Counting Everyday Objects in Everyday Scenes

Prithvijit Chattopadhyay*,[1] Ramakrishna Vedantam*,[1] Ramprasaath R. Selvaraju[1]
Dhruv Batra[2] Devi Parikh[2]
[1]Virginia Tech [2]Georgia Institute of Technology
[1]{prithv1,vrama91,ram21}@vt.edu [2]{dbatra,parikh}@gatech.edu

Abstract

We are interested in counting the number of instances of object classes in natural, everyday *images. Previous counting approaches tackle the problem in restricted domains such as counting pedestrians in surveillance videos. Counts can also be estimated from outputs of other vision tasks like object detection. In this work, we build dedicated models for counting designed to tackle the large variance in counts, appearances, and scales of objects found in natural scenes. Our approach is inspired by the phenomenon of subitizing – the ability of humans to make quick assessments of counts given a perceptual signal, for small count values. Given a natural scene, we employ a divide and conquer strategy while incorporating context across the scene to adapt the subitizing idea to counting. Our approach offers consistent improvements over numerous baseline approaches for counting on the PASCAL VOC 2007 and COCO datasets. Subsequently, we study how counting can be used to improve object detection. We then show a proof of concept application of our counting methods to the task of Visual Question Answering, by studying the 'how many?' questions in the VQA and COCO-QA datasets.*

1. Introduction

We study the scene understanding problem of counting common objects in natural scenes. That is, given for example the image in Fig. 1, we want to count the number of everyday object categories present in it: for example 4 *chairs*, 1 *oven*, 1 *dining table*, 1 *potted plant* and 3 *spoons*. Such an ability to count seems innate in humans (and even in some animals [10]). Thus, as a stepping stone towards Artificial Intelligence (AI), it is desirable to have intelligent machines that can count.

Similar to scene understanding tasks such as object detection [43, 14, 18, 37, 17, 44, 34, 29] and segmentation [4, 30, 36] which require a fine-grained understanding of the scene, object counting is a challenging problem that requires us to reason about the number of instances of objects present while tackling scale and appearance variations.

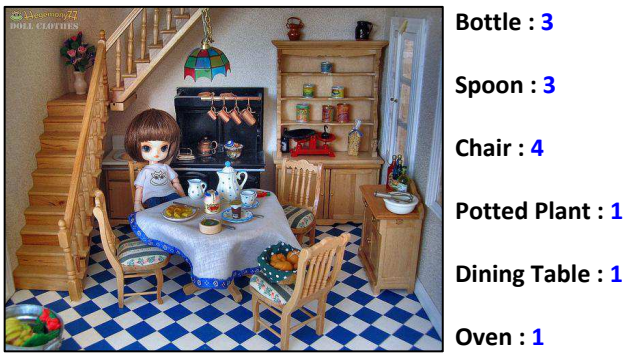

Figure 1: We study the problem of counting everyday objects in everyday scenes. Given an everyday scene, we want to predict the number of instances of common objects like bottle, chair *etc*.

Another closely related vision task is visual question answering (VQA), where the task is to answer free form natural language questions about an image. Interestingly, questions related to the count of a particular object - *How many red cars do you see?* form a significant portion of the questions asked in common visual question answering datasets [2, 35]. Moreover, we observe that end-to-end networks [2, 35, 31, 15] trained for this task do not perform well on such counting questions. This is not surprising, since the objective is often setup to minimize the cross-entropy classification loss for the correct answer to a question, which ignores ordinal structure inherent to counting. In this work we systematically benchmark how well current VQA models do at counting, and study any benefits from dedicated models for counting on a subset of counting questions in VQA datasets in Sec. 5.4.

Counts can also be used as complimentary signals to aid other vision tasks like detection. If we had an estimate of how many objects were present in the image, we could use that information on a per-image basis to detect that many objects. Indeed, we find that our object counting models improve object detection performance.

We first describe some baseline approaches to counting and subsequently build towards our proposed model.

* Denotes equal contribution.

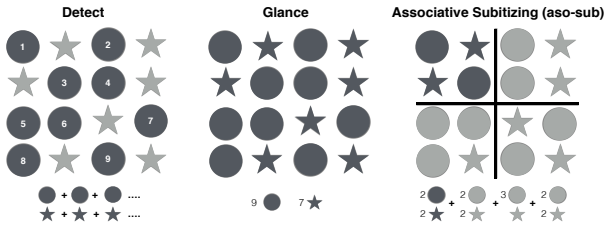

Figure 2: A toy example explaining the motivation for three categories of counting approaches explored in this paper. The task is to count the number of stars and circles. In `detect`, the idea is to detect instances of a category, and then report the total number of instances detected as the count. In `glance`, we make a judgment of count based on a glimpse of the full image. In `aso-sub`, we divide the image into regions and judge count based on patterns in local regions. Counts from different regions are added through arithmetic.

Counting by Detection: It is easy to realize that perfect detection of objects would imply a perfect count. While detection is sufficient for counting, localizing objects is not necessary. Imagine a scene containing a number of mugs kept on a table where the objects occlude each other. In order to count the number of mugs, we need not determine with pixel-accurate segmentations or detections where they are (which is hard in the presence of occlusions) as long as say we can determine the number of handles. Relieving the burden of detecting objects is also effective for counting when objects occur at smaller scales where detection is hard [18]. However, counting by detection or `detect` still forms a natural approach for counting.

Counting by Glancing: Representations extracted from Deep Convolutional Neural Networks [42, 26] trained on image classification have been successfully applied to a number of scene understanding tasks such as finegrained recognition [12], scene classification [12], object detection [12], *etc*. We explore how well features from a deep CNN perform at counting through instantiations of our glancing (`glance`) models which estimate a global count for the entire scene in a single forward pass. This can be thought of as estimating the count at one shot or *glance*. This is in contrast with `detect`, which sequentially increments its count with each detected object (Fig. 2). Note that unlike detection, which optimizes for a localization objective, the `glance` models explicitly *learn* to count.

Counting by Subitizing: Subitizing is a widely studied phenomenon in developmental psychology [8, 25, 10] which indicates that children have an ability to directly map a perceptual signal to a numerical estimate, for a small number of objects (typically 1-4). Subitizing is crucial for development and assists arithmetic and reasoning skills. An example of subitizing is how we are able to figure out the number of pips on a face of a die without having to count them or how we are able to reason about tally marks.

Inspired by subitizing, we devise a new counting approach which adopts a divide and conquer strategy, using the additive nature of counts. Note that `glance` can be thought of as an attempt to subitize from a glance of the image. However, as illustrated in Fig. 2 (center), subitizing is difficult at high counts for humans.

Inspired by this, using the divide and conquer strategy, we divide the image into non-overlapping cells (Fig. 2 right). We then subitize in each cell and use addition to get the total count. We call this method associative subitizing or `aso-sub`.

In practice, to implement this idea on real images, we incorporate context across the cells while sequentially subitizing in each one of them. We call this sequential subitizing or `seq-sub`. For each of these cells we curate real-valued ground truth, which helps us deal with scale variations. Interestingly, we found that by incorporating context `seq-sub` significantly outperforms the naive subitizing model `aso-sub` described above. (see Sec. 5.1 for more details).

Counting by Ensembling: It is well known that when humans are given counting problems with large ground truth counts (*e.g*. counting number of pebbles in a jar), individual guesses have high variance, but an average across multiple responses tends to be surprisingly close to the ground truth. This phenomenon is popularly known as the wisdom of the crowd [16]. Inspired by this, we create an ensemble of counting methods (`ens`).

In summary, we evaluate several natural approaches to counting, and propose a novel context and subitizing based counting model. Then we investigate how counting can improve detection. Finally, we study counting questions ('how many?') in the Visual Question Answering (VQA) [2] and COCO-QA [35] datasets and provide some comparisons with the state-of-the-art VQA models.

2. Related Work

Counting problems in niche settings have been studied extensively in computer vision [45, 41, 6, 27]. [6] explores a Bayesian Poisson regression method on low-level features for counting in crowds. [5] segments a surveillance video into components of homogeneous motion and regresses to counts in each region using Gaussian Process regression. Since surveillance scenes tend to be constrained and highly occluded, counting by detection is infeasible. Thus density based approaches are popular. Lempitsky and Zisserman [27] count people by estimating object density using low-level features. They show applications on surveillance and cell counting in biological images. Anchovi labs provided users interactive services to count specific objects such as swimming pools in satellite images, cells in biological images, *etc*. More recent work constructs CNN-based models for crowd counting [45, 33] and penguin counting [3] using lower level convolutional features from shallower CNN models.

Counting problems in constrained settings have a funda-

mentally different set of challenges to the counting problem we study in this paper. In surveillance, for example, the challenge is to estimate the counts accurately in the presence of large number of ground truth counts, where there might be significant occlusions. In the counting problem on everyday scenes, a larger challenge is the intra-class variance in everyday objects, and high sparsity (most images will have 0 count for most object classes). Thus we need a qualitatively different set of tools to solve this problem.

Other recent work [46] studies the problem of salient object subitizing (SOS). This is the task of counting the number of salient objects in the image (independent of the category). In contrast, we are interested in counting the number of instances of objects per category. Unlike Zhang *et al*. [46], who use SOS to improve salient object detection, we propose to improve generic object detection using counts. Our VQA experiments to diagnose counting performance are also similar in spirit to recent work that studies how well models perform on specific question categories (counting, attribute comparison, *etc*.) [22] or on compositional generalization [1].

3. Approach

Our task is to accurately count the number of instances of different object classes in an image. For training, we use datasets where we have access to object annotations such as object bounding boxes and category wise counts. The count predictions from the models are evaluated using the metrics described in Sec. 4.2. The input to the `glance`, `aso-sub` and `seq-sub` models are `fc7` features from a VGG-16 [42] CNN model. We experiment using both off-the-shelf classification weights from ImageNet [38] and the detection fine-tuned weights from our `detect` models.

3.1. Detection (`detect`)

We use the Fast R-CNN [18] object detector to count. Detectors typically perform two post processing steps on a set of preliminary boxes: non maximal suppression (NMS) and score thresholding. NMS discards highly overlapping and likely redundant detections (using a threshold to control the overlap), whereas the score threshold filters out all detections with low scores.

We steer the detector to count better by varying these two hyperparameters to find the setting where counting error is the least. We pick these parameters using grid search on a held-out val set. For each category, we first select a fixed NMS threshold of 0.3 for all the classes and vary the score threshold between 0 and 1. We then fix the score threshold to the best value and vary the NMS threshold from 0 to 1.

3.2. Glancing (`glance`)

Our `glance` approach repurposes a generic CNN architecture for counting by training a multi-layered perceptron (MLP) with a L2 loss to regress to image level counts from deep representations extracted from the CNN. The MLP has batch normalization [20] and Rectified Linear Unit (ReLU) activations between hidden layers. The models were trained with a learning rate of 10^{-3} and weight decay set to 0.95. We experiment with choices of a single hidden layer, and two hidden layers for the MLP, as well as the sizes of the hidden units. More details and ablation studies can be found in [7].

Figure 3: **Canonical counting scale:** Consider images with grids 2×2 (left) and 6×6 (right). Notice the red cells in both images: it is evident that if the cell size is too large compared to the object (left), it is difficult to estimate the large integer count of 'sheep' in the cell. However, if the cell is too small (right), it might be hard to estimate the small fractional count of 'bus' in the cell. Hence, we hypothesize that there exists a sweet spot in discretization of the cells that would results in optimum counting performance.

3.3. Subitizing (`aso-sub`, `seq-sub`)

In our *subitizing* inspired methods, we divide our counting problem into sub-problems on each cell in a non-overlapping grid, and add the predicted counts across the grid. In practice, since objects in real images occur at different scales, such cells might contain fractions of an object. We adjust for this by allowing for real valued ground truth. If a cell overlapping an object is very small compared to the object, the small fractional count of the cell might be hard to estimate. On the other hand, if a cell is too large compared to objects present it might be hard to estimate the large integer count of the cell (see Fig. 3). This trade-off suggests that at some canonical resolution, we would be able to count the smaller objects more easily by subitizing them, as well as predict the partial counts for larger objects. More concretely, we divide the image I, into a set of n non-overlapping cells $P = \{p_1, \cdots, p_n\}$ such that $I = \bigcup_{i=1}^{n} p_i$ and $p_i \cap p_{j\,(i \neq j)} = \phi$. Given such a partition P of the image I and associated CNN features $X = \{x_i, \cdots, x_n\}$, we now explain our models based on this approach:

`aso-sub` : Our naive `aso-sub` model treats each cell independently to regress to the real-valued ground truth. We train on an augmented version of the dataset where the dataset size is n-fold (n cells per image). Unlike `glance`, where feature extracted on the full image is used to regress to integer valued counts, `aso-sub` models regress to real-

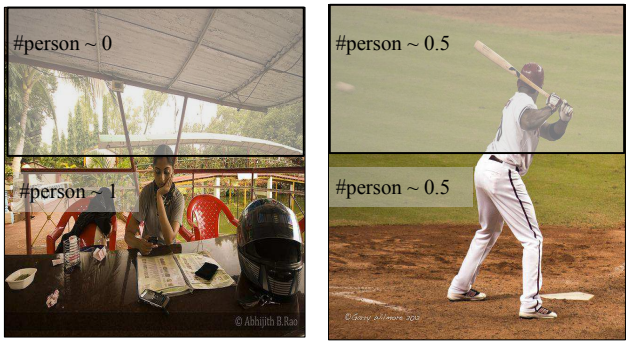

Figure 4: For both these images, the count of *person* is 1. Consider splitting this image into 2×1 cells (for illustration) for `aso-sub`. The bottom half of the left image and top half of the right image both contains similar visual signals – top-half of a person. However, the ground truth count on the cell on the left is 1, and the one on the right is 0.5. An approach that estimates counts from individual cells out of context is bound to fail at these cases. This motivates our proposed approach `seq-sub`.

valued counts on non-overlapping cells from features extracted per cell. Given class instance annotations as bounding boxes $b = \{b_1, \cdots, b_N\}$ for a category k in an image I, we compute the ground truth partial counts (c^k_{gt}) for the grid-cells (p_i) to be used for training as follows:

$$p_i : c^k_{gt} = \sum_{j=1}^{N} \frac{p_i \cap b_j}{b_j} \tag{1}$$

We compute the intersection of each box b_i with the cell p_i and add up the intersections normalized by b_i. Further, given the cell-level count predictions c_{p_i}, the image level count prediction is computed as $c = \sum_{i=1}^{n} max(0, c_{p_i})$. We use max to filter out negative predictions.

We experiment with dividing the image into equally sized 3×3, 5×5, and 7×7 grid-cells. The architecture of the models trained on the augmented dataset are the same as `glance`. For more details, refer to [7].

`seq-sub` : We motivate our proposed `seq-sub` (Sequential Subitizing) approach by identifying a potential flaw in the naive `aso-sub` approach. Fig. 4 reveals the limitation of the `aso-sub` model. If the cells are treated independently, the naive `aso-sub` model will be unaware of the partial presence of the concerned object in other cells. This leads to situations where similar visual signals need to be mapped to partial and whole presence of the object in the cells (see Fig. 4). This is especially pathological since Huber or L-2 losses cannot capture this multi-modality in the output space, since the implicit density associated with such losses is either laplacian or gaussian.

Interestingly, a simple solution to mitigate this issue is to model context, which resolves this ambiguity in counts. That is, if we knew about the partial class presence in other cells, we could use that information to predict the correct cell count. Thus, although the independence assumption in `aso-sub` is convenient, it ignores the fact that the augmented dataset is not IID. While it is important to reason at a cell level, it is also necessary to be aware of the global image context to produce meaningful predictions. In essence, we propose `seq-sub`, that takes the best of both worlds from `glance` and `aso-sub`.

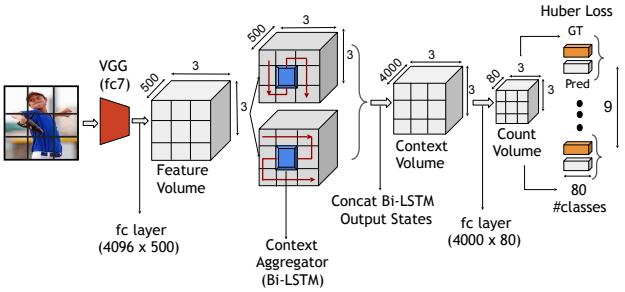

Figure 5: Architecture used for our `seq-sub` models. We extract a hidden layer representation of the `fc7` feature volume corresponding to the 3×3 discretization of the image. Subsequently, we traverse this representation volume in two particular sequences in parallel as shown via two stacked bi-LSTMs per sequence and aggregate context over the image. We get output states corresponding to each of the cells and subsequently get cell-counts via another hidden layer. The hidden layers use ReLU as non-linearity.

The architecture of `seq-sub` is shown in Fig. 5. It consists of a pair of 2 stacked bi-directional sequence-to-sequence LSTMs [40]. We incorporate context across cells as

$$c_{p_i} = h(f_1(x_1, \theta_1), \cdots, f_n(x_n, \theta_n), i, \theta) \tag{2}$$

where individual $f_i(x_i, \theta_i)$ are hidden layer representations of each cell feature with respective parameters and $h(., \theta)$ is the mechanism that captures context. This can be broken down as follows. Let H be the set containing $f_i(x_i, \theta_i)$s. Let H_{O1} and H_{O2} be 2 ordered sets which are permutations of H based on 2 particular sequence structures. The (traversal) sequences, as we move across the grid in the feature column, is decided on the basis of nearness of cells (see Fig. 5). We experiment with the sequence structures best described for a 3×3 grid as N and Z which correspond to H_{O1} and H_{O2}. Each of these feature sequences are then fed to a pair of stacked Bi-LSTMs ($L_j(., i, \theta_l)$) and the corresponding cell output states are concatenated to obtain a context vector (v_i) for each cell as $v_i = L_1(H_{O1}, i, \theta_l)||L_2(H_{O2}, i, \theta_l)$. The cell counts are then obatined as $c_{p_i} = g(v_i, \theta_g)$. The composition of $L_j(., i, \theta_l)$ and $g(., \theta_g)$ implements $h(., \theta)$.

We use a Huber Loss objective to regress to the count values with a learning rate of 10^{-4} and weight decay set to 0.95. For optimization, we use Adam [24] with a mini-batch size of 64. The ground truth construction procedure for training and the count aggregation procedure for evaluation are as defined in `aso-sub`.

4. Experimental Setup

4.1. Datasets

We experiment with two datasets depicting everyday objects in everyday scenes: the PASCAL VOC 2007 [13] and

COCO [28]. The PASCAL VOC dataset contains a train set of 2501 images, val set of 2510 images and a test set of 4952 images, and has 20 object categories. The COCO dataset contains a train set of 82783 images and a val set of $40,504$ images, with 80 object categories. On PASCAL, we use the val set as our Count-val set and the test set as our Count-test set. On COCO, we use the first half of val as the Count-val set and the second half of val as the Count-test set. The most frequent count per object category (as one would expect in everyday scenes) is 0. Although, the two datasets have a fair amount of count variability, there is a clear bias towards lower count values. Note that this is unlike the crowd-counting datasets, in particular [19] where mean count is 1279.48 ± 960.42 and also unlike PASCAL and COCO, the images have very little scale and appearance variations in terms of objects.

4.2. Evaluation

We adopt the root mean squared error (RMSE) as our metric. We also evaluate on a variant of RMSE that might be better suited to human perception. The intuition behind this metric is as follows. In a real world scenario, humans tend to perceive counts in the logarithmic scale [11]. That is, a mistake of 1 for a ground truth count of 2 might seem egregious but the same mistake for a ground truth count of 25 might seem reasonable. Hence we scale each deviation by a function of the ground truth count.

We first post-process the count predictions from each method by thresholding counts at 0, and rounding predictions to closest integers to get predictions $\hat{c_{ik}}$. Given these predictions and ground truth counts c_{ik} for a category k and image i, we compute RMSE as follows:

$$RMSE_k = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{c_{ik}} - c_{ik})^2} \tag{3}$$

and relative RMSE as:

$$relRMSE_k = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\frac{(\hat{c_{ik}} - c_{ik})^2}{c_{ik}+1}} \tag{4}$$

where N is the number of images in the dataset. We then average the error across all categories to report numbers on the dataset (**mRMSE** and **m-relRMSE**).

We also evaluate the above metrics for ground truth instances with non-zero counts. This reflects more clearly how accurate the counts produced by a method (beyond predicting absence) are.

4.3. Methods and Baselines

We compare our approaches to the following baselines:
`always-0`: predict most-frequent ground truth count (0).
`mean`: predict the average ground truth count on the Count-val set.
`always-1`: predict the most frequent non-zero value (1) for all classes.
`category-mean`: predict the average count *per* category on Count-val.
`gt-class`: treat the *ground truth* counts as classes and predict the counts using a classification model trained with cross-entropy loss.

We evaluate the following variants of counting approaches (see Sec. 3 for more details):
`detect`: We compare two methods for `detect`. The first method finds the best NMS and score thresholds as explained in Sec. 3.1. The second method uses vanilla Fast R-CNN as it comes out of the box, with the default NMS and score thresholds.
`glance`: We explore the following choices of features: (1) vanilla classification `fc7` features `noft`, (2) detection fine tuned `fc7` features `ft`, (3) `fc7` features from a CNN trained to perform Salient Object Subitizing `sos` [46] and (4) flattened `conv-3` features from a CNN trained for classification
`aso-sub`, `seq-sub`: We examine three choices of grid sizes (Sec. 3.3): 3×3, 5×5, and 7×7 and `noft` and `ft` features as above.
`ens`: We take the best performing subset of methods and average their predictions to perform counting by ensembling (`ens`).

5. Results

All the results presented in the paper are averaged on 10 random splits of the test set sampled with replacement.

5.1. Counting Results

PASCAL VOC 2007 : We first present results (Table. 1) for the best performing variants (picked based on the val set) of each method. We see that `seq-sub` outperforms all other methods. Both `glance` and `detect` which perform equally well as per both the metrics, while `glance` does slightly better on both metrics when evaluated on non-zero ground truth counts. To put these numbers in perspective, we find that the difference of 0.01 $mRMSE\text{-}nonzero$ between `seq-sub` and `aso-sub` leads to a difference of 0.19% mean F-measure performance in our counting to improve detection application (Sec. 5.3). We also experiment with `conv3` features to regress to the counts, similar to Zhang.et.al. [45]. We find that `conv3` gets $mRMSE$ of 0.63 which is much worse than `fc7`. We also tried PCA on the `conv3` features but that did not improve performance. This indicates that our counting task is indeed more high level and needs to reason about objects rather than low-level textures. We also compare our approach with the SOS model [46] by extracting fc7 features from a model trained to perform category-independent salient object subitizing. We observe that our best performing `glance` setup using

Approach	mRMSE	mRMSE-nz	m-relRMSE	m-relRMSE-nz
always-0	0.66 ± 0.02	1.96 ± 0.03	0.28 ± 0.03	0.59 ± 0.00
mean	0.65 ± 0.02	1.81 ± 0.03	0.31 ± 0.01	0.52 ± 0.00
always-1	1.14 ± 0.01	0.96 ± 0.03	0.98 ± 0.00	0.17 ± 0.03
category-mean	0.64 ± 0.02	1.60 ± 0.03	0.30 ± 0.00	0.45 ± 0.00
gt-class	0.55 ± 0.02	2.12 ± 0.07	0.24 ± 0.00	0.88 ± 0.01
detect	0.50 ± 0.01	1.92 ± 0.08	0.26 ± 0.01	0.85 ± 0.02
glance-noft-2L	0.50 ± 0.02	1.83 ± 0.09	0.27 ± 0.00	0.73 ± 0.00
glance-sos-2L	0.51 ± 0.02	1.87 ± 0.08	0.29 ± 0.01	0.75 ± 0.02
aso-sub-ft-1L-3 × 3	0.43 ± 0.01	1.65 ± 0.07	0.22 ± 0.01	0.68 ± 0.02
seq-sub-ft-3 × 3	**0.42 ± 0.01**	**1.65 ± 0.07**	0.21 ± 0.01	0.68 ± 0.02
ens	0.42 ± 0.17	1.68 ± 0.08	**0.20 ± 0.00**	**0.65 ± 0.01**

Table 1: Counting performance on PASCAL VOC 2007 Count-test Set (L implies the number of hidden layers). **Lower** is better. ens is a combination of glance-noft-2L, aso-sub-ft-1L-3 × 3 and seq-sub-ft-3 × 3.

Imagenet trained VGG-16 features outperforms the one using SOS features. This is also intuitive since SOS is a category independent task, while we want to count number of object instances of each category. Finally, we observe that the performance increment from aso-sub to seq-sub is not statistically significant. We hypothesize that this is because of the smaller size of the PASCAL dataset. Note that we get more consistent improvements on COCO (Table. 2), which is not only a larger dataset, but also contains scenes that are contextually richer.[1]

COCO : We present results for the best performing variants (picked based on the val set) of each method. The results are summarized in Table. 2. We find that seq-sub does the best on both $mRMSE$ and $m\text{-}relRMSE$ as well as their non-zero variants by a significant margin. A comparison indicates that the always-0 baseline does better on COCO than on PASCAL. This is because COCO has many more categories than PASCAL. Thus, the chances of any particular object being present in an image decrease compared to PASCAL. The performance jump from aso-sub to seq-sub here is much more compared to PASCAL. Recent work by Ren and Zemel [36] on Instance Segmentation also reports counting performance on two COCO categories - *person* and *zebra*.[2]

For both PASCAL and COCO we observe that while ens outperforms other approaches in some cases, it does not always do so. We hypothesize that this is due to the poor performance of glance. For detailed ablation studies on ens see [7].

5.2. Analysis of the Predicted Counts

Count versus Count Error : We analyze the performance of each of the methods at different count values on the COCO Count-test set (Fig. 6). We pick each count value on the x-axis and compute the $RMSE$ over all the instances at that count value. Interestingly, we find that the *subitizing* approaches work really well across a range of count values. This supports our intuition that aso-sub and seq-sub are able to capture partial counts (from larger objects) as well as integer counts (from smaller objects) better which is intuitive since larger counts are likely to occur at a smaller scale. Of the two approaches, seq-sub works better, likely because reasoning about global context helps us capture part-like features better compared to aso-sub. This is quite clear when we look at the performance of seq-sub compared to aso-sub in the count range 11 to 15. For lower count values, ens does the best (Fig. 6). We can see that for counts > 5, glance and detect performances start tailing off.

Detection : We tune the hyperparameters of Fast R-CNN in order to find the setting where the mean squared error is the lowest, on the Count-val splits of the datasets. We show some qualitative examples of the detection ground truth, the performance without tuning for counting (using black-box Fast R-CNN), and the performance after tuning for counting

[1]When the Count-val split is considered, PASCAL has an average of 1.98 annotated objects per scene, unlike COCO which has 7.22 annotated objects per scene.

[2]We compare our best performing seq-sub model with their approach. On *person*, seq-sub outperforms by 1.29 $RMSE$ and 0.24 $relRMSE$. On *zebra*, [36] outperforms seq-sub by a margin of 0.4 $RMSE$ and 0.23 $relRMSE$. A recent exchange with the authors suggested anomalies in their experimental setup, which may have resulted in their reported numbers being optimistic estimates of the true performance.

Approach	mRMSE	mRMSE-nz	m-relRMSE	m-relRMSE-nz
always-0	0.54 ± 0.01	3.03 ± 0.03	0.21 ± 0.00	1.22 ± 0.01
mean	0.54 ± 0.00	2.96 ± 0.03	0.23 ± 0.00	1.17 ± 0.01
always-1	1.12 ± 0.00	2.39 ± 0.03	1.00 ± 0.00	0.80 ± 0.00
category-mean	0.52 ± 0.01	2.97 ± 0.03	0.22 ± 0.00	1.18 ± 0.01
gt-class	0.47 ± 0.00	2.70 ± 0.03	0.20 ± 0.00	1.08 ± 0.00
detect	0.49 ± 0.00	2.78 ± 0.03	0.20 ± 0.00	1.13 ± 0.01
glance-ft-1L	0.42 ± 0.00	2.25 ± 0.02	0.23 ± 0.00	0.91 ± 0.00
glance-sos-1L	0.44 ± 0.00	2.32 ± 0.03	0.24 ± 0.00	0.92 ± 0.01
aso-sub-ft-1L-3 × 3	0.38 ± 0.00	2.08 ± 0.02	0.24 ± 0.00	0.87 ± 0.01
seq-sub-ft-3 × 3	**0.35 ± 0.00**	**1.96 ± 0.02**	0.18 ± 0.00	0.82 ± 0.01
ens	0.36± 0.00	1.98± 0.02	**0.18± 0.00**	**0.81± 0.01**

Table 2: Counting performance on COCO Count-test set (L implies the number of hidden layers). **Lower** is better. ens is a combination of glance-ft-1L, aso-sub-ft-1L-3 × 3 and seq-sub-ft-3 × 3.

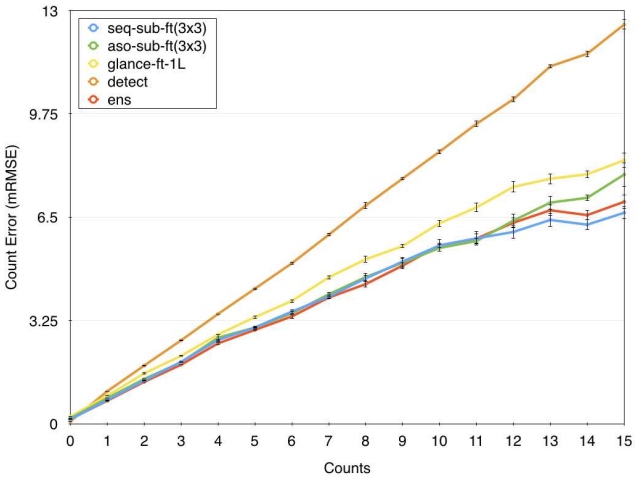

Figure 6: We plot the $mRMSE$ (across all categories) with error bars (too small to be visible) at a count against the count (x-axis) on the Count-test split of the COCO dataset. We find that the seq-sub-ft-3 × 3 and ens perform really well at higher count values whereas at lower count values the results of all the models are comparable except detect.

Figure 7: We show the ground truth count (top), outputs of `detect` with a default score threshold of 0.8 (row 1), and outputs of `detect` with hyperparameters tuned for counting (row 2). Clearly, choosing a different threshold allows us to trade-off localization accuracy for counting accuracy (see bottle image). The method finds partial evidence for counts, even if it cannot localize the full object.

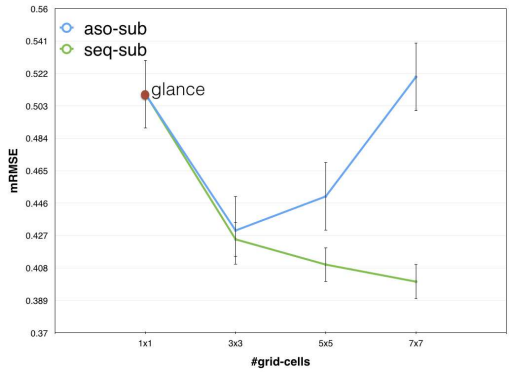

Figure 8: We plot the $mRMSE$ across all categories (y-axis) for `aso-sub` and `seq-sub` on PASCAL Count-val set against the size of *subitizing* grid cells (x-axis). As we vary the discretization we conceptually explore a continuum between `glance` and `detect` approaches. We find that for `aso-sub` there exists a sweet spot (3×3), where performance on counting is the best. Interestingly, for `seq-sub` the discretization sweet-spot is farther out to the right than `aso-sub`'s 3×3.

on the PASCAL dataset in Fig. 7. We use untuned Fast R-CNN at a score threshold of 0.8 and NMS threshold of 0.3, as used by Girshick *et al*. [18] in their demo. At this configuration, it achieves an $mRMSE$ of 0.52 on Count-test split of COCO. We find that we achieve a gain of 0.02 $mRMSE$ by tuning the hyperparameters for `detect`.

Subitizing : We next analyze how different design choices in `aso-sub` affect performance on PASCAL. We pick the best performing `aso-sub-ft-1L-`3×3 model and vary the grid sizes (as explained in Sec. 4). We experiment with 3×3, 5×5, and 7×7 grid sizes. We observe that for `aso-sub` the performance of 3×3 grid is the best and performance deteriorates significantly as we reach 7×7 grids (Fig. 8).[3] This indicates that there is indeed a sweet spot in the discretization as we interpolate between the `glance` and `detect` settings. However, we notice that for `seq-sub` this sweet spot lies farther out to the right.

5.3. Counting to Improve Detection

We now explore whether counting can help *improve* detection performance (on the PASCAL dataset). Detectors are typically evaluated via the Average Precision (AP) metric, which involves a full sweep over the range of score-thresholds for the detector. While this is a useful investigative tool, in any real application (say autonomous driving), the detector must make hard decisions at some fixed threshold. This threshold could be chosen on a per-image or per-category basis. Interestingly, if we knew *how many* objects of a category are present, we could simply set the threshold so that those many objects are detected similar to Zhang *et al*. [46]. Thus, we could use per-image-per-category counts as a prior to improve detection.

Note that since our goal is to intelligently pick a threshold for the detector, computing AP (which involves a sweep over the thresholds) is not possible. Hence, to quantify detection performance, we first assign to each detected box one ground truth box with which it has the highest overlap. Then for each ground truth box, we check if any detection box has greater than 0.5 overlap. If so, we assign a match between the ground truth and detection, and take them out of the pool of detections and ground truths. Through this procedure, we obtain a set of true positive and false positive detection outputs. With these outputs we compute the precision and recall values for the detector. Finally, we compute the F-measure as the harmonic mean of these precision and recall values, and average the F-measure values across images and categories. We call this the **mF** (mean F-measure) metric. As a baseline, we use the Fast-RCNN detector after NMS to do a sweep over the thresholds for each category on the validation set to find the threshold that maximizes F-measure for that category. We call this the `base` detector.

With a fixed per-category score threshold, the `base` detector gets a performance of 15.26% mF. With ground truth counts to select thresholds, we get a best-case `oracle` performance of 20.17%. Finally, we pick the outputs of `ens` and `seq-sub-ft` models and use the counts from each of these to set separate thresholds. Our counting methods undercount more often than they overcount[4], a high count implies that the ground truth count is likely to be even higher. Thus, for counts of 0, we default to the `base` thresholds and for the other predicted counts, we use the counts to set the thresholds. With this procedure, we get a gains of 1.64% mF and 1.74% mF over the `base` performance using `ens` and `seq-sub-ft` predictions respectively. Thus, counting can be used as a complimentary signal to aid detector performance, by intelligently picking the detector threshold in an image specific manner.

[3] Going from 1×1 to 3×3, one might argue that the gain in performance in `aso-sub` is due to more (augmented) training data. However, from the diminishing performance on increasing grid size to 5×5 (which has even more data to train from), we hypothesize that this is not the case.

[4] See [7] for more details.

Figure 9: Some examples from the Count-QA VQA subset. Given a question, we parse the nouns and resolve correspondence to COCO categories. The resolved ground truth category is denoted after the question. We show the VQA ground truth and COCO dataset resolved ground truth counts, followed by outputs from `detect`, `glance`, `aso-sub`, `seq-sub` and `ens`.

Approach	mRMSE (VQA)	mRMSE (COCO-QA)
`detect`	2.72 ± 0.09	2.59 ± 0.12
`glance-ft-1L`	2.19 ± 0.05	1.86 ± 0.12
`aso-sub-ft-1L-3` $\times 3$	1.94 ± 0.07	1.47 ± 0.04
`seq-sub-ft-3` $\times 3$	1.81 ± 0.09	$\mathbf{1.34 \pm 0.07}$
`ens`	$\mathbf{1.80 \pm 0.07}$	1.40 ± 0.08
Deeper LSTM [21]	2.71 ± 0.23	N/A
SOTA VQA [15]	3.25 ± 0.94	N/A

Table 3: Performance of various methods on counting questions in the **Count-QA** splits of the VQA dataset and COCO-QA datasets respectively (`L` implies the number of hidden layers). **Lower** is better. `ens` is a combination of `glance-ft-1L`, `aso-sub-ft-1L-3` $\times$ 3 and `seq-sub-ft-3` $\times$ 3.

5.4. VQA Experiment

We explore how well our counting approaches do on simple counting questions. Recent work [2, 35, 31, 15] has explored the problem of answering free-form natural language questions for images. One of the large-scale datasets in the space is the Visual Question Answering [2] dataset. We also evaluate using the COCO-QA dataset from [35] which automatically generates questions from human captions. Around 10.28% and 7.07% of the questions in VQA and COCO-QA are "how many" questions related to counting objects. Note that both the datasets use images from the COCO [28] dataset. We apply our counting models, along with some basic natural language pre-processing to answer some of these questions.

Given the question "how many bottles are there in the fridge?" we need to reason about the object of interest (bottles), understand referring expressions (in the fridge) *etc*. Note that since these questions are free form, the category of interest might not exactly correspond to an COCO category. We tackle this ambiguity by using word2vec embeddings [32]. Given a free form natural language question, we extract the noun from the question and compute the closest COCO category by checking similarity of the noun with the categories in the word2vec embedding space. In case of multiple nouns, we just retain the first noun in the sentence (since how many questions typically have the subject noun first). We then run the counting method for the COCO category (see Fig 9). More details can be found in the supplementary. Note that parsing referring expressions is still an open research problem [23, 39]. Thus, we filter questions based on an "oracle" for resolving referring expressions. This oracle is constructed by checking if the ground truth count of the COCO category we resolve using word2vec matches with the answer for the question. Evaluating only on these questions allows us to isolate errors due to inaccurate counts. We evaluate our outputs using the $RMSE$ metric. We use this procedure to compile a list of 1774 and 513 questions (**Count-QA**) from the VQA and COCO-QA datasets respectively, to evaluate on. We will publicly release our Count-QA subsets to help future work.

We report performances in Table. 3. The trend of increasing performance is visible from `glance` to `ens`. We find that `seq-sub` significantly outperforms the other approaches. We also evaluate a state-of-the-art VQA model [15] on the Count-QA VQA subset and find that even `glance` does better by a substantial margin.[5]

6. Conclusion

We study the problem of counting *everyday* objects in *everyday* scenes. We evaluate some baseline approaches to this problem using object detection, regression using global image features, and associative subitizing which involves regression on non-overlapping image cells. We propose sequential subtizing, a variant of the associative subitizing model which incorporates context across cells using a pair of stacked bi-directional LSTMs. We find that our proposed models lead to improved performance on PASCAL VOC 2007 and COCO datasets. We thoroughly evaluate the relative strengths, weaknesses and biases of our approaches, providing a benchmark for future approaches on counting, and show that an ensemble of our proposed approaches peforms the best. Further, we show that counting can be used to improve object detection and present proof-of-concept experiments on answering 'how many?' questions in visual question answering tasks. Our code and datasets will be made publicly available.

Acknowledgements. We are grateful to the developers of Torch [9] for building an excellent framework. This work was funded in part by NSF CAREER awards to DB and DP, ONR YIP awards to DP and DB, ONR Grant N00014-14-1-0679 to DB, a Sloan Fellowship to DP, ARO YIP awards to DB and DP, an Allen Distinguished Investigator award to DP from the Paul G. Allen Family Foundation, Google Faculty Research Awards to DP and DB, Amazon Academic Research Awards to DP and DB, and NVIDIA GPU donations to DB. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government, or any sponsor.

[5] For the column corresponding to VQA, all methods are evaluated on the subset of the predictions where [21] and [15] both produced numerical answers. For [21], there were 11 non-numerical answers and for [15] there were 3 (e.g., "many", "few", "lot")

References

[1] A. Agrawal, D. Batra, and D. Parikh. Analyzing the behavior of visual question answering models. *CoRR*, abs/1606.07356, 2016. 3

[2] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. VQA: visual question answering. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 2425–2433, 2015. 1, 2, 8

[3] C. Arteta, V. Lempitsky, and A. Zisserman. Counting in the wild. In *European Conference on Computer Vision*, 2016. 2

[4] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Semantic segmentation with second-order pooling. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 7578 LNCS, pages 430–443, 2012. 1

[5] A. B. Chan and N. Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7. IEEE, 6 2008. 2

[6] A. B. Chan and N. Vasconcelos. Bayesian poisson regression for crowd counting. In *2009 IEEE 12th International Conference on Computer Vision*, pages 545–551. IEEE, 9 2009. 2

[7] P. Chattopadhyay, R. Vedantam, R. S. Ramprasaath, D. Batra, and D. Parikh. Counting everyday objects in everyday scenes. *CoRR*, abs/1604.03505, 2016. 3, 4, 6, 7

[8] D. H. Clements. Subitizing: What is it? why teach it? *Teaching children mathematics*, 5(7):400, 1999. 2

[9] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, 2011. 8

[10] S. Cutini and M. Bonato. Subitizing and visual short-term memory in human and non-human species: a common shared system? *Frontiers in Psychology*, 3, 2012. 1, 2

[11] S. Dehaene, V. Izard, E. Spelke, and P. Pica. Log or linear? distinct intuitions of the number scale in western and amazonian indigene cultures. *Science*, 320(5880):1217–1220, 2008. 5

[12] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. 10 2013. 2

[13] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, Jan. 2015. 4

[14] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2008. 1

[15] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 457–468, 2016. 1, 8

[16] F. Galton. One Vote, One Value. 75:414, Feb. 1907. 2

[17] S. Gidaris and N. Komodakis. Object detection via a multi-region and semantic segmentation-aware cnn model. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1134–1142, 2015. 1

[18] R. Girshick. Fast r-cnn. In *International Conference on Computer Vision (ICCV)*, 2015. 1, 2, 3, 7

[19] H. Idrees, I. Saleemi, C. Seibert, and M. Shah. Multi-source multi-scale counting in extremely dense crowd images. In *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '13, pages 2547–2554, Washington, DC, USA, 2013. IEEE Computer Society. 5

[20] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 448–456, 2015. 3

[21] D. B. Jiasen Lu, Xiao Lin and D. Parikh. Deeper lstm and normalized cnn visual question answering model. https://github.com/VT-vision-lab/VQA_LSTM_CNN, 2015. 8

[22] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, and R. Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017. 3

[23] S. Kazemzadeh, V. Ordonez, M. Matten, and T. L. Berg. Referit game: Referring to objects in photographs of natural scenes. In *EMNLP*, 2014. 8

[24] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 4

[25] A. Klein and P. Starkey. Universals in the development of early arithmetic cognition. *New Directions for Child and Adolescent Development*, 1988(41):5–26, 1988. 2

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012. 2

[27] V. Lempitsky and A. Zisserman. Learning To Count Objects in Images. In *Advances in Neural Information Processing Systems*, pages 1324–1332, 2010. 2

[28] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 5, 8

[29] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. E. Reed. SSD: single shot multibox detector. *CoRR*, abs/1512.02325, 2015. 1

[30] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015. 1

[31] M. Malinowski, M. Rohrbach, and M. Fritz. Ask your neurons: A neural-based approach to answering questions about images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1–9, 2015. 1, 8

[32] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013. 8

[33] D. Oñoro Rubio and R. J. López-Sastre. Towards perspective-free object counting with deep learning. In *ECCV*, 2016. 2

[34] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1

[35] M. Ren, R. Kiros, and R. Zemel. Exploring models and data for image question answering. In *Advances in Neural Information Processing Systems*, pages 2953–2961, 2015. 1, 2, 8

[36] M. Ren and R. S. Zemel. End-to-end instance segmentation and counting with recurrent attention. *CoRR*, abs/1605.09410, 2016. 1, 6

[37] S. Ren, K. He, R. Girshick, and J. Sun. 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, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015. 1

[38] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 3

[39] A. Sadovnik, A. C. Gallagher, and T. Chen. It's not polite to point: Describing people with uncertain attributes. In *CVPR*, pages 3089–3096. IEEE, 2013. 8

[40] M. Schuster and K. K. Paliwal. Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing*, 45:2673–2681, 1997. 4

[41] S. Seguí, O. Pujol, and J. Vitrià. Learning to count with deep object features. may 2015. 2

[42] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. 9 2014. 2, 3

[43] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition (CVPR)*, 1:I—-511—-I—-518, 2001. 1

[44] L. Wan, D. Eigen, and R. Fergus. End-to-end integration of a convolution network, deformable parts model and non-maximum suppression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 851–859, 2015. 1

[45] C. Zhang, H. Li, X. Wang, and X. Yang. Cross-Scene Crowd Counting via Deep Convolutional Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 833–841, 2015. 2, 5

[46] J. Zhang, S. Ma, M. Sameki, S. Sclaroff, M. Betke, Z. Lin, X. Shen, B. Price, and R. Měch. Salient object subitizing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 3, 5, 7