

# MIST: Multiple Instance Spatial Transformer

Baptiste Angles<sup>1,\*</sup>, Yuhe Jin<sup>2,\*</sup>, Simon Kornblith<sup>3</sup>,  
 Andrea Tagliasacchi<sup>3</sup>, Kwang Moo Yi<sup>1,2</sup>

<sup>1</sup>University of Victoria, <sup>2</sup>University of British Columbia,  
<sup>3</sup>Google Research, \*equal contributions

## Abstract

We propose a deep network that can be trained to tackle image reconstruction and classification problems that involve detection of multiple object instances, without any supervision regarding their whereabouts. The network learns to extract the most significant  $K$  patches, and feeds these patches to a task-specific network – e.g., auto-encoder or classifier – to solve a domain specific problem. The challenge in training such a network is the non-differentiable top- $K$  selection process. To address this issue, we lift the training optimization problem by treating the result of top- $K$  selection as a slack variable, resulting in a simple, yet effective, multi-stage training. Our method is able to learn to detect recurring structures in the training dataset by learning to reconstruct images. It can also learn to localize structures when only knowledge on the occurrence of the object is provided, and in doing so it outperforms the state-of-the-art.

## 1. Introduction

Finding and processing multiple instances of characteristic entities in a scene is core to many computer vision applications, including object detection [37, 17, 36], pedestrian detection [9, 41, 55], and keypoint localization [29, 2]. In traditional pipelines, it is common to localize entities by selecting the top- $K$  responses in a heatmap and use their locations [29, 2, 12]. However, this type of approach does not provide a gradient with respect to the heatmap, and cannot be directly integrated into neural network-based systems.

To overcome this challenge, previous work proposed to use grids [35, 17, 8] to simplify the formulation by isolating each instance [53], or to optimize over multiple branches [33]. While effective, these approaches require additional supervision to localize instances, and do not generalize well outside their intended application domain. Other formulations, such as sequential attention [1, 15, 10] and channel-wise approaches [57] are problematic to apply when

the number of instances of the same object is large, as we show later through experiments.

Here, we introduce a novel way to tackle this problem, which we term *Multiple Instance Spatial Transformer*, or *MIST* for brevity. As illustrated in Figure 1 for the image synthesis task, given an image, we first compute a heatmap via a deep network whose local maxima correspond to locations of interest. From this heatmap, we gather the parameters of the top- $K$  local maxima, and then extract the corresponding collection of image patches via an image sampling process. We process each patch independently with a task-specific network, e.g., an image decoder, and aggregate the network’s output across patches.

Training a pipeline that includes a non-differentiable selection/gather operation is non-trivial. Thus, we propose to lift the problem to a higher dimensional one by treating the parameters defining the interest points as slack variables, and introduce a hard constraint that they must correspond to the output of the heatmap network. This constraint is realized by introducing an auxiliary function that generates a heatmap given a set of interest point parameters. We then solve for the relaxed version of this problem, where the hard constraint is turned into a soft one, and the slack variables are also optimized within the training process. Critically, our training strategy allows the network to incorporate both non-maximum suppression and top- $K$  selection. We evaluate the performance of our approach for ① recovering the basis functions that created a given texture, ② detection and classification of handwritten digits in cluttered scenes, and ③ object detection on natural images, all without any location supervision.

**Contributions** In summary, in this paper we:

- propose an end-to-end training method that allows the use of top- $K$  selection;
- show that our framework can reconstruct images as parts, as well as detect/classify instances without any location supervision;
- outperform the state of the art in various scenarios, including on natural images.

## 2. Related work

Focusing on a sub-region in an image is in an essence an attention model. Attention models and the use of localized information have been actively investigated in the literature. Some examples include discriminative tasks such as fine-grained classification [42], pedestrian detection [55], and generative ones such as image synthesis from natural language [24]. They have also been studied in the context of more traditional Multiple Instance Learning (MIL) setup [20]. We now discuss a selection of representative works, and classify them according to how they deal with multiple instances.

**Grid-based methods.** Since the introduction of Region Proposal Networks (RPN) [37], grid-based strategies have been used for dense image captioning [25], instance segmentation [17], keypoint detection [14], and multi-instance object detection [36]. Recent improvements to RPNs attempt to learn the concept of a generic object covering multiple classes [40], and to model multi-scale information [6]. The multiple transformation corresponding to separate instances can also be densely regressed via Instance Spatial Transformers [50], which removes the need to identify discrete instance early in the network. However, all these methods are fully supervised, requiring both class *labels* and object *locations* for training.

**Heatmap-based methods.** Heatmap-based methods have recently gained interest to detect features [53, 33, 8], find landmarks [57, 31], and regress human body keypoints [47, 32]. While it is possible to output a separate heatmap per class [57, 47], most heatmap-based approaches do not distinguish between instances. [53] re-formulate the problem based on each instance, but in doing so introduce a sharp difference between training and testing. Grids can also be used with heatmaps [8], but this results in an unrealistic assumption of uniformly distributed detections in the image. Overall, heatmap-based methods excel when the “final” task of the network is to generate a heatmap [31], but are problematic to use as an intermediate layer in the presence of multiple instances.

**Sequential inference methods.** Another way to approach multi-instance problems is to attend to one instance at a time in a sequentially. Training neural network-based models with sequential attention is challenging, but approaches using policy gradient [1] and differentiable attention mechanisms [15, 10] have achieved some success for images comprising *small* numbers of instances. However, Recurrent Neural Networks (RNN) often struggle to generalize to sequences longer than the ones encountered during training, and while recent results on inductive reasoning are promising [16], their performance does not scale well when the number of instances is large.

**Knowledge transfer.** To overcome the acquisition cost of

labelled training data, one can transfer knowledge from labeled to unlabeled dataset. For example, [21] train on a single instance dataset, and then attempt to generalize to multi-instance domains, while [48] attempt to also transfer a multi-class proposal generator to the new domain. While knowledge transfer can be effective, it is highly desirable to devise unsupervised methods such as ours that do not depend on an additional dataset.

**Weakly supervised methods.** To further reduce the labeling effort, weakly supervised methods have also been proposed. [49] learn how to detect multiple instances of a single object via region proposals and ROI pooling, while [45] propose to use a hierarchical setup to refine their estimates. [13] provides an additional supervision by specifying the number of instances in *each* class, while [34] and [56] localize objects by looking at the network activation maps [58, 38]. [39] introduce an adversarial setup, where detection boxes are supervised by distribution assumptions and classification objectives. Recently, [27] proposed to accompany a segmentation task to improve the two-branch architecture of [3]. However, all these methods still rely on region proposals from an existing method, or define them via a hand-tuned process.

**Differentiable top-k methods.** Recently, researchers proposed a differentiable formulation of top-K via optimal transport [52]. This method unfortunately often requires too much memory in its application, and is geared towards an image classification setup. We show in Section 5.2 that our method performs better.

## 3. MIST framework

We first explain our framework at a high level, and then detail each component in Section 4. A prototypical MIST architecture (Figure 1) is composed of two trainable components: ① the first module receives an image as input and extracts  $K$  patches, at image locations and scales given by the top  $K$  local maxima of a heatmap generated by a trainable heatmap network  $\mathcal{H}_\eta$  with weights  $\eta$ . ② the second module processes each extracted patch with a task-specific network  $\mathcal{T}_\tau$  whose weights  $\tau$  are shared across patches, and further manipulates these signals to express a task-specific loss  $\mathcal{L}_{task}$ . The two modules are connected through non-maximum suppression on the scale-space heatmap output of  $\mathcal{H}_\eta$ , followed by a top- $K$  selection process to extract the parameters defining the patches, which we denote as  $\mathcal{E}_K$ . We then sample patches at these locations through (differentiable) bilinear sampling  $\mathcal{S}$  and feed them the task module.

The defining characteristic of the MIST architecture is that they are *quasi-unsupervised*: the only supervision required is the number  $K$  of patches to extract. However, as we show in Section 5.3, our method is not sensitive to the choice of  $K$  during training. In addition, our extraction pro-

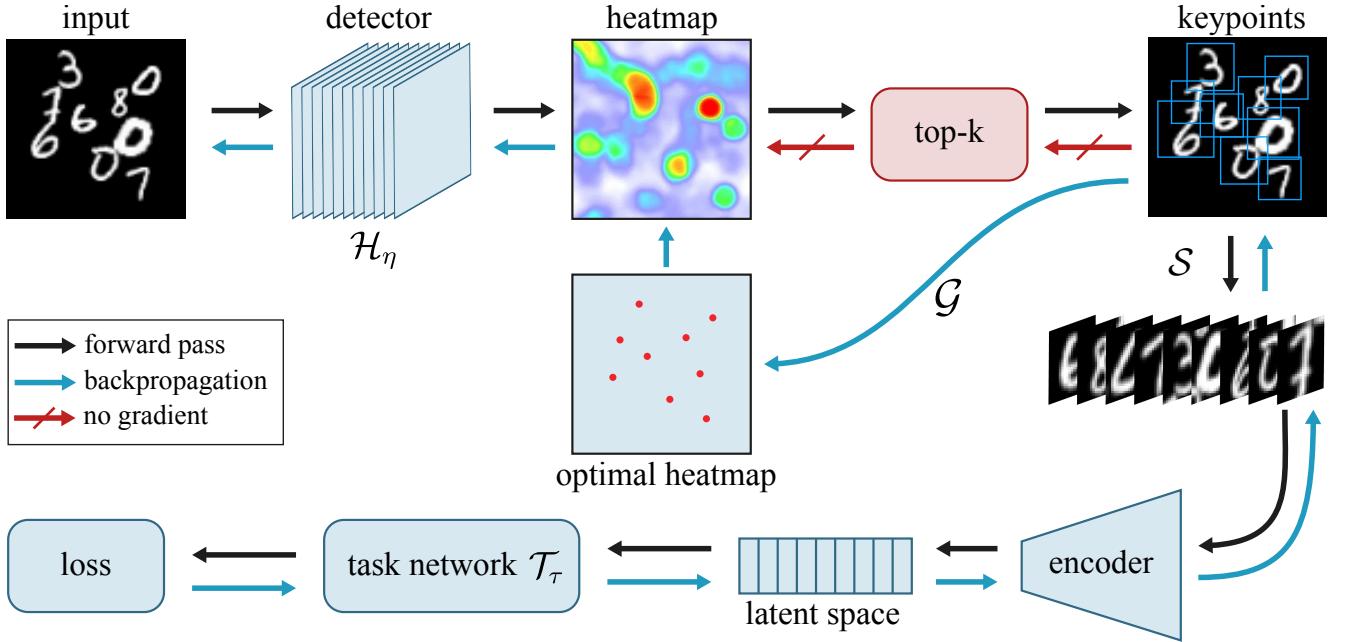


Figure 1. **The MIST architecture** – A network  $\mathcal{H}_\eta$  estimates locations and scales of patches encoded in a heatmap  $\mathbf{h}$ . Patches are then extracted via a sampler  $\mathcal{S}$ , and then fed to a task-specific network  $\mathcal{T}_\tau$ . For example, the specific task could be to re-synthesize the image as a super-position of (unknown and locally supported) basis functions, or simply classifying each patch. As top-k operation is non-differentiable, we back-propagate by lifting through  $\mathcal{G}$ ; see Section 3 for details.

cess uses a single heatmap for all instances that we extract. In contrast, existing heatmap-based methods [10, 57] typically rely on heatmaps dedicated to *each* instance, which is problematic when an image contains two instances of the same class. Conversely, we restrict the role of the heatmap network  $\mathcal{H}_\eta$  to find the “important” areas in a given image, without having to distinguishing between classes, hence simplifying learning.

Formally, the training of the MIST architecture is summarized by:

$$\underset{\tau, \eta}{\text{minimize}} \quad \mathcal{L}_{\text{task}}(\mathcal{T}_\tau(\mathcal{S}(\mathcal{E}_K(\mathcal{H}_\eta(\mathcal{I})))))) \quad (1)$$

where  $\tau, \eta$  are the network trainable parameters. Unfortunately, the patch extractor  $\mathcal{E}_K$  is non-differentiable, as it identifies the locations of the top- $K$  local maxima of a heatmap and then selects the corresponding patches from the input image. Differentiating this operation provides a gradient with respect to the input, but no gradient with respect to the heatmap. Although it is possible to smoothly relax the patch selection operation in the  $K=1$  case [53] (*i.e.*, argmax), it is unclear how this can be generalized to the case of *multiple* distinct local maxima. It is thus impossible to train the patch selector parameters directly by backpropagation. Here, we propose an alternative via lifting.

**Differentiable top-K via lifting.** The introduction of auxiliary variables to simplify the structure of an optimization problem has proven effective in a range of domains ranging from non-rigid registration [46], to robust optimization [54].

To simplify our training optimization, we start by decoupling the heatmap tensor from the optimization (1) by introducing the corresponding auxiliary variables  $\mathbf{h}$ , as well as the patch location variables  $\{\mathbf{x}_k\}$  from the top-K extractor:

$$\begin{aligned} & \underset{\eta, \tau, \mathbf{h}, \{\mathbf{x}_k\}}{\text{minimize}} \quad \mathcal{L}_{\text{task}}(\mathcal{T}_\tau(\mathcal{S}(\{\mathbf{x}_k\}))) \\ & \text{s.t.} \quad \mathbf{h} = \mathcal{H}_\eta(\mathcal{I}), \quad \{\mathbf{x}_k\} = \mathcal{E}_K(\mathbf{h}). \end{aligned} \quad (2)$$

We then relax the first constraint above to a least-squares penalty via a Lagrange multiplier  $\lambda$ :

$$\begin{aligned} & \underset{\eta, \tau, \mathbf{h}, \{\mathbf{x}_k\}}{\text{minimize}} \quad \mathcal{L}_{\text{task}}(\mathcal{T}_\tau(\mathcal{S}(\{\mathbf{x}_k\}))) + \lambda \|\mathbf{h} - \mathcal{H}_\eta(\mathcal{I})\|_2^2 \\ & \text{s.t.} \quad \{\mathbf{x}_k\} = \mathcal{E}_K(\mathbf{h}). \end{aligned} \quad (3)$$

As in many methods that use keypoint supervision to regress heatmaps [5], we assume that a good heatmap generator  $\mathcal{G}$  exists – that is  $\{\mathbf{x}_k\} \approx \mathcal{E}_K(\mathcal{G}(\{\mathbf{x}_k\}))$ . We can now rewrite our optimization as:

$$\begin{aligned} & \underset{\eta, \tau, \mathbf{h}, \{\mathbf{x}_k\}}{\text{minimize}} \quad \mathcal{L}_{\text{task}}(\mathcal{T}_\tau(\mathcal{S}(\{\mathbf{x}_k\}))) + \lambda \|\mathbf{h} - \mathcal{H}_\eta(\mathcal{I})\|_2^2 \\ & \text{s.t.} \quad \mathbf{h} = \mathcal{G}(\{\mathbf{x}_k\}). \end{aligned} \quad (4)$$

We can now drop the auxiliary variable  $\mathbf{h}$  and rewrite our optimization as:

$$\begin{aligned} & \underset{\eta, \tau, \{\mathbf{x}_k\}}{\text{minimize}} \quad \mathcal{L}_{\text{task}}(\mathcal{T}_\tau(\mathcal{S}(\{\mathbf{x}_k\}))) + \\ & \quad \lambda \|\mathcal{G}(\{\mathbf{x}_k\}) - \mathcal{H}_\eta(\mathcal{I})\|_2^2, \end{aligned} \quad (5)$$

---

**Algorithm 1** Multi-stage optimization for MISTS

---

**Require:**  $K$  : number of patches to extract,  $\mathcal{L}_{\text{task}}$  : task specific loss,  $\mathcal{I}$  : input image,  $\mathcal{G}$  : the keypoints to heatmap function,  $\mathcal{H}$  : the heatmap network,  $\eta$  : parameters of the heatmap network,  $\mathcal{T}$  : the task network,  $\tau$  : parameters of the task network,  $\mathcal{E}_K$  : the top- $K$  operator.

```

1: function TRAINMIST( $\mathcal{I}, \mathcal{L}_{\text{task}}$ )
2:   for each training batch do
3:      $\tau \leftarrow$  Optimization step for  $\mathcal{T}_\tau$  with  $\mathcal{L}_{\text{task}} + \lambda \|\{\mathbf{x}_k\} - \mathcal{E}_K(\mathcal{H}_\eta(\mathcal{I}))\|_2^2$ 
4:      $\{\mathbf{x}_k\} \leftarrow$  Optimization step for  $\{\mathbf{x}_k\}$  with  $\mathcal{L}_{\text{task}} + \lambda \|\{\mathbf{x}_k\} - \mathcal{E}_K(\mathcal{H}_\eta(\mathcal{I}))\|_2^2$ 
5:      $\eta \leftarrow$  Optimization step for  $\eta$  with  $\|\mathcal{G}(\{\mathbf{x}_k\}) - \mathcal{H}_\eta(\mathcal{I})\|_2^2$ 
6:   end for
7: end function

```

---

and then approach the problem by *block coordinate descent* – where the energy terms not containing the variable being optimized are safely dropped, and we apply  $\mathcal{E}_K$  to the penalty term of (5):

$$\begin{aligned} \underset{\tau, \{\mathbf{x}_k\}}{\text{minimize}} \quad & \mathcal{L}_{\text{task}}(\mathcal{T}_\tau(\mathcal{S}(\{\mathbf{x}_k\}))) + \\ & \lambda \|\{\mathbf{x}_k\} - \mathcal{E}_K(\mathcal{H}_\eta(\mathcal{I}))\|_2^2, \end{aligned} \quad (6)$$

$$\underset{\eta}{\text{minimize}} \quad \|\mathcal{G}(\{\mathbf{x}_k\}) - \mathcal{H}_\eta(\mathcal{I})\|_2^2. \quad (7)$$

To accelerate training, we further split (6) into two stages, and alternate between optimizing  $\tau$  and  $\{\mathbf{x}_k\}$ . Being based on block coordinate descent, this process converges smoothly, as we show in Section 5.3. The summary for the three stage optimization procedure is outlined in Algorithm 1. Notice that we are not introducing *any* additional supervision signal that is tangent to the given task.

## 4. Implementation

### 4.1. Components

**Multiscale heatmap network –  $\mathcal{H}_\eta$ .** Any network that provides localization via a heatmap can be used. Our standard implementation is a multiscale heatmap network inspired by LF-Net [33]. We improve the network by applying modifications on how the scores are aggregated over scale. The network takes as input an image  $\mathcal{I}$ , and outputs multiple heatmaps  $\mathbf{h}'_s$  of the same size for each scale level  $s$ . To limit the number of necessary scales, we use a discrete scale space with  $S$  scales, and resolve intermediate scales via interpolation. For tasks where pre-trained networks are needed – *e.g.* classification on a natural image – we utilize an architecture similar to [17]. Note that we are simply using the pretrained deep features, and these can also be obtained through a fully unsupervised setup, such as contrastive learning [7], if desired; see Section A of the appendix for details.

**Top-K patch selection –  $\mathcal{E}_K$ .** To extract the top  $K$  elements, we perform an addition cleanup through a standard

non-maximum suppression. We then find the spatial locations of the top  $K$  elements of this suppressed heatmap, denoting the spatial location of the  $k^{\text{th}}$  element as  $(x_k, y_k)$ , which now should correspond to local maxima. When using multiscale heatmaps, for each location, we also compute the corresponding scale by weighted first order moments [43] where the weights are the responses in the corresponding heatmaps, *i.e.*  $s_k = \sum_s \mathbf{h}'_s(x_k, y_k)s$

**Generative model for ideal heatmap –  $\mathcal{G}(\{\mathbf{x}_k\})$ .** For the generative model that turns keypoint and patch locations into heatmaps, we apply a simple model where the heatmap is zero everywhere except at the corresponding keypoint locations (patch centers); see Section B of the appendix.

**Patch resampling –  $\mathcal{S}$ .** As a patch is uniquely parameterized by its location and scale, *i.e.*  $\mathbf{x}_k = (x_k, y_k, s_k)$ , we can then proceed to resample its corresponding tensor via bilinear interpolation [22, 23] as  $\{\mathbf{P}_k\} = \mathcal{S}(\mathcal{I}, \{\mathbf{x}_k\})$ .

### 4.2. Task-specific networks

We now introduce two applications of the MIST framework. We use the *same* heatmap network and patch extractor for both applications, but the task-specific network and loss differ. We provide further details regarding the task-specific network architectures in Section C.1 of the appendix.

**Image reconstruction / auto-encoding.** As illustrated in Figure 1, for image reconstruction we append our patch extraction network with a *shared* auto-encoder for each extracted patch. We can then train this network to *reconstruct* the original image by inverting the patch extraction process and minimizing the mean squared error between the input and the reconstructed image. Overall, the network is designed to *jointly* model and localize repeating structures in the input signal. Specifically, we introduce the generalized inverse sampling operation  $\mathcal{S}^{-1}(\mathbf{P}_i, \mathbf{x}_i)$ , which starts with an image of all zeros, and places the patch  $\mathbf{P}_i$  at  $\mathbf{x}_i$ . We then sum all the images together to obtain the reconstructed image, optimizing the task loss

$$\mathcal{L}_{\text{task}} = \left\| \mathcal{I} - \sum_i \mathcal{S}^{-1}(\mathbf{P}_i, \mathbf{x}_i) \right\|_2^2. \quad (8)$$

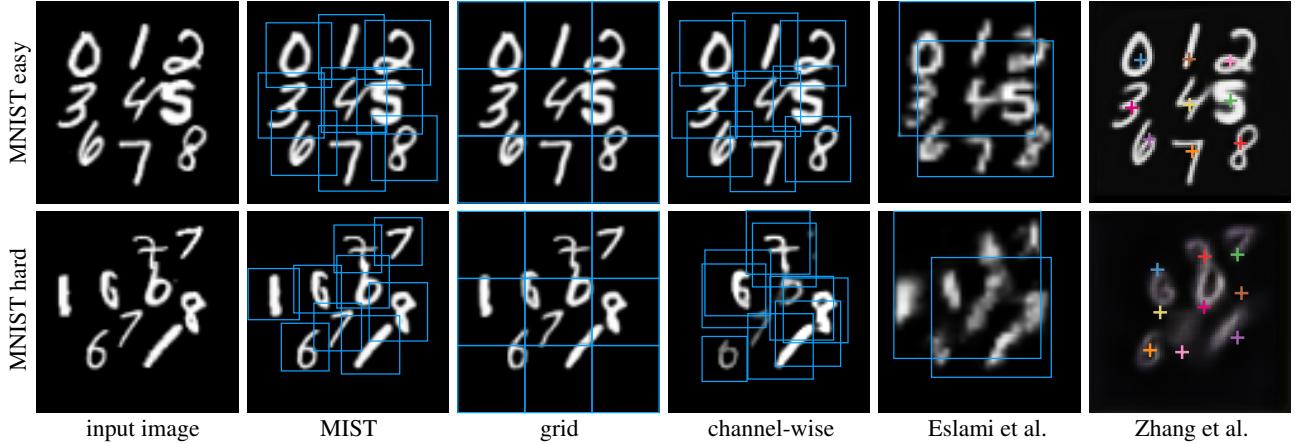


Figure 2. MNIST character synthesis examples for (top) the “easy” single instance setup and (bottom) the hard multi-instance setup. We compare the output of MISTS to grid, channel-wise, sequential Eslami *et al.* [10] and Zhang *et al.* [57].

**Multiple instance classification.** By appending a classification network to the patch extraction module, we can also perform multiple instance learning. For each extracted patch  $\mathbf{P}_k$  we apply a shared classifier network to output  $\hat{\mathbf{y}}_k \in \mathbb{R}^C$ , where  $C$  is the number of classes. In turn, these are then converted into probability estimates by the transformation  $\hat{\mathbf{p}}_k = \text{softmax}(\hat{\mathbf{y}}_k)$ .

With  $\mathbf{y}_l$  being the one-hot ground-truth labels of instance  $l$  with unit norm, we define the multi-instance classification loss as

$$\mathcal{L}_{\text{task}} = \left\| \frac{1}{L} \sum_{l=1}^L \mathbf{y}_l - \frac{1}{K} \sum_{k=1}^K \hat{\mathbf{p}}_k \right\|_2^2, \quad (9)$$

where  $L$  is the number of instances in the image. We empirically found this loss to be more stable compared to the cross-entropy loss, similar to [30]. Note that we *do not* provide supervision about the localization of instances, yet the detector network will automatically learn how to localize with minimal supervision (*i.e.*, the number of instances).

## 5. Results and evaluation

To demonstrate the effectiveness of our framework we apply MIST to two different tasks. We first perform a quasi-unsupervised image reconstruction task, where *only* the total number of instances in the scene is provided, *i.e.*,  $K$  is defined. We then show that MIST can also be applied to weakly supervised multi-instance classification, where only image-level supervision is provided. Note that, unlike region proposal based methods, our method relies only on the cues from the classifier, do not require object proposals, and can be trained from scratch.

### 5.1. Image reconstruction

From the MNIST dataset, we derive two different scenarios. In the *MNIST easy* dataset, we consider a simple

setup where the *sorted* digits are confined to a perturbed *grid* layout; see Figure 2 (top). Specifically, we perturb the digits with a Gaussian noise centered at each grid center, with a standard deviation that is equal to one-eighths of the grid width/height. In the *MNIST hard* dataset, the positions are randomized through a Poisson distribution [4], as is the identity, and cardinality of each digit. We allow multiple instances of the same digit to appear in this variant. For both datasets, we construct both training and test sets, and the test set is never seen during training.

**Comparison baselines.** We compare our method against four baselines ① the *grid* method divides the image into a  $3 \times 3$  grid and applies the same auto-encoder architecture as MIST to each grid location to reconstruct the input image; ② the *channel-wise* method uses the same auto-encoder network as MIST, but we modify the heatmap network to produce  $K$  channels as output, where each channel is dedicated to an interest point. Locations are obtained through a channel-wise soft-argmax as in [57]; ③ *Esl16* [10] is a sequential generative model; ④ *Zha18* [57] is a heatmap-based method with channel-wise strategy for unsupervised learning of landmarks. We do not compare against Xie20 [52] for the reconstruction tasks as it requires too much memory to be used for generative tasks; see Section C.2 of the appendix for more details.

**Results for “MNIST easy”.** As shown in Figure 2 (top) all methods successfully re-synthesize the image, with the exception of Eslami *et al.* [10]. As this method is sequential, with nine digits the sequential implementation simply becomes too difficult to optimize through. Note how this method only learns to describe the scene with a few large regions. We summarize quantitative results in Table 1.

**Results for “MNIST hard”.** As shown in Figure 2 (bottom), all baseline methods failed to properly represent the image. Only MIST succeeded at both localizing digits and reconstructing the original image. Although the grid method

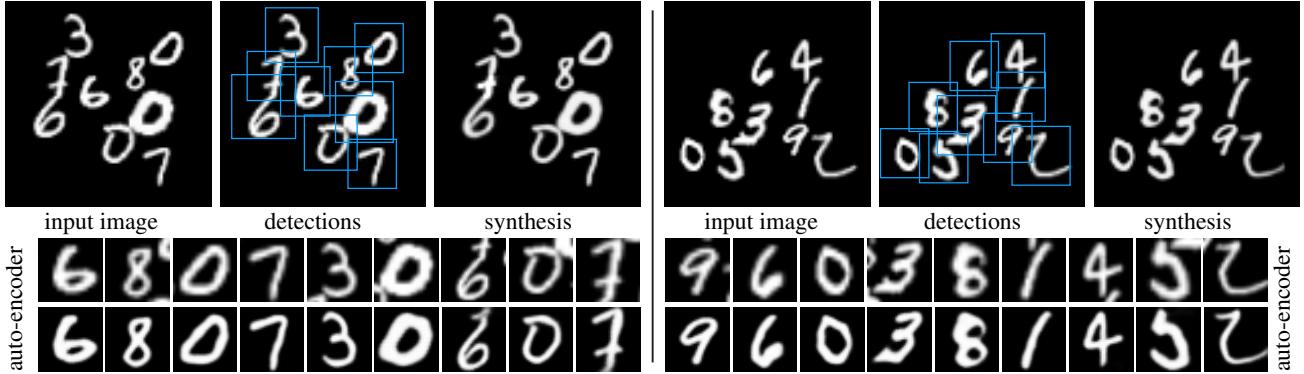


Figure 3. Two auto-encoding examples learnt from MNIST-hard. In the top row, for each example we visualize input, patch detections, and synthesis. In the bottom row we visualize each of the extracted patch, and how it is modified by the learnt auto-encoder.

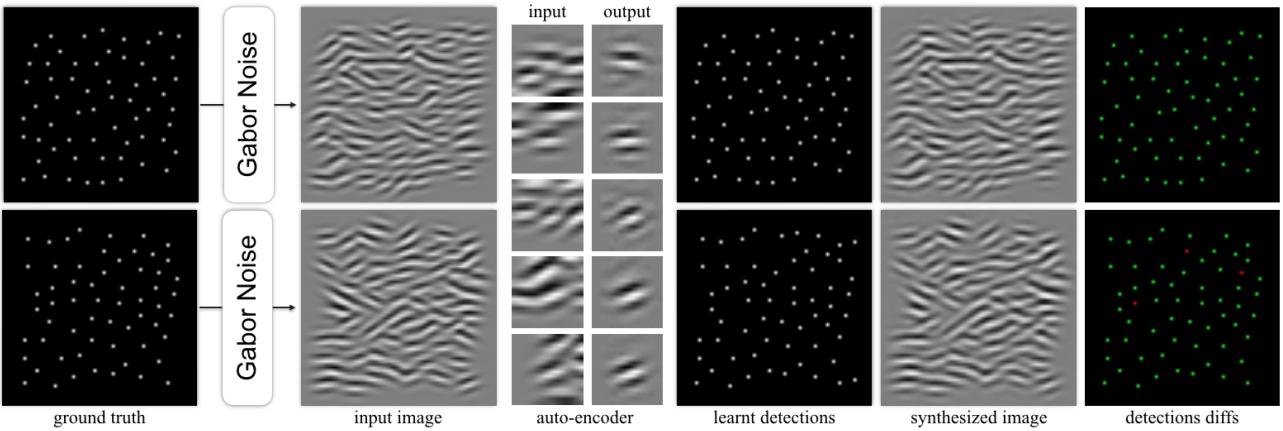


Figure 4. Inverse rendering of Gabor noise; we annotate **correct** / **erroneous** localizations.

	MIST	Grid	Ch.-wise	Esl16 [10]	Zha18 [57]
MNIST easy	.038	.039	.042	.100	.169
MNIST hard	.053	.062	.128	.154	.191
Gabor	.095	-	-	-	-

Table 1. Reconstruction error (root mean square error). Note that Grid *does not* learn any notion of digits.

accurately reconstructs the image, it has no concept of individual digits. Conversely, as shown in Figure 3, our method generates accurate bounding boxes for digits even when these digits overlap, and does so without any location supervision. For quantitative results, please see Table 1.

**Finding the basis of a procedural texture.** We further demonstrate that our methods can be used to find the basis function of a procedural texture. For this experiment we synthesize textures with procedural Gabor noise [26]. Gabor noise is obtained by convolving oriented Gabor wavelets with a Poisson impulse process. Hence, given exemplars of noise, our framework is tasked to regress the underlying impulse process and reconstruct the Gabor kernels so that when the two are convolved, we can reconstruct the original image. Figure 4 illustrates the results of our experiment. The auto-encoder learned to accurately reconstruct the Gabor

kernels, even though in the training images they are heavily overlapped. These results show that MIST is capable of generating and reconstructing large numbers of instances per image, which is *intractable* with other approaches.

## 5.2. Multiple instance classification

**Multi-MNIST – Figure 5, Table 2, and Table 3.** To test our method in a multiple instance classification setup, we rely on the *MNIST hard* dataset. To evaluate the detection accuracy of the models, we compute the intersection over union (IoU) between the ground-truth bounding box and the detection results, and assign it as a match if the IoU score is over 50%. In Table 2, we compare our method to *channel-wise* and report the number of correctly classified matches, as well as the proportion of instances that are both correctly detected and correctly classified. To provide a sense of the upper-bound, we also provide results for the case when location supervision is used. Our method clearly outperforms the *channel-wise* strategy. Note that, even without localization supervision, our method correctly localizes digits. Conversely, the *channel-wise* strategy fails to learn. This is because *multiple instances* of the same digits are present in the image. For example, in the Figure 5 (right), we have

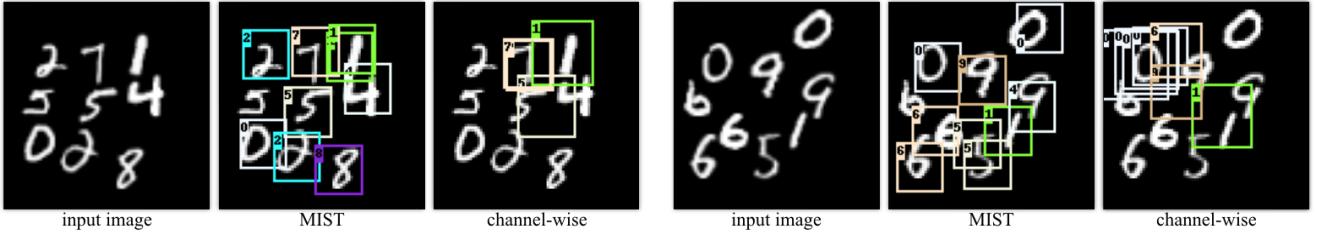


Figure 5. Two qualitative examples for detection and classification on our *MNIST-hard* dataset.

	MIST	Ch.-wise	Supervised
IOU 50%	97.8%	25.4%	99.6%
Classif.	98.8%	75.5%	98.7%
Both	<b>97.5%</b>	24.8%	98.6%

Table 2. Instance detection and classification on *MNIST-hard*.

three sixes, two zeros, and two nines. This prevents any of these digits from being detected/classified properly by a channel-wise approach.

We further compare our method to Xie20 [52], a state-of-the-art differentiable top-K formulation in Table 3. We test with the *MNIST hard* dataset, and also an easier version of this dataset with only three digits. For Xie20 [52], we modify the heatmap network to accommodate for the fact that Xie20 requires extracting *all* potential patches, thus needing extensive memory, to be used in a detection framework (Xie20 generates a differentiable top K “mask”, not a selection, which has then to be multiplied to the input to finally perform selection). Hence, we apply an architecture similar to the region proposal network [17], which we found to work better than training a heatmap network that downsamples the heatmap to a reasonable size; see Supplementary Section C.1 for details. Our method works best for both experiments in Table 3, with the gap between MIST and Xie20 widening significantly as the problem becomes more complicated.

**PASCAL + COCO – Table 4 and Figure 6.** To further test MIST in a more generalized setting, we apply MIST on PASCAL VOC 2007 dataset [11] and COCO dataset [28]. We combine the two datasets together by taking images from both datasets that contain any of the twenty Pascal VOC 2007 classes. As we are aiming for an ambitious goal of not having any location supervision at all, including the bounding box proposals used in state-of-the-art weakly-supervised works [3, 44, 27, 51], we apply additional treatments to make the task easier. We filter the combined dataset by removing images that contain objects that are too large (covering more than 30% of the image) and images with objects that are too wide or narrow (aspect ratio bigger than 2.0 or smaller than 0.5). The remaining images are random cropped to ensure that all objects within the image cover 30% of the image in average. The final dataset has 9780 images in total, which

	<i>MNIST-3 digits</i>		<i>MNIST-hard</i>	
	MIST	Xie20 [52]	MIST	Xie20 [52]
IOU 50%	99.5%	95.4%	97.8%	72.7%
Classif.	96.9%	97.4%	98.8%	93.1%
Both	<b>96.6%</b>	92.2%	<b>97.5%</b>	71.3%

Table 3. Instance level detection and classification performance comparison with a state-of-the-art differentiable top K formulation [52].

we split into training (7816 images), validation (987 images), and test (977 images) sets.

As the task is hard, we further restrict the role of MIST and Xie20 [52] to *localization* only without scale, *i.e.*, we use a single scale heatmap for localization. We use  $K=2$  for this dataset. The number of objects vary from one image to another in this dataset, thus, on top of the 20 classes in Pascal VOC 2007, we also add an additional class ‘background’ to indicate unused bounding boxes. With this setup,  $K=2$  serves as the upper bound for number of instances to attend to in an image.

Since the dataset is relatively limited, as mentioned earlier in Section 4.1, we use a pretrained ResNet34 [19] network. We use the feature maps of the fourth convolutional block and append a  $1 \times 1$  convolution layer on top to obtain the heatmap. We also resample patches from this heatmap instead of the original image to take advantage of these pretrained features. We further apply data augmentation by performing random horizontal flips, as well as randomly masking the input image with a square mask of uniform distributed random size (between  $0.15 \times$  and  $0.6 \times$  of the image size) and filling the inside of the mask with the average color within the mask. We emphasize that even with such filtering and restriction, our method is, to the best of our knowledge, the *first* to be able to learn to detect and classify without any location supervision – existing weakly supervised methods [3, 44, 27, 51] mainly rely on provided object proposals. Extending MIST to be able to deal with scale and aspect-ratio for natural images is left as future work.

We evaluate the performance of each method by computing the accuracy, recall, and F1-score of the detection outcomes. Regarding the localization, as we estimate the

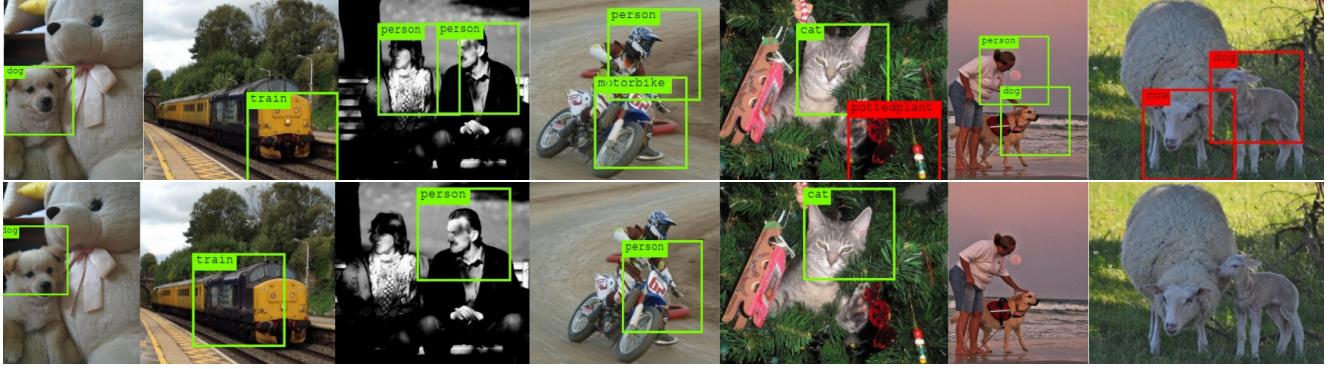


Figure 6. Qualitative results on PASCAL+COCO dataset. (Top row) we show detection results when using MIST and (bottom row) when using Xie20 [52]. Correctly localized and classified ones are shown in green, and wrong ones in red.

	Precision	Recall	F1-Score
MIST	<b>87.4%</b>	<b>55.5%</b>	<b>67.9%</b>
Xie20 [52]	72.7%	51.0%	60.0%

Table 4. Instance level localization and classification results on PASCAL+COCO dataset.

Instances during train	AP <sup>IoU=.50</sup>
{9}	<b>92.2%</b>
{6, 7, 8, 9}	90.1%
{3, 4, 5, <b>6</b> , 7, 8, 9}	90.8%

Table 5. Sensitivity experiment of  $K$  on the MNIST hard dataset. We mark the number of  $K$  used for testing in bold.

center only (scale is fixed), we consider the detection to be correct if the detection center is within the ground-truth bounding box. We report results for MIST and Xie20 [52] in Table 4, and show qualitative highlights in Figure 6. Our method outperforms Xie20 [52] by more than **13% relative**, in terms of the F1-Score, for this task.

### 5.3. Additional results

**Convergence during training.** As is typical for neural network training, our objective is non-convex and there is no guarantee that a local minimum found by gradient descent training is a global minimum. Empirically, however, the optimization process is stable, as shown in Figure 7. We report both the training losses  $\mathcal{L}_{\text{task}}$  and  $\|\mathcal{G}(\{\mathbf{x}_k\}) - \mathcal{H}_\eta(\mathcal{I})\|_2^2$  when training for *MNIST hard*. Both losses converge smoothly, demonstrating the stability of our formulation. This is unsurprising, as our formulation is a lifted version of this loss to allow gradient-based training. Note that the initial jump in Figure 7 (b) is caused by the weak response of the randomly initialized detector’s weights, producing a heatmap close to zero for every pixel.

**Sensitivity to  $K$  – Table 5.** To investigate the sensitivity to

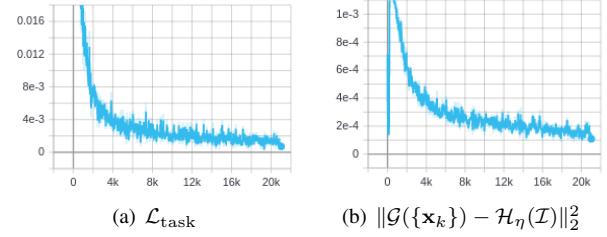


Figure 7. Evolution of the loss and heatmap during training on *MNIST hard*. (a) the task (classification) loss  $\mathcal{L}_{\text{task}}$  and (b) the heatmap related loss  $\|\mathcal{G}(\{\mathbf{x}_k\}) - \mathcal{H}_\eta(\mathcal{I})\|_2^2$  for each iteration.

the correctness of  $K$  during training, we train with varying number of instances and test with ground-truth  $K$ . For example, with  $K = 6$ , where the ground truth could be anything within  $\{3, 4, 5, 6, 7, 8, 9\}$  – we mark the  $K$  used during training with bold. Our method still is able to give accurate result with inaccurate  $K$  during training. Knowing the exact number of objects is not a strict requirement at test time, as our detector generates a heatmap for the entire image regardless of the  $K$  it was trained with. Note that while in theory sequential methods are free from this limitation, in practice they are able to deal with limited number of objects (e.g. up to three) due to their recurrent nature.

## 6. Conclusion

We have introduced the MIST framework for multi-instance image reconstruction/classification. With MIST, we show how localization of multiple instances can be learned even with the non-differentiable top-K operation by lifting. We demonstrated the effectiveness of MIST by showing its compelling performance in both reconstruction and classification tasks using synthetic data. We further demonstrated its performance in learning to localize and classify multiple objects in real-world images, without any supervision on location, and without any help from object proposal methods. MISTS are a first step towards the definition of optimizable

image decomposition networks that could be extended to a number of exciting *unsupervised* learning tasks.

## Acknowledgements

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grant, NSERC Collaborative Research and Development Grant, Google, and by Compute Canada. We would also like to thank Shahram Izadi for his great support for this project.

## References

- [1] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple Object Recognition With Visual Attention. In *International Conference on Learning Representations*, 2015. [1](#), [2](#)
- [2] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded Up Robust Features. *Computer Vision and Image Understanding*, 10(3):346–359, 2008. [1](#)
- [3] Hakan Bilen and Andrea Vedaldi. Weakly Supervised Deep Detection Networks. In *Conference on Computer Vision and Pattern Recognition*, 2016. [2](#), [7](#)
- [4] Robert Bridson. Fast Poisson Disk Sampling in Arbitrary Dimensions. In *SIGGRAPH sketches*, 2007. [5](#)
- [5] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1812.08008*, 2018. [3](#)
- [6] Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A. Ross, Jia Deng, and Rahul Sukthankar. Rethinking the Faster R-CNN Architecture for Temporal Action Localization. In *Conference on Computer Vision and Pattern Recognition*, 2018. [2](#)
- [7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In *International Conference on Machine Learning*, 2020. [4](#)
- [8] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-Supervised Interest Point Detection and Description. *CVPR Workshop on Deep Learning for Visual SLAM*, 2018. [1](#), [2](#)
- [9] Piotr Dollar, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian Detection: An Evaluation of the State of the Art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):743–761, 2012. [1](#)
- [10] S. M. Ali Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Koray Kavukcuoglu, and Geoffrey E. Hinton. Attend, Infer, Repeat: Fast Scene Understanding with Generative Models. In *Advances in Neural Information Processing Systems*, 2015. [1](#), [2](#), [3](#), [5](#), [6](#), [13](#)
- [11] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (Voc) Challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. [7](#)
- [12] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. Object Detection with Discriminatively Trained Part Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010. [1](#)
- [13] Mingfei Gao, Ang Li, Ruichi Yu, Vlad I. Morariu, and Larry S. Davis. C-WSL: Coung-guided Weakly Supervised Localization. In *European Conference on Computer Vision*, 2018. [2](#)
- [14] Georgios Georgakis, Srikrishna Karanam, Ziyang Wu, Jan Ernst, and Jana Kosecka. End-to-end Learning of Keypoint Detector and Descriptor for Pose Invariant 3D Matching. In *Conference on Computer Vision and Pattern Recognition*, 2018. [2](#)
- [15] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. DRAW: A Recurrent Neural Network For Image Generation. In *International Conference on Machine Learning*, 2015. [1](#), [2](#)
- [16] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Inductive Visual Localization: Factorised Training for Superior Generalization. In *British Machine Vision Conference*, 2018. [2](#)
- [17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *International Conference on Computer Vision*, 2017. [1](#), [2](#), [4](#), [7](#)
- [18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on Imagenet Classification. In *International Conference on Computer Vision*, 2015. [12](#), [13](#)
- [19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. [7](#)
- [20] Maximilian Ilse, Jakub M Tomczak, and Max Welling. Attention-based Deep Multiple Instance Learning. In *International Conference on Machine Learning*, 2018. [2](#)
- [21] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. Cross-Domain Weakly-Supervised Object Detection through Progressive Domain Adaptation. In *European Conference on Computer Vision*, 2018. [2](#)
- [22] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial Transformer Networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015. [4](#)
- [23] Wei Jiang, Weiwei Sun, Andrea Tagliasacchi, Eduard Trulls, and Kwang Moo Yi. Linearized Multi-Sampling for Differentiable Image Transformation. *arXiv Preprint*, 2019. [4](#)
- [24] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image Generation from Scene Graphs. In *Conference on Computer Vision and Pattern Recognition*, 2018. [2](#)
- [25] Justin Johnson, Andrej Karpathy, and Li Fei-fei. Densecap: Fully Convolutional Localization Networks for Dense Captioning. In *Conference on Computer Vision and Pattern Recognition*, 2016. [2](#)
- [26] Ares Lagae, Sylvain Lefebvre, George Drettakis, and Philip Dutré. Procedural noise using sparse gabor convolution. *ACM Transactions on Graphics*, 2009. [6](#)
- [27] Xiaoyan Li, Meina Kan, Shiguang Shan, and Xilin Chen. Weakly Supervised Object Detection With Segmentation Collaboration. In *Conference on Computer Vision and Pattern Recognition*, 2019. [2](#), [7](#)
- [28] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan,

- C. Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision*, 2014. 7
- [29] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 20(2), 2004. 1
- [30] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, Zhen Wang, and Stephen Paul Smolley. Least Squares Aenerative Adversarial Networks. In *Conference on Computer Vision and Pattern Recognition*, 2017. 5
- [31] Daniel Merget, Matthias Rock, and Gerhard Rigoll. Robust Facial Landmark Detection via a Fully-Convolutional Local-Global Context Network. In *Conference on Computer Vision and Pattern Recognition*, 2018. 2
- [32] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked Hourglass Networks for Human Pose Estimation. In *European Conference on Computer Vision*, 2016. 2
- [33] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. Lf-Net: Learning Local Features from Images. In *Advances in Neural Information Processing Systems*, 2018. 1, 2, 4, 12
- [34] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Is object localization for free? – weakly-supervised learning with convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition*, 2015. 2
- [35] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *Conference on Computer Vision and Pattern Recognition*, 2016. 1
- [36] Joseph Redmon and Ali Farhadi. YOLO 9000: Better, Faster, Stronger. In *Conference on Computer Vision and Pattern Recognition*, 2017. 1, 2
- [37] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems*, 2015. 1, 2
- [38] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. In *International Conference on Computer Vision*, 2017. 2
- [39] Yunhan Shen, Rongrong Ji, Shengchuan Zhang, Wangmeng Zuo, and Yan Wang. Generative adversarial learning towards fast weakly supervised detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5764–5773, 2018. 2
- [40] Bharat Singh, Hengduo Li, Abhishek Sharma, and Larry S. Davis. R-FCN-3000 at 30fps: Decoupling Detection and Classification. In *Conference on Computer Vision and Pattern Recognition*, 2018. 2
- [41] Russell Stewart and Mykhaylo Andriluka. End-to-End People Detection in Crowded Scenes. In *Conference on Computer Vision and Pattern Recognition*, 2016. 1
- [42] Ming Sun, Yuchen Yuan, Feng Zhou, and Errui Ding. Multi-Attention Multi-Class Constraint for Fine-grained Image Recognition. In *European Conference on Computer Vision*, 2018. 2
- [43] Supasorn Suwajanakorn, Noah Snavely, Jonathan Tompson, and Mohammad Norouzi. Discovery of Latent 3D Keypoints via End-To-End Geometric Reasoning. In *Advances in Neural Information Processing Systems*, 2018. 4
- [44] Peng Tang, Xinggang Wang, Xiang Bai, and Wenyu Liu. Multiple Instance Detection Network with Online Instance Classifier Refinement. In *Conference on Computer Vision and Pattern Recognition*, 2017. 7
- [45] Peng Tang, Xinggang Wang, Angtian Wang, Yongluan Yan, Wenyu Liu, Junzhou Huang, and Alan Yuille. Weakly Supervised Region Proposal Network and Object Detection. In *European Conference on Computer Vision*, 2018. 2
- [46] Jonathan Taylor, Lucas Bordeaux, Thomas Cashman, Bob Corish, Cem Keskin, Eduardo Soto, David Sweeney, Julien Valentin, Benjamin Luff, Arran Topalian, Erroll Wood, Sameh Khamis, Pushmeet Kohli, Toby Sharp, Shahram Izadi, Richard Banks, Andrew Fitzgibbon, and Jamie Shotton. Efficient and precise interactive hand tracking through joint, continuous optimization of pose and correspondences. *ACM Transactions on Graphics*, 2016. 3
- [47] Bugra Tekin, Pablo Márquez-Neila, Mathieu Salzmann, and Pascal Fua. Learning to Fuse 2D and 3D Image Cues for Monocular Body Pose Estimation. In *International Conference on Computer Vision*, 2017. 2
- [48] Jasper Uijlings, Stefan Popov, and Vittorio Ferrari. Revisiting Knowledge Transfer for Training Object Class Detectors. In *Conference on Computer Vision and Pattern Recognition*, 2018. 2
- [49] Fang Wan, Pengxu Wei, Jianbin Jiao, Zhenjun Han, and Qixiang Ye. Min-Entropy Latent Model for Weakly Supervised Object Detection. In *Conference on Computer Vision and Pattern Recognition*, 2018. 2
- [50] Fangfang Wang, Liming Zhao, Xi Li, Xinchao Wang, and Dacheng Tao. Geometry-Aware Scene Text Detection with Instance Transformation Network. In *Conference on Computer Vision and Pattern Recognition*, 2018. 2
- [51] Yunchao Wei, Zhiqiang Shen, Bowen Cheng, Honghui Shi, Jinjun Xiong, Jiashi Feng, and Thomas Huang. TS2C: Tight Box Mining with Surrounding Segmentation Context for Weakly Supervised Object Detection. In *European Conference on Computer Vision*, 2018. 7
- [52] Yujia Xie, Hanjun Dai, Minshuo Chen, Bo Dai, Tuo Zhao, Hongyuan Zha, Wei Wei, and Tomas Pfister. Differentiable top-k operator with optimal transport. In *NeurIPS*, 2020. 2, 5, 7, 8, 13
- [53] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. LIFT: Learned Invariant Feature Transform. In *European Conference on Computer Vision*, 2016. 1, 2, 3
- [54] Christopher Zach and Guillaume Bourmaud. Descending, Lifting or Smoothing: Secrets of Robust Cost Optimization. In *European Conference on Computer Vision*, 2018. 3
- [55] Shanshan Zhang, Jian Yang, and Bernt Schiele. Occluded Pedestrian Detection Through Guided Attention in CNNs. In *Conference on Computer Vision and Pattern Recognition*, 2018. 1, 2
- [56] Xiaolin Zhang, Yunchao Wei, Guoliang Kang, Yi Yang, and Thomas Huang. Self-produced Guidance for Weakly-

- supervised Object Localization. In *European Conference on Computer Vision*, 2018. [2](#)
- [57] Yuting Zhang, Yijie Guo, Yixin Jin, Yijun Luo, Zhiyuan He, and Honglak Lee. Unsupervised Discovery of Object Landmarks as Structural Representations. In *Conference on Computer Vision and Pattern Recognition*, 2018. [1](#), [2](#), [3](#), [5](#), [6](#), [13](#)
- [58] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning Deep Features for Discriminative Localization. In *Conference on Computer Vision and Pattern Recognition*, 2016. [2](#)

# MIST: Multiple Instance Spatial Transformer

## Supplementary Material

### A. Heatmap network architectures

**Standard architecture.** Our standard architecture is the multiscale heatmap network inspired by LF-Net [33]. We employ a fully convolutional network to produce a single heatmap for each scale indexed by  $s = 1 \dots S$ , on the input image  $\mathcal{I}$ . Specifically, for each scale we first scale the image proportionally to the inverse of the scale producing  $\mathcal{I}_s$ , execute the network  $\mathcal{H}_\eta$  on it, and finally scale the heatmap back to the original resolution. This strategy ensures that the network cannot implicitly favor a particular scale and produces scale-independent responses.

This process generates a multiscale heatmap tensor  $\mathbf{h} = \{\mathbf{h}_s\}$  of size  $H \times W \times S$  where  $\mathbf{h}_s = \mathcal{H}_\eta(\mathcal{I}_s)$ , and  $H$  is the height of the image and  $W$  is the width. For the convolutional network we use 4 ResNet blocks [18], where each block is composed of two  $3 \times 3$  convolutions with 32 channels and relu activations without any downsampling. We then perform a *local spatial softmax* operator [33] with spatial extent of  $15 \times 15$  to sharpen the responses. Then we further relate the scores across different scales by performing a “softmax pooling” operation over scale. Specifically, if we denote the heatmap tensor after local spatial softmax as  $\tilde{\mathbf{h}} = \{\tilde{\mathbf{h}}_s\}$ , since after the local spatial softmax  $\mathcal{H}_\eta(\mathcal{I}_s)$  is already an “exponentiated” signal, we do a weighted normalization without an exponential, *i.e.*  $\mathbf{h}' = \sum_s \tilde{\mathbf{h}}_s (\tilde{\mathbf{h}}_s / \sum_s (\tilde{\mathbf{h}}_s + \epsilon))$ , where  $\epsilon = 10^{-6}$  is added to prevent division by zero.

Note that differently from LF-Net [33], we do not perform a softmax along the scale dimension. The scale-wise softmax in LF-Net is problematic as the computation for a softmax function relies on the input to the softmax being *unbounded*. For example, in order for the softmax function to behave as a max function, due to exponentiation, it is necessary that one of the input value reaches infinity (*i.e.* the value that will correspond to the max), or that all other values to reach negative infinity. However, at the network stage where softmax is applied in [33], the score range from zero to one, effectively making the softmax behave similarly to averaging. Our formulation does not suffer from this drawback.

**Backbone-based heatmap network.** For experiments on natural images, we restrict the detector to only localize the objects without estimating object scales to simplify the task. Therefore, we only use a single-scale heatmap for this setting. Also, because the number of images in our dataset is limited, we leverage a pretrained ResNet34 [18] as the backbone feature extractor. Specifically, we resize the input images to have a shorter edge of 224 pixels and use the output of fourth convolution block – a tensor of  $H/16 \times W/16 \times 256$  where

$H$  and  $W$  are the height and the width of the input image, respectively, and 256 is the number of channels. We further append a  $1 \times 1$  conv layers to reduce the heatmap to have 3 channels: response,  $x$ -offset, and  $y$ -offset. We make use of offsets since our heatmap size is only  $1/16$  of the input image size, and integer pixel coordinates on the heatmap cannot provide accurate localization on the input image. We also do not use a *local spatial softmax* operator for this setting due to small heatmap size. While not using spatial softmax makes our heatmaps similar to the ones in [A3], we note that we still rely on top-K, rather than the sampling-without-replacement approach of [A3], and is therefore easy to expand to various K.

### B. Generative model for the heatmap

**Standard architecture.** To convert optimized locations into ideal heatmaps (see Section 4.1), as our standard architecture we apply a simple model where the heatmap is zero everywhere else except on the corresponding keypoint locations (patch center). However, as the optimized patch parameters can be floats corresponding to subpixel locations, we need to quantize them with care to turn them back into a heatmap. For the spatial locations we simply round to the nearest pixel, which at most creates a quantization error of half a pixel, which does not cause problems in practice. For scale however, simple nearest-neighbor assignment causes too much quantization error as our scale-space is sparsely sampled. We therefore assign values to the two nearest neighboring scales in a way that the center of mass would be the optimized scale value, making sure  $\{\mathbf{x}_k\} = \mathcal{E}_K(\mathcal{G}(\{\mathbf{x}_k\}))$ .

**Backbone-based heatmap network.** For the backbone-based network, as the feature map is very coarse, we found using a single pixel insufficient. Hence, we use a Gaussian kernel to reconstruct the first channel (response channel) of the ideal heatmap from the optimized keypoints:

$$H_{\mathbf{p}}^r = \sum_{i=1}^k \exp\left(-\frac{1}{2}(\lfloor \mathbf{x}_i \rfloor - \mathbf{p})^T \Sigma^{-1} (\lfloor \mathbf{x}_i \rfloor - \mathbf{p})\right),$$

where  $\lfloor \mathbf{x}_i \rfloor$  are the rounded optimized keypoints,  $\mathbf{p}$  is a pixel location on heatmap. We use one eighth of the patch size as the values of the diagonal covariance matrix  $\Sigma$ .

For the  $x$  and  $y$  offset channels we only supervise pixels which contain optimized keypoints:

$$H_{\mathbf{p}}^{xy} = \mathbf{x}_i - \lfloor \mathbf{x}_i \rfloor \quad \text{if } \mathbf{p} = \lfloor \mathbf{x}_i \rfloor_i \in \mathbf{x}. \quad (10)$$

To train the detector, we use the  $\ell_2$  loss for response channel as in Equation 7, and for the offset channels we use a Huber

loss [A2] as in [A1]:

$$\text{loss} = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}. \quad (11)$$

## C. Additional implementation details

### C.1. Task-specific networks

**MIST auto-encoder network.** The input layer of the autoencoder is  $32 \times 32 \times C$  where  $C$  is the number of color channels. We use five up/down-sampling levels. Each level is made of three standard non-bottleneck ResNet v1 blocks [18] and each ResNet block uses a number of channels that doubles after each downsampling step. ResNet blocks use  $3 \times 3$  convolutions of stride 1 with ReLU activation. For downsampling we use 2D max pooling with  $2 \times 2$  stride and kernel. For upsampling we use 2D transposed convolutions with  $2 \times 2$  stride and kernel. The output layer uses a sigmoid function, and we use layer normalization before each convolution layer.

**MIST classification network for MNIST dataset.** We use the same architecture as the encoder part of the auto-encoder and append a dense layer to it to map the latent space to the score vector of our 10 digit classes.

**MIST classification network for PASCAL+COCO dataset.** We crop patches at keypoint locations on the *feature map* from fourth convolution block and feed the patches into a single ResNet block – same as the one used for the auto-encoder network – followed by a global average pooling layer, and a dense layer that converts the output into 21 logit values for classification.

### C.2. Implementations of compared methods

**Baseline unsupervised reconstruction methods.** To implement the *Esl16* [10] baseline, we use a publicly available third-party implementation<sup>1</sup>. We note that their method was originally applied to a dataset consisting of images of 0, 1, or 2 digits with equal probability. We found that the model failed to converge unless it was trained with examples where various number of total digits exist, so for fair comparison, we populate the training set with images consisting of all numbers of digits between 0 and 9. For the *Zha18* [57] baseline, we use the authors’ implementation and hyperparameters.

**Differentiable top-K (Xie20) [52].** We implemented their method as a PyTorch module according to the pseudo-code provided in [52]. In addition to the provided pseudo-code, we add a small epsilon clipping for the division operations within the equations for stability. The differentiable

<sup>1</sup><https://github.com/aakhundov/tf-attend-infer-repeat>

top-K operation in [52] outputs a top-k selection mask  $\mathbf{m} \in (0, 1)^N$ , where  $N$  is number of elements to select from – top-K elements have a mask score close to 1, and non selected elements have a mask score close to 0. To apply this method to our task, one needs to then apply this mask to the classification results of patches at all heatmap location, as the operation is not indexing, but rather masking – this is how differentiability is obtained in this method. It is therefore necessary that all results that gets masks to stay in memory, and requires a smaller heatmap to be trained on reasonable system – we use a *GeForce RTX 2080 Ti* with 11GB memory. In addition, we modify the classification loss in Eq.(9) to incorporate the selection mask:

$$\mathcal{L}_{\text{task}} = \left\| \frac{1}{L} \sum_{l=1}^L \mathbf{y}_l - \frac{1}{K} \hat{\mathbf{p}} \times \mathbf{m} \right\|_2^2, \quad (12)$$

where  $L$  is the number of instances in the image,  $\hat{\mathbf{p}}$  is the  $C \times N$  classification score matrix,  $\mathbf{m}$  is the mask vector of size  $N$ , with  $C$  being the number of classes. Note that the only difference here is that the top-K selection via indexing in the main paper has now been transformed into a mask-based selection.

## D. Non-uniform distributions



Figure 8. Examples with uneven distributions of digits.

Although the images we show in Figure 2 involve small displacements from a uniformly spaced grid, our method does not require the keypoints to be evenly spread. As shown in Figure 8, our method is able to successfully learn even when the digits are placed unevenly. Note that, as our detector is fully convolutional and local, it does not learn the absolute location of keypoints. In fact, we weakened the randomness of the locations for fairness against [57], which is not designed to deal with severe displacements.

## Additional References

- [A1] Ross Girshick. Fast R-CNN. In *International Conference on Computer Vision*.
- [A2] Peter J. Huber. Robust estimation of a location parameter. *Ann. Math. Statist.*, 35(1):73–101, 03 1964.
- [A3] Angelos Katharopoulos and François Fleuret. Processing Megapixel Images with Deep Attention-Sampling Models. In *International Conference on Machine Learning*, 2019.