Functionality-Driven Object Discovery in Videos

Ian Huang
Columbia University
iyh2110@columbia.edu

Jingwei Ji
Stanford University
jingweij@stanford.edu

Juan Carlos Niebles
Stanford University
jniebles@cs.stanford.edu

Abstract

While deep neural networks have shown considerable promise in large-scale object recognition, their applications to real-world systems are oftentimes limited by their closed set of trained object classes. In this paper, we address the problem of zero-shot object "hierarchical placement" in videos. This is the task of finding the closest super-class of an object from an unknown class within a taxonomy of known object classes. Specifically, we assert that object functionality provides valuable information to infer the placement of unknown object classes within a hierarchy. We present a formulation of functionality that entangles object classes, and introduce an architecture that leverages spatio-temporal information to learn representations of functions of known object classes. At test time, this model utilizes these representations to robustly infer the closest known super-class of unknown classes. Our comparisons with prior works and our own baselines show large improvements, and results from our ablation experiments suggest the value of learning and encoding object functionality for the task of accurately inferring zero-shot hierarchical placement.

1. Introduction

Object recognition [16, 30, 31] and detection [15, 11, 28] have achieved impressive performances with convolutional neural networks [20], and have been thus far utilized in many real-world and commercialized applications, such as robotics and surveillance. In applications to the real world, these systems are often faced with a number of object classes in the order of thousands. These extreme classification problems are oftentimes tackled by exploiting the hierarchical taxonomy present in object classes [22].

More challenging is the "Open-World" setting where instead of a closed set of noun classes, the system encounters unfamiliar object classes. The ability of intelligent systems to make assumptions about the nature of unknown objects remains a challenge. A system that can infer such relationships between unseen object classes and a set of "learned" object classes will allow for more robust interactions between robotic systems and the world, among other important applications.

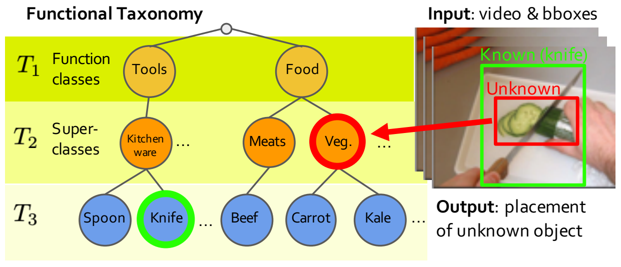

Figure 1. The problem of "hierarchical placement" is that of correctly identifying the closest known object category of an unknown object class. Our approach takes as input video data and bounding boxes of known objects and *one instance* of the unknown object, and uses both the static visual features and the human-object interaction through video to achieve accurate hierarchical placement. Here, the zucchini is an unknown class, and the knife is a known class. The task is to predict the unknown object is a vegetable in a zero-shot manner. Our model exploits a representation of "knife" that entangles other object super-classes to combine with the static visual features of the "zucchini" to predict that it is firstly a food and secondly a vegetable.

Of many possible ways to express this relationship, this paper will focus on the prediction of a new object class's closest known superclass in the taxonomy. The equivalent problem of identifying the placement of a new class within an existing hierarchy has been studied only in static images (see Section 2) and has been coined as the *"hierarchical novelty detection task"* [21]. To disambiguate from novel object class detection, we shall refer to this as "hierarchical placement". We present an approach that is capable of predicting its hierarchical placement among known superclasses from observing its interactions with the other objects around it. See Figure 1 for a concrete example.

Our approach is motivated by cognitive science research that indicate the importance of object functionality in the categorization of object classes within taxonomies [33, 29, 32]. While previously untapped for the purpose of hierarchical placement, object functionality is learned and exploited in our model. We introduce a novel representation of functionality that entangles the knowledge of an object's functionality with the concepts of all other known objects. This is notably effective in video data, as the interactions are not always human-to-object, but also frequently object-to-object. Our formulation permits us to use the knowledge of interactions between known object classes to infer the hierarchical placement of unknown object classes. Concretely, if our model has observed interactions between peelers with many vegetable and fruit classes, then an interaction in which a peeler is used on an unknown object class should bias the model towards predicting the general category of vegetables or fruits. For this to happen, the representation of an object must be entangled with the identity of other object classes.

In order to learn object functionality, we use the spatio-temporal data available in videos to model manipulations of objects and thereby discover their functionality. For the same reason, ego-centric videos are positioned as a prime media, as the human-object interactions they offer can be used to build robust representations of object functionality. For the model presented in this paper, they are sufficiently extracted using the input features of hand pose, hand location and the visual features of the objects upon which they act.

Our evaluation done on the EPIC-Kitchens Dataset shows that our model is able to leverage the temporal information in ego-centric cooking videos to learn helpful representations of object functionality that lead to strong results in zero-shot hierarchical placement. To the best of our knowledge, this is the first system to propose zero-shot hierarchical placement in video data, and moreover the first to do hierarchical placement by analysis of object manipulation and functionality. In comparison with its static-image counterparts on the same novel object classes, we find that our model achieves superior accuracy. Furthermore, our ablation experiments further justify our focus on embedding object functionality and our choice of using hand pose information for the purposes of modeling object functionality. Across our experiments, we notice that our architecture radically improves the marginal accuracy of predictions in deep layers of the taxonomy. Ablation studies corroborate this observation. This tells us that while static-visual features suffice for hierarchical placement in lower depths of the taxonomy, placement in object categories of higher granularity requires a more nuanced understanding of object function and interactions with known object classes, both of which our model addresses.

The main contribution of this paper is two-fold. Firstly, we introduce a formulation of object functionality that not only represents the function of a single object as a combination of other objects and human-object interactions, but also lends itself to a relation-graph embedding approach. Secondly, we present a novel architecture that uses said representations to robustly predict hierarchical placement of unknown object classes.

2. Related Work

Hierarchical classification in images. The technique of using the taxonomy of classes to give better predictions for extreme classification problems has been well studied. Whereas many works have developed good classifiers for classes with latent hierarchical structures by exploiting these structures [5, 4, 24, 25, 13, 9], some have developed embeddings to directly learn the similarity between super classes of object classes [34]. More recently, Lee et al. [21] developed a method that aims to find the closest super class in a hierarchical taxonomy of known classes. To train their model to insert unknown object classes in a hierarchy, they employed a method by which a subset of the known classes are considered to be novel classes. While this work is closest to ours, we extend the "hierarchical novelty detection" task to the video domain through representations of object functionality provided by rich spatio-temporal information inherent in the video modality.

Zero-shot object classification and detection in images. Works that focus on zero-shot object classification can be grouped into models that use visual features like shape, color, pose or geographical information and multi-modal approaches [3]. Some examples of the former include techniques used in [10, 18, 19] for classification. On the other hand, multi-modal approaches rely on the information available about the semantic similarity of the unknown classes to the known classes [1, 2].

More recently, Bansal et al. [3], Rahman et al. [27] and Zhu et al. [38] respectively introduced models that each use semantic information available in word embeddings to do zero-shot classification of the objects. While their experimentation suggests promise of their approach, our approach uses information in the RGB video only, and does not assume anything about the availability of information on the unknown classes in other modalities. This thus makes our setup more suitable for an open-world setting, where unknown classes are truly novel, and not "seen" in some other modality (e.g. introduction of a new consumer product). Additionally, our goal of predicting super classes of unknown object classes instead of the classes themselves, as in [21], makes the problem well-defined without some prior notion of these unknown classes encoded in other modalities.

Modeling object functionality. The relationship between object identity and functionality has long been explored in the community of computer vision. Main approaches include a decomposition of an object's functional components [12] or using human-object interactions (HOIs) involving

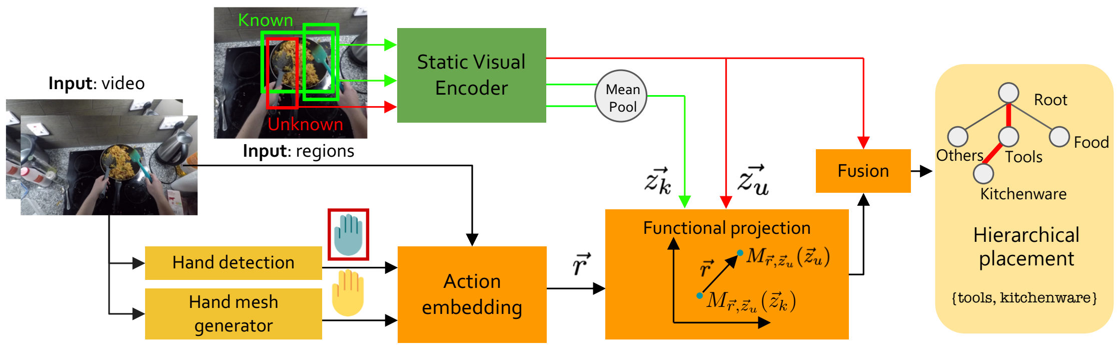

Figure 2. Architecture of our full system. The yellow modules are pretrained, and frozen during training. The green module is the static visual encoder, and is pretrained on a held-out subset of frames featuring classes in K. The orange modules correspond to trainable parameters in the Functionality-Driven Hierarchical Placement (FDHP) network. It works to use spatio-temporal data to recover object functionality, and then use this to correct the embeddings outputs by the static visual encoder. The output hierarchical placement corresponds to that of the unknown object class in the red bounding box on the left.

individual objects [37, 17]. Unlike us, none of the aforementioned works attempt the task of inferring the hierarchical placement of unknown object classes using object functionality. In the works of Yao et al. [37] on static images and Jelodar et al. [17] in videos, functionality is defined by human-object interaction and actions, respectively, and does not consider interactions with other object classes. In this paper, we formulate the functionality of objects to be intertwined with other object classes. We observe that in real life, functionality of objects is often defined by their interactions with other objects. For example, a screwdriver cannot be purely understood in terms of the HOIs. It must also be entangled with the concept of a screw. This formulation naturally provides more information in video data, where useful cues can be obtained through the observation of object-to-object interactions mundane in video data. Thus, our formulation is essential to zero-shot hierarchical placement in video data.

Modeling affordances. Fang et al. [8] predicts affordances by using an embedding that encapsulates both visual and HOI information. However, they do not consider object functionality or attempt to infer the hierarchical placement of unknown object classes. Additionally, Nagarajan et al. [26] presented a model that learns human-object interactions (HOI) and predicts the "hotspots" of the object at which these interactions occur. They show that their models is also able to anticipate hotspots of HOI on unknown object classes. This suggests that knowledge of the affordances and function of objects may be nicely transferable between the domains of seen and unseen object classes, and thus justifies our focus on learning to extract and represent the function of objects as a key boost for hierarchical classification of unknown object classes.

Action-driven object classification. Prior works in object classification in videos have shown that actions are able to provide weak supervision on the object category. Yang et al. [36] present that this additional weak supervision through the use of action labels allows their approach to outperform then-SOTA object classification methods by more than 6% mAP on the charades video dataset. This suggests that there is potential in using representations of actions to supplement missing knowledge about object classes.

3. Approach

See Figure 2 for an overview of the architecture. Given a sequence of frames, a set of bounding boxes for known objects and a bounding box for an unknown object in a single frame, we would like to build a model that accurately performs hierarchical placement. To that end, we formulate such a problem as a two-stage classification system. First, a static-visual encoder pretrained on the set of known classes projects the unknown object class into its latent space. Then, a model that considers object functionality and human object interaction corrects this projection and uses the resultant embedding to achieve hierarchical placement. In the following sections, we will first present the formal formulation of the problem, then provide further details to the design of both the pretrained static-visual encoder and the model as a whole.

3.1. Formulation

Let C be the universal set of classes, and $U, K, \tilde{U}$ be disjoint partitions of C. U and K are respectively known and unknown sets accessible during training, and $\tilde{U}$ is a set of unknown object classes used during testing. Each class within C is represented as a leaf node under a taxonomy T of depth d. T_i shall be used to denote the set of object categories at depth i, with T_0 being a singleton set containing only the root, as shown in Figure 1. Using known and unknown sets K and U during training, we

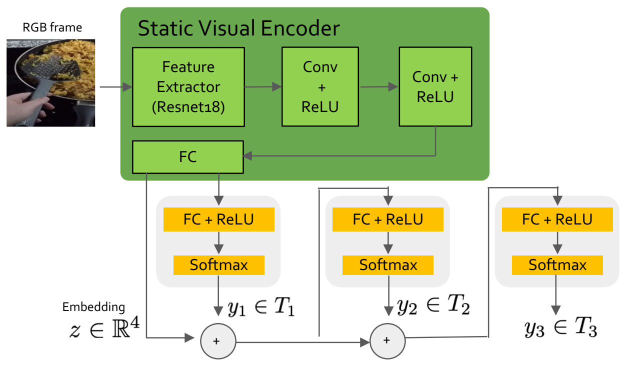

Figure 3. Architecture of the static visual encoder and classification head. The green modules make up the static visual encoder in Figure 2). Note the skip connections in the classification head, and how it allows each deeper level of hierarchical placement to be based on the placement at the previous depth.

would like our model F to produce the correct predictions for $T_1, T_2 ... T_{d-1}$ over all unseen classes in $\tilde{U}$ (e.g. to decide that an unknown object is firstly a food, then a fruit ...etc). That is, we would formally like to find a function

$$F : X_{\tilde{U}} \times \prod_{i=1}^{t} B_k^{(i)} \times \hat{B}_u \to \prod_{i=1}^{d-1} T_i \, , \tag{1}$$

where $X_{\tilde{U}}$ is the space of sequences of frames featuring objects of unknown class $u \in \tilde{U}$ and T_i is the set of hierarchical categories at depth i. Note that the root of the hierarchy T_0 is excluded and the leaf nodes T_d are also excluded, since u is not yet a leaf node in the hierarchy of known classes. $B_k^{(i)} \subset \mathbb{R}^4$ is the bounding box at the ith frame bounding a known object. $\hat{B}_u$ is the bounding box localizing the unknown object in a single frame.

We form F as a composite of two maps $f \circ g$. While g captures the information of known classes, f is uses the outputs of g to infer the placement of unknown classes. To this end, g is a neural network that is pretrained on the set K using only on static visual features and follows a neural architecture indicated in Figure 3. Let us call g the *static visual encoder*. See complete details in Section 3.2. We shall refer to f as the *Functionality-Driven Hierarchical Placement* (FDHP) network.

Since visual features are insufficient in predicting object functionality, we use human-object interaction features to augment the representational output of the static visual encoder. In parallel with the static visual encoder, FDHP uses video data to develop representations of functions of different known objects (bottom row of Figure 2). We assert that these representations of object functionality are more robust to visual domain shift when new classes are shown. At test time, visual information from known objects that interact with the unknown objects will be used to retrieve a representation for the most likely subjects upon which they act, thus utilizing information from the affordances of the unknown object (see **Object functionality** and **Learning and using functional projections** in Section 3.3). This representation is then used to correct outputs of the static visual encoder to arrive at a more accurate hierarchical placement.

3.2. Static visual encoder

The static visual encoder g is used to encode the set of classes in K using a condensed representation of these classes. To develop these representations, we give g the auxiliary task of predicting the hierarchical categories from $i = 1$ to $i = d$. That is, $g : V_K \times B_k \to \prod_{i=1}^{d} T_i$, where V_K is the space of static images featuring an object of class $k \in K$, and B_k is a bounding box localizing the object instance.

After training the static visual encoder to perform hierarchical placement using the architecture in Figure 3, we use the output of the static visual encoder during the training and testing of FDHP. The architecture's skip connections in the classification head allow for the predictions for T_i to depend on the outputs for $T_1 ... T_{i-1}$ as illustrated in Figure 3.

To train this network, we use a loss inspired by the hierarchical loss [35]. To prevent the network from taking advantage of the class imbalance, we apply a class reweighting inverse to the frequency of $y \in \prod_{i=1}^{d} T_i$:

$$\mathcal{L}(\theta) = -\frac{1}{|V_K|} \sum_{(x,y) \in V_K} \left[\log\left(\frac{1}{Pr(y)}\right) \times \log\left(\sum_{i=1}^{d} 2^{-i} \right.\right. \\ \left.\left. Pr(g(x;\theta)_i = y_i) + 2^{-d} Pr(g(x;\theta)_d = y_d) \right) \right] \tag{2}$$

During our experiments, g is pretrained on a subset of frames featuring classes in K, and this subset is not used during the training of zero-shot hierarchical placement.

3.3. Hierarchical placement with FDHP

Object functionality: Key to our method of inferring the low-depth hierarchical categorization of unknown objects is an understanding of the functionality of known objects. If an unknown object is seen interacting with a known object, the nature of that interaction, if similar to ones observed before in the training data, gives valuable information to hierarchical placement of unknown object classes. To that end, we say that the functionality of an object class is parametrized by the set of all observed interactions with other object classes. Formally, the functionality of an object class can be considered as a probability distribution over the set of tuples (p, a, q) for an actor-action-recipient relation, where $p, q \in K \cup U$ and a is a representation of an action. In order for such a representation to be learned and used in

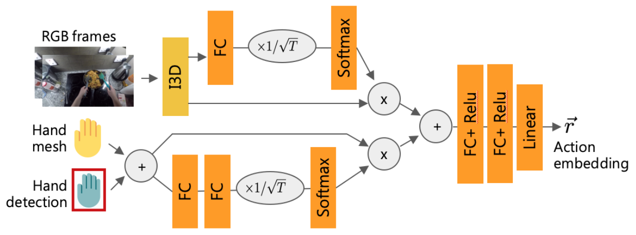

Figure 4. Architecture of action embedding module. Indices corresponding to classes in T_1, T_2 and T_3 are outputted. The prediction for T_i depends on the prediction of shallower layers $T_1 \ldots T_{i-1}$ through skip connections.

a neural network, the network learns to predict the vector representation of q given those of p and a. Let the mapping to these vector representations of q and p be M. The functionality of an an object class p should then be recoverable by $\{(p, a, M(p) + a) \mid \forall a \in A_p\}$, where A_p is the set of actions observed with object class p. To use this knowledge of object functionality, it is necessary to correctly determine a, and use M to find the most likely vector representation of the recipient of the observed action. This representation is then used to correct the output of the static visual encoder.

Action embedding: Using hand mesh prediction networks [14], we are able to get over all frames hand pose information $h_p \in \mathbb{R}^{t \times D_p}$ and hand position information $h_b \in \mathbb{R}^{t \times D_b}$. We fuse the RGB video modality $v \in \mathbb{R}^{t \times 3 \times W \times H}$ and hand information together through an architecture shown in Figure 4 that implements self-attention on both modalities separately before applying fully connected layers to the concatenation of both outputs. The resultant action embedding is the representation of a used to find the appropriate functional projection transform.

Learning and using functional projections: The task of representing object functionality lends itself to the problem of representations for relational graph [23]. Rather than using a discrete set of actions, we use a continuous action embedding $\vec{r}$ output by the action embedding module with RGB data, hand pose and location.

For each $\vec{r}$ and $\vec{z}_u$, we define linear transformation $M_{\vec{r},\vec{z}_u}$ under which embeddings $\vec{z}_u$ and $\vec{z}_k$ are $\vec{r}$ apart in the embedded relation space. That is,

$$M_{\vec{r},\vec{z}_u}(z_u) = M_{\vec{r},\vec{z}_u}(z_k) + \vec{r} \tag{3}$$

We define $M_{\vec{r},\vec{z}_u}(x) = (A\vec{r})(B\vec{z}_u)^T x$, where A and B are learned parameters. At test time, the module outputs $M_{\vec{r},\vec{z}_u}(\vec{z}_k) + \vec{r}$.

Since many interactions can exist between the same objects in videos (e.g. the tuples $(\text{hand}, \text{washes}, \text{apples})$ and $(\text{hand}, \text{throws}, \text{apples})$ are equally as legal), it would be inadequate to map the entities into a singular relation space. [23] Moreover, as we'd like our action embedding spaces to be continuous, we want similar action embeddings to yield similar transformations (the mapping from the action embedding space to the space of linear transformations must also be continuous). This is indeed the case by our construction.

Fusion module: As stated 3.1, we view the role of the fusion model as a correction on z_u given by $\delta = FC([M_{\vec{r},\vec{z}_u}(\vec{z}_k) + \vec{r}, \vec{z}_u])$. The intuition behind this is that part of the dense layers will learn something akin to a projection back to the entity space from the relationship in which $\vec{r}$ resides, and the remainder of the fusion module will be used to moderate between $\vec{z}_u$ and $M_{\vec{r},\vec{z}_u}(\vec{z}_k) + \vec{r}$. The final output is

$$\tilde{\vec{z}}_u = \vec{z}_u + \delta$$

$\tilde{\vec{z}}$ is then used to obtain a level-wise category prediction using the same architecture as the classification head of g (refer to Section 3.2). Note that during training, the weights are not shared between the two.

Since the task of FDHP is similar to that of the static visual encoder, it is trained to a loss with a minor change that measures only its efficacy up to depth $d-1$ of T over the set of clips featuring unknown classes. That is,

$$\mathcal{L}(\theta) = -\frac{1}{|X_U|} \sum_{(x,y) \in X_U} \left[\log\left(\frac{1}{P(y)}\right) \times \log\left(\sum_{i=1}^{d-1} 2^{-i} \right.\right. \\ \left.\left. Pr(f(x;\theta)_i = y_i) + 2^{-(d-1)} Pr(f(x;\theta)_{d-1} = y_{d-1}) \right) \right] \tag{4}$$

4. Experiments

4.1. Dataset

All experiments are done on the EPIC Kitchens Dataset [6], chosen for its size and hierarchical diversity of items, distributed across 331 noun classes. The dataset website supplies a three-layer hierarchy (root, general class category, and classes), upon which we impose additional hierarchical structure to end with the hierarchical tree displayed in Figure 5. Note that the layer we inserted is the set of nodes $\{\text{food}, \text{tools}, \text{others}\}$.

To determine the splits for U, K, $\tilde{U}$, we firstly assert that since our feature extractor [16] is pre-trained on Imagenet, the intersection $C' = C \cap (C_{Imagenet})$ must be a subset of K. In practice, we make $C' = C \cap (C_{Imagenet} \cup C_{MSCOCO})$, just for good measure. This is to ensure that the training unknown U and testing unknowns $\tilde{U}$ are not contaminated by samples that the visual feature extractor have already seen. The cardinality $|C'| = 76$. As the label names for the same categories are oftentimes different across datasets, finding this intersection was done manually.

In the experiments that follow, we let the set $K = C'$. U and $\tilde{U}$ are thereafter equal partitions of $C - K$. If the set of children of category $s \in T_k$ is denoted $\mathcal{C}(s)$ and if for set of classes A, $A_s = \{a \mid a \in A, a \in C(s)\}$, we require that

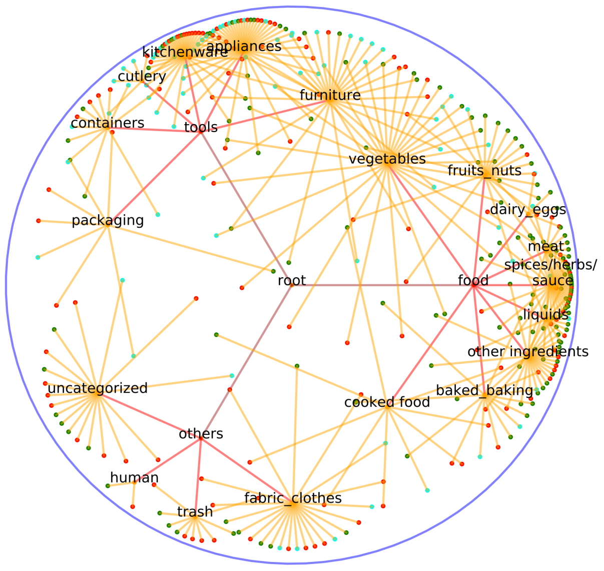

Figure 5. Hyperbolic embedding plot of the object class hierarchy of EPIC Kitchen, by Sarkar's Construction [7]. For the purposes of readability, the leaf node labels have been omitted. The full list of object classes can be found in [6] or on their website. Cyan leaf nodes represent classes in K, green leafs represent classes in U, and red leafs represent classes in $\tilde{U}$. Best viewed in color.

Table 1. Comparisons against baseline variants on function classes and super-classes. The results in italics correspond to oracles pretrained on the test data, and as such they serve as performance upper bounds. We achieve better accuracy than both variants of the static visual encoder in both classification of function classes and of super-classes.

Model	T_1 acc	T_2 marg. acc.	T_2 cum. acc.
Stat. Visual Encoder	0.51	0.24	0.12
Stat. Video Encoder	0.44	0.32	0.14
Stat. Visual Oracle	*0.69*	*0.42*	*0.29*
Stat. Video Oracle	*0.74*	*0.54*	*0.40*
FDHP (ours)	**0.53**	**0.53**	**0.28**

partitions U and $\tilde{U}$ be chosen such that $\forall s \in T_{d-1}, \mathcal{C}(s) - K_s = U_s \cup \tilde{U}_s$ and $|U_s| = |\tilde{U}_s|$. This construction would guarantee that during test time, unknown objects would not be the first of a brand new category. In practice, $|U_s| \approx |\tilde{U}_s|$.

A distribution of the different hierarchical categories in K, U and $\tilde{U}$ can be seen in Figure 5. T is a tree of depth 3, and we shall refer to T_1. T_2 and T_3 as the set of "function classes", "super-classes" and "noun classes", respectively.

4.2. Pretraining the static visual encoder

Conforming to the aforementioned splits, pretraining of g is done using a held-out subset of X_K. That is, g is trained on a subset of static frames featuring the object of a class in K. Since g is only meant to learn embeddings of visual features, we crop these frames according to the ground truth bounding boxes of the classes, and then resize it to a 224×224 image using bilinear interpolation.

For the base network, we choose Resnet18 [16] as g's feature extractor, and train it using the weighted empirical hierarchical loss in Equation 2.

4.3. Hierarchical placement with FDHP

During the end-to-end training of FDHP, we provide the output embeddings $\vec{z}_u$ and $\vec{z}_k$ given by the correct bounding boxes. We choose the known and unknown bounding boxes for $\vec{z}_u$ and $\vec{z}_k$ as the ones closest to the locations of hands in the image, as given by the hand detection modules in Figure 2. In corner cases when no hands are available during a frame or no known objects exist, we set both resultant feature vectors to $\vec{0}$.

During training, hierarchical placement over the set X_U is only done to depth $d-1$, thus leaving out the requirement that it outputs the correct object class (as this is clearly unknown). We train the network in Figure 2 using the loss in Equation 4.

During testing, we report the accuracy score at each layer of f's predictions on $X_{\tilde{U}}$. Note that because of the sequential nature in which our models predict the category, the a "correct" prediction (in terms of logit output of a neural network) at depth i is only considered correct if it has correctly selected the higher level category at depth $i-1$.

4.3.1 Comparison with the static visual encoder

Comparison with static visual encoder: We compare our model with the static visual encoder g (and the pretrained classification head). That is, we evaluate the performance of g on $V_{\tilde{U}}$ with the performance of f on $X_{\tilde{U}}$.
Comparison with static video baseline: To ensure that the difference in performance is not due to the different dataset or the existence of temporal information in one versus the other, we also compare against a video baseline using the static visual encoder. The latter makes a prediction on $x \in X_{\tilde{U}}$ by classifying every instance of the unknown object within frames of x and returning a majority-vote decision across all layers of all predictions. This will be referred to as the *static video encoder*.
Comparison with static visual oracle $\hat{g}$: As an upper bound for the performance, we pretrain the static visual encoder on the test data $V_{\tilde{U}}$, and call this network $\hat{g}$. Comparison of our hierarchical placement model FDHP against the static visual oracle will tell us the relative accuracy that FDHP achieves in relation to the upperbound of accuracy attainable using the architecture in Figure 3. A static video oracle, $\hat{g}$, is similarly constructed using the same approach described for the static video encoder.

See Table 4.2 for comparisons with the baselines on the

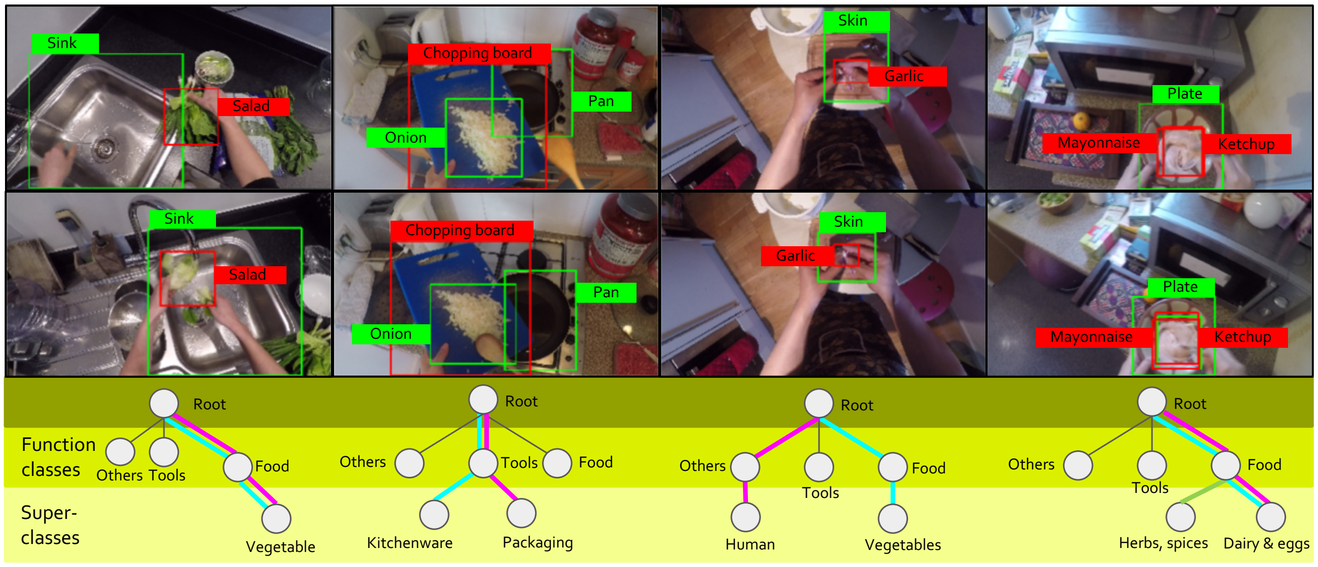

Figure 6. Qualitative samples of predictions by our model. Each column represents a two frames from a test sample (columns from left to right are (a), (b), (c) and (d)). All cyan indicates the predictions of our model whereas magenta indicates that of our static visual encoder g. Known objects at test time are in green and unknown are in red. The final hierarchical placements correspond to the unknown objects bounded in red. (a) Our model correctly infers that the unknown object belongs in the category of "vegetables", but so does the vision baseline model. This is an instance where the visual characteristics alone are enough to achieve correct hierarchical placement. (b) Our model correctly infers that the unknown item falls under the super-class of kitchenware. Meanwhile, our static visual encoder is correct at the function class level but incorrect at the super-class level. (c) Our model correctly infers that the object is a "vegetable", under the category of foods. Since the garlic is largely occluded by the hands, this shows that our model's performance is not hinged on visual features alone. In contrast, we see that the static-visual feature fails at both layers of the hierarchy. (d) Both models predict that the object in question is in the category of "dairy & eggs", but are both incorrect. The ground truth is indicated using the green edge. Best viewed in color.

accuracy over function classes and super-classes. We see that in terms of function class accuracy, our model offers a slight boost in performance (2%) over the static visual encoder. The discrepancy in performance between our model and both static and video variants of the static visual encoder is larger for the super-classes (e.g. vegetables vs. fruits, appliances vs. utensils), with an increase of 6% in comparison to the static visual encoder and 13% in comparison to the static video baseline. This shows that object functionality information is especially important in differentiating between categories of higher granularity in the taxonomy. That is, static visual features like color and texture may easily be enough to differentiate between food and tools (thus explaining the small marginal benefit that object functionality provides for placement among function classes), but are not enough to accurately differentiate objects for lower levels of the hierarchy (e.g. differentiating between fruits and vegetables). In this setting, the information about the function of known objects provides large benefit to the hierarchical placement among super classes. See Figure 6 for some qualitative samples during testing. They show that in the presence of known objects, FDHP uses object functionality to achieve superior zero-shot placement.

4.3.2 Comparison with prior works

Comparison with prior work: Lee et al. [21] proposed a method using static visual features. Since they are the only related work within recent years to address hierarchical placement in static images, they are our biggest benchmark. As mentioned in [21], they leverage WordNet to generate hierarchies based on noun labels in static image datasets. For this experiment, our hierarchy (Figure 5) is used instead. Three variants of their approach - TD, TD+LOO, and LOO - were retrained on the static images of $V_{\tilde{U}}$, exactly as according to their implementation. Note that their architecture does not adopt the same layer-wise prediction formulation as ours, but rather predicts the identity of the closest parent of the unknown object class in a single layer. Thus, for the purposes of evaluating layer-wise accuracy, we adopt the following rules: 1) if their model predicts the right super-class, then it is correct in all layers before that parent. 2) if the model predicts the wrong parent in T_2, then we evaluate whether the parent of this super-class is the same as the parent of the ground truth in T_1. This allows us to develop layer-wise accuracies for their model.

Table 4.3.2 shows the layer wise accuracy of their ap-

Table 2. Comparisons with Lee et al.'s Top-Down (TD), Leave-One-Out (LOO) and combinational (TD+LOO) approaches. We achieve higher accuracy than the 3 variants of Lee et al.'s approach on these of known classes. We observe that the marginal accuracy for super-classes is much higher than that of the static image counterparts, as also observed in Table 4.2.

Model	T_1 acc	T_2 marg. acc.	T_2 cum. acc.
TD [21]	0.10	0.28	0.03
LOO [21]	0.70	0.32	0.22
TD+LOO [21]	**0.72**	0.31	0.22
FDHP (ours)	0.53	**0.53**	**0.28**

Table 3. Comparison with the best performing variant of Lee et al.'s approaches, as from Table 4.3.2 and results of [21], but replacing their Resnet101 feature extractor with our static visual encoder. We see that our model performs noticeably better in super-class accuracy, and also achieve superior results in function class accuracy.

Model	T_1 acc	T_2 marg. acc.	T_2 cum. acc.
TD+LOO [21]	0.44	0.17	0.08
FDHP (ours)	**0.53**	**0.53**	**0.28**

proaches. We provide our model's performance as a reference. We achieve a 6% improvement on their best approaches in cumulative accuracy, and achieve a substantial increase (21%) in the marginal super-class accuracy that enables the aforementioned net 6% increase, despite the lower function-class accuracy. As observed in the comparisons against static visual baselines in Table 4.2, our model's strength lies in providing much stronger marginal accuracy to lessen the decay of accuracy as the model decides between categories at increasing depths. This may suggest that the additional spatio-temporal and action-related features are more valuable in differentiating between lower-level hierarchical categories.

Comparison with prior work using the static visual encoder: Note that while we use Resnet18 as a backbone, their models by default use Resnet101. In order to fairly evaluate the two approaches' efficacy in *correcting* the embedding space of a pretrained static visual encoder, we use the same encoder for their model and ours. We compare with their highest scoring variant from [21] and in Table 4.3.2. Results of this experiment is shown in Table 4.3.2. Once both models are given the same feature extractor (our static visual encoder pretrained on the known classes K), we find that our model achieves a 20% improvement from Lee et al.'s approach. Under the conditions of a simpler feature extractor, we achieve higher accuracy across all metrics, including accuracy for function classes.

Table 4. Ablation studies show the importance of the function encoding modules, action embedding, and the importance of hand position and pose in the marginal accuracy. The functional projection module in Figure 2 is very important, as its ablation caused both a notable drop in function class accuracy and 22% drop in super-class accuracy. These results also show that the model's super-class accuracy degrades by more than half without action embeddings, thus showing its importance in our formulation of object functionality representation. We see that overall, the marginal accuracy is the most volatile to ablations. This furthermore shows that our model's largest contributions are at super-class level.

Model	T_1 acc	T_2 marg. acc.	T_2 cum. acc.
w/o action embedding	0.51	0.21	0.11
w/o functional projection	0.43	0.15	0.06
w/o unknown vis. feat.	0.50	0.08	0.04
w/o known vis. feat.	0.52	0.47	0.24
w/o hand features	0.51	0.14	0.07
w/o video features	0.53	0.09	0.05
FDHP (ours)	**0.53**	**0.53**	**0.28**

4.3.3 Ablation Experiments

We ablate different modules and inputs of the model separately to gauge the importance of the different components. Ablations of modules are implemented by concatenating all inputs and then relayed as outputs. Ablations of inputs are simply implemented by replacing them with zeros. Table 4.3.3 shows the results of individually ablating different modules and inputs of our architecture. Note that the pretrained static visual encoder was untouched.

By these results, we see that our model's ability to differentiate super-classes depends heavily on the action embedding and function projection modules. We furthermore note that the small fluctuation in function class accuracy corroborates the understanding that static features (provided by the un-ablated static visual encoder throughout all ablation experiments) can provide enough information to differentiate among higher levels of the hierarchy, but are insufficient in lower layers. We see especially that the removal of hand-pose and hand location data is detrimental to the marginal accuracy of super-classes. These results therefore show the importance of using hand-related features to understand actions and function of objects, and show that they are crucial during hierarchical placement.

5. Conclusion

Our method addresses the problem of hierarchical placement in a way that exploits cues beyond the static visual in ego-centric videos. We show that by representing the interactions between known objects, such interactions can naturally give rise to an understanding of object function that provides useful information for the zero-shot hierarchical placement of unknown object classes, leading to higher placement accuracy.

References

[1] Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid. Label-embedding for attribute-based classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 819–826, 2013.

[2] Zeynep Akata, Scott Reed, Daniel Walter, Honglak Lee, and Bernt Schiele. Evaluation of output embeddings for fine-grained image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2927–2936, 2015.

[3] Ankan Bansal, Karan Sikka, Gaurav Sharma, Rama Chellappa, and Ajay Divakaran. Zero-shot object detection. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[4] A Beygelzimer, J Langford, Y Lifshits, G Sorkin, and A Strehl. Cond. prob. tree estimation analysis and algorithms. In *UAI*, 2009.

[5] Lijuan Cai and Thomas Hofmann. Hierarchical document categorization with support vector machines. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 78–87. ACM, 2004.

[6] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 720–736, 2018.

[7] Christopher De Sa, Albert Gu, Christopher Ré, and Frederic Sala. Representation tradeoffs for hyperbolic embeddings. *Proceedings of machine learning research*, 80:4460, 2018.

[8] Kuan Fang, Te-Lin Wu, Daniel Yang, Silvio Savarese, and Joseph J Lim. Demo2vec: Reasoning object affordances from online videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2139–2147, 2018.

[9] Rob Fergus, Hector Bernal, Yair Weiss, and Antonio Torralba. Semantic label sharing for learning with many categories. In *European Conference on Computer Vision*, pages 762–775. Springer, 2010.

[10] Vittorio Ferrari, Manuel Marin-Jimenez, and Andrew Zisserman. Pose search: retrieving people using their pose. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2009.

[11] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[12] Kevin Green, D Eggert, Louise Stark, and K Bowyer. Generic recognition of articulated objects by reasoning about functionality. In *Proceedings of 12th International Conference on Pattern Recognition*, volume 1, pages 847–849. IEEE, 1994.

[13] Gregory Griffin and Pietro Perona. Learning and using taxonomies for fast visual categorization. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[14] Yana Hasson, Gul Varol, Dimitrios Tzionas, Igor Kalevatykh, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning joint reconstruction of hands and manipulated objects. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015.

[17] Ahmad Babaeian Jelodar, David Paulius, and Yu Sun. Long activity video understanding using functional object-oriented network. *IEEE Transactions on Multimedia*, 2018.

[18] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 951–958. IEEE, 2009.

[19] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):453–465, 2013.

[20] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[21] Kibok Lee, Kimin Lee, Kyle Min, Yuting Zhang, Jinwoo Shin, and Honglak Lee. Hierarchical novelty detection for visual object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1034–1042, 2018.

[22] Li-Jia Li, Chong Wang, Yongwhan Lim, David M. Blei, and Fei-Fei Li. Finding task-relevant features for few-shot learning by category traversal. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[23] Zhihui Li, Lina Yao, Xiaoqin Zhang, Xianzhi Wang, Salil Kanhere, and Huaxiang Zhang. Zero-shot object detection with textual descriptions. In *The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)*.

[24] Marcin Marszalek and Cordelia Schmid. Semantic hierarchies for visual object recognition. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7. IEEE, 2007.

[25] Marcin Marszałek and Cordelia Schmid. Constructing category hierarchies for visual recognition. In *European Conference on Computer Vision*, pages 479–491. Springer, 2008.

[26] Tushar Nagarajan, Christoph Feichtenhofer, and Kristen Grauman. Grounded human-object interaction hotspots from video. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[27] Shafin Rahman, Salman Khan, and Fatih Porikli. Zero-shot object detection: Learning to simultaneously recognize and localize novel concepts. In *Asian Conference on Computer Vision (ACCV-2018)*, May 2019.

[28] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Proceedings of the 28th International*

Conference on Neural Information Processing Systems - Volume 1, NIPS'15, pages 91–99, Cambridge, MA, USA, 2015. MIT Press.

[29] Eleanor Rosch. Principles of categorization. *Concepts: core readings*, 189, 1999.

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations*, 2015.

[31] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[32] Barbara Tversky and Kathleen Hemenway. Categories of environmental scenes. *Cognitive psychology*, 15(1):121–149, 1983.

[33] Barbara Tversky and Kathleen Hemenway. Objects, parts, and categories. *Journal of experimental psychology: General*, 113(2):169, 1984.

[34] Nakul Verma, Dhruv Mahajan, Sundararajan Sellamanickam, and Vinod Nair. Learning hierarchical similarity metrics. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2280–2287. IEEE, 2012.

[35] Cinna Wu, Mark Tygert, and Yann LeCun. Hierarchical loss for classification. *arXiv preprint arXiv:1709.01062*, 2017.

[36] Zhenheng Yang, Dhruv Mahajan, Deepti Ghadiyaram, Ram Nevatia, and Vignesh Ramanathan. Activity driven weakly supervised object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2917–2926, 2019.

[37] Bangpeng Yao, Jiayuan Ma, and Li Fei-Fei. Discovering object functionality. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2512–2519, 2013.

[38] Pengkai Zhu, Hanxiao Wang, and Venkatesh Saligrama. Zero shot detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 2019.