. Interestingly, the largest fraction of activations are annotated by all 5 annotators and the second largest are annotated by 0 annotators. This confirms, once again, that for most significant activations, either all annotators choose to assign a name or none of them do. There is strong agreement about the set of activations that should be named.

We now take a more nuanced look at the agreement between visual concepts that different annotators created for each category. In particular, we want to know how similar are these concepts and find potential translations between the concepts of different annotators. Are there one-to-one correspondences, are there subsumption relationships, or are there cases of purely incompatible concepts?

Given two namings N_i and N_j produced by two annotators i and j, we are interested

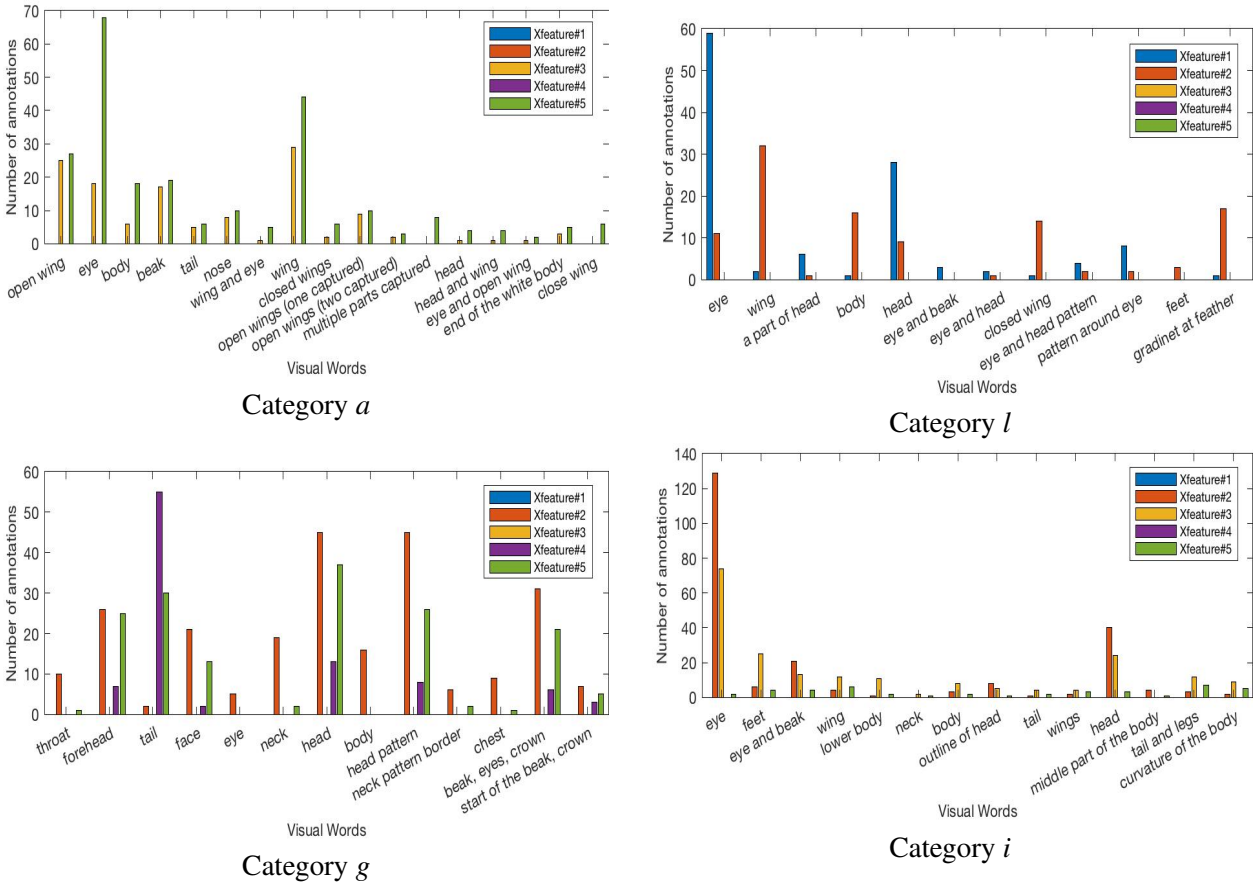


Figure 4.13: The visual concepts that correspond to each X-feature and their frequency

in matching the clusters, or visual concepts, between the namings. For this purpose, we follow the cluster matching framework of [Cazals et al., 2017] to find a best match. The framework first defines "the intersection graph" G of N_i and N_j. G is a bipartite graph where the vertices in each partite set correspond to the clusters of N_i and N_j respectively. There is a weighted edge between each cluster of N_i and N_j, with the weight equal to the size of the intersection of the clusters. Thus, a large weight between two visual concepts indicates that they represent many of the same activations in the test set.

According to [Cazals et al., 2017], a D-family matching is a partition of all nodes (that belong to either naming) of the bipartite graph into some number of disjoint sets $S_1, S_2, ...$

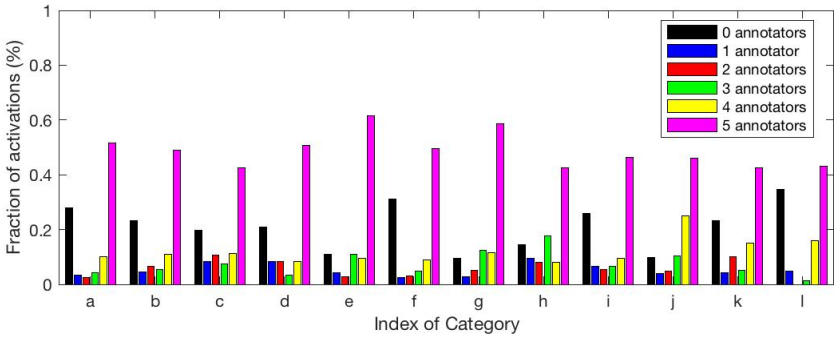


Figure 4.14: Fraction of significant activations that are named by exactly n of the annotators, where $n \in \{0, 1, 2, 3, 4, 5\}$

such that the diameter of all subgraphs of G over the nodes in S_i is $\leq D$. The best D-family matching maximizes the sum of the weights of all edges in all the subgraphs.

Although restriction of diameter sounds odd, it has a natural interpretation. $D = 1$ corresponds to the standard 1-1 matching between bipartite graphs found by the "Hungarian algorithm" [Kuhn and Yaw, 1955]. $D = 2$ allows one cluster to be refined into multiple clusters on the other side, but disallows further splitting or merging of these smaller sets. Importantly, the clusters can be refined in both directions simultaneously as long as they do not overlap with each other. Thus, when one visual concept in N_i is matched to multiple concepts in N_j for $D = 2$, we can view this as an approximate subsumption relationship among the concepts. In this paper we compute agreement scores based on both $D = 1$ and $D = 2$ for each pair of annotators.

Figure 4.15 shows examples of matching between 3 pairs of users with $D = 1$ and $D = 2$. It is worth noting that there are one-to-one mappings as well as one-to-many mappings from both sides for $D = 2$. The words that are associated with each visual concept are the linguistic names assigned by the annotators to those concepts. In the first mapping of Figure 6, for example, both 'Open Wing' and 'body' on the left map to 'Wing' on the right. On the other hand, the single cluster 'Eye' maps to 'Head', 'Eye' and 'Head and Wing' on the right.

Overall, many of the matched visual concepts are sensible based on the linguistic descriptions provided by the annotators. This is especially true for $D = 1$. Many of the one-to-many matches in $D = 2$ also appear to made sense based on the words. These

Table 4.7: Test set explanation summary for one annotator for a sample of categories

Category	Visual concepts combinations
a	('eye', 'close wing') 56.6667% , ('close wing') 10% , ('open wing', 'eye') 6.6667% , ('eye', 'end of the white body') 5% , ('nose') 5% , ('eye') 3.3333% , ('open wing') 1.6667% , ('end of the white body') 1.6667% , ('unlabeled') 10%
b	('crown') 25% , ('beak', 'wing') 25% , ('wing') 25% , ('crown', 'neck') 4.5455% , ('unlabeled') 4.5455% , ('neck') 4.5455% , ('beak', 'feet') 4.5455% , ('beak', 'crown') 2.2727% , ('wing', 'neck') 2.2727% , ('beak') 2.2727%
c	('feet', 'eye and beak') 32.2034% , ('eye and beak') 28.8136% , ('eye', 'eye and beak') 25.4237% , ('eye', 'feet') 5.0847% , ('feet') 3.3898% , ('unlabeled') 3.3898% , ('eye') 1.6949% ,
d	('red spot in wing', 'tail') 60% , ('tail', 'forehead') 11.6667% , ('red spot in wing', 'forehead') 11.6667% , ('unlabeled') 8.3333% , ('forehead') 3.3333% , ('red spot in wing') 3.3333% , ('tail') 1.6667% ,
e	('eyes and beak') 56.6667% , ('eye', 'wing') 26.6667% , 13.3333% , ('wing') 1.6667% , ('eye') 1.6667% ,
f	('throat', 'forehead') 64.9123% , ('throat') 17.5439% , ('unlabeled') 8.7719% , ('throat', 'tail') 3.5088% , ('forehead', 'tail') 3.5088% , ('forehead') 1.7544%
g	('tail') 30% , ('feet') 25% , ('feet', 'wing') 15% , ('wing') 15% , ('wing', 'head') 5% , ('tail', 'head') 3.3333% , ('head') 3.3333% , ('feet', 'head') 1.6667% , ('neck', 'head') 1.6667% ,
h	('tail') 26.6667% , ('feet', 'tail') 25% , ('feet') 18.3333% , ('eye and beak') 5% , ('wing') 5% , ('eye', 'eye and beak') 3.3333% , ('wing', 'feet') 3.3333% , ('eye', 'feet') 3.3333% , ('feet', 'eye and beak') 3.3333% , ('eye', 'wing') 1.6667% , ('eye') 1.6667% , 1.6667% , ('wing', 'eye and beak') 1.6667%
i	('feet', 'tail') 11.6667% , ('wing', 'head pattern') 11.6667% , ('beak', 'eye', 'wing') 10% , ('tail') 6.6667% , ('beak') 6.6667% , ('beak and feet', 'beak', 'wing') 5% , ('head pattern', 'tail') 5% , ('feet', 'beak', 'wing') 5% , 3.3333% , ('beak', 'wing') 3.3333% , ('feet') 3.3333% , ('beak and feet', 'feet', 'wing') 3.3333% , ('eye', 'head pattern', 'tail') 3.3333% , ('beak and feet', 'wing') 3.3333% , ('beak and feet') 3.3333% , ('beak and feet', 'tail') 1.6667% , ('beak', 'eye') 1.6667% , ('beak', 'tail') 1.6667% , ('feet', 'wing') 1.6667% , ('head pattern') 1.6667% , ('feet', 'head pattern') 1.6667% , ('eye') 1.6667% , ('beak', 'head pattern') 1.6667% , ('feet', 'beak') 1.6667%
j	('legs and head', 'head and wing') 25% , ('legs', 'head and wing') 18.3333% , ('wing', 'legs and head') 15% , ('legs and head') 11.6667% , ('head', 'legs and head') 8.3333% , ('wing', 'head and wing') 5% , ('head and wing') 3.3333% , ('legs') 3.3333% , ('head') 3.3333% , ('wing') 1.6667% , ('wing', 'head') 1.6667% , ('head', 'head and wing') 1.6667% , ('head', 'legs') 1.6667%
k	('feet') 45% , ('head', 'feet') 23.3333% , ('eye') 6.6667% , ('wing', 'head') 5% , ('eye', 'wing') 5% , ('eye', 'feet') 3.3333% , 3.3333% , ('wing', 'feet') 3.3333% , ('wing') 3.3333% , ('head') 1.6667%
l	('eye') 88.2353% , ('head') 5.8824% , ('wing') 1.9608% , ('unlabeled') 1.9608% , ('eye', 'wing') 1.9608%

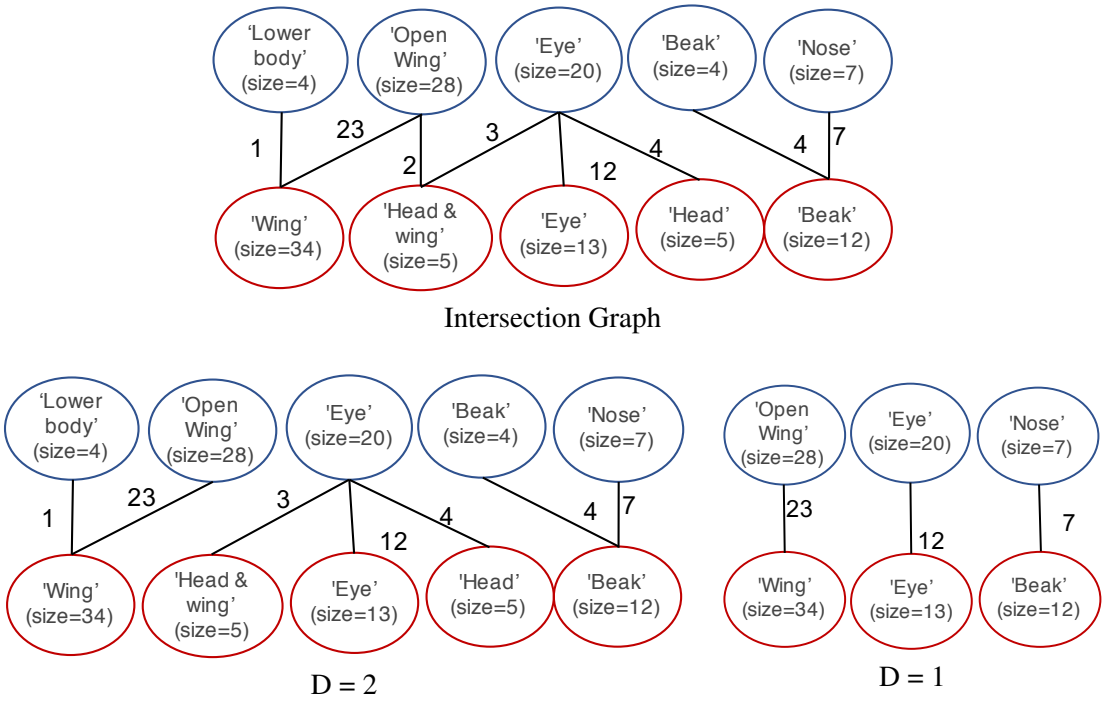


Figure 4.15: An example of pairwise similarity matching between two annotators. The blue and red nodes of the graphs are the clusters/namings generated by annotator #i and annotator #j, respectively. In each node of the graphs the name and the size of a cluster is shown.

one-to-many mappings indicate different choices in the resolution of assigning names. There are some other cases, where the matchings do not appear to make semantic sense based on the names. For example, 'Eye' matching to 'Head and wing' might not appear to be a good match. However, this can simply reflect somewhat arbitrary labeling conventions used by different annotators. For example, one annotator may consider activation maps that are "brightest" on the eye to be labeled 'eye', while another annotator may also pay attention to parts of the activation map that are not brightest, but still active.

Since not all significant activations are labeled by all annotators, we first try to characterize the fraction of common annotations between pairs of annotators. Let us denote by A_i and A_j the number of significant activations throughout the test set labeled by Annotator #i and Annotator #j, respectively. we use the Jaccard index, which is the ratio of the intersection to the union of the two sets, i.e., $\frac{|A_i \cap A_j|}{|A_i \cup A_j|}$, to measure the fraction of the images

both annotators annotated. This is shown in the last column of Table 4.15 averaged over different pairs of annotators. The Jaccard index is fairly high for all categories, indicating that there is a good overlap between the sets of activations chosen by different annotators to annotate.

We compute the agreement between the two annotators for $D = 1$ and $D = 2$ as the total weight of all edges in the D-family matching as a fraction of the number of activations labeled by both annotators. If we interpret the matchinges as translations between namings, then the agreement is the fraction of activations that are translatable between namings. The columns labeled "Agreement" in Table 4.15 shows the statistics of 1-family and 2-family agreements for each category over the set of all annotator pairs. The average and standard deviation as well as min and max of the scores are shown. The agreement numbers are fairly high across most categories, although the minimum values for some categories for $D = 1$ are low. Since 2-family matching is more permissive than 1-family matching, the agreement numbers are higher for $D = 2$ as we expect. Even for $D = 1$ the agreement in most categories is reasonably high, which shows that there is reason to be optimistic about developing a common ontology for explanations.

Test Set Explanation Summaries. One of the motivations for naming a test set is to produce summaries of the explanation types used to predict test images. Table 4.7 gives examples of test set summaries for 4 categories that are produced based on the namings of a single annotator. For example, we see that for category 'a' over 56% of the test set predictions had the explanation ('eye', 'close wing'), which indicates that the network was focusing on the bird eye and closed wing area for those examples. As another example, for category 'l' over 88% of the predictions had the explanation ('eye'), which means the network only looked at the eye area to make the prediction. This type of insight may cause a practitioner to question the robustness of the classifier if they have reason to believe the eye alone is not discriminative enough. Alternatively, an expert may gain insight from this explanation and realize that the eye is discriminative enough for the task.

Linguistic Compatibility of Matched Concepts. We now consider the compatibility between the linguistic labels given to the visual concepts by the annotators and the semantic mapping found by our matching system. This gives some indication of how well the automated matching computations correspond to the annotator's attempts to attach linguistically

Table 4.8: Pairwise comparison between clusters generated by annotators over all categories of Birds dataset.

		Pairwise similarity scores		
Category		**Agreement (D=1)**	**Agreement (D=2)**	**Jaccard index**
a	min	0.6	0.88	0.78
	average	0.74±0.09	0.96±0.04	0.82±0.04
	max	0.9	1	0.89
b	min	0.74	0.86	0.73
	average	0.82±0.05	0.91±0.04	0.81±0.04
	max	0.9	0.97	0.87
c	min	0.7	0.82	0.7
	average	0.79±0.07	0.92±0.05	0.75±0.05
	max	0.9	1	0.86
d	min	0.92	0.97	0.75
	average	0.96±0.02	0.98±0.01	0.8±0.04
	max	1	1	0.88
e	min	0.79	0.89	0.76
	average	0.88±0.06	0.97±0.04	0.83±0.05
	max	1	1	0.91
f	min	0.58	0.82	0.81
	average	0.77±0.13	0.94±0.04	0.86±0.04
	max	0.99	0.99	0.91
g	min	0.57	0.77	0.72
	average	0.69±0.07	0.91±0.07	0.8±0.05
	max	0.79	1	0.85
h	min	0.59	0.76	0.6
	average	0.73±0.06	0.86±0.07	0.7±0.08
	max	0.8	0.95	0.82
i	min	0.6	0.75	0.71
	average	0.69±0.07	0.85±0.07	0.8±0.04
	max	0.81	0.94	0.86
j	min	0.57	0.77	0.58
	average	0.74±0.1	0.85±0.05	0.75±0.13
	max	0.86	0.92	0.91
k	min	0.52	0.85	0.64
	average	0.71±0.1	0.92±0.05	0.77±0.08
	max	0.85	1	0.88
l	min	0.67	0.86	0.74
	average	0.86±0.09	0.95±0.05	0.85±0.06
	max	1	1	0.96
Global Average		0.78	0.91	0.79

Table 4.9: Agreement between the automatically produced matchings between clusters (i.e. visual concepts) and the linguistic names assigned to clusters by the annotators.

		Exact compatibility		**Subset compatibility**	
Category		**(D = 1)**	**(D = 2)**	**(D = 1)**	**(D = 2)**
a	min	0.45	0.34	0.9	0.87
	average	0.65±0.3	0.56±0.46	0.99±0.01	0.95±0.02
	max	0.92	0.85	1	1
b	min	0.84	0.75	0.9	0.87
	average	0.91±0.03	0.83±0.04	0.97±0.01	0.97±0.02
	max	1	0.93	1	1
c	min	0	0	0.92	0.9
	average	0.56±1.18	0.51±0.88	0.98±0.01	0.96±0.01
	max	1	0.83	1	1
d	min	0.92	0.87	0.97	0.94
	average	0.98±0.01	0.96±0.01	0.99±0	0.99±0
	max	1	1	1	1
e	min	0.15	0.13	1	1
	average	0.64±1.23	0.58±0.97	1±0	1±0
	max	1	0.92	1	1
f	min	0.19	0.16	1	0.93
	average	0.37±0.43	0.35±0.46	1±0	0.98±0.01
	max	0.93	0.92	1	1
g	min	0.2	0.16	0.92	0.87
	average	0.61±0.28	0.49±0.26	0.99±0	0.96±0.01
	max	0.89	0.8	1	1
h	min	0.09	0.07	0.96	0.97
	average	0.49±0.51	0.42±0.35	0.99±0	0.99±0
	max	0.96	0.86	1	1
i	min	0.06	0.27	0.98	0.97
	average	0.57±0.47	0.53±0.34	1±0	0.99±0
	max	0.97	0.88	1	1
j	min	0.12	0.11	1	1
	average	0.43±0.85	0.42±0.69	1±0	1±0
	max	0.92	0.89	1	1
k	min	0	0	0.37	0.66
	average	0.3±0.6	0.32±0.31	0.94±0.36	0.95±0.09
	max	0.62	0.64	1	1
l	min	0	0	0.59	0.68
	average	0.35±1	0.41±0.77	0.88±0.31	0.91±0.18
	max	1	1	1	1
Global Average		0.57	0.53	0.978	0.971

meaningful descriptions to concepts. By convention we will put quotes around individual linguistic names assigned to clusters, which often involve the conjunction of multiple terms. For example, ‘Wing and Eye’, is a name that corresponds to activations that covered the eye and parts of the wing.

We consider two types of compatibility – 1) exact compatibility and 2) subset compatibility – between the linguistic labels given to two matched clusters. The word labels are exactly compatible if the words used are considered semantically equivalent: e.g., ’beak’ and ’nose,’ and ’Eye’ and ’Eye’. Subset compatibility is when one of the words seems to describe a subset of the other word. For example, ’Open Wing’ and ’Wing,’ or ’Wing’ and ’Wing and Eye’.

The linguistic names that were judged as not compatible can be grouped into two categories: 1) Names that have semantically different meanings, for example, ’Tail’ and ’Wing’, ’Feet’ and ’wing shape’ , ’Eye and beak’ and ’Body’, and 2) Names that are very ambiguous and are difficult to match, for example: ’multiple parts captured’ , ’bottom body’ , ’gradinet at feather’ ,or ’middle body/Feathers’.

Table 4.9 shows the comparison of exact compatibility and subset compatibility over all categories. Given a matching graph of a pair of annotators, we compute the sum of the edge weights for those edges that are semantically compatible divided by the sum of all edge weights. The numbers in the table are averaged over all pairs of annotators in each category. The Table shows that subset compatible word pairs score highly for both $D = 1$ and $D = 2$ for almost all categories. They are also quite consistent across annotators. However, exact compatibility between name pairs displays a wide range across annotators for almost all categories. This suggests that the meanings of word labels are not exactly consistent across annotators and should not be trusted, although the subsumption relationships between the words seem to hold more consistently across the annotators and categories.

4.5.4 RQ4: Visual Concept Generalization

How well do the visual concepts of different annotators and categories generalize? To answer this question we ran SVM and k-Nearest Neigbor (kNN) classifiers on the examples of labeled X-feature activation maps. The I-features of the penultimate layer of the DNN

Table 4.10: Comparison of KNN and SVM on the ground-truth generated by 5 different annotators over 12 bird categories, 4-folds cross validation

		Index of bird category											
User	Classifier	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*
#1	SVM	**0.69**	**0.75**	**0.93**	**0.84**	**0.75**	**0.93**	**0.71**	**0.51**	**0.46**	**0.68**	**0.82**	0**.81**
	KNN (K=3)	0.36	0.5	0.75	0.44	0.75	0.72	0.51	0.27	0.22	0.42	0.54	0.33
	KNN (K=4)	0.42	0.48	0.72	0.44	0.75	0.75	0.49	0.22	0.27	0.45	0.47	0.38
	KNN (K=5)	0.44	0.48	0.7	0.42	0.75	0.69	0.46	0.3	0.202	0.45	0.57	0.38
	KNN (K=6)	0.31	0.52	0.72	0.44	0.75	0.72	0.44	0.35	0.27	0.45	0.43	0.45
#2	SVM	0.63	0.82	0.61	1.0	0.73	0.57	0.56	0.45	0.41	0.61	0.63	0.71
	KNN (K=3)	0.27	0.58	0.5	0.71	0.71	0.32	0.43	0.21	0.20	0.53	0.39	0.34
	KNN (K=4)	0.30	0.55	0.51	0.71	0.71	0.33	0.48	0.17	0.23	0.53	0.4	0.3
	KNN (K=5)	0.37	0.55	0.51	0.71	0.71	0.36	0.48	0.22	0.20	0.56	0.37	0.36
	KNN (K=6)	0.38	0.53	0.51	0.71	0.71	0.37	0.47	0.26	0.20	0.55	0.39	0.36
#3	SVM	0.5	0.5	0.62	0.34	0.66	0.69	0.45	0.49	0.47	0.63	0.45	0.25
	KNN (K=3)	0.28	0.33	0.42	0.29	0.65	0.58	0.35	0.34	0.30	0.47	0.33	0.19
	KNN (K=4)	0.23	0.36	0.44	0.30	0.65	0.60	0.37	0.31	0.32	0.46	0.34	0.18
	KNN (K=5)	0.27	0.34	0.42	0.30	0.65	0.55	0.33	0.29	0.33	0.47	0.34	0.22
	KNN (K=6)	0.32	0.35	0.42	0.30	0.65	0.54	0.35	0.32	0.35	0.49	0.31	0.25
#4	SVM	0.66	0.76	0.89	0.6	0.78	0.93	0.8	0.54	0.67	0.57	0.80	0.73
	KNN (K=3)	0.24	0.51	0.80	0.63	0.76	0.72	0.72	0.35	0.42	0.53	0.66	0.33
	KNN (K=4)	0.24	0.47	0.80	0.63	0.76	0.72	0.7	0.25	0.50	0.53	0.59	0.49
	KNN (K=5)	0.3	0.47	0.80	0.63	0.76	0.72	0.69	0.32	0.51	0.51	0.60	0.49
	KNN (K=6)	0.28	0.47	0.80	0.63	0.76	0.72	0.7	0.31	0.49	0.53	0.53	0.49
#5	SVM	0.71	0.64	0.52	0.84	0.68	0.60	0.57	0.47	0.40	0.61	0.78	0.75
	KNN (K=3)	0.33	0.52	0.424	0.70	0.68	0.43	0.48	0.21	0.27	0.49	0.51	0.57
	KNN (K=4)	0.34	0.49	0.46	0.57	0.68	0.47	0.51	0.28	0.27	0.5	0.52	0.53
	KNN (K=5)	0.34	0.44	0.40	0.57	0.68	0.49	0.48	0.23	0.24	0.53	0.47	0.53
	KNN (K=6)	0.35	0.48	0.40	0.47	0.68	0.46	0.48	0.24	0.22	0.52	0.45	0.55

are given as the inputs and the labels of the visual concepts are given as the outputs for the classifiers. The results are shown in Table 4.17.

The accuracy of SVM is generally better than that of kNN for k ranging from 2 thru 6. The accuracies vary significantly across the different annotators and across the different categories. The best accuracy was 0.933 (annotator #1, category f) and the worst was 0.3 (annotator #3, category l). Different annotators appear better for different categories in giving easily learnable labels, e.g., annotators #1 and #4 do better on categories c and i

respectively.

4.6 Human Subject Study 2

The second user study is similar to the first user study with the following differences: we generated the significant activations using the MSX method, we created the visualization using the I-GOS algorithm, and we conducted our experiments on both the 12 categories of the Birds dataset and the 5 categories of the CEL dataset. We had the activation maps of different images of 12 categories of Birds dataset, annotated by 10 different subjects using the annotation interface. Also, we had the activation maps of different images of 5 categories of CEL dataset, annotated by 5 different subjects.

4.6.1 RQ1: Coverage of Interactive Naming

Figure 4.16 (top) and Figure 4.16 (bottom) show the mean of the fraction of significant activations that are named by annotators for each category of Birds and CEL dataset, respectively. In addition, the bar labeled “Any Annotator", shows the fraction of significant activations that were assigned to a visual concept by at least one annotator. We see that more than 80% of the signficant activations are named by the annotators on average in the birds dataset and more than 90% are named in all but one class in the CEL dataset. However, in *Hemispherebleb* category of CEL dataset in Figure 4.16 (bottom), in average 40% of the activations were not labeled by the users. We observed that the activations of this category are more difficult to annotate in comparison to the others.

4.6.2 RQ2: X-feature-Visual Concept correspondence

As it may be clear, both CX-purity and XC-purity vary from annotator to annotator and from category to category. Table 4.11 and Table 4.12 show the CX-purity and XC-purity values for each annotator across all categories of Birds dataset, respectively. Moreover, Table 4.13 and Table 4.12 show the CX-purity and XC-purity values for each annotator across all categories of CEL dataset, respectively.

Our first observation is that the standard deviations of all these numbers are low,

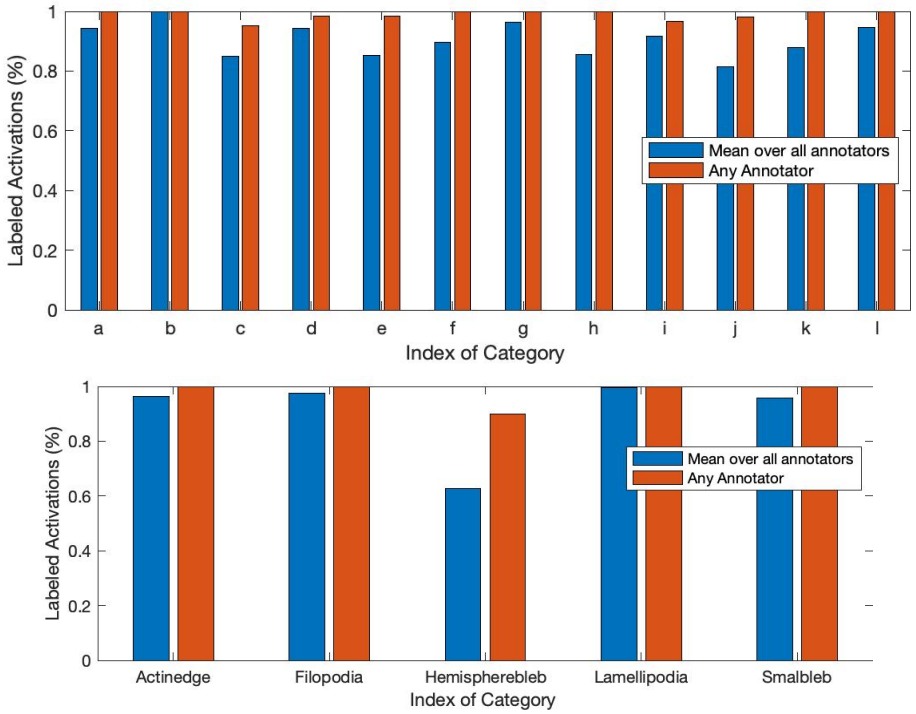


Figure 4.16: Fraction of labeled significant activations for each category of Birds dataset (top) and CEL dataset (bottom).

suggesting inter-annotator consistency. The CX-purity numbers in Table 4.11 are generally high ranging from 0.82 to 1.0 with an average of 0.92, with the exception of Category b where it is 0.61. The XC-purity numbers in Table 4.12 are generally worse than CX-purity, going down to 0.47 in one case (Category k) and a maximum of 0.96, with an average of 0.74. This suggests that the mapping from X-features to visual concepts is more one-to-many than the other way around in Birds dataset. There are two reasons for this. First the Birds dataset only has 5 X-features, which forces each X-feature to represent multiple concepts. Second, in most categories of Birds dataset, the significant activations are mostly covered by a single X-feature (Figure 4.8), which further reduces the number of available significant X-features to cover the visual concepts. Except for the difficult category of $Filopodia$, the CX-purity numbers vary from 0.82 to 0.98 and the XC-purity varies from 0.89 to 0.99 for the remaining categories of CEL dataset with averages of 0.82 and 0.86

respectively (Table 4.13 and Table 4.14).

Table 4.11: CX-purity in Birds dataset: Shows the degree to which the activations of a visual concept map to a single X-feature.

Category	Annotator										
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	average
a	0.83	0.89	0.83	0.83	0.83	0.83	0.89	0.84	0.83	0.82	0.84
b	0.68	0.59	0.59	0.59	0.59	0.7	0.57	0.59	0.59	0.58	0.61
c	0.88	0.96	0.9	0.86	0.89	0.87	0.9	0.88	0.87	0.89	0.89
d	1	1	1	1	1	1	1	1	1	1	1
e	0.95	0.94	0.95	0.96	0.97	0.95	0.95	0.94	0.96	0.96	0.95
f	0.83	0.8	0.81	0.83	0.83	0.84	0.83	0.81	0.83	0.81	0.82
g	1	1	1	1	1	1	1	1	1	1	1
h	0.95	0.89	0.94	0.94	0.94	0.94	0.91	0.93	0.94	0.94	0.93
i	1	1	1	1	1	1	1	1	1	1	1
j	1	1	1	1	1	1	1	1	1	1	1
k	1	1	1	1	1	1	1	1	1	1	1
l	1	1	1	0.98	1	0.98	0.98	0.98	0.98	1	0.99
average	0.93	0.92	0.92	0.92	0.92	0.93	0.92	0.91	0.92	0.92	

Table 4.12: XC-purity in Birds dataset: Shows the degree to which the activations of an X-feature map to a single visual concept.

Category	Annotator										
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	average
a	0.7	0.67	0.64	0.68	0.9	0.7	0.67	0.66	0.68	0.63	0.69
b	0.59	0.55	0.91	0.57	0.91	0.64	0.93	0.91	0.64	0.93	0.76
c	0.87	0.91	0.88	0.76	0.87	0.85	0.87	0.79	0.8	0.87	0.85
d	0.93	0.93	1	0.9	0.88	0.93	0.93	0.95	0.92	0.9	0.93
e	0.67	0.77	0.8	0.79	0.68	0.67	0.89	0.87	0.82	0.81	0.78
f	1	0.9	0.88	0.94	0.83	1	0.88	0.88	0.88	0.91	0.91
g	0.79	0.72	0.83	0.4	0.85	0.83	0.81	0.85	0.8	0.82	0.77
h	0.53	0.52	0.57	0.56	0.56	0.52	0.72	0.7	0.6	0.58	0.58
i	0.55	0.39	0.54	0.58	0.33	0.62	0.39	0.54	0.6	0.57	0.51
j	0.5	0.59	0.89	0.75	0.6	0.5	0.59	0.89	0.89	0.6	0.68
k	0.36	0.44	0.34	0.92	0.34	0.36	0.33	0.38	0.86	0.38	0.47
l	1	1	1	0.93	1	0.94	0.93	0.93	0.91	1	0.96
average	0.71	0.7	0.77	0.73	0.73	0.71	0.74	0.78	0.78	0.75	

Table 4.13: CX-purity in CEL dataset: Shows the degree to which the activations of a visual concept map to a single X-feature.

Category	Annotator					
	#1	#2	#3	#4	#5	average
Actinedge	0.96	1	0.98	0.98	0.97	0.98
Filopodia	0.56	0.55	0.58	0.56	0.55	0.56
Hemispherebleb	0.92	0.79	1	0.84	0.82	0.87
Lamellipodia	0.87	0.87	0.86	0.86	0.86	0.86
Smalbleb	0.92	0.81	0.81	0.82	0.81	0.83
average	0.85	0.80	0.84	0.81	0.80	

Table 4.14: XC-purity in CEL dataset: Shows the degree to which the activations of an X-feature map to a single visual concept.

Category	Annotator					
	#1	#2	#3	#4	#5	average
Actinedge	0.94	0.95	0.95	0.89	0.87	0.92
Filopodia	0.62	0.56	0.74	0.65	0.55	0.62
Hemispherebleb	0.98	1	1	0.78	0.72	0.89
Lamellipodia	0.98	1	1	0.98	0.98	0.99
Smalbleb	0.93	0.96	0.95	0.97	0.97	0.96
Average	0.89	0.89	0.93	0.85	0.82	

4.6.3 RQ3: Inter-annotator Agreement

First, we consider annotator agreement about which significant activations should be named. Figure 4.17 shows, for each bird category, the fraction of significant activations that were named by different numbers of annotators - 0 thru 5. Interestingly, the largest fraction of activations are annotated by all 5 annotators and the second largest are annotated by 0 annotators. This confirms, once again, that for most significant activations, either all annotators choose to assign a name or none of them do. There is strong agreement about the set of activations that should be named.

Next, we characterize the fraction of images annotated by pairs of annotators using the Jaccard index. This is shown in the last column of Table 4.15 averaged over different pairs of annotators. The Jaccard index is fairly high for all categories, indicating that there is a good overlap between the sets of activations chosen by different annotators to annotate.

Next we consider the extent to which the namings of different annotators can be trans-

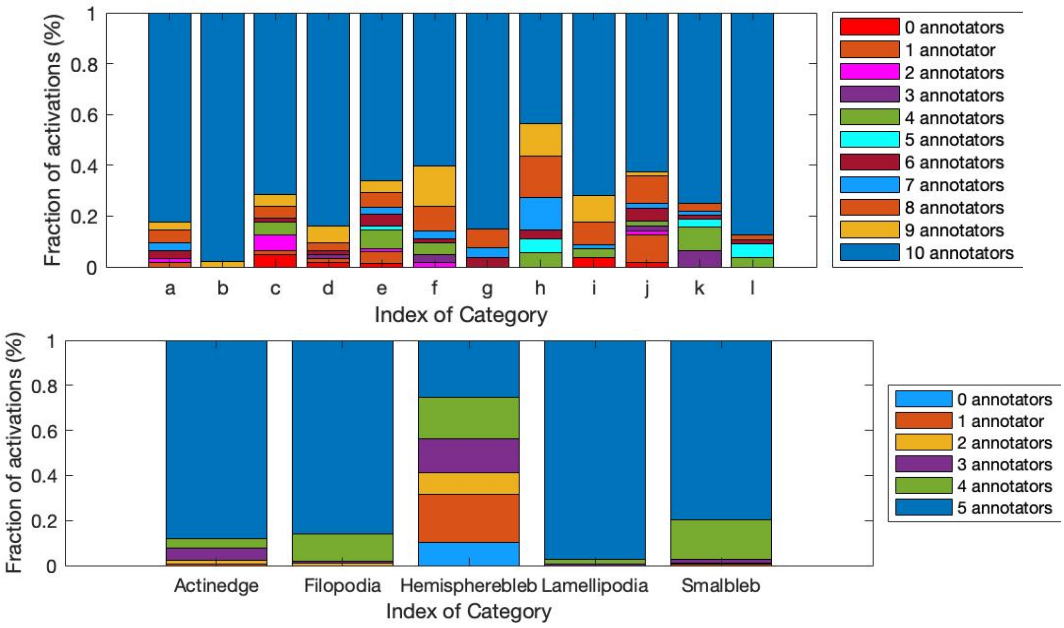


Figure 4.17: Fraction of significant activations that are named by exactly n of the annotators in Birds dataset (top) and CEL dataset (bottom).

lated to one another by looking for one-to-one correspondences, subsumption relationships and cases of purely incompatible concepts?

The columns labeled "Agreement" in Table 4.15 and Tables 4.16 show the statistics of 1-family and 2-family agreements for each category over the set of all annotator pairs for Birds and CEl datasets, respectively. The agreement numbers are fairly high across most categories, although the minimum values for some categories for $D = 1$ are low. Since 2-family matching is more permissive than 1-family matching, the agreement numbers are higher for $D = 2$ as we expect. Even for $D = 1$ the agreement in most categories is reasonably high, which shows that there is reason to be optimistic about developing a common ontology for explanations.

4.6.4 RQ4: Visual Concept Generalization

Finally, we turn to the question of how well the visual concepts of different annotators and categories generalize? To answer this question we ran LibSVM classifiers [Chang and Lin,

Table 4.15: Pairwise comparison between clusters generated by annotators over all categories of Birds dataset

	Pairwise similarity scores		
Category	Agreement (D=1)	Agreement (D=2)	Jaccard index
a	0.83±0.12	0.95±0.03	0.93±0.04
b	0.73±0.17	0.95±0.1	0.96±0.04
c	0.95±0.04	0.98±0.02	0.89±0.05
d	0.96±0.02	0.98±0.02	0.95±0.03
e	0.88±0.07	0.94±0.04	0.86±0.06
f	0.9±0.05	0.97±0.03	0.86±0.09
g	0.8±0.16	0.94±0.03	0.92±0.05
h	0.9±0.09	0.97±0.03	0.77±0.16
i	0.79±0.13	0.91±0.05	0.92±0.05
j	0.8±0.13	0.99±0.02	0.86±0.07
k	0.71±0.24	0.92±0.06	0.87±0.07
l	0.99±0.01	1±0.01	0.93±0.04
Global Average	0.85	0.957	0.8924

Table 4.16: Pairwise comparison between clusters generated by annotators over all categories of CEL dataset.

	Pairwise similarity scores		
Category	Agreement (D=1)	Agreement (D=2)	Jaccard index
Actinedge	0.91±0.03	0.96±0.02	0.94±0.02
Filopodia	0.63±0.1	0.75±0.07	0.94±0.03
Hemispherebleb	0.89±0.08	0.99±0.04	0.58±0.07
Lamellipodia	0.98±0.01	0.99±0.01	0.99±0.01
Smalbleb	0.96±0.02	0.98±0.01	0.91±0.08
Global Average	0.874	0.934	0.872

2011] on the examples of labeled significant activation maps. As in the previous user study, we developed the *I-feature map* for each X-feature activation. This involved first blurring the original image and adding the region of the original image that has been highlighted by the X-feature activation. We then gave this masked image to the learned DNN and xNN and saved the values of the penultimate layer of the networks as the I-feature map.

To deal with the unbalanced classes, we applied weighted SVM with RBF Kernel classification algorithm to learn the visual concepts. The results of 5-fold cross-validation for the two datasets based on the namings by 5 annotators each are shown in Table 4.17 (top)

and Table 4.17 (bottom). The classification accuracies into visual concepts is reasonably high in the Birds dataset given that the number of concepts is about 3-4 per category. With the exception of the difficult category of *Filopodia*, the accuracies are higher in CEL dataset as it contains more images and approximately 2 concepts per class on average. In some categories of the CEL dataset, some users just created a single concept (e.g., *Filopodia* and Annotator #3) which made the task trivial.

Table 4.17: SVM accuracy result for each category of Birds dataset (top) and CEL dataset (bottom) cross 5 annotaors.

	Index of bird category											
User	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*
#1	0.77	0.571	0.94	0.96	0.66	0.96	0.8	0.581	0.66	0.62	0.42	0.94
#2	0.84	0.7	0.93	0.925	0.78	0.93	0.82	0.36	0.44	0.714	0.46	0.96
#3	0.65	0.99	0.89	0.84	0.85	0.78	0.82	0.6	0.58	0.93	0.5	0.96
#4	0.76	0.625	0.8	0.95	0.854	0.96	0.49	0.6	0.46	0.81	0.95	0.90
#5	0.847	0.99	0.99	0.8	0.69	0.69	0.84	0.58	0.76	0.75	0.65	0.96

	Annotator				
Category	#1	#2	#3	#4	#5
Actinedge	0.94	0.94	0.936	0.88	0.847
Filopodia	0.6	0.65	1	0.649	0.548
Hemispherebleb	0.967	1	1	0.764	0.6727
Lamellipodia	0.976	1	1	0.979	0.97
Smalbleb	0.92	0.96	0.953	0.973	0.974

4.7 Discussion and Conclusions

In this chapter we studied the problem of understanding the decisions of DNNs in terms of human-recognizable visual concepts. Our interactive-naming approach involved augmenting the original DNN with a sparser xNN, visualizing the significant activation maps for each decision of the xNN on a test set, and then allowing annotators to flexibly group the activations into recognizable visual concepts, while attaching names to the concepts if desired. The visual concepts can then be used as the basis for producing concise meaningful

explanations for test set images. Our interactive naming interface allowed annotators to flexibly produce and name the clusterings, which allowed for semantically meaningful explanations to be computed for each test image.

One of the motivations for naming a test set is to produce summaries of the explanation types used to predict test images.

We reported on our experience of having annotators use our interface for DNNs trained to recognize different bird species and cell types. Our results showed the following.

1. In most cases, the xNN is able to localize the signficant explanations to a single feature, We also introduced metrics for measuring the feature-concept correspondence and used them to characterize the nature of mappings between X-features and visual concepts;

2. The annotators are able to assign names to a very high fraction of significant activations, which allows for at least partial semantic explanations for most test images. We defined the metrics of full and partial explanation coverage, and used them to characterize the coverage of examples by human-recognizable concepts.

3. The annotators had strong agreement about which significant activations should and should not be named. In addition, there was a non-trivial amount of agreement between the namings produced by different annotators, as indicated by the automatically produced matchings, or translations, between different namings and their consistency with the linguistic names assigned to clusters by the annotators.

4. It is possible to automatically learn the visual concepts from a modest amount of annotated data using the significant activations. Particularly, when the size of the clustered significant activations for each visual concept is large, learning visual concepts would be easier with the classifiers.

5. Comparing the two user-studies, one can conclude that applying MSX instead of 90% impact in classification method will provide less significant activations. However, using MSX there will be fewer un-explained significant activations. Moreover, since in user study 2 the results were mostly better than user study 1, one can

conclude that the techniques that we provided for selecting significant activations, creating I-features, and providing visualization are more useful than the corresponding techniques that we used in user study 1. Recall that we used I-GOS instead of ExcitationBP for generating visualization, blurring technique instead of in-painintg for providing I-features, and MSX instead of 90% impact in classification method for providing significant activations in user study 2.

This formative study has set the stage for a variety of future work. Our current interactive naming interface is flexible, but does not attempt to actively reduce the annotator effort. Thus, there is potential to speed up the naming process via active learning techniques. It would also be interesting to interactively train the system based on named concepts, which might further improve the alignment between the human concepts and the X-features. Investigations on larger varieties of visual concepts is important for understanding the general characteristics of annotator produced namings.

Chapter 5: Conclusions and Future Work

This chapter summarizes the contributions of the thesis and makes some suggestions for future research.

5.1 Summary of the Contributions

The goal of the thesis was to study and develop solutions for three problems in active learning from examples, queries and explanations. The contributions of the dissertation can be summarized as follows:

- Chapter 2 developed a framework for active learning of hierarchical policies in the form of PSDGs from trajectories generated by a hierarchical policy. This framework contains a novel two-level active learning algorithm, where the top level selects a trajectory and the lower level actively queries the teacher about the intentional structure at selected points along the trajectory. It developed a new information-theoretically justified heuristic, called *cost-normalized information*, for selecting trajectories, and employed Bayesian active learning for the lower-level query selection. Experimental results on five benchmark problems indicate that our cost-normalized information heuristic compares favorably to a number of baselines for learning hierarchical policies in a query-efficient manner.

- Chapter 3 extended the framework of adapative submodular optimization to a setting where the available queries are randomly constrained and gave a simple greedy algorithm with a near-optimal performance bound. It applied the new framework to the problem of active multi-label learning, where each expert can only label a subset of labels and only some experts are available at any time. Experimental results on six benchmark of multi-label problems showed promising results based on the Gibbs error heuristic.

- Chapter 4 presented an interactive naming approach for the problem of understanding the decisions of DNNs in terms of human-recognizable visual concepts. This approach involved augmenting the original DNN with a sparser xNN, visualizing the significant activation maps for each decision of the xNN on a test set, and then allowing annotators to flexibly group the activations into recognizable visual concepts, while attaching names to the concepts if desired. The visual concepts can then be used as the basis for producing concise meaningful explanations for test set images. It explored six research questions via two user studies, where annotators used the interactive naming interface for DNNs trained to recognize different bird species and cell types. Experimental results showed that the annotators were able to cluster a large fraction of the significant activations. Moreover, there was a significant agreement between the namings produced by different annotators. It was possible to automatically learn the visual concepts from a modest amount of annotated data, given the annotated clusters of visual concepts generated by the user.

  Finally, comparing two different user studies, we conclude that applying I-GOS instead of ExcitationBP as a visualization technique, MSX instead of 90% impact on classification method as a significant activation selection, and blurring instead masking for I-feature creation, would provide more promising results.

5.2 Future work

In this section, we summarize some of the main future research directions identified in different chapters.

5.2.1 Richer forms of demonstrations for active imitation learning

One direction for future research is to allow richer and more natural forms of demonstrations from the human experts. In the second chapter of this thesis, where the goal is to learn the hierarchical policies from demonstrations, we assumed that the trajectories contain the termination symbol of a sub-task that just terminated. A future direction of this research is to solve the problem without having these termination symbols in the trajectories. Another

way to have a natural form of demonstrations is to support very long trajectories. To achieve this, one solution would be to apply more efficient and generalized parsing algorithms than the CKY algorithm.

5.2.2 Handling noisy and sub-optimal humans

In all chapters of this thesis, we assumed that the machine interacts with a human expert. However, in real-world applications, it is not always easy to find an expert or an optimal teacher to annotate the data, while it is easy to find several sub-optimal experts. Thus, one future direction is to solve the problem of noisy and sub-optimal annotators or teachers.

5.2.3 Implementation on real robots

In most of the robotics applications, which apply hierarchical task structures for policies, it is assumed that the task hierarchy is given as an input to the system. It would be interesting to apply the active imitation learning of hierarchical policies algorithm on a real robot.

5.2.4 Further generalizations of adaptive submodularity

In the third chapter of this thesis on adaptive submodularity, we assumed that the cost of labeling of each expert is the same, and the expert's label size is the same for all experts. One direction of further research is to allow a more natural setting where there are a lot of experts with different skills and labeling costs, for example, by allowing the sizes of the label sets that each expert can annotate to be different.

5.2.5 Active example selection for interactive naming

Our current interactive naming interface is flexible but does not attempt to actively reduce the annotator effort by actively selecting examples. Thus, there is a potential to speed up the naming process via active learning techniques.

5.2.6 Re-train xNNs to align with visual concepts

A potential future work for interactive naming is to re-train the xNN to align with visual concepts. Given the visual concepts generated by a human using the interactive naming, the goal is to re-train the xNN so that its output will be based on the visual concepts. There are a number of challenges here. For examle, how to generate the negative and positive examples for re-training the xNN? How to define the cost function for re-training the xNN? How to design the structure of the neural network? We expect that future work will build on the current thesis to explore these questions in more depth.

Bibliography

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS One*, 10, 7 2015. ISSN 1932-6203. doi: 10.1371/journal.pone.0130140.

J. Baldridge and M. Osborne. Active learning for hpsg parse selection. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 17–24, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Computer Vision and Pattern Recognition*, 2017.

H. Blockeel and L. De Raedt. Top-down induction of first-order logical decision trees. *Artif. Intell.*, 101(1-2):285–297, May 1998. ISSN 0004-3702.

K. Brinker. On active learning in multi-label classification. In M. Spiliopoulou, R. Kruse, C. Borgelt, A. NÃijrnberger, and W. Gaul, editors, *In From Data and Information Analysis to Knowledge Engineering*, pages 206–213. Springer Berlin Heidelberg, 2005.

R W Byrne and A E Russon. Learning by imitation: a hierarchical approach. *Behavioral and Brain Sciences*, 21:667–84; discussion 684–721, 1998. URL `http://www.ncbi.nlm.nih.gov/pubmed/10097023`.

Frédéric Cazals, Dorian Mazauric, Romain Tetley, and Rémi Watrigant. Comparing two clusterings using matchings between clusters of clusters. Research Report RR-9063, INRIA Sophia Antipolis - Méditerranée ; Universite Cote d'Azur, April 2017. URL `https://hal.inria.fr/hal-01514872`.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Y. Chen and A. Krause. Near-optimal batch mode active learning and adaptive submodular optimization. In Sanjoy Dasgupta and David Mcallester, editors, *Proceedings of the 30th*

International Conference on Machine Learning (ICML-13), volume 28, pages 160–168. JMLR Workshop and Conference Proceedings, 2013.

N. V. Cuong. *Near-optimality and robustness of greedy algorithms for Bayesian pool-based active learning*. PhD dissertation, National University of Singapore, 2015.

N. V. Cuong, W. Sun Lee, N. Ye, K. Ming Adam Chai, and H. L. Chieu. Active learning for probabilistic hypotheses using the maximum gibbs error criterion. In *Advances in Neural Information Processing Systems.*, pages 1457–1465, 2013.

N. V. Cuong, W. Sun Lee, and N. Ye. Near-optimal adaptive pool-based active learning with general loss. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 122–131, 2014.

Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. In *NIPS*, 2017.

I. Dagan and S. P. Engelson. Committee-based sampling for training probabilistic classifiers. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 150–157. Morgan Kaufmann, 1995.

Sanjoy Dasgupta. Analysis of a greedy active learning strategy. In Lawrence K. Saul, Yair Weiss, and LÃl'on Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 337–344. MIT Press, Cambridge, MA, 2004.

Thomas G. Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *J. Artif. Intell. Res. (JAIR)*, 13:227–303, 2000.

C. Z. Eddy, X. Wang, F. Li, and B. Sun. The morphodynamics of 3d migrating cancer cells. *arXiv:1807.10822*, 2018.

Martin Erwig, Alan Fern, Magesh Murali, and Anurag Koul. Explaining deep adaptive programs via reward decomposition. *IJCAI/ECAI Workshop on Explainable Artificial Intelligence*, Jan 2018. URL `http://par.nsf.gov/biblio/10096985`.

R. C. Fong and A. Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3449–3457, 2017.

D. Golovin and A. Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research (JAIR)*, 42:427–486, 2011.

Daniel Golovin, Andreas Krause, and Debajyoti Ray. Near-optimal bayesian active learning with noisy observations. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 766–774, 2010.

Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. Generating visual explanations. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, 2016.

B. Hengst. Discovering hierarchy in reinforcement learning with hexq. In *Maching Learning: Proceedings of the Nineteenth International Conference on Machine Learning*, pages 243–250. Morgan Kaufmann, 2002.

G. E. Hinton, J. L. McClelland, and D. E. Rumelhart. Distributed representations. In D. E. Rumeihart, J. L. McClelland, and the PDP Research Group, editors, *Parallel distributed processing: Explorations in the microstructure of cognition: Vol. 1. Foundations*, pages 77âĂŞ–109. MIT Press, Cambridge, MA, 1986.

C. Hogg, U. Kuter, and H. Munoz-Avila. Learning hierarchical task networks for nondeterministic planning domains. In *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence*, pages 1708–1714. AAAI Press, 2009.

S.J. Huang and Z.H. Zhou. Active query driven by uncertainty and diversity for incremental multi-label learning. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, pages 1079–1084, 2013.

S.J. Huang, R. Jin, and Z.H. Zhou. Active learning by querying informative and representative examples. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36 (10):1936–1949, 2014.

S.J. Huang, S. Chen, and Z.H. Zhou. Multi-label active learning: Query type matters. In Qiang Yang and Michael Wooldridge, editors, *IJCAI*, pages 946–952. AAAI Press, 2015.

R. Hwa. Sample selection for statistical parsing. *Computational Linguistics*, 30:253–276, 2004.

A. Jonsson and A.G. Barto. Causal graph based decomposition of factored mdps. *Journal of Machine Learning Research*, 7:2259–2301, 2006.

Omar Zia Khan, Pascal Poupart, and James P. Black. Minimal sufficient explanations for factored markov decision processes. In *In Proceedings of the 19th International Conference on Automated Planning and Scheduling (ICAPSâĂŹ09*, 2009.

Roni Khardon. Learning to take actions. *Mach. Learn.*, 35(1):57–90, April 1999. ISSN 0885-6125. doi: 10.1023/A:1007571119753. URL http://dx.doi.org/10.1023/A:1007571119753.

P.-J. Kindermans, K. T. Schütt, M. Alber, K.-R. Müller, D. Erhan, B. Kim, and S. Dähne. Learning how to explain neural networks: PatternNet and PatternAttribution. *ArXiv e-prints*, 2017.

E. Koechlin and T. Jubault. Broca's area and the hierarchical organization of human behavior. *Neuron*, 50(6):963–974, June 2006.

H. W. Kuhn and Bryn Yaw. The hungarian method for the assignment problem. *Naval Res. Logist. Quart*, pages 83–97, 1955.

Todd Kulesza, Saleema Amershi, Rich Caruana, Danyel Fisher, and Denis Charles. Structured labeling for facilitating concept evolution in machine learning. ACM, May 2014.

K. Lari and S.J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56, 1990.

Benjamin J. Lengerich, Sandeep Konam, Eric P. Xing, Stephanie Rosenthal, and Manuela M. Veloso. Towards visual explanations for convolutional neural networks via input resampling. In *ICML Workshop on Visualization for Deep Learning*, 2017.

S.Y. Li, Y. Jiang, and Z.H. Zhou. Multi-label active learning from crowds. *CoRR*, abs/1508.00722, 2015.

X. Li and Y. Guo. Active learning with multi-label SVM classification. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, 2013a.

X. Li, L. Wang, and E. Sung. Multilabel svm active learning for image classification. In *2004 International Conference on Image Processing (ICIP '04)*, volume 4, pages 2207–2210, 2004.

Xin Li and Yuhong Guo. Active learning with multi-label svm classiiňĄcation, 2013b. URL https://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6509.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. 2017.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008. ISBN 0521865719, 9780521865715.

A. Mcgovern and A.G. Barto. Automatic discovery of subgoals in reinforcement learning using diverse density. In *Proceedings of the eighteenth international conference on machine learning*, pages 361–368. Morgan Kaufmann, 2001.

Neville Mehta, Soumya Ray, Prasad Tadepalli, and Thomas G. Dietterich. Automatic discovery and transfer of MAXQ hierarchies. In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, pages 648–655, 2008.

S. Natarajan, S. Joshi, P. Tadepalli, K. Kersting, and J. Shavlik. Imitation learning in relational domains: a functional-gradient boosting approach. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1414–1420. AAAI Press, 2011. ISBN 978-1-57735-514-4.

D. Nau, T.-C. Au, O. Ilghami, U. Kuter, W. Murdock, D. Wu, and F.Yaman. SHOP2: An HTN planning system. *J. Artif. Intell. Res. (JAIR)*, 20:379–404, 2003.

N. Nejati, P. Langley, and T. Konik. Learning hierarchical task networks by observation. In *Proceedings of the Twenty-Third International Conference on Machine Learning*, pages 665–672. ACM Press, 2006.

M. L. Puterman. *Markov Decision Processes*. Wiley, 1994.

David V. Pynadath and Michael P. Wellman. Probabilistic state-dependent grammars for plan recognition. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, UAI'00, pages 507–514, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1-55860-709-9. URL `http://dl.acm.org/citation.cfm?id=2073946.2074005`.

G.J. Qi, X.S. Hua, Y. Rui, J. Tang, and H.-J. Zhang. Two-dimensional multilabel active learning with an efficient online adaptation model for image classification. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(10):1880–1897, 2009.

Zhongang Qi and Fuxin Li. Embedding deep networks into visual explanations. *CoRR*, abs/1709.05360, 2017. URL `http://arxiv.org/abs/1709.05360`.

Zhongang Qi, Saeed Khorram, and Fuxin Li. Embedding deep networks into visual explanations. *CoRR*, abs/1709.05360, 2017.

Zhongang Qi, Saeed Khorram, and Fuxin Li. Visualizing deep networks by optimizing with integrated gradients. *CoRR*, abs/1905.00954, 2019. URL http://arxiv.org/abs/1905.00954.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1135–1144, New York, NY, USA, 2016. ACM.

Advait Sarkar, Cecily Morrison, Jonas F. Dorn, Rishi Bedi, Saskia Steinheimer, Jacques Boisvert, Jessica Burggraaff, Marcus D'Souza, Peter Kontschieder, Samuel Rota Bulò, Lorcan Walsh, Christian P. Kamm, Yordan Zaykov, Abigail Sellen, and Siân Lindley. Setwise comparison: Consistent, scalable, continuum labels for computer vision. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 261–271, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-3362-7. doi: 10.1145/2858036.2858199.

R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017.

Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 1070–1079, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1613715.1613855.

Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. Not just a black box: Learning important features through propagating activation differences. *CoRR*, abs/1605.01713, 2016. URL http://arxiv.org/abs/1605.01713.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *ICLR Workshop*, 2014.

M. Singh, E. Curran, and P. Cunningham. Active learning for multi-label image annotation. Technical report, 2009.

J.T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. In *ICLR Workshop*, 2015. URL http://lmb.informatik.uni-freiburg.de/Publications/2015/DB15a.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, pages 3319–3328. PMLR, 2017.

Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res.*, 2:45–66, March 2002. ISSN 1532-4435. doi: 10.1162/153244302760185243. URL `http://dx.doi.org/10.1162/153244302760185243`.

Kurt VanLehn. Learning one subprocedure per lesson. *Artif. Intell.*, 31(1):1–40, 1987.

D. Vasisht, A. Damianou, M. Varma, and A. Kapoor. Active learning for sparse bayesian multilabel classification. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 472–481. ACM âĂŞ Association for Computing Machinery, 2014.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

A. Whiten, E. Flynn, K. Brown, and T. Lee. Imitation of hierarchical action structure by young children. *Dev Sci*, 9(6):574–82, 2006.

J. Wu, V.S. Sheng, J. Zhang, P. Zhao, and Z. Cui. Multi-label active learning for image classification. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 5227–5231, 2014.

B. Yang, J. Sun, T. Wang, and Z. Chen. Effective multi-label active learning for text classification. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 917–926, New York, NY, USA, 2009. ACM.

C. Ye, J. Wu, V. S. Sheng, S. Zhao, P. Zhao, and Z. Cui. Multi-label active learning with chi-square statistics for image classification. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pages 583–586, 2015.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 818–833, Cham, 2014. Springer International Publishing.

B. Zhang, Y. Wang, and W. Wang. Batch mode active learning for multi-label image classification with informative label correlation mining. In *IEEE Workshop on Applications of Computer Vision, WACV 2012, Breckenridge, 2012*, pages 401–407, 2012.

Jianming Zhang, Zhe Lin, Jonathan Brandt, Xiaohui Shen, and Stan Sclaroff. Top-down neural attention by excitation backprop. In *European Conference on Computer Vision*, pages 543–559. Springer, 2016.

Ning Zhang, Ryan Farrell, Forrest Iandola, and Trevor Darrell. Deformable part descriptors for fine-grained recognition and attribute prediction. In *Proceedings of the 2013 IEEE International Conference on Computer Vision*, ICCV '13, pages 729–736, Washington, DC, USA, 2013. IEEE Computer Society. ISBN 978-1-4799-2840-8.

Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Interpretable basis decomposition for visual explanation. In *Computer Vision – ECCV 2018*, 2018.

APPENDICES

This appendix is inoperable.