From Object Classification to Naming: Modeling Entry-level Names for Real-world Objects

Anonymous ACL submission

Abstract

Object detection and image classification models from Computer Vision are typically trained towards predicting labels of objects. Our goal is to extend such models to entry-level naming, where the task is to predict the natural name of an object that most humans would choose, instead of an arbitrary name chosen by at best a few annotators. Based on large-scale, thoroughly verified object naming data from many annotators, we conduct an extensive analysis of the ability of object detection and classification models to account for entry-level object naming. We show that standard object classification models "spontaneously" learn good entry-level names and perform better on predicting entry-level than arbitrary names, despite being trained on the latter, and that we can use transfer learning to train entry-level classifiers on the naming data. Our detailed error analysis reveals that models generally make more mistakes when human agreement is low, with inadequate predictions due to bounding box issues being a major source of errors.

1 Introduction

Objects, being members of many categories, can be called by many names (e.g., a duck can be called *duck*, *bird*, *animal*, etc.). The task of *object naming*—generating not just a formally correct but also an appropriate, *natural* name for an object—is distinct from the closely related tasks of object detection and image classification (see below). It has been studied in psycholinguistics (e.g., Rosch 1978), but has received comparatively little attention in language & vision (L&V) and computational linguistics (CL) research. We address the following question:

Q: Can standard object detection/image classification models (be fine-tuned to) exhibit human-like object naming behavior?

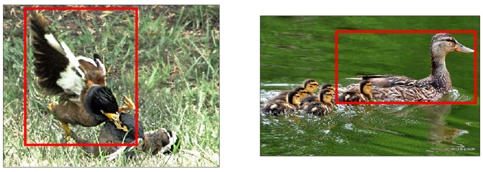


Figure 1: Different naming preferences (frequencies) for instances of the same class *duck* in ManyNames.

In particular, we test several models for their ability to predict *entry-level names*, i.e., the names people prefer when calling a specific object (e.g., *bird* in Figure 1, left image). A second contribution of this paper is instrumental: for a proper evaluation we must first augment an existing human object naming dataset (ManyNames, Anonymous, under review) with extensive quality control, the resulting data of which we also describe here.[1]

In L&V, object naming underlies virtually all tasks that model how speakers refer to objects in the world, such as image description or visual question answering. L&V methods are generally based on object detection or image classification models from computer vision (CV) that were pre-trained towards predicting the single label of each object that is deemed correct by the dataset. The labels used often seem unintuitive from a human perspective, e.g., some are very basic, natural names (*bus*) while others are much more specific (*goblet*, *gyromitra*), cf. the label inventory of the ILSVRC challenges (Russakovsky et al., 2015). Pre-training on such labels can enable CV models to learn rich, discriminative visual feature representations which capture fine-grained differences in object appearances (e.g., sharp-pointed vs. slightly pointed). But it raises the issue of whether these models could be

[1] Data and code for this paper will be made available upon publication.

directly used for more human-like object naming.

Psycholinguistic studies have found that humans have a preference towards a particular name, defined as the *entry-level name*, when naturally naming an object (Rosch et al., 1976; Rosch, 1978; Jolicoeur et al., 1984). However, this research has traditionally focused on classes and/or their prototypical/schematic depictions (e.g., Rossion and Pourtois 2004), as opposed to the situated instances in naturalistic images with which L&V is mostly concerned. But humans may prefer different names for instances of the same class (e.g., Figure 1), and even disagree in their choice for the same instance (see also Graf et al. 2016).

Hence, both the single-ground-truth view of CV models and the category-level approach predominant in psycholinguistics are insufficient for assessing the effectiveness of L&V models for human-like, entry-level object naming. The ManyNames (MN) dataset was designed to help mend this gap: it consists of names entered by 36 annotators in parallel, for 25K objects in naturalistic images from VisualGenome (Krishna et al., 2016). However, the annotations of ManyNames have not been verified to filter out incorrect names or names for the wrong object, a problem that is in some sense the opposite of that posed by the single-ground-truth approach.

Our main contributions are to (i) run the ManyNames data through a rigorous annotation round to quantify both the adequacy of its names and the types of errors they contain, in order to (ii) test and compare object detection and image classification models at the task of *instance-based, entry-level object naming*. More precisely we define entry-level name as follows:

Entry-level name: the most common name given to an object instance according to ManyNames

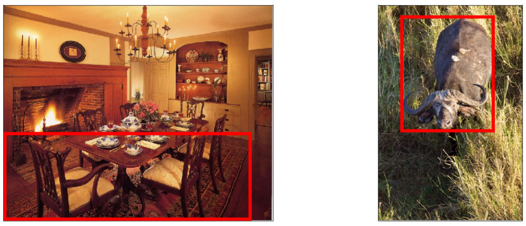
We evaluate the effectiveness of different models on predicting these entry-level names, and our additional verification annotations enable us to understand the kinds of mistakes the models make (e.g., a 'wrong' prediction may be a less preferred name as opposed to a true error). We show that generic object classification models on their own already learn to provide entry-level names for objects, but also that they can transfer-learn on ManyNames, successfully using pre-trained features from object detection and image classification models to train entry-level naming models.

2 Related Work

In psycholinguistics, studies of picture naming have found that humans identify objects at a preferred level of abstraction, the so-called basic-level (Rosch et al., 1976; Jolicoeur et al., 1984). The typicality of the object with respect to this basic-level is important for determining the preferred name, i.e. the entry-level name: typical objects (e.g., a robin) are simply named by their basic-level class (*bird*), while atypical objects (e.g., *penguin*) are preferably named by the more specific class. Ordonez et al. (2016) adopt this and train a model to map an object's class detected by an object recognition system to a name, guided by external resources like corpus statistics. By contrast, we use ManyNames (details below) to directly test and fine-tune object recognition and image classification systems, without presupposing a taxonomical relationship between the possible names for an object.

Our approach is closer in this regard to Zarrieß and Schlangen (2017), who train an object naming model from names produced in referring expressions for real-world objects. However, they do not have access to name annotations from (many) different annotators, whereas the additional annotations we collect enable more fine-grained evaluation. The ManyNames dataset on which we rely is akin in spirit to older picture naming datasets (e.g., Rossion and Pourtois 2004) but focuses on naturalistic images of real-world objects instead of prototypical line drawings.

In Computer Vision a major focus is visual object recognition, with object classification being a core task. Neural image classification systems are trained to label the most salient object in an image and often use the ImageNet (Deng et al., 2009) benchmark that labels images/objects with 1,000 fine-grained classes (Szegedy et al., 2015; He et al., 2016). These neural classifiers have proven extremely useful as pre-trained feature extractors in object detection systems that localize objects and predict their labels (Ren et al., 2015). For instance, a commonly used object detector in L&V is Anderson et al. (2018)'s so-called bottom-up model, which is based on a pre-trained ResNet classifier and fine-tuned Faster-RCNN (Ren et al., 2015) for object detection and classification on VisualGenome. We will use pre-trained ResNet and Faster-RCNN as models in this paper, extending and testing them for entry-level object naming.



Entry-level: rug (21)	animal (9)
Same-object: carpet (4, 0.7)	yak (3,0.8), buffalo (2,0.7)
Error-box: chair (5, 0.5), floor (3, 0.5)	
Error-visual:	cow (6,0.5), ram (3,0.7), bull (2,0.7)
FRCNN$_{\text{VG1600}}$: floor	animal
FRCNN$_{\text{VG1600–VGMN}}$: rug	cow

Figure 2: Name (no. of annotations, adequacy rating), verification and predictions for two objects

The ManyNames Dataset The ManyNames dataset (MN, Anonymous, under review), which builds upon object annotations in VisualGenome (VG), provides up to 36 name annotations for 25K objects in images, see e.g. Figure 1. The large number of name annotations per object in ManyNames gives us a way to reliably derive entry-level names, defined as the most frequent name for an object instance. Its total vocabulary of names has 7,970 types; restricted to entry-level names, it has 442 types (still covering over 50% of the object instances in VisualGenome). That these objects are diverse and visually situated in real-world scenes aligns with our view that entry-level names should be characterized at an instance-level, not at the class-level. However, our manual inspection of the ManyNames data revealed a range of annotation problems, e.g., annotators occasionally named an object different from the one highlighted by the bounding box, or the right object but with an incorrect name (see Figure 2, *error-box* and *error-visual*). Although these errors are interesting human data, many of them would not occur in actual language use but are artifacts of the crowdsourcing methodology that uses simple graphical boxes to point workers to the annotation target. As such, they pose challenges for our research goal, which is to see whether the object-naming ability of computational models resembles that of actual language users. To overcome these, we collected a new layer of verification annotations on top of ManyNames, as follows.

3 Adequacy and Errors in ManyNames

Collection Procedure We recruited annotators from Amazon Mechanical Turk[2] (AMT) to categorize the object-name pairs from ManyNames. Since the task was multiple-choice we could conduct rigorous quality control, ensuring the reliability of these annotations (see the supplementary material for details). We ran all the ManyNames data (response sets) through our verification, with two exceptions: response sets with 100% agreement (i.e., all annotators gave the same name for an object) are treated as correct, and all names with a count of 1 considered unreliable. Each of the remaining 19,427 images, with the target object highlighted with a bounding box, were presented to 3 annotators, with a list of all its names to be verified (Figure 1 in supplementary material; on average 4 names per object). They were asked to judge the adequacy[3] of each name among "perfectly adequate", "there may be a (slight) inadequacy" and "totally inadequate" (encoded below as 1.0, 0.5 and 0.0, respectively). If they did not select "adequate" they had to specify the type of inadequacy from "linguistic mistake", "named object not tightly in bounding box", "named object mistaken for something it's not", and "something else". Lastly, annotators had to judge which names were likely intended for the same object, through an interface where they would assign the same color to names intended for the same object. Overall, 255 participants annotated 69,356 object-name pairs (each pair receiving 3 annotations).

Judgment Aggregation We aggregate our name adequacy judgments by taking the mean, and inadequacy type judgments by taking the majority. Average name adequacy, on the scale from totally inadequate (0) to adequate (1), is 0.80 (std=0.093). The proportion of annotators agreeing on adequacy ratings is 88%, agreeing on inadequacy type is 86%, and agreeing on whether two names were intended for the same object is 94%. This is high given the complexity of the task and the fact that pictures and bounding boxes are sometimes unclear. Figure 3 shows the proportion of adequate names and inadequacy types.

We aggregate the judgments of whether two names were intended for the same object in two ways, for different purposes: SameObject and Clus-

[2] https://mturk.com

[3] In the task we provided the definition that *a name is "adequate" if there is an object in the image, whose visible parts are tightly circumscribed by the red bounding box, that one could reasonably call by that name.*

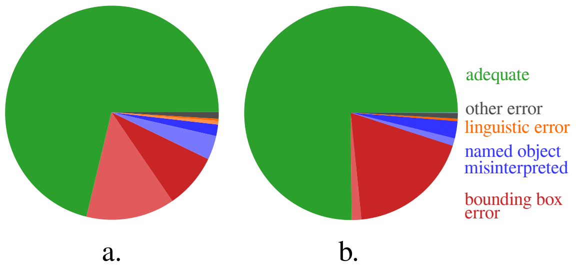


Figure 3: Verification results: a. counting individual annotations; b. counting image-name pairs with their aggregated scores. Lighter shade within a hue indicates slight/possible error of that type; darker shade severe error.

ter. First, for analysing computational object naming models (Section 5), we want to know primarily whether the name predicted by a model, if it isn't the entry-level name, was at least intended for the same object as the entry-level name. This **SameObject** measure we compute by a simple majority rule: true if name1 and name2 were judged to be for the same object by a majority of annotators, and false otherwise. Second, for a better understanding of the verified ManyNames data itself (analysis below) we want to be able, e.g., to assess the mean number of names per object, and for that we need to do clustering (the pairwise majority rule does not result in a proper clustering). We compute the **Cluster** of a name by agglomerative, complete-linkage clustering with a distance threshold of .5, using as distance between two names the proportion of annotators saying they do not name the same object (hence all pairs of names in a cluster must name the same object according to a majority of annotators).[4]

Analysis The verification we conducted is necessary for our main research aim—to evaluate object naming models—but they also enable a deeper understanding of the ManyNames data. Table 1 provides an overview. We focus on our two main reasons for testing models on ManyNames: a dataset where (i) actual contextualized instances are named (not prototypes or classes), and (ii) every instance is named by many annotators.

objects in MN	25,315
total vocabulary	7,970
entry-level vocabulary	442
objects in VG with name in entry-level vocab	$\sim$ 2M (50%)
av. agreement all names	34.9%
av. agreement entry-level names	75.2%
av. agreement entry-level cluster	42.3%
av. adequacy all names	0.81
av. adequacy entry-level names	0.97
av. adequacy entry-level cluster	0.94

Table 1: Basic statistics for entry-level names in ManyNames. Av.agreement denotes the proportion of annotators (out of 36) that annotated a given name for an object, averaged over all names of a relevant subset.

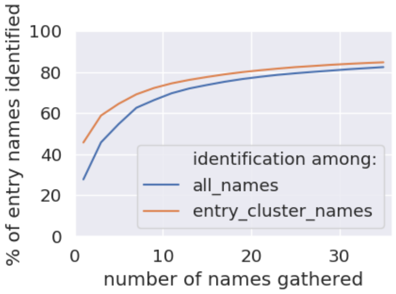


Figure 4: % of objects for which the entry-level name can be confidently identified after gathering N names.

Regarding (i), recall that a traditional view of object naming regards all instances of a class to have the same entry-level name. This predicts that we will find a stable ordering of entry-level names. We observe instead that many pairs of entry-level names do not have a fixed preference ordering across instances (e.g., Figure 1). We quantify this by focusing on cases in ManyNames where there is a 'clear winner', namely, objects where the entry-level name is at least twice as frequent in the response set as the next most frequent name (covering 281 of the 442 entry-level names of ManyNames). These names, which are a clear winner at least once, are so still on average only in 46% of cases where there is a clear winner at all (meaning in the other 54% of cases they are a clear loser). Importantly, the verified results support the same conclusion: even restricting attention to cases where the name is adequate and names the right object, there is still a clear winner only in 66% of cases where there is one (std= .35, skewed with about half over 80%).

Regarding our second reason (ii) mentioned above, Figure 4 shows that gathering many annotations is necessary for reliably identifying entry-level names. For each object we estimated, on the basis of its ManyNames response set, how many

[4] The 'hard' clustering we use may occasionally lead us to *under*estimate the number of names for an object (this is also why we will use SameObject and not Cluster in model analysis). The alternative, *soft* clustering (where each name could potentially be in multiple clusters), would risk *over*estimating the number of names per object, as well as the number of named objects in an image.

names should have been gathered for the entry-level name (as defined on the basis of 36 names) to be the majority name among them with 95% probability. The blue line shows that, even after gathering 10 names, the entry-level name is sufficiently likely to be the majority name for only 68% of objects (i.e., for 32% of objects, there's a greater than 5% chance of getting a different majority name). Gathering 20 names increases this to 78%. Our verification results let us see that most of the remaining indeterminacy occurs within the entry-level name's Cluster: the red line is computed by taking only names from the Cluster of the entry-level name into account, showing that, by discarding names for other objects, the entry-level name for the intended object can be confidently identified more easily (but not by much, since names for the wrong object tend not to be frequent anyway).

Note that both lines in Figure 4 stay well below 100%, reflecting that for 15%-20% of objects even gathering 36 names is not enough for the entry-level name to come out as majority (95% probability): differences in frequencies between the contenders are simply too small. That is, for 15%-20% of objects we aren't sure that the *sample* entry-level name we derive from ManyNames—what we defined as the entry-level name—in fact corresponds to the *population* entry-level name. This is not expected to affect our model comparison, but in principle gathering even more annotations could help for these cases—although it is also possible that some instances do not have a single, clear entry-level name within a mixed population, or even for a given individual.

4 Predicting Entry-Level Names

Given the entry-level naming data from ManyNames, which extends the more arbitrary naming data from VisualGenome (VG), we now have a variety of possibilities to train, re-train or fine-tune existing object detection or classification models, to capture natural naming. Moreover, we can systematically vary the vocabulary for these models during training: a standard object naming vocabulary compiled from a frequency list as in (Anderson et al., 2018), or a set of entry-level names as aggregated from the ManyNames data. Our experiments will assess these various modeling decisions for the task of entry-level naming, in the current section by using accuracy as a standard evaluation metric. Section 5 will analyze the results with respect to the different types of errors identified by our verification data.

4.1 Experimental Setup

Data We split ManyNames into training ($21K$ objects) and test data ($1,072$), where the test data corresponds to the overlap of ManyNames (MN) and the VisualGenome (VG) test split used by Anderson et al. (2018). The target vocabulary of the test data is MN442: the 442 entry-level names we derived from ManyNames. For transfer learning we train models on the ground truth entry-level names provided by ManyNames, and compare this to training on VisualGenome labels.

Models We test two object detection models trained on objects and names from the original VisualGenome data, using the Faster R-CNN architecture (Ren et al., 2015), initialized with features that were pre-trained on 1K ImageNet classes with the ResNet-101 classification model (He et al., 2016). We use and retrain these two models with different target vocabularies:

- FRCNN$_{\text{VG1600}}$: This is (Anderson et al., 2018)'s model[5] that is optimized for a set of 1600 object names in VisualGenome. This set was obtained by first taking the 2500 most frequent names in VisualGenome and then (manually) filtering certain names.
- FRCNN$_{\text{MN442}}$: we pre-process the VisualGenome data as in Anderson et al., but restrict the object classes to the 442 entry-level names found in ManyNames (Section 3). For training, we use a PyTorch reimplementation of Faster. R-CNN[6]

The two former models are trained on VisualGenome, which has almost 4 million labeled objects. ManyNames is too small to train object detection models on it from scratch, which requires learning millions of parameters. On the other hand, obtaining enough naming data to be able to estimate entry-level names for larger datasets is very costly and probably unrealistic. Therefore, we investigate the possibility of obtaining effective entry-level naming models via transfer learning, fine-tuning existing models with ManyNames data.

[5] We used the code and model available at `github.com/peteanderson80/bottom-up-attention`. The model is trained with an additional output over attributes, but we only use the object class prediction layer.

[6] `github.com/jwyang/faster-rcnn.pytorch`

We do so with the standard methodology of taking the last hidden layer of existing Computer Vision models as input, and train object classification models to predict entry-level names (MN442).

We experiment with two kinds of input features, as well as different training vocabularies and ground-truth names for instances, similar to the detection models above. This yields the following 4 transfer-learning-based naming models:

- FRCNN$_{\text{VG1600–VGMN}}$: This model extracts features from the object classification model FRCNN$_{\text{VG1600}}$[7]. We expect these to be very good features for our task, as they have been obtained with the same task (object classification) and dataset (VisualGenome). For each object (i.e., bounding box) in ManyNames, we extract its activation features from FRCNN's last hidden layer before the output layer. The model uses the VGMN vocabulary (overlap of the full ManyNames vocabulary with VG).

- ResNet-VGMN: For comparison, we consider a much less informed input, namely the features from the ResNet-101 image classification model, that was used to initialize the FR-CNN models above, and pretrained on ImageNet (Deng et al., 2009). Results should of course be worse, and our aim is to assess the feasibility of using such generic object representations to obtain entry-level names. ResNet-VGMN uses the VGMN vocabulary; we trained two models, optimized on VG and MN ground truth names, respectively, but report only the one optized on VG (the other one yielded similar output).

- ResNet-MN442 (two models): The same model as ResNet-VGMN, but trained with a vocabulary of 442 entry-level names; we optimized for on VG or MN ground truth names respectively.

Note that these four models are optimized on the 21K objects in ManyNames. To predict the entry-level name of a target object o, the classifier, a multi-layer neural network, is fed the object's features, and applies softmax activation to the output layer that corresponds to the target vocabulary.

Model$_{\text{Vocab}}$	GTtrain	GTtest VG	 MN
FRCNN$_{\text{MN442}}$	VG	65.4	71.2
FRCNN$_{\text{VG1600}}$	VG	67.3	**74.5**
Classifiers: Transfer learning on MN			
Features$_{\text{Vocab}}$			
FRCNN$_{\text{VG1600-VGMN}}$	MN	**70.8**	**80.6**
ResNet$_{\text{MN442}}$	MN	61.7	69.6
ResNet$_{\text{VGMN}}$	VG	62.4	62.9
ResNet$_{\text{MN442}}$	VG	63.7	63.7

Table 2: Model accuracy (in %) on the ManyNames test objects. Vocab denotes the dataset used to induce the target vocabulary for training (the numbers are vocabulary size). GTtrain gives the dataset providing the ground truth labels for training, and GTtest for testing .

4.2 Results

Table 2 shows the accuracy of the models on the ManyNames test data. We compare model effectiveness on predicting the entry-level name (column MN) against predicting the VG name .

A pervasive trend is that all models perform better at predicting the entry-level name (MN) than the VG name (VG), regardless of which vocabulary (Vocab) and which ground-truth (GTtrain) they are trained on. Note that the VG names were derived on the basis of 1-2 annotations per object, which, as shown in Section 3, is not enough to robustly estimate entry-level names. The results therefore indicate that, even when not trained to do so (FRCNN$_{\text{VG1600}}$), models tend to predict entry-level names. This is likely because entry-level names better reflect generalizations across objects.

The overall best model is FRCNN$_{\text{VG1600-VGMN}}$ (the transfer learning classifier with FRCNN$_{\text{VG1600}}$ features, trained on the ManyNames). Fine-tuning on entry-level names also helps in the original task of predicting VG names, as FRCNN$_{\text{VG1600-VGMN}}$ also improves in the VG evaluation (70.8% accuracy, vs. 67.3% without fine-tuning). As expected, the classifiers using ResNet features obtain the worst results, but are still remarkably effective considering that they were trained on a fraction of FRCNN's training size using a simple transfer learning classifier.

In sum, we find that models learn entry-level naming even when being trained on arbitrary naming data, obtaining better results on entry-level prediction than VG name prediction for the same instances, and that they benefit from transfer learning with entry-level names. Instead, restricting

[7] As experimental results will show, FRCNN$_{\text{VG1600}}$ is more effective than FRCNN$_{\text{MN442}}$, so we only use the former.

the training vocabulary to entry-level names is not effective (all MN422 models are worse or comparable to VGMN/VG1600 models).

4.3 Discussion

As Section 3 showed, object naming is a linguistically complex phenomenon which exhibits a fair amount of agreement and also variation and potential errors. Hence, it is striking that even models that are trained on arbitrary names capture entry-level naming to a good extent. However, our results also suggest that current *evaluation* methods might not be very reliable when applied on arbitrary names, probably punishing models for 'errors' that actually constitute plausible alternatives. The next section elaborates on this and proposes a fairer assessment of naming quality.

5 Analysis of Model Naming Behavior

The previous section looked at the performance of the different naming models using a standard single-label evaluation scheme. This way of evaluating object naming is unsatisfactory: the ManyNames data, and the verification data, clearly shows that other names than the entry-level name may be adequate too. In the following, we use our verification data to assess the severity of errors in different naming models.

Error Categories We categorize model predictions into the following types (see also Figure 2 for examples). Besides *Hit* (the entry-level name) and *Off* (none of the following categories, representing a wrong name), we distinguish between different *Human-like mishits*:

SameObject:	alternative name for the same object, estimated from ManyNames and including WordNet synonyms and hypernyms (*house/building*; Fellbaum 1998).
OtherObject:	name produced by annotators, but not an alternative of the entry-level name
LowCount:	name produced by 1 annotator
Related:	a hyponym or co-hyponym of the entry-level (*jacket/shirt*; also estimated from WordNet)

How Human-Like are Model Predictions and Errors? Table 3 shows the breakdown of the results for the error categories explained above. The proportion of the human-like errors that related inadequate alternatives is surprisingly constant across all models. Both object detection models achieve essentially the same portion of human-like mishits. Thus, the main difference of $\text{FRCNN}_{\text{MN442}}$ to $\text{FRCNN}_{\text{VG1600}}$ is that the former makes more nonsensical errors (column Off), which might be an effect of the reduction in training data. Importantly, the transferred classification model $\text{FRCNN}_{\text{VG1600–VGMN}}$ achieves more hits and also fewer nonsensical errors (other) compared to $\text{FRCNN}_{\text{VG1600}}$. However, it also predicts fewer valid alternatives (SameObject). This suggests that the transfer learning approach is most effective when the object detection model achieves a valid prediction, which is then calibrated to a perfect hit (it predicts the entry-level) by the fine-tuned classification model. The proportion of instances for which the ResNet models predict a 'related' category is high, which suggests that they tend to confuse invalid names that are semantically similar to a valid name.

Low Human Agreement, Worse Predictions Table 4 shows a break-down of model predictions and errors for objects with high and low naming agreement in ManyNames, i.e. objects where more (viz. less) than 90% of the annotators opted for the entry-level name. Here, we generally find that for all models the portion of hits is extremely high when human agreement is also high. Note that this holds even for the ResNet-based classifiers that generally achieve lower accuracy. When human agreement is lower, all models produce fewer hits and more valid alternatives, indicating a similarity between what is difficult for models and humans.

Table 5 focuses on the predicted ManyNames names that were judged as inadequate (i.e., adequacy score < 0.5) and their inadequacy types (see Section 3). Most of the inadequate name predictions are due to the model naming an object that does not properly fit the bounding box. The overall best model $\text{FRCNN}_{\text{VG1600–VGMN}}$ has the smallest proportion of these bounding box errors, and the highest proportion of *visual* errors (i.e., named object misinterpreted for something it's not). An example for this is shown in Figure 2 (right): $\text{FRCNN}_{\text{VG1600–VGMN}}$ predicts the more specific name *cow* for an object that is not

		Hit	Human-like				Off
Model	GT		SameObject	OtherObject	LowCount	Related	
FRCNN$_{\text{VG1600}}$	VG	74.8	13.7	2.9	2.4	1.7	4.5
FRCNN$_{\text{MN442}}$	VG	71.1	13.6	3.4	2.0	2.6	7.4
FRCNN$_{\text{VG1600–VGMN}}$	MN	80.7	9.0	2.0	1.8	3.4	3.3
ResNet101$_{\text{MN442}}$	MN	69.7	9.9	3.2	2.7	7.0	7.6
ResNet101$_{\text{VGMN}}$	VG	62.8	11.6	2.1	3.2	10.3	10.1
ResNet101$_{\text{MN442}}$	VG	63.8	11.9	2.4	3.3	8.8	9.9

Table 3: Model results for different categories of errors.

		MN agreement > 0.9			MN agreement ≤0.9		
model	GT	hit	human-like	off	hit	human-like	off
FRCNN$_{\text{VG1600}}$	VG	94.8	2.6	2.6	63.6	30.9	5.5
FRCNN$_{\text{MN442}}$	VG	89.6	4.2	6.2	60.7	31.3	8.0
FRCNN$_{\text{VG1600–VGMN}}$	MN	94.5	4.2	1.3	72.9	22.7	4.4
ResNet101$_{\text{MN442}}$	MN	90.1	4.9	4.9	58.2	32.8	9.0
ResNet101$_{\text{VGMN}}$	VG	88.3	6.2	5.5	48.5	38.8	12.7
ResNet101$_{\text{MN442}}$	VG	88.6	5.2	6.2	49.9	38.2	12.0

Table 4: Break-down of the results (in %) according to the agreement level of the MN name: Categorization of a predicted name $\hat{n}$ into either a *hit*, *human-like* error (less preferred name, synonym, hypernym/hyponym), or *off*

Model	GT	visual	linguistic	box	other
FRCNN$_{\text{VG1600}}$	VG	4.3	0.0	95.7	0.0
FRCNN$_{\text{MN442}}$	VG	3.4	0.0	96.6	0.0
FRCNN$_{\text{VG1600–VGMN}}$	MN	6.7	0.0	93.3	0.0
ResNet101$_{\text{MN442}}$	MN	3.0	0.0	97.0	0.0
ResNet101$_{\text{VGMN}}$	VG	5.6	0.0	94.4	0.0
ResNet101$_{\text{MN442}}$	VG	4.8	0.0	95.2	0.0

Table 5: Break-down of the instances (in %) for which the models predicted an *inadequate* name according to the type of inadequacy.

strictly speaking a cow and which most humans and FRCNN$_{\text{VG1600}}$ named *animal*. This illustrates a tricky case of entry-level name: the model needs to 'understand' that humans will have difficulties naming that object with a specific or basic-level name and opt instead for an unusually general name.

None of the models predicted an inadequate name classified as a "linguistic error" or as "other error". Recall that all predictions we are analyzing here concerned the ManyNames data, i.e., at least two people gave the predicted, but inadequate name for the object. That the models make similar mistakes to humans (problems in identifying or recognizing the target object) on MN illustrates the challenges involved in collecting object names from image data to model natural object naming, and the relevance of thorough verification and quality control as we did in our work.

6 Conclusions

In this paper, we have investigated instance-level entry-level naming of objects in images. To the best of our knowledge, our work is the first to investigate entry-level naming on such a large-scale. For models in L&V, the complexity of human naming data we observed brings up an important question: how should a naming models best fine-tune and calibrate the *vocabulary* of object detection and classification models from Computer Vision, in order to capture natural naming behavior? We showed that the best way to achieve this is to first retrain an object detector on a large set of arbitrary names and then fine-tune via transfer learning to entry-level names. In future work, we plan to develop this approach further and assess, e.g., how well the learned naming vocabulary generalizes for other, external datasets. We also showed that naming data elicited for boxed objects comes with certain challenges, which have to be considered during evaluation (and potentially during modeling).

More generally, we believe that the verified object naming data as well as our modeling approach affords tremendous opportunities to study factors affecting object naming that have not been studied to date, such as the prototypicality of the instance itself, or the effect of the visual context.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering. In *Proceedings of CVPR*.

Jia Deng, W. Dong, Richard Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

Christiane Fellbaum. 1998. *WordNet*. Wiley Online Library.

Caroline Graf, Judith Degen, Robert XD Hawkins, and Noah D Goodman. 2016. Animal, dog, or dalmatian? level of abstraction in nominal referring expressions. In *Proceedings of the 38th annual conference of the Cognitive Science Society*. Cognitive Science Society.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. pages 770–778.

Pierre Jolicoeur, Mark Gluck, and Stephen Kosslyn. 1984. Pictures and names: Making the connection. *Cognitive psychology*, 16:243–275.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, Michael Bernstein, and Li Fei-Fei. 2016. Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations.

Vicente Ordonez, Wei Liu, Jia Deng, Yejin Choi, Alexander C. Berg, and Tamara L. Berg. 2016. Learning to Name Objects. *Commun. ACM*, 59(3):108–115.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *NIPS'2015*, pages 91–99.

Eleanor Rosch. 1978. Principles of Categorization. In Eleanor Rosch and Barbara B. Lloyd, editors, *Cognition and Categorization*, pages 27—-48. Lawrence Erlbaum, Hillsdale, N.J., USA.

Eleanor Rosch, Carolyn B Mervis, Wayne D Gray, David M Johnson, and Penny Boyes-Braem. 1976. Basic objects in natural categories. *Cognitive psychology*, 8(3):382–439.

Bruno Rossion and Gilles Pourtois. 2004. Revisiting snodgrass and vanderwart's object pictorial set: The role of surface detail in basic-level object recognition. *Perception*, 33(2):217–236.

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

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *CVPR 2015*, Boston, MA, USA.

Sina Zarrieß and David Schlangen. 2017. Obtaining referential word meanings from visual and distributional information: Experiments on object naming. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 243–254, Vancouver, Canada. Association for Computational Linguistics.