

Teaching Compositionality to CNNs*

Austin Stone
Yi Liu

Huayan Wang
D. Scott Phoenix

Michael Stark
Dileep George

Vicarious FPC, San Francisco, CA, USA

{austin, huayan, michael, yi, scott, dileep}@vicarious.com

Abstract

Convolutional neural networks (CNNs) have shown great success in computer vision, approaching human-level performance when trained for specific tasks via application-specific loss functions. In this paper, we propose a method for augmenting and training CNNs so that their learned features are compositional. It encourages networks to form representations that disentangle objects from their surroundings and from each other, thereby promoting better generalization. Our method is agnostic to the specific details of the underlying CNN to which it is applied and can in principle be used with any CNN. As we show in our experiments, the learned representations lead to feature activations that are more localized and improve performance over non-compositional baselines in object recognition tasks.

1. Introduction

Convolutional neural networks (CNNs) have shown remarkable performance in many computer vision tasks [21, 20, 42, 37, 35] including image classification [20], object class detection [41, 12], instance segmentation [13], image captioning [18, 44], and scene understanding [6]. Their success is typically attributed to two factors; they have large enough capacity to make effective use of the ever-increasing amount of image training data available today, while at the same time managing the number of free parameters through the use of inductive biases from neuroscience. Specifically, the interleaving of locally connected filter and pooling layers [15] bears similarity to the visual cortex’s interleaving of simple cells, which have localized receptive fields, and complex cells, which have wider receptive fields and greater local invariance.

Recently, researchers have investigated more inductive biases from neuroscience to improve CNN architectures. Examples include learning representations from video sequences [2, 10, 17], encouraging the utilization of depth in-

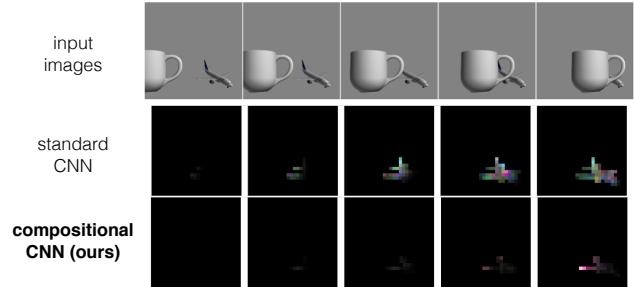


Figure 1: For a standard CNN (VGG, [37]), the presence of a nearby object (cup) greatly affects the activations in the region of an object of interest (airplane). In contrast, a CNN trained with our method demonstrates better compositionality in its feature representations – the activations in the airplane region represent primarily the airplane and are therefore less affected by the presence of the cup.

formation [14], and using physical interaction with the environment [31] to bias representations.

In this paper, we follow a similar philosophy, but focus our attention on the inductive bias of *compositionality*: the notion that *the representation of the whole should be composed of the representation of its parts* (we give a precise formal definition of this notion in Sect. 3). Intuitively, encouraging this property during training results in representations that are more robust to re-combination (e.g., when seeing a familiar object in a novel context) and less prone to focusing on discriminative but irrelevant background features. It is also in line with findings from neuroscience that suggest separate processing of figure and ground regions in the visual cortex [16, 32]. Note that a typical CNN does not exhibit this property (Fig. 1 visualizes the difference in activations between a CNN trained without (VGG [37]) and with our compositionality objective¹).

¹Fig. 1 shows the activation difference in the airplane region between the current frame and a frame where the airplane is shown in isolation. Activations are taken from intermediate conv. layers with spatial resolution 28×28 . We marginalize over feature channels to create visualization.

*Preprint appearing in CVPR 2017.

In contrast to previous work that designs compositional representations from the ground up [34, 45, 43, 47], our approach does not mandate any particular network architecture or parameterization – instead, it comes in the form of a modified training objective that can be applied to *teach any standard CNN* about compositionality in a soft manner. While our current implementation requires object masks for training, it allows apples to apples comparison of networks trained with or without the compositionality objective. As our experiments show (Sect. 4), the objective consistently improves performance over non-compositional baselines.

This paper makes the following specific contributions: First, we introduce a novel notion of *compositionality* as an inductive bias for training arbitrary convolutional neural networks (CNNs). It captures the intuition that *the representation of a partial image* should be equal to *the partial representation* of that image. Second, we implement that notion in the form of a modified CNN training objective, which we show to be straightforward to optimize yet effective in learning compositional representations. Third, we give an extensive experimental evaluation on both synthetic and real-world images that highlights the efficacy of our approach for object recognition tasks and demonstrates the contributions of different components of our objective.

2. Related work

Our work is related mostly to three major lines of research: compositional models, inductive biases, and the role of context in visual recognition.

Compositional models. Compositional models have existed since the early days of computer vision [24] and have appeared mainly in two different varieties. The first flavor focuses on the creation of hierarchical feature representations by means of statistical modeling [8, 27, 48, 47], reusable deformable parts [49, 29], or compositional graph structures [36, 45]. The second flavor designs neural network-based representations in the form of recursive neural networks [38], imposing hierarchical priors on Deep Boltzman Machines [34], or introducing parametric network units that are themselves compositional [43].

The basis for our work is a notion of compositionality (Sect. 3.1) that is distinct from all these approaches in that it does not have to be baked into the design of a model but can be applied as a soft constraint to a CNN. Recent work [28] constrains CNN activations to lie within object masks in the context of weakly-supervised localization. Our compositional objective (Sect. 3.3) goes beyond this formulation: it consists of multiple components that not only suppress background activations, but also explicitly encourage object activations to be invariant to both background clutter and adjacent objects. Our experiments verify that each component is important for performance (Sect. 4.3).

Inductive biases. A recent line of work on neural network architectures takes inspiration from human learning in its design of training regimen. It has demonstrated improved performance when training from video sequences instead of still images [2, 17], assuming an object-centric view [10], integrating multimodal sensory side information [14], or even being in control of movement [31]. The benefit arises from providing helpful inductive biases to the learner that regularize the learned representations. The inductive bias of compositionality presented in this work (Sect. 3.1) follows a similar motivation but is largely complementary to the biases explored by these prior approaches.

The role of context in visual recognition. It is well known that context plays a major role in visual recognition, both in human and artificial vision systems [9, 26, 5]. Our environment tends to be highly regular, and making use of regularities in the occurrence of different object and scene classes has been shown to be beneficial for recognizing familiar objects [25, 4], objects in unusual circumstances [3], and recurring spatial configurations [7, 30, 11, 46]. At the extreme, object classes can be successfully recognized even in the absence of local information by relying exclusively on scene context [33].

While CNN-based representations typically support the use of context implicitly (by including pixels indiscriminately in a receptive field), they lack the ability to explicitly address context and non-context information. The notion of compositionality proposed in this work (Sect. 3.1) is a step towards making CNN-based representations more amenable to explicit context modeling through an external mechanism (by cleanly separating the representation of objects from their context). The experiments in this paper (Sect. 4) do not further elaborate on this aspect, but indicate that the compositional objective (i) elicits a performance improvement, (ii) the improvement is similar for objects appearing in and out-of-context, and (iii) the improvement is least pronounced for very small object instances.

3. Teaching compositionality to CNNs

This section describes our approach to encouraging CNNs to learn compositional representations. To that end, we proceed from introducing our notion of compositionality (Sect. 3.1) to describing network architecture (Sect. 3.2) and training procedure (Sect. 3.3) to giving technical details of our implementation (Sect. 3.4).

3.1. Compositionality notion

The goal of our notion of compositionality is to encourage the representation of a part of an image to be similar to the corresponding part of the representation of that image. More formally, let X be an image, m a binary mask that

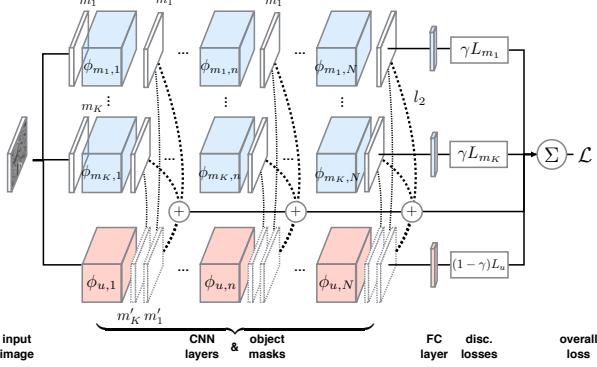


Figure 2: Architecture and loss function (\mathcal{L}) computation for encouraging compositionality when multiple objects are present in a training image (see Sect. 3.2 and Sect. 3.3). The original CNN (red) is enhanced by K additional masked CNNs (blue), all of them sharing weights. $\phi_{\cdot, \cdot}$ represent feature maps, L loss functions, and γ is a hyper-parameter. Masks m_k (solid) are applied to feature map outputs, while masks m'_k (dotted) are only applied for computing losses. For simplicity, we depict layers and masks with equal sizes.

identifies part of X (i.e., m is a tensor of the same shape as X with 1s indicating part affiliation), ϕ a mapping from an image onto an arbitrary feature layer of a CNN, and p the projection operator onto the feature map represented by ϕ . We define ϕ to be *compositional* iff the following holds:

$$\phi(m \cdot X) = p(m) \cdot \phi(X). \quad (1)$$

Here, the \cdot operator represents element-wise multiplication. The projection operator, p , down-samples the object mask to the size of the output of ϕ . E.g., if $\phi(X)$ is the activations of a convolutional layer with size (h, w, c) (the first two dimensions are spatial and c is the number of feature channels), p will down-sample the object mask to size (h, w) and then stack c copies of the down-sized object mask on top of each other to produce a mask of size (h, w, c) .

Note that in practice we do not require Eq. (1) to hold for all possible masks m , as this would constrain ϕ to be the identity map. Instead, we apply the inductive bias selectively to image parts that we would like to be treated as a unit – obvious choices for these selected parts include objects or object parts. In the following, we use object masks (as provided by standard data sets such as MS-COCO [23]) as the basis for compositionality.

3.2. Enhanced network architecture

To encourage a network to satisfy the compositionality property of Eq. (1) (Sect. 3.1), we devise an enhanced architecture and corresponding objective function. Note that this enhancement is non-destructive in nature and leaves the

original network completely intact; it merely makes virtual copies of the original network, Fig. 2.

When there is only one object in the input image, teaching compositionality takes the form of ensuring that the activations within the region of that object remain invariant regardless of what background the object appears on. With multiple objects, we also explicitly ensure that the activations of each object remain the same as if that object were shown in isolation (i.e., activations should be invariant to the other objects within the respective object mask).

To implement this notion, we create $K + 1$ weight-sharing CNNs where K is the number of objects shown in the scene. K of these CNNs take as input a different object instance, each shown against a blank background (we apply the mask for the k th object instance to the input image before giving the input image to the k th CNN). We refer to these K CNNs as “masked CNNs,” and we denote the mapping onto layer n of the k th masked CNN as $\phi_{m_k, n}$.

Each of these K masked CNNs have their respective object mask reapplied to their activations at multiple layers in the hierarchy (see Sect. 3.4), zeroing out activations outside of the object region. These masked activations are then passed on to higher layers (which might also re-apply the mask again in the same way). This constrains the masked CNNs to only use activations within the object mask region when classifying the input image. The final ($K + 1$ th) CNN receives as input the original image with no masks applied, and we refer to it subsequently as the “unmasked CNN”. We denote the mapping onto layer n of this CNN as $\phi_{u, n}$. We denote the total number of layers as N .

3.3. Training procedure

We train the architecture of Sect. 3.2 for compositionality by introducing an objective function that combines an application-specific discriminative CNN loss with additional terms that establish dependencies between the different masked and unmasked CNNs.

Discriminative loss. To encourage correct discrimination, we add separate discriminative loss terms for both the K masked and the one unmasked CNN, denoted L_{m_k} and L_u , respectively. Their relative contributions are controlled by the hyperparameter $\gamma \in [0, 1]$, to yield

$$\mathcal{L}_d = \frac{1}{K} \left(\sum_k \gamma L_{m_k} \right) + (1 - \gamma) L_u. \quad (2)$$

Compositional loss. To encourage compositionality, we add $K \times N$ terms that establish dependencies between the responses of corresponding layers of the masked and unmasked CNNs, respectively. Specifically, on all layers at which an object mask is applied, we take the l_2 difference

between the activations of the masked CNN and the activations of the unmasked CNN. We then multiply this difference by a layer specific penalty hyper-parameter (denoted as λ_n) and add this to our compositional loss:

$$\mathcal{L}_c = \frac{1}{K} \sum_k \sum_n \lambda_n \|\phi_{m_k, n} - \phi_{u, n} m'_k\|_2^2. \quad (3)$$

The final objective can then be stated simply as $\mathcal{L} = \mathcal{L}_d + \mathcal{L}_c$. Because the unmasked CNN sees all objects and will naturally have different activations from the k th masked CNN due to the presence of the objects other than the k th object, we apply a mask to the unmasked CNN’s activations before computing the penalty term. We denote this mask as m'_k . However, we do not pass these masked activations ($\phi_{u, n} m'_k$) up to higher layers as was done for the masked CNNs; we only use them to compute the compositional penalty term on layer n .

Design choices. The above objective leaves degrees of freedom w.r.t. choosing the precise nature of the masks m'_k , and the corresponding choices do have an impact on performance (Sect. 4.3). First, to penalize background activations outside of the regions of objects of interest, we can make m'_k be a tensor of 1s but with the locations of all objects other than the k th object filled with 0s. Second, we can penalize any shifts in activations within the region of the k th object without discouraging background activations by making m'_k equal to m_k .

3.4. Implementation details

Our experiments (Sect. 4) use the following network architectures: MS-COCO-sub (Sect. 4.4): conv1-conv3 ($224 \times 224 \times 64$), pool1, conv4-conv6 ($128 \times 128 \times 128$), pool2, conv7-conv9 ($64 \times 64 \times 256$), pool3, conv10-conv12 ($32 \times 32 \times 512$), pool4, fc1 (131072×20). 3D-Single (Sect. 4.3): conv1-conv3 ($128 \times 128 \times 64$), pool1, conv4-conv6 ($64 \times 64 \times 128$), pool2, conv7-conv9 ($32 \times 32 \times 256$), pool3, conv10-conv12 ($16 \times 16 \times 512$), pool4, fc1 (32768×14). MNIST (Sect. 4.3): conv1-conv3 ($120 \times 120 \times 32$), pool1, conv3-conv4 ($60 \times 60 \times 64$), pool2, conv5-conv6 ($30 \times 30 \times 128$), pool3, fc1 (28800×10).

The discriminative loss functions L_{m_k} and L_u are instantiated as softmax-cross entropy or element-wise sigmoid-cross entropy for joint or independent class prediction, respectively. Since \mathcal{L}_c is of a standard form, we can optimize it like any CNN via SGD (specifically, using the ADAM optimizer [19] and Tensorflow [1]).

Empirically, we find that best performance is achieved when applying \mathcal{L}_c to the topmost convolutional and pooling layers of the network (i.e., λ_n is zero on most early layers). We believe this to be an artifact of the CNN needing a certain minimum number of layers and corresponding representational power to successfully discriminate between relevant and irrelevant (background) pixels.

In practice, we create only two weight-sharing CNNs (independent of the number of object training instances): one which sees 1 randomly selected out of K objects in the input image, and one which sees the entire scene. Empirically, this model is only about 50% slower to train than a standard CNN. The parameter space is just that of a single CNN due to weight sharing. γ is fixed to .5.

4. Experiments

In this section, we give a detailed experimental evaluation of our approach for teaching compositionality to CNNs, highlighting its ability to improve performance over standard CNN training on both synthetic (Sect. 4.3) and real images (MS-COCO [23], Sect. 4.4). Our emphasis lies on providing an in-depth analysis of the contributions of different components of our compositional objective as well as quantifying the impact of object context on performance.

4.1. Datasets and metrics

Rendered 3D objects. We perform diagnostic experiments on two novel datasets of rendered 3D objects. We use rendered datasets so we have maximum control over the statistics of our image data in terms of depicted objects and context (notably, segmentation masks come for free in this setting). Specifically, the datasets are based on 12 3D object classes (e.g., car, bus, boat, or airplane), with ≈ 20 object instances per category, each rendered from ≈ 50 different viewpoints (uniform sampling of the upper viewing half-sphere) in front of 20 different real-image backgrounds.

The first dataset, termed 3D-Single, has 1,600 images depicting single objects in front of random backgrounds. The second dataset, termed 3D-Multi, has 800 images of multiple objects with varying degrees of occlusion (see Fig. 1). For both datasets, we distinguish between a category-level recognition setting and easier variants (3D-Single-Inst, 3D-Multi-Inst) that allow the set of 3D object instances seen during training and test to be non-empty (whereas the set of views of the same instance has to be empty). In both cases, we ensure that the backgrounds seen in training (80% of the images) and test (20%) are distinct.

MNIST. We create two variants of the popular MNIST dataset [22], in analogy to the two aforementioned 3D object datasets. The first variant, MNIST-Single, depicts individual MNIST characters in front of randomized, cluttered backgrounds (we use the standard train/test split). The second variant, MNIST-Multi, depicts multiple characters with varying degrees of overlap, against these backgrounds.

MS-COCO-sub. MS-COCO [23] constitutes a move away from “iconic” views of objects towards a dataset in which objects frequently occur in their respective natural

contexts. For most experiments, we focus on subsets of MS-COCO training and validation (for testing) images that contain at least one of 20 diverse object classes² (see Fig. 4a) and further restrict the set of images to ones with sufficiently large object instances of at least 7,000 pixels. This results in 22,476 training and 12,245 test images.

In addition, we quantify the impact of context on classification performance by defining two further test sets. The first test set is the full validation set of MS-COCO. Here, we measure classification performance on object instances of different sizes (small, medium, large) as defined for the MS-COCO detection challenge³. In order to make the performance comparable, we stratify the number of positive and negative examples by randomly sampling 20 negatives for each positive example.

The second test set examines object instances *in* and *out* of *context* (see Fig. 5 (b)). We start with all test images from MS-COCO-sub. For each object instance o of a category c occurring in that set, we create two positive examples, one by cropping o and placing it in front of a new random test image that does not have c in it (this will be the out-of-context set), and one by leaving o in its original context (the in-context set). For both, we add as negative examples all images where c does not occur.

Metrics. All experiments consider image-level classification tasks, not object class detection or localization. For diagnostic experiments (Sect. 4.3), we evaluate performance as the average fraction of correctly predicted object classes among the top- k scoring network predictions, where k is the number of objects in a given image. For MS-COCO-sub (Sect. 4.4), we treat object classes separately and report (mean) average precision (AP) over independent binary classification problems. In all cases, we monitor performance on a held-out test set over different epochs as training progresses, and report both the resulting curves and best achieved values per method (Fig. 3, Fig 4).

4.2. Methods

In this section, we evaluate the following baselines and variations on our compositional training technique (see Sect. 3.3). For the sake of clean comparison, we always train all networks from scratch (i.e., we do not use pre-training of any form).

COMP-FULL. Our main architecture, where m'_k is chosen to be equal to a block of all 1s but with the locations of objects other than the k th object set to 0s.

COMP-OBJ-ONLY. Like COMP-FULL, but with m'_k equal to m_k (this penalizes any shifts in activations within the object region but does not discourage background activations).

COMP-NO-MASK. Like COMP-FULL, except that the

²The first 20 classes in the original MS-COCO ordering w/o *person*.

³<http://mscoco.org/dataset/#detections-eval>

masked CNNs do not apply m_k to any of their activations.

BASELINE. Architecture with the same layer sizes as COMP-FULL but without compositional objective terms – a “standard” CNN.

BASELINE-AUG. Like BASELINE, except for each batch we make half of the images be a single object shown in isolation against a black background and the other half be the raw images of the same objects in the same locations against cluttered background. This method has access to the same information as COMP-FULL (it knows about the object mask), but without any compositional objective.

BASELINE-REG. Like BASELINE, but with dropout [40] and l_2 -regularization.

BASELINE-AUG-REG. Like BASELINE-AUG, but with dropout and l_2 -regularization.

4.3. Diagnostic experiments on synthetic data

We commence by comparing the performance of different variants of our compositional objective and the corresponding baselines (Sect. 4.2) in a diagnostic setting on synthetic data. In order to assess both best case performance and convergence behavior, we plot test performance vs. training epochs in Fig. 3a through 3f. The respective best performance per curve is given in parentheses in plot legends. Fig. 5 and 6 give qualitative results.

Rendered 3D objects. In Fig. 3, we observe that all variants of compositional CNNs (blue curves) perform consistently better than the baselines (red curves), both per epoch and in terms of best case performance.

Our full model, COMP-FULL, performs overall best (blue-solid). It outperforms the best baseline by between 17.1% (3D-Multi, Fig. 3d) and 35.2% (3D-Single-Inst, Fig. 3a). Performance drops for COMP-OBJ-ONLY (blue-dashed) by 14.7% (3D-Single-Inst, Fig. 3a), 7.3% (3D-Single, Fig. 3b), 4.4% (3D-Multi-Inst, Fig. 3c), and 2.9% (3D-Multi, Fig. 3d), respectively. COMP-NO-MASK (blue-dotted) performs worst among our models, but still outperforms the best baseline by 0.3% (3D-Single-Inst, Fig. 3a), 6.8% (3D-Single, Fig. 3b), 26.6% (3D-Multi-Inst, Fig. 3c), and 17.0% (3D-Multi, Fig. 3d), respectively.

As expected, the baseline benefits from observing additional masked training data mostly for images with multiple objects: BASELINE (red-dashed) and BASELINE-AUG (red-solid) perform comparably on 3D-Single-Inst and 3D-Single, but BASELINE-AUG improves over BASELINE on 3D-Multi-Inst (by 6.2%) and 3D-Multi (by 7.8%). In terms of convergence, the compositional CNNs (blue curves) tend to stabilize later than the baselines (red curves).

MNIST. In Fig. 3e and 3f, the absolute performance differences between our compositional CNNs and the corresponding baselines are less clear cut, but still highlight

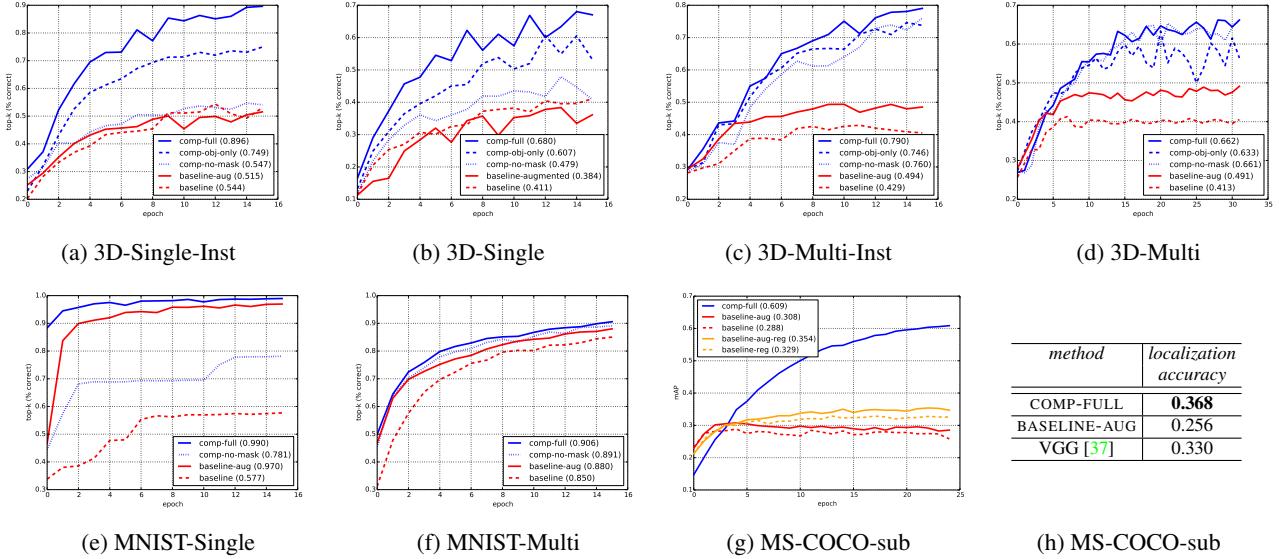


Figure 3: Test performance on rendered 3D objects (a-d), MNIST (e-f), and MS-COCO-sub (g), as training progresses over epochs (best performance per curve given in parentheses; see Sect. 4.3 and 4.4). Localization accuracy (h) (Sect. 4.4).

the importance of the compositional objective when object masks are used (COMP-FULL outperforms BASELINE-AUG by 2.0% on MNIST-Single and by 2.6% on MNIST-Multi). Without reapplying the masks to the activations, performance decreases, but the trend remains (COMP-NO-MASK is better than BASELINE by 20.4% and 4.1%, respectively).

4.4. Experiments on real-world data (MS-COCO)

We proceed to evaluating our best performing method COMP-FULL on the real-world images of MS-COCO (Sect. 4.1). We compare to the same baselines as before, plus two baselines with dropout [40] and l_2 -regularization (see Sect. 4.2). Specifically, we report performance for COMP-FULL at convergence (last epoch, see Fig. 3g for convergence behavior); for all baselines we consider the best performing model over all epochs. Fig. 4 gives details w.r.t. individual object categories (4a), amount of training data (4b), size of object instances (4c), and context (4d).

MS-COCO-sub. In Fig. 3g, COMP-FULL (blue-solid) outperforms the best baseline BASELINE-AUG-REG (orange-solid) by a significant margin of 25.5%, confirming the benefit of the compositionality objective in a real-world setting. The added regularization improves the performance of the baselines only moderately, by 4.6% (BASELINE-AUG) and 4.1% (BASELINE), respectively. In Fig. 4a, we see that COMP-FULL performs better than the baselines for every single category, improving performance by up to 32% (for *stop sign*). Fig. 4b gives results for varying amounts of training data (5, 10, 20, 50, 75, 100%).

method	in-context	out-of-context	ratio
COMP-FULL	0.660	0.256	0.39
BASELINE-AUG	0.356	0.131	0.37
BASELINE	0.334	0.116	0.35
BASELINE-AUG-REG	0.389	0.144	0.37
BASELINE-REG	0.374	0.128	0.34

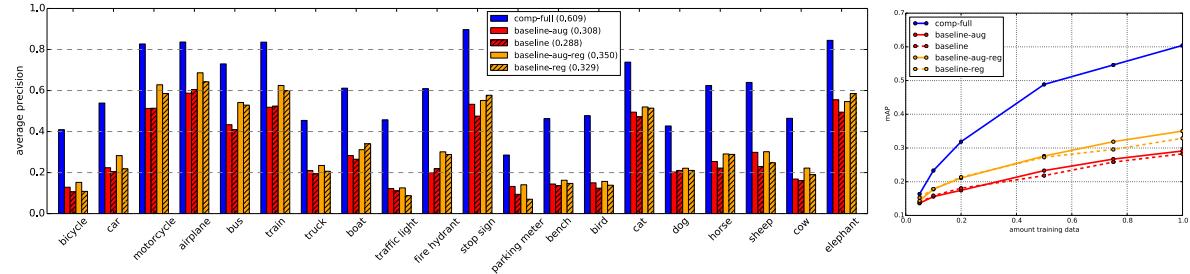
Table 1: Relative performance ratio on MS-COCO-sub.

COMP-FULL (blue-solid) clearly outperforms the baselines (orange and red curves) for all plotted amounts, with an increasing performance gap as training data increases.

Object sizes and context. Fig. 4c gives the performance when testing the respective models trained on the training portion of MS-COCO-sub on all images of MS-COCO and evaluating them on object instances of different sizes (small, medium, large, all; see Sect. 4.1).

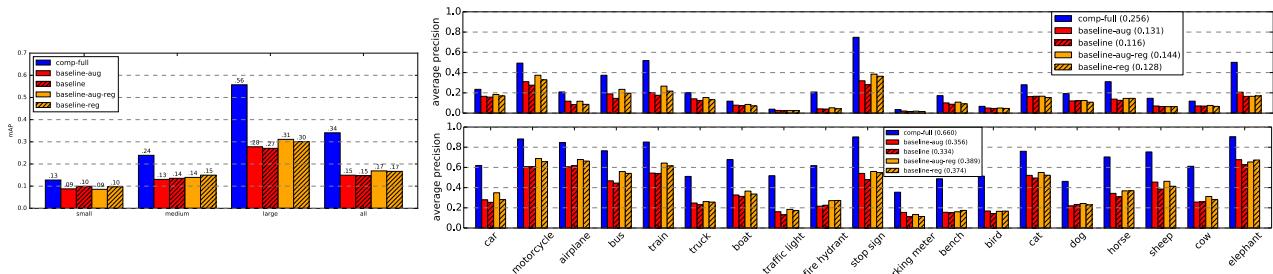
We observe that the compositional objective improves performance over the baselines consistently for all sizes. The improvement is most pronounced for large object instances (25%, COMP-FULL vs. BASELINE-AUG-REG), decreases for medium (9%, BASELINE-REG), and almost vanishes for small objects (3%, BASELINE-REG). This ordering is in line with the intuition that the compositionality objective encourages activations to be context invariant: as context becomes more important with decreasing object size, the advantage of context invariance decreases.

Fig. 4d explicitly examines the role of context, by comparing the performance on the in- (Fig. 4d (bottom)) and out-of-context (Fig. 4d (top)) test sets defined in Sect. 4.1



(a) MS-COCO-sub test performance (AP) per object category (corresponding to the last epoch for COMP-FULL / best case performance for baselines reported in Fig. 3g).

(b) Performance (mAP) for fractions of training data.



(c) MS-COCO-sub performance for objects of different sizes (small, medium, large).

(d) MS-COCO-sub performance for objects in-context (bottom) and out-of-context (top). See Sect. 4.4 for details.

Figure 4: MS-COCO-sub performance per object class (a), training set size (b), object instance size (c), and context (d).

(Fig. 5 (b) gives examples). Indeed, COMP-FULL improves performance over the baselines in all cases: COMP-FULL outperforms BASELINE-AUG-REG by 27.1% (in-context) and BASELINE-AUG-REG by 11.2% (out-of-context). The relative performance ratio (Tab. 1) between in- and out-of-context objects is slightly more favorable for COMP-FULL (0.39) than for BASELINE-AUG-REG (0.37).

Localization accuracy. Fig. 5 and 6 give qualitative results that highlight two distinct properties of our compositional objective COMP-FULL. First, it leads to bottom-up network activations that are better localized than for BASELINE-AUG, as indicated visually by the differences in masked and unmasked activations (Fig. 5). Second, it also leads to better localization when backtracing classification decisions to the input images, which we implement by applying guided backpropagation [39] (Fig. 6). Fig. 3h quantifies this on all test images of MS-COCO-sub, by computing the percentage of “mass” of the back-trace heat-map inside the ground-truth mask of the back-traced category, averaged over categories. COMP-FULL outperforms both BASELINE-AUG and VGG [37] by considerable margins.

Discussion. To our knowledge, only [28] reports classification (not detection) performance on MS-COCO, achieving 62.8% mAP on the full set of 80 classes using fixed lower-layer weights from ImageNet pre-training [20] and an elaborate multi-scale, sliding-window network architecture.

In comparison, our COMP-FULL achieves 34% on 20 classes (Fig. 4c, ‘all’ column) when trained entirely from scratch using only a small fraction of the full data (6% with area above 7,000) and a single, fixed scale window (the original image), outperforming the best baseline BASELINE-REG by 17%. We believe this to be an encouraging result that is complementary to the gains reported by [28] and leave the combination of both as a promising avenue for future work.

5. Conclusion

We have introduced an enhanced CNN architecture and novel loss function based on the inductive bias of compositionality. It follows the intuition that the representation of part of an image should be similar to the corresponding part of the representation of that image and is implemented as additional layers and connections of an existing CNN.

Our experiments indicate that the compositionality bias aids in the learning of representations that generalize better when training networks from scratch, and improves the performance in object recognition tasks on both synthetic and real-world data. Obvious next steps include the application to tasks that explicitly require spatial localization, such as image parsing, and combination with pre-trained networks.

Acknowledgments. We thank John Bauer and Robert Hafner for support with experiment infrastructure.

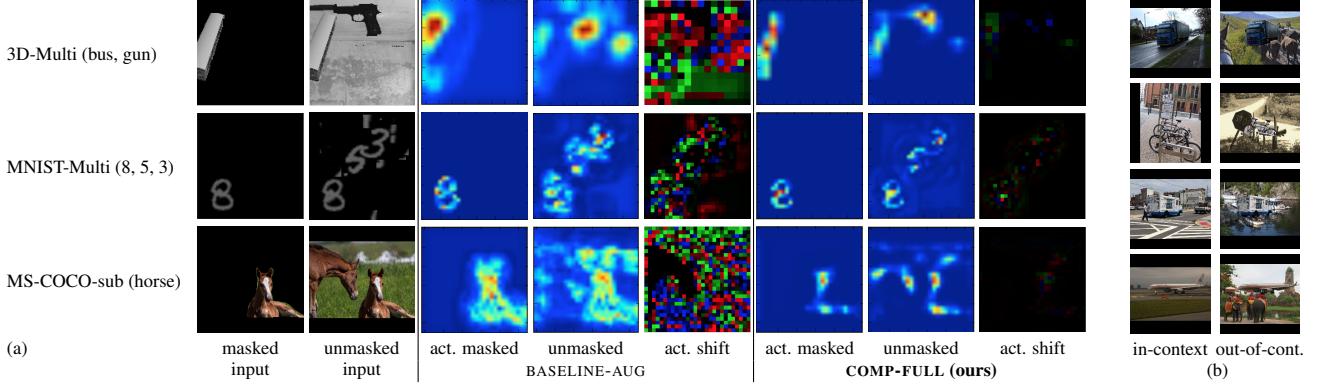


Figure 5: Shifts in conv-12 activations on test images (a). When the object context contains other objects in addition to the isolated object in the first column, we apply the mask for these additional objects to the visualizations of the activation shifts. Example images in-context and out-of-context (b).

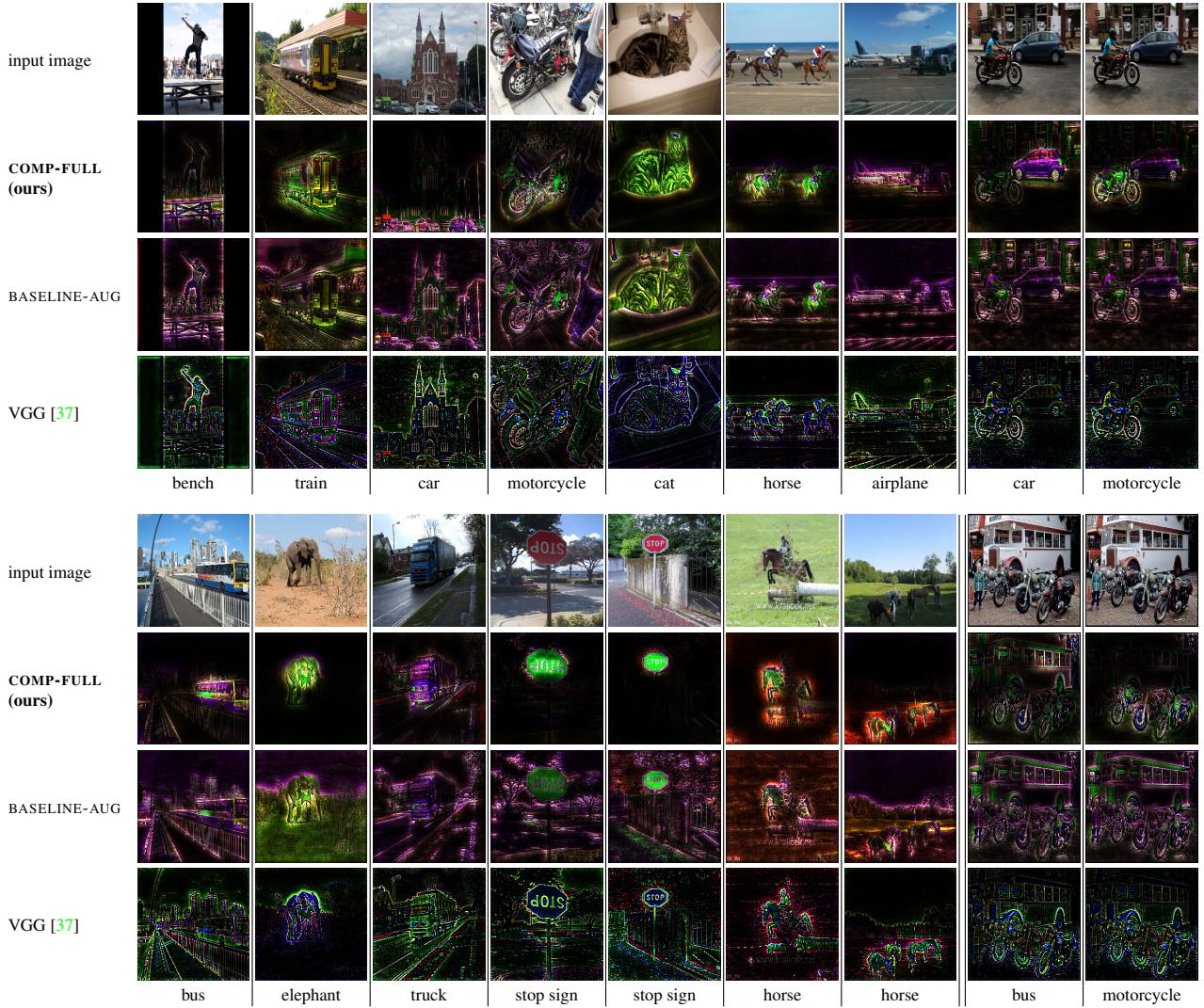


Figure 6: Backtracing classification activations (MS-COCO categories, denoted by column labels) to test images using guided backpropagation [39]. Please note the ability of COMP-FULL to backtrace to different object categories in one image, whereas BASELINE-AUG and VGG produce very similar outputs (rightmost 2 columns). Since VGG was trained on ImageNet categories, which are different from MS-COCO categories, we either backtrace from a semantically close category (identified manually) or VGG's top classification decision when a semantically close category cannot be found.

References

- [1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. [4](#)
- [2] P. Agrawal, J. Carreira, and J. Malik. Learning to see by moving. In *ICCV*, 2015. [1](#) [2](#)
- [3] M. J. Choi, A. Torralba, and A. S. Willsky. Context models and out-of-context objects. *Pattern Recognition Letters*, 2012. [2](#)
- [4] R. G. Cinbis and S. Sclarof. Contextual object detection using set-based classification. In *ECCV*, 2012. [2](#)
- [5] S. K. Divvala, D. Hoiem, J. H. Hays, A. A. Efros, and M. Hebert. An empirical study of context in object detection. In *CVPR*, 2009. [2](#)
- [6] S. M. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, K. Kavukcuoglu, and G. E. Hinton. Attend, infer, repeat: Fast scene understanding with generative models. In *NIPS*, 2016. [1](#)
- [7] A. Farhadi and M. A. Sadeghi. Recognition using visual phrases. In *CVPR*, 2011. [2](#)
- [8] S. Fidler and A. Leonardis. Towards scalable representations of object categories: Learning a hierarchy of parts. In *CVPR*, 2007. [2](#)
- [9] C. Galleguillos and S. Belongie. Context based object categorization: A critical survey. *CVIU*, 2010. [2](#)
- [10] R. Gao, D. Jayaraman, and K. Grauman. Object-centric representation learning from unlabeled videos. In *ACCV*, 2016. [1](#) [2](#)
- [11] S. Gupta, B. Hariharan, and J. Malik. Exploring person context and local scene context for object detection. *arXiv*, 2015. [2](#)
- [12] B. Hariharan, P. Arbelaez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *ECCV*, 2014. [1](#)
- [13] B. Hariharan, P. Arbelaez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *CVPR*, 2015. [1](#)
- [14] J. Hoffman, S. Gupta, and T. Darrell. Learning with side information through modality hallucination. In *CVPR*, 2016. [1](#) [2](#)
- [15] D. H. Hubel and T. N. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of Physiology*, 160:106–154, 1962. [1](#)
- [16] J. M. Hup, A. James, B. R. Payne, S. G. Lomber, P. Girard, and J. Bullier. Cortical feedback improves discrimination between figure and background by v1, v2 and v3 neurons. *Nature*, 394:784–787, 1998. [1](#)
- [17] D. Jayaraman and K. Grauman. Slow and steady feature analysis: Higher order temporal coherence in video. In *CVPR*, 2016. [1](#) [2](#)
- [18] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, 2015. [1](#)
- [19] D. P. Kingma and J. Lei Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. [4](#)
- [20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *NIPS*, 2012. [1](#) [7](#)
- [21] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989. [1](#)
- [22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, pages 2278–2324, 1998. [4](#)
- [23] T.-Y. Lin, M. Maire, S. Belongie, J. H. P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. [3](#) [4](#)
- [24] D. Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. MIT Press, 1982. [2](#)
- [25] K. Murphy, A. Torralba, and W. T. Freeman. Using the forest to see the trees: A graphical model relating features, objects, and scenes. In *NIPS*, 2003. [2](#)
- [26] A. Oliva and A. Torralba. The role of context in object recognition. *TRENDS in Cognitive Sciences*, 2007. [2](#)
- [27] B. Ommer and J. M. Buhmann. Learning the compositional nature of visual object categories for recognition. *PAMI*, 2010. [2](#)
- [28] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Is object localization for free? weakly-supervised learning with convolutional neural networks. In *CVPR*, 2015. [2](#) [7](#)
- [29] P. Ott and M. Everingham. Shared parts for deformable part-based models. In *CVPR*, 2011. [2](#)
- [30] B. Pepik, M. Stark, P. Gehler, and B. Schiele. Occlusion patterns for object class detection. In *CVPR*, 2013. [2](#)
- [31] L. Pinto, D. Gandhi, Y. Han, Y.-L. Park, and A. Gupta. The curious robot: Learning visual representations via physical interactions. In *ECCV*, 2016. [1](#) [2](#)
- [32] J. Poort, M. W. Self, B. v. Vugt, H. Malkki, and P. R. Roelfsema. Texture segregation causes early figure enhancement and later ground suppression in areas v1 and v4 of visual cortex. *Cerebral Cortex*, pages 3964–3976, 2016. [1](#)
- [33] B. C. Russell, A. Torralba, C. Liu, R. Fergus, and W. T. Freeman. Object recognition by scene alignment. In *NIPS*, 2007. [2](#)
- [34] R. Salakhutdinov, J. B. Tenenbaum, and A. Torralba. Learning with hierarchical-deep models. *PAMI*, 2013. [2](#)
- [35] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015. [1](#)
- [36] Z. Si and S.-c. Zhu. Learning and-or templates for object recognition and detection. *PAMI*, 35:2189–2205, 2013. [2](#)
- [37] K. Simonyan and A. Zisserman. Very deep convolutional networks for large scale image recognition. *ICLR*, 2015. [1](#), [6](#), [7](#), [8](#)
- [38] R. Socher, C. C.-Y. Lin, A. Y. Ng, and C. D. Manning. Parsing natural scenes and natural language with recursive neural networks. In *ICML*, 2011. [2](#)

- [39] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. In *ICLR-WS*, 2015. [7](#), [8](#)
- [40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 2014. [5](#), [6](#)
- [41] C. Szegedy, A. Toshev, and D. Erhan. Deep neural networks for object detection. In *NIPS*, 2013. [1](#)
- [42] L. Szegedy, C., P. W., Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CVPR*, 2015. [1](#)
- [43] D. Tabernik, M. Kristan, J. L. Wyatt, and A. Leonardis. Towards deep compositional networks. *arxiv*, 2016. [2](#)
- [44] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *ICML*, 2015. [1](#)
- [45] J. Wang and A. Yuille. Semantic part segmentation using compositional model combining shape and appearance. In *CVPR*, 2015. [2](#)
- [46] T. Wu, B. Li, and S. Zhu. Learning and-or models to represent context and occlusion for car detection and viewpoint estimation. *PAMI*, 2016. [2](#)
- [47] A. Yuille and R. Mottaghi. Complexity of representation and inference in compositional models with part sharing. *JMLR*, 2016. [2](#)
- [48] L. Zhu, Y. Chen, A. Torralba, W. Freeman, and A. Yuille. Part and appearance sharing: Recursive compositional models for multi-view multi-object detection. In *CVPR*, 2010. [2](#)
- [49] L. Zhu, Y. Chen, A. Yuille, and W. Freeman. Latent hierarchical structural learning for object detection. In *CVPR*, 2010. [2](#)