Detection and Identification of Rare Audiovisual Cues

Insperata accident magis saepe quam quae speres.
(Things you do not expect happen more often than
things you do expect) Plautus (ca 200(B.C.))

Project no: 027787

DIRAC

Detection and Identification of Rare Audio-visual Cues

Integrated Project
IST - Priority 2

DELIVERABLE NO: D4.15

MODELING CORTICAL CELLS WITH DISTANCE FUNCTIONS

Date of deliverable: 30.6.2010
Actual submission date: 12.8.2010

Start date of project: 01.01.2006 Duration: 60 months

Organization name of lead contractor for this deliverable: ***HUJI***

Revision [0]

Project co-funded by the European Commission within the Sixth Framework Program (2002-2006)		
Dissemination Level		
PU	Public	
PP	Restricted to other program participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	X
CO	Confidential, only for members of the consortium (including the Commission Services)	

D4.15 - Building object hierarchies for knowledge transfer

THE HEBREW UNIVERSITY OF JERUSALEM (HUJI)

Abstract:

In this work we propose a hierarchical model of tasks. The model captures both part-membership and class-membership hierarchies. We provide an algorithm for discovering hierarchical structure in data. We extend the notion of task relatedness in the context of multi-task learning to an hierarchical task relatedness approach, based on hierarchy induced invariances. Based on the hierarchical multi-task framework we analyze under what conditions learning a single task can benefit from multiple tasks organized in an hierarchical structure.

Insperata accident magis saepe quam quae speres.
(Things you do not expect happen more often than things you do expect) Plautus (ca 200(B.C.))

Table of Content:

1. Introduction

We present a general hierarchical model of tasks. Providing an algorithm for inferring the model from data and an analysis of the conditions under-which such a model can be beneficial when learning the tasks. A task is considered as a probabilistic binary mapping of an input space. Categorization tasks, classifying if a sample does or does not belong to a category can be viewed as a specific type of a task. Our model and structure inferring algorithm assume a binary set representation of tasks, where each task is represented by a specific set of properties defining the task. The theoretical analysis is not limited to the binary set representation.

Hierarchical approaches are common in various models dealing with perception both in biological/phycological research of human perception and in artificial intelligence domains such as machine learning and computer vision. It has been argued that humans rely on a hierarchal representation of objects in the world, in the process of recognizing and referring to objects. Various studies have shown different aspects of hierarchal organization of object categories in humans. A well known example of such a study of the nature of human hierarchical representation is Rosch's Basic-Level theory [17]. A different line of work, conducted recently by Kiani et al. [14] shows neurophysiological findings of visual object hierarchy. In the field of machine learning general hierarchical approaches such as CART [7] and hLDA [6] have been proposed for learning and modeling hierarchical relations in data. The use of hierarchy has proven to be beneficial in increasing classification performance for tasks such as visual object categories [2, 16, 20]. Furthermore, explicitly modeling hierarchical relations among categories achieves a richer more informative representation which can enable different responses for different tasks given the same sample, e.g. a Dog, is not only a Dog, it is a specific type of a Dog such as a Collie-Dog, it is also an Animal and can also be considered as Food. On the one hand hierarchies enable richer responses to what is known, e.g. from a learning point of view the hierarchical information can be used to calibrate the loss function [9], on the other hand the hierarchical structure enables incongruence detections which enable dealing with the unknown, as presented in [19].

The proposed representation presented in section 2 consists of a set of properties for which it is possible to measure the existence of the property in a data sample (e.g. image). This representation can be regarded as a generalization of many existing representation approaches. For example, in the following we focus on computer vision- object-class/image representation approaches. The bag-of-words model [18, 8] is a set of feature space clusters where for each cluster it is possible to measure if local features sampled from an image fall into the cluster or rather how many such features fall into the cluster. Spatial models such as part based star shape models [1, 11] can be viewed as sets of properties were each property is an appearance (region in feature space) of a part at a specific location. Attribute representation such as [15] where images are characterized by a set of supervised high level attributes such as "striped" or "yellow" for which their appearance can be learnt and applied to testing images by specifying whether the attribute exists or not. Each such attribute after the learning phase can be regarded as a measurable property given the learnt model. Part-Hierarchy models such as [12, 10] can be regarded as sets where parts at higher levels of the hierarchy are co-occurring lower level properties at relative proximate regions in images from a training dataset. The set of properties is not restricted to the visual domain, for example cues in the text of an image caption [5] or audio cues simultaneously heard while viewing an image [13] can also be formulated as properties representing an object class.

The set of properties representation enables the combination of many properties from different domains in a single representation: local appearance, global appearance, spatial relation, 3D cues, scene cues, different modalities, object class classifiers stating if an object exists in an image, supervised attributes and many others, all can be formalized as measurable properties and considered together in the set of properties representation. Obviously the number of properties can be infinite but as evident in current research, even a selected sample of such properties can yield relatively good results [18, 8, 15, 1, 11].

The basic partial order representation presented in 2 will be defined using properties for which it is possible to measure whether they exist or not in a data sample. One of the merits of the model in 2 is the dynamic nature in which adding properties as well as adding well defined tasks is straight forward and does not require recalculation of known data, opposed to a model such as SVM where in the presence of a new feature the model should be retrained.

When considering categorization tasks, the model presented in 2 captures both the part-membership and class-membership hierarchies [19] in a single hierarchy. Given a representation of an object class as a set of measurable properties, a grouping of several classes into a single more abstract class defines a co-occurring set of properties shared by these classes. On the other hand a set of co-occurring properties defines a grouping of one or more classes sharing these properties. The connection between these two concepts is achieved by the notion of partial order. A hierarchy of classes defines a partial order among the classes, while inclusion relations between sets of properties define partial order over these sets.

By modeling a general level task using the co-occuring set of properties all more specific level tasks belonging to the same general level share these properties, hence are invariant with respect to these properties. In 5 we give a precise definition of invariance. Using the notion of invariance we will extend the multi-task learning framework based on relatedness [4] to an hierarchical multi-task learning framework and analyze under what conditions such a framework can be beneficial. In the context of multi-task learning our contribution goes beyond the hierarchical extension of the framework in [4] the model of the hierarchy and algorithm for building such an hierarchy can be considered as a means of discovering related tasks automatically; a crucial step in multi-task learning in cases where the tasks beneficial for transfer are not known a-priori.

In Section 2 we will present the unifying partial order graph representation and in Section 3 we will describe the algorithm discovering the hierarchical representation based on given data. In Section 4 we discuss a statistical extension of the model- from the case where each task is represented by a set of properties to the case where each task is represented by a set of samples and each sample is represented by a set of properties. This scenario fits the standard supervised approach in learning tasks. In section 5 we will present the hierarchical approach to multi-task learning.

2. Partial Order Representation

Our hierarchical model is a partial order model capturing both the part-membership and class-membership hierarchies [19]. In the following we regard the part-membership hierarchy as a property co-occurrences hierarchy. We view the task of finding an object class hierarchy as finding commonalities among object classes, see section 3. Finding such commonalities can be thought of as computing the intersection of the sets of properties representing each class. The set of properties representation is a general representation which can be reduced to many of

the recent object class representation approaches. This line of thought lead us to propose the following dual representation model:

Given a finite set of properties $\mathbf{P} = \{\mathbf{a}, \mathbf{b}, \mathbf{d}, \mathbf{e}...\}$ and a matching set of boolean functions (predicates) $\theta = \{\psi_{\mathbf{x}} | \mathbf{x} \in \mathbf{P}\}$ each denoting if a given sample contains a property, thus: $\psi_{\mathbf{a}}(\mathbf{s}) = 1$ iff the property $\mathbf{a}$ is present in $\mathbf{s}$. We represent a single class by the set of properties $\mathbf{P_c}$ each instance in this class contains, thus $\mathbf{s} \in \mathbf{C}$ iff $\psi_{\mathbf{x}}(\mathbf{s}) = 1\ \forall \mathbf{x} \in \mathbf{P_c}$. This can also be expressed as a boolean function which is a conjunction of all property functions, thus $\mathbf{s} \in \mathbf{C}$ iff $\psi^{\mathbf{c}}(\mathbf{s}) = 1$ where $\psi^{\mathbf{c}}(\mathbf{s}) = \bigwedge_{\mathbf{x} \in \mathbf{P_c}} \psi_{\mathbf{x}}(\mathbf{s})$; we call $\psi^{\mathbf{c}}(\mathbf{s})$ the membership function of class $\mathbf{C}$.

In the class-membership hierarchy we say that $\mathbf{B} \succeq \mathbf{A}$ iff $\forall \mathbf{s} \in \mathbf{A} \Rightarrow \mathbf{s} \in \mathbf{B}$ in the part-membership hierarchy, or equivalently we say $\mathbf{B} \succeq \mathbf{A}$ iff $\forall \mathbf{x} \in \mathbf{P_B} \Rightarrow \mathbf{x} \in \mathbf{P_A}$. Given our representation of classes as set of properties one can easily see that defining the Specific level class representation as the union of set of properties of all its General level classes is equivalent to setting the samples belonging to a Specific level class to the intersection of the set of samples belonging to all of its general level classes. This results in a partial order which is consistent with both class-membership and part-membership hierarchies.

We shall note that by representing a specific class $\mathbf{c}$ as the union of the set of properties of its more general classes $\mathbf{P}a_{\mathbf{c}}$ we get that $\psi^{\mathbf{c}}(\mathbf{s}) = \bigwedge_{\mathbf{p} \in \mathbf{P}a_{\mathbf{c}}} \bigwedge_{\mathbf{x} \in \mathbf{p}} \psi_{\mathbf{x}}(\mathbf{s})$, thus the membership function of class $\mathbf{c}$ is the conjunction of the membership functions of classes $\mathbf{P}a_{\mathbf{c}}$. This fits our intuitive example of part-membership hierarchy from above - if something has a leg, tail and head (conjunctions) then it is a dog, but just knowing that there is a leg does not imply that there is a dog. Thus the conjunction of the part detections is more specific then each separate part detection. Even more, given a set of specific classes which we know share a common general class, we can represent the general class using the intersection of property sets of all its specific classes, thus we can compute the membership function. On the other hand knowing which samples belong to all its specific classes we cannot deduce all samples belonging to the general class as the (from our definition) union of all the samples belonging to the specific classes is merely contained in the set of samples belonging to the general class. This point is critical for later use of novel class detection. Thus the equality in $DOG = AFGHAN \cup BEAGEL \cup COLLIE$ holds only in a fully observable world assumption where we know all categories.

Based on the set of properties representation of a class and the definition above for specific and general levels we represent the partial order using a graph $\mathbf{G} = < \mathbf{V}, \mathbf{E} >$. Each node $\mathbf{v} \in \mathbf{V}$ represents a class and is associated with two sets: $\mathbf{R_v}$ - the property set which is the class representation, and $\mathbf{I_v}$ - all instances belonging to the class represented by node $\mathbf{v}$. Each edge $(\mathbf{v}, \mathbf{u}) \in \mathbf{E}$ represents that v implies u, thus $\mathbf{u} \succeq \mathbf{v}$. Let $\mathbf{C}h_v$ denote the set of children of node $\mathbf{v}$ we define:

1. $\mathbf{R_v} = \cup_{\mathbf{c} \in \mathbf{C}h_v} \mathbf{R_c}$
2. $\mathbf{I_v} = \cap_{\mathbf{c} \in \mathbf{C}h_v} \mathbf{I_c}$

In case $\mathbf{C}h_v = \emptyset$:

1. $\mathbf{R_v} = \{\mathbf{x}\}$, where $\mathbf{x}$ is a single property from the property set.
2. $\mathbf{I_v} = \{\mathbf{s} | \psi_{\mathbf{x}}(\mathbf{s}) = 1\}$

Note, that as the representation of a node is based on the union of the representation of its children we get that for each property used to represent at least one of the classes there has to be a single node represented by this property alone. Let's denote the special set of nodes for which $\mathbf{C}h_v = \emptyset$ as $\mathbf{L}_1$ and the set of nodes representing all known classes we call $\mathbf{L}_2$. We note that $\mathbf{L}_1 \cup \mathbf{L}_2 \subseteq \mathbf{V}$.

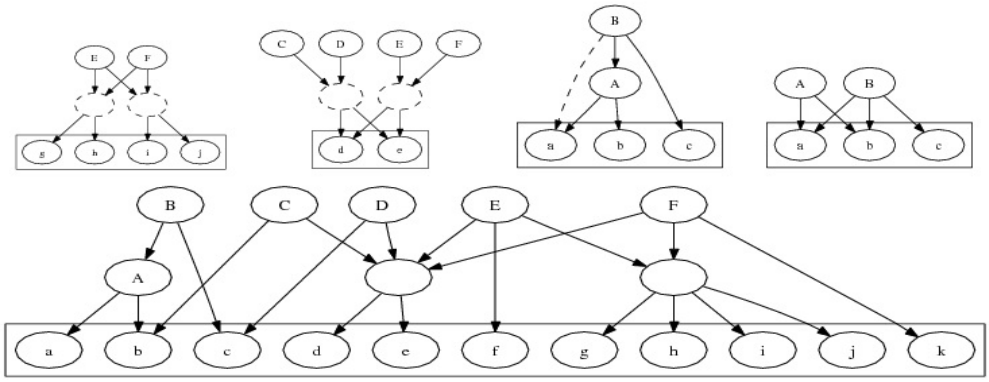

Figure 1. Possible graphical structures for a given set of properties: $\mathbf{P} = \{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{f}, \mathbf{g}, \mathbf{h}, \mathbf{i}, \mathbf{j}, \mathbf{k}\}$, and a set of labeled classes $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{E}, \mathbf{F}\}$. Each class has the following sub set of properties: $\mathbf{P_A} = \{\mathbf{a}, \mathbf{b}\}$, $\mathbf{P_B} = \{\mathbf{a}, \mathbf{b}, \mathbf{c}\}$, $\mathbf{P_C} = \{\mathbf{b}, \mathbf{d}, \mathbf{e}\}$, $\mathbf{P_D} = \{\mathbf{c}, \mathbf{d}, \mathbf{e}\}$, $\mathbf{P_E} = \{\mathbf{d}, \mathbf{e}, \mathbf{f}, \mathbf{g}, \mathbf{h}, \mathbf{i}, \mathbf{j}\}$ and $\mathbf{P_F} = \{\mathbf{d}, \mathbf{e}, \mathbf{g}, \mathbf{h}, \mathbf{i}, \mathbf{j}, \mathbf{k}\}$. $\mathbf{L}_1$ nodes are labeled with lower case. $\mathbf{L}_2$ nodes are labeled with upper case. Dashed lines (nodes and edges) represent constraint violations. Bottom row shows a graph representation of the data which satisfies all constraints. Top row shows possible subgraphs each has a single constraint violated- 2, 3, 5 or 4 from left to right respectively.

For a given set of known classes and properties the description so far does not describe a unique graph representation, as shown in figure 1. From all possible graphs representing a legal partial order, we define the following constraints which seem desirable for a "good" graph:

1. Data Consistency: Given a set of properties and a set of classes, we would like the graph representation to maintain the known relation and not add any new ones.

$$\forall \mathbf{u} \in \mathbf{L}_1 \; and \; \mathbf{v} \in \mathbf{L}_2 \exists path(\mathbf{v}, \mathbf{u}) \Leftrightarrow \mathbf{R_u} \subseteq \mathbf{P_v} \tag{1}$$

2. Vertex Minimality Constraints: from the representation point of view, there are two types of redundant vertices - those that represent the same class and those that have the same representation. First constraint - no two vertices may represent the same class:

$$\neg \exists \mathbf{s}, \mathbf{v} \in \mathbf{V} \; such \; that \; \{\mathbf{u} : path(u, s) \subset \mathbf{E}, \mathbf{u} \in \mathbf{L}_2\} = \{\mathbf{u} : path(u, v) \subset \mathbf{E}, \mathbf{u} \in \mathbf{L}_2\} \tag{2}$$

Second constraint - no two vertices may have the same representation:

$$\neg \exists \mathbf{s}, \mathbf{v} \in \mathbf{V} \; such \; that \; \{\mathbf{u} : path(s, u) \subset \mathbf{E}, \mathbf{u} \in \mathbf{L}_1\} = \{\mathbf{u} : path(v, u) \subset \mathbf{E}, \mathbf{u} \in \mathbf{L}_1\} \tag{3}$$

3. Maximal order: We would like to represnt all order relations within the set of known classes.

   For example in figure 1, all graphs represent a valid partial order, but the rightmost graph in the top line does not show a maximal partial order representation as there is no edge going from $\mathbf{B}$ to $\mathbf{A}$ so we cannot deduce from the graph that $\mathbf{A} \succeq \mathbf{B}$, while the bottom graph does show this relation. Thus when presented with a sample from $\mathbf{B}$ represented by $\mathbf{P_B} = \{\mathbf{a}, \mathbf{b}, \mathbf{c}\}$ their will be two satisfied nodes $\mathbf{A}$ and $\mathbf{B}$ where neither one is more specific then the other.

$$\{\mathbf{v} : path(s, v) \in \mathbf{E}, \mathbf{v} \in \mathbf{L}_1\} \subset \{\mathbf{v} : path(u, v) \in \mathbf{E}, \mathbf{v} \in \mathbf{L}_1\} \Leftrightarrow path(\mathbf{u}, \mathbf{s}) \subset \mathbf{E} \tag{4}$$

4. Edge Minimality Constraint: Let $\mathbf{v}^a$ denote the node represented by only one property $\mathbf{a}$, and $\mathbf{v}^{ab}$, $\mathbf{v}^{ac}$ and $\mathbf{v}^{abc}$ denote classes represented by the sets of properties $\{\mathbf{a}, \mathbf{b}\}$, $\{\mathbf{a}, \mathbf{c}\}$ and $\{\mathbf{a}, \mathbf{b}, \mathbf{c}\}$ respectively. A possible graph would have the following edges: $(\mathbf{v}^{abc}, \mathbf{v}^{ab})$,$(\mathbf{v}^{abc}, \mathbf{v}^{ac})$,$(\mathbf{v}^{abc}, \mathbf{v}^{a})$. These edges are redundant in the sense that $(\mathbf{v}^{abc}, \mathbf{v}^{a})$ can be removed without effecting the representation of $\mathbf{v}^{abc}$ or violating the maximal order constraint.

$$\neg\exists \mathbf{e} = (\mathbf{u}, \mathbf{v}) \in \mathbf{E} \; such \; that \; \mathbf{G}^* =< \mathbf{V}, \mathbf{E} \backslash \mathbf{e} > \tag{5}$$
$$maintains \; the \; maximal \; order \; constraint \; and \; \mathbf{R}^*_u = \mathbf{R}_u$$

3. Hierarchy Discovery Algorithm

In our above discussion we dealt with representing the partial order given a set of classes. In this section we deal with these arising two problems:

a) Not all possible labels in the partial order are given in advance.

b) How to find this partial order among a set of classes.

How do we deal with the case where not all possible labels in the partial order are given in advance? For example an implicit description of the general level 'Motorbike' class can be given by more specific motorbikes such as 'Cross-Motorbike' and 'Sport-Motorbike', without stating explicitly that both belong to the same general level class 'Motorbike'. In such a case were a general class is not stated explicitly we'll refer to it as a hidden general class. In the following we will deal with discovering all the possible hidden general classes by finding commonalities among known given classes. Given any two classes we say these classes have a general class in common if the intersection of feature sets used as representation of both classes is not empty. Formally, We say a class $\mathbf{C}$ is a hidden general class with respect to a given set of classes Γ iff $\exists \mathbf{A}, \mathbf{B} \in \Gamma$ such that $\mathbf{P_C} \neq \emptyset$ and $\mathbf{P_C} = \mathbf{P_A} \cap \mathbf{P_B}$.

Under this definition finding all hidden general classes requires we find all possible intersections of representation sets between all known classes.

We propose algorithm 1 for finding all possible hidden general level classes while simultaneously building a graph representation $\mathbf{G}^{out} =< \mathbf{V}^{out}, \mathbf{E}^{out} >$ consistent with the proposed graph partial order representation. In

order to do so we start with an initial graph representation of a given set of classes which is consistent with the above representation. For each feature in $\mathbf{x} \in \cup_{\mathbf{c}\in\Gamma}\mathbf{P}_c$ we define a vertex $\mathbf{v}, \mathbf{v_R} = \{\mathbf{x}\}\ and\ \mathbf{C}h_v = \emptyset$. For each class $\mathbf{C} \in \Gamma$ we define a vertex $\mathbf{u}$ with a set of outgoing edges denoted $\mathbf{O}\hat{ut}(u)$, where for each $\mathbf{x} \in \mathbf{P_C}$ there is a corresponding edge $(\mathbf{u}, \mathbf{v}) \in \mathbf{O}\hat{ut}(u)$. We denote the vertices corresponding to features as $\mathbf{L}_1$ and the vertices corresponding to classes as $\mathbf{L}_2$. Thus the input graph $\mathbf{G}^{in} =< \mathbf{V}^{in}, \mathbf{E}^{in} >$ is defined by $\mathbf{V}^{in} = \mathbf{L}_1 \cup \mathbf{L}_2$ and $\mathbf{E}^{in} = \cup_{\mathbf{u}\in\mathbf{L}_2}\mathbf{O}\hat{ut}(u)$, see fig for an illustration. We assume for now that in a given set of classes Γ there aren't any two class $\mathbf{A}, \mathbf{B} \in \Gamma$ where $\mathbf{P_A} \subseteq \mathbf{P_B}$ or $\mathbf{P_B} \subseteq \mathbf{P_A}$ (we'll deal with this later).

Now we shell describe the four basic operation carried out by the algorithm: Split, Foward-Merge,Backward-Merge and Maximal-Ordering. These four operations enable the creation of new nodes based on the set of vertices connected by outging edges $\mathbf{O}ut(u)$ and the set of vertices connected by incoming edges $\mathbf{I}n(u)$ of each vertex $\mathbf{u} \in \mathbf{V}$.

Split(s,in(s)): The split operation creates a new node for each pair of incoming edges in $\mathbf{in}(\mathbf{s}) \subset \mathbf{I}n(s)$ of node $\mathbf{s} \in \mathbf{V}$. Intuitively this helps us mark an intersection between the representation of two nodes as equivalent to $\mathbf{R_s}$ (the representation of node $\mathbf{s}$).

Formally:

$\forall \mathbf{u}, \mathbf{v} \in \mathbf{V}$ such that $\{\mathbf{u}, \mathbf{v}\} \in \mathbf{in}(\mathbf{s})$ do:

Create a new node $\mathbf{t}$ with

- $\mathbf{O}ut(t) = \{\mathbf{s}\}$
- $\mathbf{I}n(t) = \{\mathbf{u}, \mathbf{v}\}$

Foward-Merge(U): The Forward-Merge operation merges all nodes in a specific set of nodes $\mathbf{U}$ which share the same incoming edges. Intuitively by doing the merge operation after the split operation we find the maximum set of intersection between the representation of any two vertices. This operation is essential for maintaining constraint 2 as will be proven later. We denote $\mathbf{E}_{in}$ as the group of maximal equivalence sets of nodes according to the incoming edges. Thus, ${\mathbf{E}_{in}}^i \in \mathbf{E}_{in} \Leftrightarrow \forall \mathbf{u}, \mathbf{v} \in {\mathbf{E}_{in}}^i\ \mathbf{I}n(u) \equiv \mathbf{I}n(v)$ and $\forall {\mathbf{E}_{in}}^i, {\mathbf{E}_{in}}^j \in \mathbf{E}_{in}, i \neq j \Leftrightarrow \forall \mathbf{u} \in {\mathbf{E}_{in}}^i$ and $\mathbf{v} \in {\mathbf{E}_{in}}^j\ \mathbf{I}n(u) \neq \mathbf{I}n(v)$

Formally:

1. Compute the group of $\mathbf{E}_{in}$ over $\mathbf{U}$.
2. $\forall {\mathbf{E}_{in}}^i \in \mathbf{E}_{in}$ if $|{\mathbf{E}_{in}}^i| > 1$
   Create a new node $\mathbf{n}$ with
   - $\mathbf{O}ut(n) = \{\mathbf{s} | \mathbf{s} \in \mathbf{O}ut(u)\ \forall \mathbf{u} \in {\mathbf{E}_{in}}^i\}$
   - $\mathbf{I}n(n) = \mathbf{I}n(u)$ where $\mathbf{u} \in {\mathbf{E}_{in}}^i$

As the **Foward-Merge(U)** is done sequentially after the **Split(s,in(s))** operation, each node $\mathbf{u} \in \mathbf{U}$ has exactly two incoming edges- $\{(\mathbf{x}, \mathbf{u}), (\mathbf{y}, \mathbf{u})\}$, where $\mathbf{x}, \mathbf{y} \in \cup_{\mathbf{s}}\mathbf{in}(\mathbf{s})$. We can compute $\mathbf{E}_{in}$ using the algorithm described in 6.1 after sorting the two incoming edges to each node given some predefined order

over all possible incoming edges. As input to the algorithm in 6.1 we pass $\mathbf{U}$ as the group of sets, and $\cup_{\mathbf{s}}\mathbf{in}(\mathbf{s})$ as the group of possible elements in each set. Thus the runtime of **Foward-Merge(U)** is O($|\cup_{\mathbf{s}}\mathbf{in}(\mathbf{s})| + |\mathbf{U}|$).

Backward-Merge(U): The Backward-Merge operation merges all node in a specific set of nodes $\mathbf{U}$ which share the same outgoing edges. Intuitively by doing the Backward-Merge operation after the Foward-Merge we find the maximal set of nodes which share the same representation. This operation is essential for maintaining constraint 3 as will be proven later. We denote $\mathbf{E}_{out}$ as the group of maximal equivalence sets of nodes according to the outgoing edges. Thus, ${\mathbf{E}_{out}}^i \in \mathbf{E}_{out} \Leftrightarrow \forall \mathbf{u},\mathbf{v} \in {\mathbf{E}_{out}}^i\ \mathbf{O}ut(u) \equiv \mathbf{O}ut(v)$ and $\forall {\mathbf{E}_{out}}^i, {\mathbf{E}_{out}}^j \in \mathbf{E}_{out}, i \neq j \Leftrightarrow \forall \mathbf{u} \in {\mathbf{E}_{out}}^i$ and $\mathbf{v} \in {\mathbf{E}_{out}}^j\ \mathbf{O}ut(u) \neq \mathbf{O}ut(v)$

Formally:

1. Compute the group of $\mathbf{E}_{out}$ over $\mathbf{U}$.
2. $\forall {\mathbf{E}_{out}}^i \in \mathbf{E}_{out}$
   Create a new node $\mathbf{n}$ with
   - $\mathbf{O}ut(n) = \mathbf{O}ut(u)$ where $\mathbf{u} \in {\mathbf{E}_{out}}^i$
   - $\mathbf{I}n(n) = \{\mathbf{s}|\mathbf{s} \in \mathbf{I}n(u)\ \forall \mathbf{u} \in {\mathbf{E}_{out}}^i\}$

For the computation of $\mathbf{E}_{out}$ we can again use the algorithm described in 6.1, where $\mathbf{U}$ is the group of sets and $\cup_{\mathbf{s}-was-split}\mathbf{s}$ is the group of possible elements in each set (outgoing edges). As mentioned in 6.1 for each node $\mathbf{u} \in \mathbf{U}$ the elements (outgoing edges) should be sorted. Contrary to the computation of $\mathbf{E}_{in}$ in the **Foward-Merge(U)** operation where each node has exactly two elements (incoming edges) thus sorting the elements of each node can be done in constant time, the size of the possible elements considered for each node at this stage can be at most $|\cup_{\mathbf{s}-was-split}\mathbf{s}|$. In order to over come the need of sorting outgoing edges of each node at this stage we would like to keep the $\mathbf{O}ut(u)$ of each node, ordered according to some chosen order of the original $\mathbf{s}$ nodes. This can be accomplished easily by keeping each ${\mathbf{E}_{in}}^i$ during the **Foward-Merge(U)** operation ordered according to the order of **Split(s,in(s))**. The runtime of **Backward-Merge(U)** is O($|\cup_{\mathbf{s}-was-split}\mathbf{s}| + \sum_{\mathbf{u}} |\mathbf{O}ut(u)|$).

EdgeMin(U): Given a set of nodes $\mathbf{U}$ it may be the case that $\exists \mathbf{u},\mathbf{v} \in \mathbf{U}$ for which $\mathbf{R_u} \subset \mathbf{R_v}$. Under our partial order interpretation of the graph and in order to maintain constraints 4 and 5 we would like to connect by an edge each node to the **most** specific node representing a more general level and when doing so we would like to delete all edges to nodes more general then the newly connected node. This can be achieved by checking for each pair of new nodes $\exists \mathbf{u},\mathbf{v} \in \mathbf{U}$ if w.l.o.g $\mathbf{R_u} \subset \mathbf{R_v}$, in such a case all edges connecting $\mathbf{u}$ to nodes in the intersection of $\mathbf{I}n(u)$ and $\mathbf{I}n(v)$ should be deleted. This will ensure that $\forall \mathbf{s} \in \mathbf{I}n(u) \cap \mathbf{I}n(v)$ for which it follows that- $\mathbf{R_u} \subset \mathbf{R_v} \subset \mathbf{R_s}$, $\mathbf{s}$ will be connected to the most specific general node, $\mathbf{v}$. Hence maintaing the maximal ordering 4 by maintaining the connectivity between $\mathbf{s}$ to $\mathbf{v}$ and $\mathbf{u}$, and maintaing the edge minimality 5 by deleting redundant edges- $(\mathbf{s},\mathbf{u})$.

Formally:

1. for each $\mathbf{p} \in \cup_{\mathbf{s}-was-split}\mathbf{s}$ create a list L(p)
2. for each $\mathbf{u}^i \in \mathbf{U}$ go over all $\mathbf{p} \in \mathbf{O}ut(u^i)$, and add i to L(p).
3. for each i choose $\mathbf{p}^* = arg\,min_{\mathbf{p}\in\mathbf{O}ut(u^i)}$ L(p)
   (a) for each $\mathbf{p} \in \mathbf{O}ut(u^i)$ mark all $j \neq i \in$ L(p*) which appear in L(p).
      i. if j appears in all L(p), conclude that $\mathbf{O}ut(u^i) \subseteq \mathbf{O}ut(u^j)$
4. for each j such that $\mathbf{O}ut(u^i) \subseteq \mathbf{O}ut(u^j)$ do $\mathbf{I}n(u^i) = \mathbf{I}n(u^i) \backslash \mathbf{I}n(u^i) \cap \mathbf{I}n(u^j)$

Note that in following algorithm EdgeMin(U) is preformed sequentially after Backward-Merge(U) thus for any two $\mathbf{u}^i, \mathbf{u}^j \in \mathbf{U}$ there cannot be an identity between $\mathbf{O}ut(u^i)$ and $\mathbf{O}ut(u^j)$.

Initializing all L(p) takes O($|\cup_{\mathbf{s}-was-split}\mathbf{s}|$). Updating all L(p) for all $\mathbf{u}^i \in \mathbf{U}$ takes O($\sum_i |\mathbf{O}ut(u^i)|$). We assume a predefined order over $\mathbf{u}^i \in \mathbf{U}$. Finding $\mathbf{p}^*$ for each i takes O($|\mathbf{O}ut(u^i)|$). Going over all other $\mathbf{p} \in \mathbf{O}ut(u^i)$ and checking for each $j \neq i \in$ L(p*) if it appears in L(p) takes O($\sum_{\mathbf{p}\in\mathbf{O}ut(u^i)} |L(p)|$), this can be achieved as we assume that L(p*) and L(p) are ordered, so comparing both lists can be done in a single pass with runtime O($|L(p)|$) as $|L(p*)| < |L(p)|$. Thus, finding all containment relations among all $\mathbf{u}^i \in \mathbf{U}$ takes O($\sum_i |\mathbf{O}ut(u^i)| + \sum_i \sum_{\mathbf{p}\in\mathbf{O}ut(u^i)} |L(p)|$). Finding and deleting the edges in the intersection of both nodes, step 4, can take O($|\mathbf{I}n(u^i)| + |\mathbf{I}n(u^j)|$), given a hash table of size $|\cup_i \mathbf{I}n(u^i)|$ (we do not assume $\mathbf{I}n(u)$ is ordered). Thus, in total we can conclude the runtime of the EdgeMin operation is O($\sum_i |\mathbf{O}ut(u^i)| + \sum_i \sum_{\mathbf{p}\in\mathbf{O}ut(u^i)} |L(p)| + |\mathbf{I}n(u^i)| + |\mathbf{I}n(u^j)|$).

4. Statistical Model

Till now we assumed a scenario where a class/category is described deterministically by a set of properties $\mathbf{P_c}$ which should appear in all instances of the class. We would now like to relax this definition to deal with the case where group of properties belonging to a class, appear with a certain probability in each of its instances. Thus, instead of requiring $\mathbf{x}\in\mathbf{P_c} \Rightarrow \psi_{\mathbf{x}}(\mathbf{s}) = 1\ \forall \mathbf{s} \in \mathbf{C}$, we denote $\delta_{\mathbf{x}}^{\mathbf{C}} \equiv \mathbf{P}(\psi_{\mathbf{x}}(\mathbf{s}) = 1|\mathbf{s} \in \mathbf{C})$, the class dependent probability of a specific type of property- $\mathbf{x}$ and require: $\mathbf{x}\in\mathbf{P_c} \Rightarrow \delta_{\mathbf{x}}^{\mathbf{C}} >> \rho_{\mathbf{x}}$, in such a case we shall refer to the property $\mathbf{x}$ as "typical" to class $\mathbf{C}$. In addition properties can appear in classes where they are not typical, in such a case we will refer to them as noisy properties. We define a property as noisy with respect to a given class if $\rho_{\mathbf{x}} \geq \delta_{\mathbf{x}}^{\mathbf{C}}$. $\rho_{\mathbf{x}}$ is a class independent probability of a specific property $\mathbf{x}$ to appear in instances of classes where this property is not "typical".

We say $\mathbf{x}$ is "similarly typical" between classes if the class dependent values $\delta_{\mathbf{x}}^{\mathbf{C}}$ are similar, formally we shell denote $\mathbf{X_i}$ a group of classes for which $\mathbf{x}$ is similarly typical- $\delta_{\mathbf{x}}^{\mathbf{C}} \in [\delta_{\mathbf{xi}} - \lambda_{\mathbf{x}}, \delta_{\mathbf{xi}} + \lambda_{\mathbf{x}}], \forall \mathbf{C} \in \mathbf{X_i}$.

For example, lets denote $\Gamma = [\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{E}]$ a group of classes, and [0.1,0.4,0.45,0.78,0.8] the corresponding class dependent probabilities $[\delta_{\mathbf{x}}^{\mathbf{A}}, \delta_{\mathbf{x}}^{\mathbf{B}}, \delta_{\mathbf{x}}^{\mathbf{C}}, \delta_{\mathbf{x}}^{\mathbf{D}}, \delta_{\mathbf{x}}^{\mathbf{E}}]$ of property $\mathbf{x}$, where $\rho_{\mathbf{x}} = 0.2$ and $\lambda_{\mathbf{x}} = 0.1$. We will say that $\mathbf{x}$ is "typical" to classes $\mathbf{B}, \mathbf{C}, \mathbf{D}$ and $\mathbf{E}$ where $\mathbf{B}$ is "similarly typical" to $\mathbf{C}$ with respect to $\mathbf{x}$ and also $\mathbf{D}$ is "similarly typical" to $\mathbf{E}$ with respect to $\mathbf{x}$.

Similarly we can extend the notions of "typical" and "similarly typical" properties to a "typical group" of co-couring properties, where $\mathbf{x}$ now denotes a group of properties, $\mathbf{P_c}$ a group of groups of properties and $\delta_{\mathbf{x}}^{\mathbf{C}} \equiv \mathbf{P}(\bigwedge_{\mathbf{x}'\in\mathbf{x}} \psi_{\mathbf{x}'}(\mathbf{s}) = 1|\mathbf{s} \in \mathbf{C})$.

Algorithm 1 SIPO: Set Intersection Partial Order

Input :

$\mathbf{G}^{in} =< \mathbf{V}^{in}, \mathbf{E}^{in} >$ where $\mathbf{V}^{in} = \mathbf{L}_1 \cup \mathbf{L}_2$ and $\mathbf{E}^{in} = \cup_{\mathbf{u} \in \mathbf{L}_2} \mathbf{O}ut(u)$

Output:

$\mathbf{G}^{out} =< \mathbf{V}^{out}, \mathbf{E}^{out} >$

1. Init:
   - $\mathbf{V}^{out} = \mathbf{V}^{in}$,$\mathbf{E}^{out} = \mathbf{E}^{in}$,
   - $\mathbf{S} = \mathbf{L}_1$,$\forall \mathbf{s} \in \mathbf{S}$ $\mathbf{in(s)} = \mathbf{I}n(s)$,
   - $\mathbf{FWMerg} = \emptyset, \mathbf{BWMerg} = \emptyset, \mathbf{tmpS} = \emptyset$
2. While $\exists \mathbf{s} \in \mathbf{S}$ such that $\mathbf{in(s)} \neq \emptyset$ do:
   (a) $\forall \mathbf{s} \in \mathbf{S}$ do:
      i. **Split(s,in(s))**
      ii. add all nodes created by Split(s) to $\mathbf{tmpS}$
      iii. $\mathbf{in(s)} = \emptyset$
   (b) **Foward-Merge(tmpS)** and add all newly created nodes to $\mathbf{FWMerg}$
   (c) **Backard-Merge(FWMerg)** and add all newly created nodes to $\mathbf{BWMerg}$
   (d) $\mathbf{V}^{out} = \mathbf{V}^{out} \cup \mathbf{BWMerg}$
   (e) **EdgeMin(BWMerg)**
   (f) $\forall \mathbf{s} \in \mathbf{BWMerg}$ and $\forall \mathbf{u} \in \mathbf{I}n(s), \forall \mathbf{v} \in \mathbf{O}ut(s)$ do:
      i. $\mathbf{E}^{out} = \mathbf{E}^{out} \backslash (\mathbf{u}, \mathbf{v})$
      ii. $\mathbf{E}^{out} = \mathbf{E}^{out} \cup \{(\mathbf{u}, \mathbf{s}), (\mathbf{s}, \mathbf{v})\}$
      iii. $\mathbf{in(v)} = \mathbf{in(v)} \cup \mathbf{s}$
3. $\forall \mathbf{v} \in \mathbf{V}^{out}$ such that $|\mathbf{O}ut(v)| == 1$ do:
   (a) for $\mathbf{s} = \mathbf{O}ut(v)$ and $\forall \mathbf{u} \in \mathbf{I}n(v)$
      i. $\mathbf{O}ut(u) = \mathbf{O}ut(u) \cup \mathbf{O}ut(v)$
      ii. $\mathbf{I}n(s) = \mathbf{I}n(s) \cup \mathbf{I}n(v)$
      iii. $\mathbf{E}^{out} = \mathbf{E}^{out} \cup (\mathbf{u}, \mathbf{s})$
   (b) $\mathbf{V}^{out} = \mathbf{V}^{out} \backslash \mathbf{v}$

In order to apply the graphical model and graph construction algorithm to this statistical scenario we restrict the probability model using the following assumptions:

1. Noisy properties of a class are statistically independent.

2. If a group of properties is "typical" to a class then each individual property is also "typical".

3. A group of properties is said to be "typical" to a class if $\delta^{\mathbf{C}}_{\mathbf{x}} \geq \prod_{\mathbf{x}' \in \mathbf{x}} \delta^{\mathbf{c}}_{\mathbf{x}'}$ where $\mathbf{x}$ denotes the group and $\mathbf{x}'$ an individual property in the group.

Assumption 2 is a strong restriction as we might want to allow two or more properties to be "typical" only when they occur together but not separately, in such a case we can define a new property as the conjunction of these property, the new property will be typical, but the old properties and their conjunction wont.

In the deterministic case, we presented algorithm 1 for finding all possible general classes. This algorithm is based on finding commonalities among classes by computing the intersection between the property sets of each class. The notion of intersection between the sets $\mathbf{P_c}$ can be extended to the statistical scenario by regarding any "similarly typical" group of properties between two or more classes as a partial intersection between the group of classes. In the deterministic case the intersection of a given group of classes contains all properties shared by all classes in the group, in the statistical case two groups of properties $\mathbf{x}$ and $\mathbf{y}$ which are both "similarly typical" among a group of classes Γ, may not be "similarly typical" together with the respect to the same group of classes, hence $\mathbf{t} = \mathbf{x} \cup \mathbf{y}$ is not "similarly typical" in Γ. In such a case we will say the intersection between a given group of classes is a set of sets of "similarly typical" properties, opposed to the deterministic scenario where the intersection is a single group. Thus, we will say that the statistical intersection, $\mathbf{S}I$, of Γ is $\{\mathbf{x}, \mathbf{y}\}$, denoted $\mathbf{S}I(\Gamma) = \{\mathbf{x}, \mathbf{y}\}$. Given the set Υ of all "similarly typical" groups with respect to any group of classes Γ, we define the statistical intersection of Γ as a set of "similarly typical" groups of properties with the following two conditions:

1. $\forall \mathbf{x} \in \Upsilon \; \exists \mathbf{y} \in \mathbf{S}I(\Gamma)$ such that $\mathbf{x} \subseteq \mathbf{y}$.
2. $\forall \mathbf{y} \in \mathbf{S}I(\Gamma) \; \nexists \mathbf{x} \in \Upsilon$ such that $\mathbf{y} \subset \mathbf{x}$.

It can be said that $\mathbf{S}I$ is the maximal set of "similarly typical" groups with respect to any group of classes.

5. Hierarchical Multi-Task learning

In this section we introduce an hierarchical view of multi-task learning. We extend the notion of task relatedness [4] to an hierarchy of task relations in 5.2. In section 5.3 we extend the learning paradigm presented in [4] to deal with the hierarchical setting, we present IMT-ERM (Iterative Multi Task-ERM) which is an hierarchical generalization of MT-ERM. We recall the generalization analysis of MT-ERM and motivate a specific learning approach based on a cascade of classifiers, such an approach is often applied in hierarchical learning algorithms (e.g. CART [7]). Limiting our-selfs to a more restrictive approach we are able to propose a constructive paradigm for which under two assumptions, *transformation-independance* and *optimality-preservation* we can derive explicit learning bounds, opposed to the general lower and upper bounds on the generalization bound of the MT-ERM approach. The cascaded approach and its required assumption are presented in 5.4.

5.1. Background

We start by restating the multi-task learning scenario and Ben-David's et al. [4] notion of relatedness. As in [4] we view multitask learning as having a single task that one wishes to learn and the extra related tasks are just an aid towards learning the main task. We follow the notations and restate briefly the definitions from [4], where

its relevant we refer the reader to the specific definitions in [4]. Formally, the multi-task learning scenario can be stated as follows: Given domain $\mathcal{X}$ and unknown distributions P_1,...P_n over $\mathcal{X}$ x $\{0,1\}$, a learner is presented with a sequence of random samples S_1,...S_n drawn from these P_i's respectively, and has to come up with a hypothesis $h : \mathcal{X} \to \{0,1\}$ such that, for (x,b) drawn randomly from P_1, $h(x) = b$ with high probability. As in [4] we focus on the extent to which the samples S_i, for $i \neq 1$ can be utilized to help find a good hypothesis for predicting the labels of P_1.

Task relatedness is defined based on a set $\mathcal{F}$ of transformations $f : \mathcal{X} \to \mathcal{X}$. Tasks P_1 and P_2 are said to be $\mathcal{F}$-related if $P_1(x,b) = P_2(f(x),b)$ or $P_2(x,b) = P_1(f(x),b)$. For the formal definition we refer the reader to definition 1 in [4]. Given an hypothesis space, $\mathbb{H}$, over domain $\mathcal{X}$ we assume $\mathcal{F}$ acts as a group over $\mathbb{H}$, namely $\mathcal{F}$ is closed under function composition and $\mathbb{H}$ is closed under transformations from $\mathcal{F}$, for a formal definition see definition 2 in [4]. Two hypothesis $h_1, h_2 \in \mathbb{H}$ are said to equivalent under $\mathcal{F}$ iff there exists $f \in \mathcal{F}$ such that $h_2 = h_1 \circ f$, hypothesis equivalence under $\mathcal{F}$ is denoted by $\sim_{\mathcal{F}}$. $[h]_{\sim_{\mathcal{F}}} = \{h \circ f : f \in \mathcal{F}\}$ denotes a set of hypothesis which are equivalent up to transformations in $\mathcal{F}$. $\mathbb{H}/\sim_{\mathcal{F}}$ denotes the family of all equivalence classes of $\mathbb{H}$ under $\sim_{\mathcal{F}}$, namely- $\mathbb{H}/\sim_{\mathcal{F}} = \{[h]_{\sim_{\mathcal{F}}} : h \in \mathbb{H}\}$.

The learning scenario in [4] assumes the learner gets samples $\{S_i : i \leq n\}$, each S_i is a set of samples drawn i.i.d from P_i. The probability distributions are assumed to be pairwise $\mathcal{F}$-related. The learner knows the set of indices of the distributions, $\{1,...,n\}$ and the family of functions $\mathcal{F}$, but does not know the data-generating distribution nor which specific function f relates any given pair of distributions. In this setting, Ben-David et al. [4] proposed exploiting the relatedness among tasks by first finding all aspects of the tasks which are invariant under $\mathcal{F}$ then focus on learning the specific $\mathcal{F}$-sensitive elements of the single objective task. The potential benefit lays both in the reduction of the search space from the original $\mathbb{H}$ to a smaller $\mathcal{F}$-sensitive subspace $[h]_{\sim_{\mathcal{F}}}$, and from the bigger sample size available to learn the $\mathcal{F}$-invariant aspects of the task. Though, the second potential benefit is not guaranteed and depends on the complexity of finding a single $[h]_{\sim_{\mathcal{F}}}$ in $\mathbb{H}/\sim_{\mathcal{F}}$. The complexity of finding $[h]_{\sim_{\mathcal{F}}}$ is formalized in [4] using the notion of generalized VC-dimension from [3]. See [4] for an analysis and example of the complexity of finding $[h]_{\sim_{\mathcal{F}}}$.

5.2. Multi-Task Hierarchical Setting

Now we shall extend the multi-task learning setting to an hierarchical multi-task learning setting. In the hierarchical setting our objective is the same as in the original multi-task, that of learning a single task by exploiting extra related tasks. Our approach extends the original by assuming that the group of tasks, indexed- $\{1,...,n\}$ and the corresponding family of transformation functions $\mathcal{F}$ can be decomposed hierarchically. We denote by l a single level in the hierarchy, $0 \leq l \leq L$ and $\mathcal{T}_l \subseteq \{1,...,n\}$ the group of related tasks in the l's level of the hierarchy. $\mathcal{F}_l$ denotes a family of transformations for which all task in $\mathcal{T}_l$ are pairwise $\mathcal{F}_l$-related.

We assume that the set of transformation for each level $0 \leq l \leq L-1$ can be written as a concatenation of the set of domain transformations corresponding to the proceeding level $l+1$, $\mathcal{F}_{l+1}$, and a set of domain transformations $\mathcal{G}_l$, hence- $F_l = \{g \circ f : g \in \mathcal{G}_l, f \in \mathcal{F}_{l+1}\}$.

Definition 1 We say that the group of tasks $\mathcal{T}_{l+1}$ is invariant with respect to the set of transformation $\mathcal{G}_l$ iff:

$\forall i \in \{1..n\} \setminus \mathcal{T}_{l+1}, \exists g \in \mathcal{G}_l$ such that $\forall j \in \mathcal{T}_{l+1}$, $\exists f^{ij} \in \mathcal{F}_{l+1}$ for which: $P_i(x,b) = P_j(g(f^{ij}(x)),b)$

We call this property an invariance, as all tasks in $\mathcal{T}_{l+1}$ share the same transformation from $\mathcal{G}_l$ when transforming into tasks which are not in the group $\mathcal{T}_{l+1}$. For example, lets consider n rectangles in R^d each is marked with an index from $\{1..n\}$, now lets assume that some of these rectangles share the same size lengths but differ in their location, lets mark this group by $\mathcal{T} \subset \{1..n\}$. Now we shall consider two different sets of transformations- $\mathcal{F}$ shifts in R^d and $\mathcal{G}$ scaling in R^d. Clearly the group of rectangles $\mathcal{T}$ is invariant with respect to the transformations in $\mathcal{G}$ but not to $\mathcal{F}$.

Definition 2 We say $\{\mathcal{T}_l, \mathcal{F}_l\}_{l=0}^{L}$ is an hierarchical decomposition of a set of $\mathcal{F}$-related tasks, $\{1, ..., n\}$ iff:

1. $\mathcal{T}_0 = \{1, .., n\}$

2. $\mathcal{T}_L = \{1\}$, hence $\mathcal{T}_L$ represents the single objective task.

3. $\mathcal{F}_L = \{f\}$, where f is the identity transformation, hence $f(x) = x$.

4. for all $0 \leq l \leq L-1$:

- **a.** $\mathcal{T}_{l+1} \subset \mathcal{T}_l$
- **b.** $\forall i, j \in \mathcal{T}_l$, exists $f \in \mathcal{F}_l$ such that $P_i(x, b) = P_j(f(x), b)$
- **c.** $\mathcal{T}_{l+1}$ is invariant to $\mathcal{G}_l$
- **d.** $\mathcal{F}_l = \{g \circ f : g \in \mathcal{G}_l, f \in \mathcal{F}_{l+1}\}$.
- **e.** $\mathcal{F}_l$ and $\mathcal{G}_l$ act as a group over $\mathbb{H}$.

From the definition of the hierarchical decomposition we see (4.c) that the set of transformations for level l can be obtained by concatenating the set of invariances of levels $l+1$ till L. This point will be crucial in understanding the benefit of the cascaded hierarchical approach.

Lemma 1 $\mathcal{F}_{l+1} \subset \mathcal{F}_l$, for all $0 \leq l \leq L-1$.

From the fact that $\mathcal{G}_l$ is a group, hence contains the identity transformation and from definition 2, setion 4.c

Lemma 2 Given $h \in \mathbb{H}$, $[h]_{\sim_{\mathcal{F}_{l+1}}} \subset [h]_{\sim_{\mathcal{F}_l}}$, for all $0 \leq l \leq L-1$.

This is an immediate consequence of Lemma 1 and the definition of $[h]_{\sim_{\mathcal{F}}}$.

5.3. Learning paradigm and Generalization analysis

Now we shall extend Ben-Davis's et al. [4] two stage learning paradigm, MT-ERM (Multi-Task Empirical Risk Minimization), to an $L+1$ stage learning paradigm, denoted IMT-ERM (Iterative MT-ERM). Both are empirical risk minimization approaches. We follow standard notation and denote the empirical error an hypothesis for a sample set S as:

$$\hat{Er}^S(h) = \frac{|\{(x, b) \in S : h(x) \neq b\}|}{|S|}.$$

the true error of an hypothesis, as:

$$Er^P(h) = P(\{(x,b) \in \mathcal{X} \times \{0,1\} : h(x) \neq b\}).$$

and following [4] we define the error of any hypothesis space, $\mathbb{H}$ as:

$$Er^P(\mathbb{H}) = \inf_{h \in \mathbb{H}} Er^p(h).$$

For notation convenience we shall denote $|\mathcal{T}_l|$ (the number of tasks in each level) by t_l, and l_i corresponds to the i'th task in $\mathcal{T}_l$.

Definition 3 Given $\mathbb{H}$, n tasks hierarchically decomposed by $\{\mathcal{T}_l, \mathcal{F}_l\}_{l=0}^{L}$ and their sequence of labeled sample sets, $S_1, .., S_n$, the IMT-ERM paradigm works as follows:

1. $\mathbb{H}^0 = \mathbb{H}$.

2. for $l = 0..L$

a. Pick $h \in \mathbb{H}^l$ that minimizes $\inf_{h_{l_1},...,h_{l_{t_l}} \in [h]_{\sim_{\mathcal{F}_l}}} \sum_{i=1}^{t_l} \hat{Er}^{S_{l_i}}(h_{l_i})$ over all $[h]_{\sim_{\mathcal{F}_l}} \in \mathbb{H}^l / \sim_{\mathcal{F}_l}$

b. $\mathbb{H}^{l+1} = [h]_{\sim_{\mathcal{F}_l}}$

3. output $h^\diamond$ the single hypothesis in $[h]_{\sim_{\mathcal{F}_L}}$ as the learner's hypothesis.

First we state that the fact that $h^\diamond$ is the single hypothesis in $[h]_{\sim_{\mathcal{F}_L}}$, is derived directly from the definition of $\mathcal{F}_L$ as containing only the identity transformation.

Following the definition of hierarchical decomposition one can easily be convinced that for $L = 1$ the IMT-ERM is exactly the MT-ERM. The first iteration corresponds to the first step of the MT-ERM- learning the aspects which are invariant under $\mathcal{F}_0$. The second iteration corresponds to the second stage of MT-ERM where only the training samples coming from the target task, the single task in $\mathcal{T}_L$, are used to find a single predictor, the single hypothesis in $[h]_{\sim_{\mathcal{F}_L}}$.

The learning complexity of step 2.a, picking $[h]_{\sim_{\mathcal{F}_l}} \in \mathbb{H}^l / \sim_{\mathcal{F}_l}$, is analyzed in [4] using the notion of generalized VC-dimension from [3], denoted by $d_{\mathbb{H}^l/\sim_{\mathcal{F}_l}}(n)$, where n refers to the number of tasks, t_l in our setting. This measure has a lower bound of $\sup\{VC - dim([h]_{\sim_{\mathcal{F}_l}}) : [h]_{\sim_{\mathcal{F}_l}} \in \mathbb{H}^l / \sim_{\mathcal{F}_l}\}$ and in the general case an upper-bound of $VC - dim(\mathbb{H}^l)$, see proposition 1 in [4]. This analysis captures the interrelation between the set of transformations $\mathcal{F}_l$ and the hypothesis space $\mathbb{H}^l$.

From lemma 1 and lemma 2 we know that both $\sup\{VC - dim([h]_{\sim_{\mathcal{F}_l}}) : [h]_{\sim_{\mathcal{F}_l}} \in \mathbb{H}^l / \sim_{\mathcal{F}_l}\}$ and $VC - dim(\mathbb{H}^l)$ monotonically decrease with l, which may imply the potential of the hierarchical approach. But, these measures only act as lower and upper-bounds on the search complexity of step 2.a. In order to account for the potential benefit in the hierarchical approach we shall limit our following discussion to a scenario were we can precisely calculate the search complexity of step 2.a.

Furthermore, the measure $d_{\mathbb{H}^l/\sim_{\mathcal{F}_l}}(n)$ is used in [4] theorem 3, to bound the minimal amount of samples needed for each task in $\mathcal{T}_l$, separately. In the following we shall consider the case where we are concerned with the total amount of samples from related tasks, this will enable richer learning scenarios where we can still benefit from tasks with very few samples as long as the shared aspects of the tasks can be learnt in conjunction with tasks with many samples.

Finally, the MT-ERM paradigm and its hierarchical extension IMT-ERM, don't provide a constructive way of preforming step 2.a. In the following we shall consider the conditions under which step 2.a. can be preformed by searching over the possible set of invariance's $\mathcal{G}_l$ yielding a constructive approach with search complexity of $VC - dim([h]_{\sim_{\mathcal{G}_l}})$.

5.4. Multi-Task Cascade

In this section we analyze the conditions under-which an optimal hypothesis of a single task can be found via searching for a cascade of invariants, each invariant fits a single level in the hierarchical decomposition described above. We start by analyzing specific properties of the optimal transformations given our hierarchical decompositions, lemma 3 and 4. In section 5.4.1 we present the cascaded approach defining a new task for each level in the hierarchy. In sections 5.4.2 and 5.4.3 we present two properties of a cascade of invariances which are the basis of the assumptions under which a cascaded approach can reach optimality. Finally in section 5.4.4 we state the assumptions and prove the cascaded approach can reach optimality.

We shall start by recalling that from lemma 2 in [4] we can deduce that-

$$Er^{P_{l_j}}([h]_{\sim_{\mathcal{F}_l}}) = \inf_{h_1,\ldots h_{|\mathcal{T}_l|}\in[h]_{\sim_{\mathcal{F}_l}}} \frac{1}{|\mathcal{T}_l|}\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h_i) \tag{6}$$

In the above hierarchical decomposition definition (definition 2) we assumed that $\mathcal{T}_l$ is invariant to $\mathcal{G}_{l-1}$. In the following we show that any $g \in \mathcal{G}_{l-1}$ which is optimal for a single task $j \in \mathcal{T}_l$ in the sense that it minimizes $Er^{P_{l_j}}([h \circ g]_{\sim_{\mathcal{F}_l}})$, is optimal for all other tasks in $\mathcal{T}_l$.

Lemma 3 For each $j \in \mathcal{T}_l, g^* = \arg\min_{g\in\mathcal{G}_{l-1}} Er^{P_{l_j}}([h\circ g]_{\sim_{\mathcal{F}_l}}) \Leftrightarrow \forall i \in \mathcal{T}_l$ and $\forall g \in \mathcal{G}_{l-1}\ Er^{P_{l_i}}([h\circ g^*]_{\sim_{\mathcal{F}_l}}) \leq Er^{P_{l_i}}([h \circ g]_{\sim_{\mathcal{F}_l}})$

Proof $\Leftarrow$: This direction is trivial, if g^* attains minima for all $i \in \mathcal{T}_l$ it does so also for j. □

$\Rightarrow$: Assume that $g^* = \arg\min_{g\in\mathcal{G}_{l-1}} Er^{P_{l_j}}([h \circ g]_{\sim_{\mathcal{F}_l}})$. Let $f^g \in \mathcal{F}_l$ be the transformation which minimizes $Er^{P_{l_j}}([h \circ g]_{\sim_{\mathcal{F}_l}})$ hence by definition $Er^{P_{l_j}}(h \circ g \circ f^g) = Er^{P_{l_j}}([h \circ g]_{\sim_{\mathcal{F}_l}})$. $\mathcal{F}_l$ is the family of transformation between tasks in $\mathcal{T}_l$, thus $\forall i \in \mathcal{T}_l\ \exists f_{ij} \in \mathcal{F}_l$, such that $Er^{P_{l_j}}(h \circ g \circ f^g) = Er^{P_{l_i}}(h \circ g \circ f^g \circ f_{ij}), \forall g \in \mathcal{G}_{l-1}$.

By the definition of $\mathcal{F}_l$ as a group we know that $f^g \circ f_{ij} \in \mathcal{F}_l$, lets assume that $Er^{P_{l_i}}(h\circ g\circ f^g\circ f_{ij}) \neq Er^{P_{l_i}}([h\circ g]_{\sim_{\mathcal{F}_l}})$, thus exists $z \in \mathcal{F}_l$ such that $Er^{P_{l_i}}(h\circ g\circ z) = Er^{P_{l_i}}([h\circ g]_{\sim_{\mathcal{F}_l}})$ and $Er^{P_{l_i}}(h\circ g\circ z) < Er^{P_{l_i}}(h\circ g\circ f^g\circ f_{ij})$. Let $f_{ji} \in \mathcal{F}_l$ be the transformation from task i to task j. Thus, $Er^{P_{l_j}}(h \circ g \circ z \circ f_{ji}) < Er^{P_{l_j}}(h \circ g \circ f^g)$ which contradicts the definition of f^g, as $z \circ f_{ji} \in \mathcal{F}_l$. So we can derive that- $Er^{P_{l_i}}(h \circ g \circ f^g \circ f_{ij}) = Er^{P_{l_i}}([h \circ g]_{\sim_{\mathcal{F}_l}})$.

We know that $Er^{P_{l_j}}(h \circ g^* \circ f^{g^*}) \leq Er^{P_{l_j}}(h \circ g \circ f^g)\ \forall g \in \mathcal{G}_{l-1}$, thus $Er^{P_{l_i}}(h \circ g^* \circ f^{g^*} \circ f_{ij}) \leq Er^{P_{l_i}}(h \circ g \circ f^g \circ f_{ij})\ \forall i \in \mathcal{T}_l$. Thus, $\forall i \in \mathcal{T}_l$ and $\forall g \in \mathcal{G}_{l-1}\ Er^{P_{l_i}}([h \circ g^*]_{\sim_{\mathcal{F}_l}}) \leq Er^{P_{l_i}}([h \circ g]_{\sim_{\mathcal{F}_l}})$. □

Lemma 4 If $g' = \arg\min_{g \in \mathcal{G}} Er^P([h \circ g]_{\sim_{\mathcal{F}}})$ and $f' = \arg\min_{f \in \mathcal{F}} Er^P(h \circ g' \circ f)$, then $g' = \arg\min_{g \in \mathcal{G}} Er^P(h \circ g \circ f')$

Proof For each $g \in \mathcal{G}$ lets denote by f^g the optimal $f \in \mathcal{F}$ with respect to g, thus $f^g = \arg\min_{f \in \mathcal{F}} Er^P(h \circ g \circ f)$. From the definition of g' we know:

$$Er^P(h \circ g' \circ f') \leq Er^P(h \circ g \circ f^g), \forall g \in \mathcal{G}. \tag{7}$$

From the definition of f^g we know:

$$Er^P(h \circ g \circ f^g) \leq Er^P(h \circ g \circ f'), \forall g \in \mathcal{G}. \tag{8}$$

From 7 and 8 we get that:

$$Er^P(h \circ g' \circ f') \leq Er^P(h \circ g \circ f'), \forall g \in \mathcal{G}.$$

Hence- $g' = \arg\min_{g \in \mathcal{G}} Er^P(h \circ g \circ f')$. □

5.4.1 Single Unified Task

Our cascaded approach is based on defining a single task for each level in the hierarchy and optimizing the solution based on this task. We start by defining the task-

For each level in the hierarchy $0 \leq l \leq L-1$, we define the task representing the level as the union of all tasks in $\mathcal{T}_l$. As before we mark the i'th task in $\mathcal{T}_l$ as $l_i \in \{1..n\}$. Let P_1,...P_n be the probability distributions over $\mathcal{X}$ x $\{0,1\}$ of tasks $\{1..n\}$ respectively. Hence, for $j \in \{1..n\}$, $P_j = P(x,b|j)$. We shall now define the probability distribution over $\mathcal{X}$ x $\{0,1\}$,which fits the single task $\mathbf{a}_l$ representing the union of all tasks in $\mathcal{T}_l$, as the average of distribution of tasks in $\mathcal{T}_l$, hence:

$$P_{\mathbf{a}_l}(x,b) = \frac{1}{|\mathcal{T}_l|} \sum_{i=1}^{|\mathcal{T}_l|} P_{l_i}(x,b) \tag{9}$$

Lemma 5

$$Er^{P_{\mathbf{a}_l}}(h) = \frac{1}{|\mathcal{T}_l|} \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h) \tag{10}$$

This is straight-forward from the definition of $P_{\mathbf{a}_l}$ and Er^P.

In our hierarchical approach we assume that $\mathcal{F}_{l-1} = \{g \circ f : g \in \mathcal{G}_{l-1}, f \in \mathcal{F}_l\}$ is the set of pairwise transformations of group $\mathcal{T}_{l-1}$. Thus an hypothesis $h' \in [h]_{\sim_{\mathcal{F}_{l-1}}}$, can be written as $h' = h \circ g \circ f$, where $g \in \mathcal{G}_{l-1}$ and $f \in \mathcal{F}_l$. Developing this another level, $h'' \in [h']_{\sim_{\mathcal{F}_l}}$, we can write $h'' = h' \circ g' \circ f'$, where $g' \in \mathcal{G}_l$ and $f' \in \mathcal{F}_{l+1}$, which can also be written as $h'' = h \circ g \circ g' \circ f'$, thus we get that $h'' \in [h \circ g]_{\sim_{\mathcal{F}_{l+1}}}$, where $g \in \mathcal{G}_{l-1}$.

In the cascaded approach we propose finding the optimal $g \in \mathcal{G}_{l-1}$ for task $\mathbf{a}_l$, where $0 \leq l \leq L-1$. The potantial gain in such an approach is that each learning step is governed by $VC - dim([h]_{\sim_{\mathcal{G}_{l-1}}})$ and the number of samples available for the unified task is larger.

In the following we analyze under what conditions this approach guarantees optimality with respect to the original tasks, hence: finding $\arg\min_{g \in \mathcal{G}_{l-1}} Er^{P_{l_j}}([h \circ g]_{\sim_{\mathcal{F}_l}})$, by choosing $g \in \mathcal{G}_{l-1}$ which minimizes the error of task $\mathbf{a}_l$: finding $\arg\min_{g \in \mathcal{G}_{l-1}} Er^{P_{\mathbf{a}_l}}(h \circ g)$.

5.4.2 Transformation Independence

Definition 4 Two transformations taken from two families of transformations $g \in \mathcal{G}$ and $f \in \mathcal{F}$ are said to be independent with respect to hypothesis $h \in \mathbb{H}$ iff $h \circ g \circ f(x) = h \circ g(x) \cdot h \circ f(x)$.

It is easy to see that $\{x | h \circ g(x) \cdot h \circ f(x) = 1\} = \{x | h \circ g(x) = 1\} \cap \{x | h \circ f(x)\}$. For illustration lets consider rectangles in R^2, $\mathcal{G}$ is the family of scale and translation of the first dimension and $\mathcal{F}$ is the family of scale and translation of the second dimension. We shall parametrize the rectangle by $[c1, c2, s1, s2]$ where c1 and c2 correspond to the center of the rectangle and [s1,s2] correspond to the size of each arc (width and height of rectangle). Let $h \in \mathbb{H}$ be the rectangle $[3, 3, 2, 2]$, and let $g \in \mathcal{G}$ be translation of 0.5 and scale of 0.5 of the first dimension, let $f \in \mathcal{F}$ have the same values of g applied on the second dimension. Hence: $h \circ g$ is rectangle $[3.5, 3, 1, 2]$ and $h \circ f$ $[3, 3.5, 2, 1]$ the intersection of their support corresponds to rectangle $[3.5, 3.5, 1, 1]$ which is the same as $h \circ g \circ f(x)$. On the other hand if we were to consider just translations but no scaling we would get that $h \circ g$ is rectangle $[3.5, 3, 2, 2]$ and $h \circ f$ $[3, 3.5, 2, 2]$ the intersection of their support would be the rectangle parametrized by $[3.25, 3.25, 1.75, 1.75]$, thus will not equal $h \circ g \circ f(x)$ which is rectangle $[3.5, 3.5, 2, 2]$.

The transformation independence property will enable us to write $Er^P(h \circ g \circ f)$ as $Er^P(h \circ g)$ plus a residual term, describing the gain of adding the transformation from $\mathcal{F}$.

Definition 5 We shall denote the residual term by R^{gf} hence-

$$\begin{aligned} R^{gf} &= P(\{(x, b) \in \mathcal{X} \times \{0, 1\} : b = 1, h \circ g(x) = 1, h \circ f(x) = 0\}) \\ &\quad - P(\{(x, b) \in \mathcal{X} \times \{0, 1\} : b = 0, h \circ g(x) = 1, h \circ f(x) = 0\}) \end{aligned} \tag{11}$$

This term specifics the amount of errors introduced when adding transformation f minus the amount of corrections this transformation makes with respect to transformation g. Under the transformation independence assumption, adding a transformation f can change the overall classification value only for points in the support of $h \circ g$.

Lemma 6 If transformations $g \in \mathcal{G}$ and $f \in \mathcal{F}$ are independent with respect to hypothesis $h \in \mathbb{H}$ then:

$$Er^P(h \circ g \circ f) = Er^P(h \circ g) + R^{gf} \tag{12}$$

for notational convenience in the proof following we shall omit the domain $\mathcal{X} \times \{0,1\}$ when referring to (x,b), and $h \circ f(X)$ we shall write shortly as $hf(X)$.

Proof

$$\begin{aligned}
&Er^P(h \circ g \circ f) = P(\{(x,b) : hgf(x) \neq b\}) = \\
&P(\{(x,b) : hgf(x) \neq b, hf(x) = 1\}) + P(\{(x,b) : hgf(x) \neq b, hf(x) = 0\}) = \qquad (13)\\
&P(\{(x,b) : hg(x)hf(x) \neq b, hf(x) = 1\}) + P(\{(x,b) : hgf(x) \neq b, hf(x) = 0\}) = \\
&P(\{(x,b) : hg(x) \neq b, hf(x) = 1\}) + P(\{(x,b) : hgf(x) \neq b, hf(x) = 0\}) = \\
&P(\{(x,b) : hg(x) \neq b, hf(x) = 1\}) + P(\{(x,b) : hgf(x) \neq b, hf(x) = 0\}) + \\
&P(\{(x,b) : hg(x) \neq b, hf(x) = 0\}) - P(\{(x,b) : hg(x) \neq b, hf(x) = 0\}) = \\
&P(\{(x,b) : hg(x) \neq b\}) + P(\{(x,b) : hgf(x) \neq b, hf(x) = 0\}) - \\
&P(\{(x,b) : hg(x) \neq b, hf(x) = 0\}) = \qquad (14)\\
&P(\{(x,b) : hg(x) \neq b\}) + P(\{(x,b) : b = 1, hg(x) = 1, hf(x) = 0\}) + \\
&P(\{(x,b) : b = 1, hg(x) = 0, hf(x) = 0\}) - P(\{(x,b) : b = 1, hg(x) = 0, hf(x) = 0\}) - \\
&P(\{(x,b) : b = 0, hg(x) = 1, hf(x) = 0\}) = \\
&P(\{(x,b) : hg(x) \neq b\}) + P(\{(x,b) : b = 1, hg(x) = 1, hf(x) = 0\}) - \\
&P(\{(x,b) : b = 0, hg(x) = 1, hf(x) = 0\}) = \\
&Er^P(h \circ g) - P(\{(x,b) : b = 0, hg(x) = 1, hf(x) = 0\}) + \\
&P(\{(x,b) : b = 1, hg(x) = 1, hf(x) = 0\}) = \\
&Er^P(h \circ g) + R^{gf}.
\end{aligned}$$

Equalities 13 and 14 are due to the transformation independence. □

Choosing a transformation $f \in \mathcal{F}$ which minimizes $Er^P(h \circ g \circ f)$ implies that $R^{gf} \leq 0$ as it can only decrease the overall error, this is stated formally in the following lemma-

Lemma 7 $f = \arg\min_{f' \in \mathcal{F}} Er^P(h \circ g \circ f') \Rightarrow R^{gf} \leq 0$

Proof We know that the identity transformation $f'(x) = x$ is in $\mathcal{F}$ ($\mathcal{F}$ is a group), thus $Er^P(h \circ g \circ f) - Er^P(h \circ g) \leq 0$, from lemma 6 we know that $R^{gf} = Er^P(h \circ g \circ f) - Er^P(h \circ g)$. □

5.4.3 Optimality-Preservation

As we've shown in lemma 6, under the *transformation-independance* assumption the error $Er^P(h \circ g \circ f)$ can be decomposed into the error obtained by choosing $g \in \mathcal{G}$, $Er^P(h \circ g)$, and a residual term R^{gf} referring to the change in the overall classification value of points in the support of $h \circ g$, when adding a transformation $f \in \mathcal{F}$. If an optimal transformation remains optimal when adding another transformation from a different family we shall say the added transformation has the *optimality-preservation* property, formally-

Definition 6 A transformation $f \in \mathcal{F}$ is said to have the *optimality-preservation* property with respect to a distribution P, an hypothesis h and family of transformations $\mathcal{G}$, if for $g^* = \arg\min_{g \in \mathcal{G}} Er^P(h \circ g)$ the following holds:

$$R^{g^* f} - R^{gf} \leq Er^P(h \circ g) - Er^P(h \circ g^*), \forall g \in \mathcal{G}$$

To illustrate a scenario where such a property occurs, lets consider rectangles in R^d, and two families of transformations $\mathcal{G}$ and $\mathcal{F}$ each scales and translates a different set of coordinates. Lets assume that any two transformations $g \in \mathcal{G}$ and $f \in \mathcal{F}$ are statistically independent. Hence:

$$\begin{aligned} &P(\{(x,b) : h \circ g(x) = a, h \circ f(x) = c\}) = \\ &P(\{(x,b) : h \circ g(x) = a\}) \cdot P(\{(x,b) : h \circ f(x) = c\}) \end{aligned} \tag{15}$$

Now we shall assume a realizable learning scenario where for each rectangle r, $P^r(x, b = 1) = 1$ iff x is within r otherwise $P^r(x, b = 1) = 0$. The family of hypothesis we consider, $\mathbb{H}$, is the family of functions $h(x) \in \{0, 1\}$ defined over all possible rectangles in R^d, hence $h^r(x) = 1$ iff x is within the rectangle r. In this scenario for each r there exits an hypothesis $h^* \in \mathbb{H}$ where $Er^{P^r}(h^*) = 0$.

From the realizability of the learning scenario, we can conclude that for $g^* = \arg\min_{g \in \mathcal{G}} Er^P(h \circ g)$, g^* is minimal in both types of errors separately , has lower false postives- $P(\{(x,b) : b = 0, hg^*(x) = 1\})$ and lower false negatives- $P(\{(x,b) : b = 1, hg^*(x) = 0\})$ then any other $g \in \mathcal{G}$. This can easily be seen by assuming that there exists some g for which this is not the case, changing g^* according to g, can only reduce the error- making the rectangle bigger in order to include the false negative will not cause false positives, while making the rectangle smaller in-order to exclude false positives will not cause false negatives. This claim is straight forward for the scaling transformation and also holds for translations as they can be viewed as two consecutive non uniform scaling operations.

Corollary 1 For the task of learning a rectangle r given the hypothesis space $\mathbb{H}$, and transformations $\mathcal{G}$ and $\mathcal{F}$ as defined above, each $f \in \mathcal{F}$ which is statistical independent of any $g \in \mathcal{G}$ has also the optimal preservation property with respect to P^r, $\mathcal{G}$ and any $h \in \mathbb{H}$.

Proof Let $g^* = \arg\min_{g\in\mathcal{G}} Er^P(h\circ g)$ and f which is statistical independent of any $g \in \mathcal{G}$ then:

$$R^{g^*f} - R^{gf} = \tag{16}$$
$$P(\{(x,b) : b = 1, hg^*(x) = 1, hf(x) = 0\}) - P(\{(x,b) : b = 0, hg^*(x) = 1, hf(x) = 0\}) -$$
$$(P(\{(x,b) : b = 1, hg(x) = 1, hf(x) = 0\}) - P(\{(x,b) : b = 0, hg(x) = 1, hf(x) = 0\})) = \tag{17}$$
$$[P(\{(x,b) : b = 1, hg^*(x) = 1\}) - P(\{(x,b) : b = 1, hg(x) = 1\})]P(\{(x,b) : b = 1, hf(x) = 0\}) -$$
$$[P(\{(x,b) : b = 0, hg^*(x) = 1\}) - P(\{(x,b) : b = 0, hg(x) = 1\})]P(\{(x,b) : b = 0, hf(x) = 0\}) =$$
$$[P(\{(x,b) : b = 1, hg(x) = 0\}) - P(\{(x,b) : b = 1, hg^*(x) = 0\})]P(\{(x,b) : b = 1, hf(x) = 0\}) +$$
$$[P(\{(x,b) : b = 0, hg(x) = 1\}) - P(\{(x,b) : b = 0, hg^*(x) = 1\})]P(\{(x,b) : b = 0, hf(x) = 0\}) \leq \tag{18}$$
$$[P(\{(x,b) : b = 1, hg(x) = 0\}) - P(\{(x,b) : b = 1, hg^*(x) = 0\})] +$$
$$[P(\{(x,b) : b = 0, hg(x) = 1\}) - P(\{(x,b) : b = 0, hg^*(x) = 1\})] = \tag{19}$$
$$Er^P(h\circ g) - Er^P(h\circ g^*).$$

Equality 16 is due to the definition of the residual, R^{gf}. Equality 17 is due to the statistical independence of $f \in \mathcal{F}$ and any $g \in \mathcal{G}$. From the above argument and the optimality of g^* we know that both the false negative and false postive of g^* are smaller then any other g, hence the components specifying the false negative and false positive differences, between g and g^*, are positive, thus mulitplying each of them by a bigger number increases their sum as in 18. Equality 19 is due to the definition of the error as the sum of the false positive and false negative.□

In the following we show that for a group of tasks and transformations maintaining the *optimality-preservation* property the optimal transformation with respect to the unified task and a single set of transformations $\mathcal{G}$ is optimal also when considering each of the tasks separately and adding a transformation from $\mathcal{F}$. For this we shall denote:

- $g^{\mathbf{a}_l} = \arg\min_{g\in\mathcal{G}_{l-1}} Er^{P_{\mathbf{a}_l}}(h\circ g)$, the optimal transformation with respect to the task $\mathbf{a}_l$, representing the union of tasks in T_l.
- $g^l = \arg\min_{g\in\mathcal{G}_{l-1}} Er^{P_{l_i}}([h\circ g]_{\sim_{\mathcal{F}_l}}), \forall i \in |\mathcal{T}_l|$. the optimal transformation which is shared among all of the tasks in $\mathcal{T}_l$ (from lemma 3 we know g^l exists).
- $f^{l_i} = \arg\min_{f\in\mathcal{F}_l} Er^{P_{l_i}}(h\circ g^l\circ f), \forall i \in |\mathcal{T}_l|$. the optimal transformation which is specific to each of the tasks in $\mathcal{T}_l$.

Lemma 8 Under the transformation independence assumption between $g^{\mathbf{a}_l}$ and $f^{l_i}\forall i \in |\mathcal{T}_l|$, and g^l and $f^{l_i}\forall i \in |\mathcal{T}_l|$,

$$\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h\circ g^{\mathbf{a}_l}]_{\sim_{\mathcal{F}_l}}) = \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h\circ g^l]_{\sim_{\mathcal{F}_l}})$$
$$\Leftrightarrow \sum_{i=1}^{|\mathcal{T}_l|} R^{g^{\mathbf{a}_l}f^{l_i}} - \sum_{i=1}^{|\mathcal{T}_l|} R^{gf^{l_i}} \leq \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h\circ g) - \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h\circ g^{\mathbf{a}_l}), \forall g \in \mathcal{G}_{l-1}$$

Proof

$$\sum_{i=1}^{|\mathcal{T}_l|} R^{g^{\mathbf{a}_l} f^{l_i}} - \sum_{i=1}^{|\mathcal{T}_l|} R^{g f^{l_i}} \leq \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g) - \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g^{\mathbf{a}_l}), \forall g \in \mathcal{G}_{l-1} \Leftrightarrow$$

$$\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g^{\mathbf{a}_l}) + R^{g^{\mathbf{a}_l} f^{l_i}} \leq \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g) + R^{g f^{l_i}}, \forall g \in \mathcal{G}_{l-1} \Leftrightarrow_1 \quad (20)$$

$$\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g^{\mathbf{a}_l} \circ f^{l_i}) \leq \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g \circ f^{l_i}), \forall g \in \mathcal{G}_{l-1} \Leftrightarrow \quad (21)$$

$$\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g^{\mathbf{a}_l} \circ f^{l_i}) \leq \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g^{l} \circ f^{l_i}) \Leftrightarrow_2 \quad (22)$$

$$\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g^{\mathbf{a}_l} \circ f^{l_i}) = \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g^{l} \circ f^{l_i}) \Leftrightarrow_3 \quad (23)$$

$$\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g^{\mathbf{a}_l} \circ f^{l_i}) = \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h \circ g^{l}]_{\sim_{\mathcal{F}_l}}) \Leftrightarrow_3 \quad (24)$$

$$\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h \circ g^{\mathbf{a}_l}]_{\sim_{\mathcal{F}_l}}) = \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h \circ g^{l}]_{\sim_{\mathcal{F}_l}}) \quad (25)$$

20 $\Leftrightarrow$ 21 due to lemma 6. 21 $\Rightarrow$ 22 due to lemma 4. 22 $\Rightarrow$ 21 and 22 $\Rightarrow$ 23 from the optimality of g^l . 23 $\Leftrightarrow$ 24 is derived from the definition of f^{l_i} and the definition of $Er^P([h]_{\sim_{\mathcal{F}}})$. 24 $\Leftrightarrow$ 25 otherwise from the definition of $Er^P([h]_{\sim_{\mathcal{F}}})$ we'll get that $\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h \circ g^{\mathbf{a}_l}]_{\sim_{\mathcal{F}_l}}) < \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g^{\mathbf{a}_l} \circ f^{l_i})$, thus $\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h \circ g^{\mathbf{a}_l}]_{\sim_{\mathcal{F}_l}}) < \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h \circ g^{l}]_{\sim_{\mathcal{F}_l}})$ which will contradict the optimality of g^l .□

5.4.4 Cascade Optimality

Now we are ready to state the following two assumptions under-which the optimal transformation $g^l \in \mathcal{G}_{l-1}$ for each of the tasks in $\mathcal{T}_l$ is the same as the optimal transformation $g^{\mathbf{a}_l} \in \mathcal{G}_{l-1}$ for task $\mathbf{a}_l$.

Assumption 1

- $g^{\mathbf{a}_l}$ and f^{l_i} are *transformation-independent* $\forall i \in |\mathcal{T}_l|$.
- g^{l} and f^{l_i} are *transformation-independent* $\forall i \in |\mathcal{T}_l|$.

Assumption 2 $\forall i \in |\mathcal{T}_l|$, f^{l_i} has the *optimality-preservation* property with respect to a distribution P_{l_i}, the underline hypothesis h and family of transformations $\mathcal{G}$.

Theorem 1 Under assumptions 1 and 2:

$$\arg\min_{g\in\mathcal{G}_{l-1}} Er^{P_{\mathbf{a}_l}}(h\circ g) = \arg\min_{g\in\mathcal{G}_{l-1}} Er^{P_{l_i}}([h\circ g]_{\sim_{\mathcal{F}_l}}), \forall i \in \mathcal{T}_l$$

Proof From assumption 2 we can deduce that:

$$\sum_{i=1}^{|\mathcal{T}_l|} R^{g^{\mathbf{a}_l} f^{l_i}} - \sum_{i=1}^{|\mathcal{T}_l|} R^{g f^{l_i}} \leq \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h\circ g) - \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h\circ g^{\mathbf{a}_l}), \forall g \in \mathcal{G}_{l-1}$$

putting this together with assumption 1 we know from lemma 8 that:

$$\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h\circ g^{\mathbf{a}_l}]_{\sim_{\mathcal{F}_l}}) = \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h\circ g^{l}]_{\sim_{\mathcal{F}_l}})$$

thus for $g^{\mathbf{a}_l} = \arg\min_{g\in\mathcal{G}_{l-1}} Er^{P_{\mathbf{a}_l}}(h\circ g)$ also attains minimum for the sum over the error of all tasks in $\mathcal{T}_l$, thus:

$$g^{\mathbf{a}_l} = \arg\min_{g\in\mathcal{G}_{l-1}} \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h\circ g]_{\sim_{\mathcal{F}_l}})$$

Thus, from equation (6) above we can deduce that:

$$g^{\mathbf{a}_l} = \arg\min_{g\in\mathcal{G}_{l-1}} Er^{P_{l_i}}([h\circ g]_{\sim_{\mathcal{F}_l}}), \forall i \in \mathcal{T}_l. \square$$

We note, that there might be more then one transformation achieving the minimum.

5.5. Multi-Task Cascade ERM

In the previous section we showed that the optimal transformation of a task can be found by considering a group of tasks together, in case this group is invariant with respect to this transformation. This implies a learning scenario were searching for the optimal invariance of a single task can be done by considering the group of tasks sharing this invariance, thus leveraging the bigger sample size from all the group. Till now we discussed the optimal solution, now we shell extend this framework to the case where we cannot guarantee optimality of the solution. For this we will need to extend our assumptions so that we can derive that a near optimal solution for the group of tasks considered together is also near optimal for each of the tasks considered separately, thus permitting a derivation of an ERM approach.

Firstly, instead of considering the *transformation-independance* only for the optimal choice of transformations, we will assume *transformation-independance* between $f^{l_i} \in \mathcal{F}_l$, the optimal transformation for task $l_i \in \mathcal{T}_l$, and any transformation $g \in \mathcal{G}_{l-1}$.

Secondly, the *optimality-preservation* assumption needs to be extended, adding a lower bound to the difference

of residuals . Thus, the improvement achieved by adding the transformation to a non optimal invariance is close to the improvement when adding a transformation to an optimal invariance.

In order to extend our cascaded approach from two levels to the L levels in the hierarchy, we extend the assumptions to the whole hierarchy. For this we will denote:

- $f^{l_i} = \arg\min_{f' \in \mathcal{F}_l} Er^{P_{l_i}}(h \circ g^l \circ f'), \forall i \in [1..|\mathcal{T}_l|]$,for $l = 0..L-1$
- $g^{\mathbf{a}_l} = \arg\min_{g \in \mathcal{G}_{l-1}} Er^{P_{\mathbf{a}_l}}(h \circ g)$, for $l = 1..L-1$

Cascade assumptions: for $l = 0..L-1$ and $h \in \mathbb{H}^{l-1}$ the chosen hypothesis for Pa_{l-1}:

1. $\forall g \in \mathcal{G}_{l-1}$ and $f^{l_i}, \forall i \in |\mathcal{T}_l|$, are independent with respect to h , hence:

$$h \circ g \circ f^{l_i}(x) = h \circ g(x) \cdot h \circ f^{l_i}(x)$$

2. $|\sum_{i=1}^{|\mathcal{T}_l|} R^{g^{\mathbf{a}_l} f^{l_i}} - \sum_{i=1}^{|\mathcal{T}_l|} R^{g f^{l_i}}| \leq \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g) - Er^{P_{\mathbf{a}_l}}(h \circ g^{\mathbf{a}_l}), \forall g \in \mathcal{G}_{l-1}$

For simplicity, in the following we shall refer to the strong form of assumptions 1,2 declared above as *transformation-independance* and *optimality-preservation* respectively.

We can now analyze the bound of the error of an hypothesis composed of some original hypothesis, h, a shared invariance learnt via an ERM process, $g^\diamond$, and the optimal transformation for each task, f^{l_i}, given the hypothesis-$h \circ g^\diamond$.

Theorem 2 Let $d = VC - dim([h]_{\sim_{\mathcal{G}_{l-1}}})$, $h^* = \arg\min_{h \in [h]_{\sim_{\mathcal{G}_{l-1}}}} Er^{P_{\mathbf{a}_l}}(h)$ and $h^\diamond \in [h]_{\sim_{\mathcal{G}_{l-1}}}$ be the output of a standard ERM algorithm trained on task $\mathbf{a}_l$, with $m^{\mathbf{a}_l}$ samples. Then for every ϵ and $\delta > 0$, if $m^{\mathbf{a}_l} \geq c_0(\frac{1}{\epsilon} log\frac{1}{\delta} + \frac{d}{\epsilon} log\frac{1}{\epsilon})$, with probability greater then $(1-\delta)$,

$$\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h^\diamond]_{\sim_{\mathcal{F}_l}}) \leq \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h^*]_{\sim_{\mathcal{F}_l}}) + 2\epsilon$$

Proof We note that from the above definition of $g^{\mathbf{a}_l}$, $h^* = h \circ g^{\mathbf{a}_l}$, we shall also write $h^\diamond$ as the original hypothesis h and the chosen transformation $g^\diamond \in \mathcal{G}_{l-1}$ such that $h^\diamond = h \circ g^\diamond$. We know that for a standard ERM algorithm, with probability greater then $(1-\delta)$-

$$Er^{P_{\mathbf{a}_l}}(h \circ g^\diamond) \leq Er^{P_{\mathbf{a}_l}}(h \circ g^{\mathbf{a}_l}) + \epsilon \tag{26}$$

From Lemma 5 we can write-

$$\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g^\diamond) \leq \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g^{\mathbf{a}_l}) + \epsilon$$

From the *transformation-independance* assumption and Lemma 6 we can write-

$$\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h \circ g^{\diamond}]_{\sim_{\mathcal{F}_l}}) - \sum_{i=1}^{|\mathcal{T}_l|} R^{g^{\diamond} f^{l_i}} \leq \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h \circ g^{\mathbf{a}_l}]_{\sim_{\mathcal{F}_l}}) - \sum_{i=1}^{|\mathcal{T}_l|} R^{g^{\mathbf{a}_l} f^{Pa_{l_i}}} + \epsilon \tag{27}$$

where $f^{Pa_{l_i}} \in \mathcal{F}_l$ are the optimal transformations for all $l_i \in \mathcal{T}_l$, given hypothesis $h \circ g^{\mathbf{a}_l}$. and $f^{l_i} \in \mathcal{F}_l$ are the optimal transformations for all $l_i \in \mathcal{T}_l$, given hypothesis $h \circ g^{\diamond}$.

From the strong form of the *optimality-preservation* assumption we know that-

$$|\sum_{i=1}^{|\mathcal{T}_l|} R^{g^{\mathbf{a}_l} f^{l_i}} - \sum_{i=1}^{|\mathcal{T}_l|} R^{g^{\diamond} f^{l_i}}| \leq \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}(h \circ g^{\diamond}) - Er^{P_{\mathbf{a}_l}}(h \circ g^{\mathbf{a}_l}) \tag{28}$$

as $f^{Pa_{l_i}} \in \mathcal{F}_l$ are the optimal transformations for all $l_i \in \mathcal{T}_l$, given hypothesis $h \circ g^{\mathbf{a}_l}$ we get that $\forall i$-

$$R^{g^{\mathbf{a}_l} f^{Pa_{l_i}}} \leq R^{g^{\mathbf{a}_l} f^{l_i}} \tag{29}$$

From 26, 28 and 29 we can write-

$$\epsilon \leq \sum_{i=1}^{|\mathcal{T}_l|} R^{g^{\mathbf{a}_l} f^{Pa_{l_i}}} - \sum_{i=1}^{|\mathcal{T}_l|} R^{g^{\diamond} f^{l_i}} \tag{30}$$

Putting together 27 and 30-

$$\sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h \circ g^{\diamond}]_{\sim_{\mathcal{F}_l}}) \leq \sum_{i=1}^{|\mathcal{T}_l|} Er^{P_{l_i}}([h \circ g^{\mathbf{a}_l}]_{\sim_{\mathcal{F}_l}}) + 2\epsilon \tag{31}$$

□

Now we are ready to propose CMT-ERM (Cascaded MT-ERM) paradigm, extending the two stage cascaded approach to an L stage cascade. In this approach each stage searches for an invariance which is shared by all tasks grouped into a level in the hierarchy.

Definition 7 Given $\mathbb{H}$, n tasks hierarchically decomposed by $\{\mathcal{T}_l, \mathcal{F}_l\}_{l=0}^{L}$ and their sequence of labeled sample sets, $S_1, .., S_n$, the CMT-ERM paradigm works as follows:

1. $\mathbb{H}^0 = \mathbb{H}$.

2. for $l = 0..L$

a. Pick $h \in \mathbb{H}^l$ that minimizes $Er^{P_{\mathbf{a}_l}}(h)$

b. $\mathbb{H}^{l+1} = [h]_{\sim_{\mathcal{G}_l}}$

3. output $h^{\diamond}$ the single hypothesis in $[h]_{\sim_{\mathcal{F}_L}}$ as the learner's hypothesis.

In the first iteration we choose an hypothesis and not a transformation as in the remaining stages, in order to apply the same analysis we consider $h \in \mathbb{H}^0$ the chosen hypothesis for Pa_0, define $\mathcal{G}_{-1}$ the set of transformations for which $[h]_{\sim_{\mathcal{G}_{-1}}} = \mathbb{H}^0$ and $g^{Pa_0} \in \mathcal{G}_{-1}$ as the identity transformation. We will also define $\mathbb{H}^{-1} = \mathbb{H}^0$ and Pa_{-1} as the same task as Pa_0, thus h is also the chosen hypothesis for Pa_{-1}. We note, that as we don't actually search over $[h]_{\sim_{\mathcal{G}_{-1}}}$ but rather $\mathbb{H}^0, \mathcal{G}_{-1}$ need not exist and can be considered just as a notation for writing each hypothesis $h' \in \mathbb{H}^0$ as $h' = h \circ g'$.

6. Appendix

6.1. Finding Equivalence Sets

In this section we present an algorithm for finding all equivalencies between sets within a given group of sets $\mathbf{S}$. Each set $\mathbf{s}^i$ were $i \in \mathbf{S}$, contains several elements from a finite set $\mathbf{P}$, thus $\mathbf{s}^i \subseteq \mathbf{P}$. Two sets are said to be equivalent if they are identical in the elements that they contain. The output of the algorithm is a grouping $\mathbf{E_P}$ of the sets into maximal equivalence sets. Thus, $\mathbf{E}^i_\mathbf{P} \in \mathbf{E_P} \Leftrightarrow \forall \mathbf{u}, \mathbf{v} \in \mathbf{E}^i_\mathbf{P}\ \mathbf{u} \equiv \mathbf{v}$ and $\forall \mathbf{E}^i_\mathbf{P}, \mathbf{E}^j_\mathbf{P} \in \mathbf{E}, i \neq j \Leftrightarrow \forall \mathbf{u} \in \mathbf{E}^i_\mathbf{P}$ and $\mathbf{v} \in \mathbf{E}^j_\mathbf{P}\ \mathbf{u} \neq \mathbf{v}$. The elements in each set in the input are assumed to be sorted according to some predefined ordering of P. Given such sorted sets equivalence can be found simply by a sequential pass over all elements in each sets, each time comparing the k'th element in all sets of size k or bigger. Sets which are equivalent will be identical on all elements. An efficient implementation has runtime of $O(|\mathbf{P}| + \sum_i |\mathbf{s}^i|)$.

References

[1] A. Bar-Hillel, T. Hertz, and D. Weinshall. Efficient learning of relational object class models. *Proc. ICCV*, 2005. 2, 3

[2] A. Bar-Hillel and D. Weinshall. Subordinate class recognition using relational object models. *Proc. NIPS*, 19, 2006. 2

[3] J. Baxter. A model of inductive bias learning. *J. Artif. Intell. Res. (JAIR)*, 12:149–198, 2000. 12, 14

[4] S. Ben-David and R. Borbely. A notion of task relatedness yielding provable multiple-task learning guarantees. *Machine Learning*, 73(3):273–287, 2008. 3, 11, 12, 13, 14, 15

[5] T. Berg, A. Berg, J. Edwards, and D. Forsyth. Whos in the Picture? In *Advances in neural information processing systems 17: proceedings of the 2004 conference*, page 137. The MIT Press, 2005. 2

[6] D. Blei, T. Griffiths, M. Jordan, and J. Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process. *Advances in neural information processing systems*, 16:106, 2004. 2

[7] L. Brieman, J. Friedman, R. Olshen, and C. Stone. Classification and regression trees. *Wadsworth Inc*, 67, 1984. 2, 11

[8] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, volume 1, page 22. Citeseer, 2004. 2, 3

[9] O. Dekel. Distribution-Calibrated Hierarchical Classification. 2

[10] B. Epshtein and S. Ullman. Semantic hierarchies for recognizing objects and parts. In *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR'07*, pages 1–8, 2007. 2

[11] R. Fergus, P. Perona, and A. Zisserman. A sparse object category model for efficient learning and exhaustive recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005*, volume 1, 2005. 2, 3

[12] S. Fidler and A. Leonardis. Towards scalable representations of object categories: Learning a hierarchy of parts. In *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR'07*, pages 1–8, 2007. 2

Algorithm 2 Find Equivalence

Input :

$\mathbf{P}$ set of possible elements

$\mathbf{S}$ group of sets, for each set $i \in \mathbf{S}$, $\mathbf{s}^i \subseteq \mathbf{P}$

Output:

$\mathbf{E_P}$ A grouping of the sets in $\mathbf{S}$ into maximal equivalence sets according to identity on elements of $\mathbf{P}$.

1. Lists = $|\mathbf{P}|$ empty lists.
2. $\mathbf{E_P}$ = FindEq($\mathbf{S}$,Lists,1);

FindEq($\mathbf{S}$,Lists,ind)

1. init empty list $\mathbf{E_P}$
2. add all $\mathbf{s}^i$ for which $|\mathbf{s}^i| = ind - 1$ to of $\mathbf{E_P}$
3. add each $\mathbf{s}^i$ with $|\mathbf{s}^i| \geq ind$ to Lists($\mathbf{s}^i(ind)$)
4. add each non empty list as a list to TempList, and empty Lists
5. for each list in TempList compute $\mathbf{E}^k_\mathbf{P}$ = FindEq(TempList(k),Lists,ind+1)
6. add all $\mathbf{E}^k_\mathbf{P}$ to $\mathbf{E_P}$

---

[13] L. Jie, B. Caputo, A. Zweig, J. Bach, and J. Anemuller. Object category detection using audio-visual cues. In *Proc. Internat. Conf. on Computer Vision System, Santorini, Greece*, 2008. 2

[14] R. Kiani, H. Esteky, K. Mirpour, and K. Tanaka. Object Category Structure in Response Patterns of Neuronal Population in Monkey Inferior Temporal Cortex. *Journal of Neurophysiology*, 97(6):4296, 2007. 2

[15] C. Lampert, H. Nickisch, and S. Harmeling. Learning To Detect Unseen Object Classes by Between-Class Attribute Transfer. In *CVPR*, 2009. 2, 3

[16] M. Marszałek and C. Schmid. Constructing category hierarchies for visual recognition. In *European Conference on Computer Vision*, volume IV of *LNCS*, pages 479–491. Springer, oct 2008. 2

[17] E. Rosch. Basic Objects in Natural Categories. *Cognitive Psychology*, 8(3):382–439, 1976. 2

[18] J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. Discovering object categories in image collections. In *Proc. ICCV*, volume 2, 2005. 2, 3

[19] D. Weinshall, H. Hermansky, A. Zweig, J. Luo, H. Jimison, F. O, and M. Pavel. Beyond Novelty Detection: Incongruent Events, when General and Specific Classifiers Disagree. *NIPS*, 2008. 2, 3

[20] A. Zweig and D. Weinshall. Exploiting object hierarchy: Combining models from different category levels. In *IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007. 2