

MULTI-TASK LEARNING BY A TOP-DOWN CONTROL NETWORK - SUPPLEMENTARY MATERIALS

Anonymous authors

Paper under double-blind review

We first describe the three datasets used our experiments, then add full architecture descriptions, training hyperparameters, detailed results (5 repetitions to each experiment) and examples of attention maps, produced on both of CLEVR and CUB200 evaluation dataset, demonstrating our inherent ability to extract useful spatial information in a task dependent manner.

1 DATASETS

1.1 MULTI-MNIST

MultiMNIST (Sabour et al., 2017) is a multi-task learning version of the MNIST dataset in which multiple MNIST images are placed on the same image. We use 2, 4 and 9 classes experiments built as suggested by Sener & Koltun (2018). In the 2-classes experiment the tasks are: classifying the digit on the top-left (task-TL) and classifying the digit on the bottom-right (task-BR). We correspondently add tasks (TM, TR, L, C, R, BL and BM) for classifying the digits on the top-mid, top-right, left, center, right, bottom-left and bottom-mid on the 4 and 9-classes experiments. The digits in each position are independently chosen. We use 60K examples and directly apply LeNet (LeCun et al., 1998) as the underlying backbone in our experiments.

1.2 CLEVR

CLEVR is a synthetic dataset, consists of 70K training images and 15K validation images, mainly used as a diagnostic dataset for VQA. The dataset includes images of 3D primitives, with multiple attributes (shape, size, color and material) and a set of corresponding (question-answer) pairs. We followed the work Liu et al. (2019), which suggested to use CLEVR not as a VQA dataset, but rather as a so-called referring expression dataset, and further adapt it to a multi-task learning methodology. The tasks in our setup consist of 40 questions that query an attribute of an object that is to the (left, right, up, down) of a referred object. Typical tasks in our experiment are: "What is the color of the object to the right of the metal cylinder?" and "What is the shape of the object to the left of the small object?", where metallic cylinder and small object are the referred objects respectively. Notice that given an image, only some of the questions are allowed (a question is valid only if the image includes exactly one instance of the referred object). In our experiment, given a batch of images, we randomly chose the corresponding questions from the set of valid questions of each image. To demonstrate our ability to scale the number of tasks, we further extended our set of tasks to include 80, 160 and 1645 (all possible questions in the described structure) questions. results are in the main text. To demonstrate our interpretability capability, we also used the coordinates of the object to the (left, right, up, down) of the referred objects, annotated by a single point, as an auxiliary target at the end of the TD stream.

1.3 CELEB-A

CELEB-A (Liu et al., 2015) is a set of real-world celeb face images, intensively used in the scope of MTL (e.g., Sener & Koltun (2018), Strezoski et al. (2019)) on attribute classification tasks. The dataset consists of 200K images with binary annotations on 40 face attributes related to expression, facial parts, etc.

1.4 CUB-200

CUB-200 (Welinder et al., 2010) is a fine grained recognition dataset that provides 11,788 bird images (equally divided for training and testing) over 200 bird species with 312 binary attribute annotations, most of them referring to the colors of specific birds' parts. In contrast to other work (Strezoski et al., 2019) that used all the 312 attributes as yes/no questions, we re-organized the attributes as a

multi-task problem of 12 tasks (for 12 annotated bird’s parts) each with 16 classes (the annotated colors + an unknown class) and trained using a multi-class cross-entropy loss. To demonstrate our interpretability capability, we further used the parts’ location, annotated by a single point to each seen part, as an auxiliary target at the end of the TD stream.

2 TRAINING & HYPERPARAMETERS

We use LeNet, VGG-11, VGG-7 and resnet-18 as our backbone BU architectures for the Multi-MNIST, CLEVR, CELEB-A and CUB-200 experiments respectively. Each of the backbones has been divided to two parts; a first part that consists mainly of the convolutional layers of the backbone and a second part with the fully connected layers (including the classifier).

In our architecture, both BU streams consist of the first part of the backbone and share their weights. The TD stream, unless specified otherwise, is a replica of the BU stream combined with upsampling layers. The classifier is only attached to the BU2 stream. Information is passed between the BU1, TD and BU2 streams using lateral connections implemented as 1x1 convolutions. A task embedding layer (a fully connected layer) is added on the top of the TD stream. During training, the learning optimizes all the weights along the BU and TD streams, shared by all tasks, as well as the task specific embedding parameters. Learning uses a standard backpropagation. See an illustration of the full scheme in the main text and a detailed architecture description in the next section.

In training time, the network is supplied with an input image and a selected task, drawn at random from the different tasks. During testing, the different tasks are applied sequentially to each test image.

2.1 MULTI-MNIST

We use the Multi-MNIST dataset to demonstrate our performance for 2, 4, and 9 tasks recognition problems. All models trained using a standard LeNet architecture. We used a batch size of 512 images trained on 1 GPU with learning rate of $1e^{-3}$ using the Adam optimizer. We followed Sener & Koltun (2018) and decrease the learning rate by a factor of 2 every 30 epochs. For a fair comparison, all models were trained with the same amount of training examples per task (60K examples per task in an epoch) for 100 epochs.

2.2 CLEVR

We used the CLEVR dataset to test performance while scaling the number of tasks (up to 1645) with a fixed model size.

We trained all models using a VGG-11 architecture but decreased the number of channels in the output of the last convolutional layer from 512 to 128 to allow training with larger batch size. We used a batch size of 128 images trained on 2 GPUs with learning rate of $1e^{-4}$ using the Adam optimizer and decreased the learning rate by a factor of 2 every 30 epochs. An auxiliary localization loss of the referred objects was added to our architecture, detailed on the next section and illustrated in figures 5 and 6.

2.3 CELEB-A

We used the CELEB-A dataset to test performance on higher level classification tasks on real world images. We trained all models using a VGG-7 architecture (with 32, 64, 128, 128, 128, 128, 128 channels). We used a batch size of 512 images trained on 4 GPUs with learning rate of $1e^{-3}$ using the Adam optimizer and decreased the learning rate by a factor of 2 after 30 epochs.

2.4 CUB-200

We used the CUB200 dataset to test performance on real-world images with low-level features, and to demonstrate our use of interpretability. We trained all models using a Resnet-18 architecture. We used a batch size of 128 images trained on 2 GPUs with learning rate of $1e^{-4}$ using the Adam optimizer for 200 epochs. We added an auxiliary loss at the end of the TD stream. The target in this case is a 224x224 mask, where a single pixel, blurred by a Gaussian kernel with a standard deviation of 3 pixels, indicated the part’s location. Training one task at a time, we minimize both the cross-entropy loss at the top of BU2 (classification loss) and the cross-entropy loss taken over the 224x224 image at the end of the TD softmax output (which encourages a small detected area) with

Table 1: Detailed architectures description in term of convolutional and FC layers for the M-MNIST experiment. **K** (bold) is the number of tasks. (a) multi-branch architecture, (b) channel modulation architecture and (c) our TD control network.

(a) Multi-Branch architecture					(c) Our TD control network				
SubModule	Layer	kernel	channels	remarks	SubModule	Layer	kernel	channels	remarks
BU	conv2d	5	1→10		BU1	conv2d	5	1→10	
	conv2d	5	10→20			conv2d	5	10→20	
	FC		320→50			FC		320→50	
branch	FC		50→50		laterals1	conv2d	1	10→10	
	FC		50→10	K times		conv2d	1	20→20	
						conv2d	1	20→20	
(b) channel-modulation architecture									
SubModule	Layer	kernel	channels	remarks	EMB	FC		K →320	
EMB	FC		K→10		TD	conv2d	5	20→10	
	FC		K→20			conv2d	5	10→1	
	FC		K→20		laterals2	conv2d	1	1→1	
BU	conv2d	5	1→10			conv2d	1	10→10	
	conv2d	5	10→20			conv2d	1	20→20	
	FC		320→50		BU2	conv2d	5	1→10	shared weights
branch	FC		50→50			conv2d	5	10→20	
	FC		50→10			FC		320→50	
					branch	FC		50→50	
						FC		50→10	

the appropriate targets for each task. Applying the localization loss in train time allows us to obtain an attention map in inference time, helping interpretability by locating the reference parts attended by the network (see illustrations of correctly predicted tasks in figure 3 and of failure cases in figure 4).

3 IMPLEMENTATION DETAILS

3.1 DETAILED ARCHITECTURES DESCRIPTION FOR THE MULTI-MNIST EXPERIMENT

For the MultiMNIST experiments, we use an architecture based on LeNet. We follow Sener & Koltun (2018) and use the two 5x5 convolutional layers and the first fully-connected layer as the shared BU backbone module and the two other fully-connected layers as the branch module. The "single task" architecture uses several duplication, same as the number of tasks, of this basic structure, each consisting of a backbone and a branch. The "multi-branch" architecture uses several task specific branch modules, same as the number of tasks, on top of a shared BU module. Table 1a summarizes this architecture in detail.

The "channel modulation" architecture consists of a BU module and a single branch module. Here, three trainable FC layers (EMB module) create the task embeddings (channel weights), later integrated into the BU stream by a multiplication operation. See table 1b for more details.

Our task-based TD control network for the Multi-MNIST experiment is specified in detail in table 1c. We use two BU streams with shared weights; BU1 is task independent while BU2 is task dependent, modified by the TD stream for an accurate prediction of the specific task. The TD stream consists of successive convolutional and interpolation layers and, unless stated otherwise, is a replica of the BU stream in terms of layers types and number of channels. We use the convolutional layers in the TD network as an efficient way to induce the modification tensors, which multiply the feature-maps along BU2. The task is supplied to the network at the top of the TD stream as an one-hot-vector, passes through an embedding layer. A single branch is attached to the top of BU2.

A similar implementation was used for the CLEVR, CELEB-A and CUB200 experiments while using VGG-11, VGG-7 and ResNet-18 as backbones. The exact number of parameters in the architectures is listed in table 2.

3.2 LOCALIZATION AUXILIARY LOSS

In our architecture, a fine-resolution localization loss can be naturally integrated at the end of the top-down stream. This can be useful for interpretability and visualization and may also help attending relevant objects in the image, based on the current task and the image content.

Table 2: Number of parameters in the M-MNIST, CLEVR, CELEB-A and CUB200 architectures

Module / Architecture	Multi-MNIST	CLEVR	CELEB-A	CUB-200
Recognition backbone	21,250	7,448,256	387,936	4,900,032
Each branch	3,000	7,473,152	819,456	8,192
TD with laterals	6,651	8,306,688	180,224	3,052,544
task embedding	320	1568	1568	1568
Single-Task architecture	24,250	14,921,408	1,207,392	4,908,224
Multi-branched architecture	$21,250 + 3000 \cdot K$	$7,448,256 + 7,473,152 \cdot K$	$387,936 + 819,456 \cdot K$	$4,900,032 + 8,192 \cdot K$
TD modulation architecture	$30,901 + 320 \cdot K$	$23,228,096 + 1568 \cdot K$	$1,387,616 + 1568 \cdot K$	$7,960,768 + 1568 \cdot K$

We used this option in the CLEVR and CUB200 experiments, where a single dot marked the ground truth location of the center of the objects (CLEVR) or the specific bird part (CUB200). Specifically, in training our architecture, we add an auxiliary loss at the end of the TD stream. The target is a 224x224 mask, where a single pixel, blurred by a Gaussian kernel with a standard deviation of 3 pixels, is marked. Training one task at a time, we minimize the cross-entropy loss over the 224x224 image at the end of the TD softmax output. This TD output allows us to create a visual attention map in inference time, which illustrates the relative weights assigned by the network to different locations in the image. Examples of interest are given in section 4.3 for the CLEVR and CUB200 experiments.

For a fair comparison we have also added an auxiliary localization loss to the multi-branch and to the channel modulation architectures, by changing the branch structure and adding an additional FC layer to predict the object’s location. We used a regression loss with respect to the x, y coordinates of the ground-truth dot annotations of the relevant objects, normalized to the range of $(-1, 1)$. Adding the auxiliary loss was found beneficial in the CLEVR and CUB200 experiments; however, the accuracies were still lower than ours (see main text) and lack an immediate visual interpretation. A possible explanation for the accuracy gap, is that in our architecture, the auxiliary loss explicitly guides an early attention process of the main recognition network, while in the other architectures the regression loss induces a late attention process only.

3.3 COMPARISON OF FOUR ARCHITECTURES - DESIGN CHOICES

Figures 1a and 1b demonstrate the average accuracy of the 9-classes (M-MNIST) and 40-tasks (CLEVR) experiments as a function of the number of parameters in four types of architectures. The large markers correspond to the architectures that have been used in the main text. The exact design choices of all the other architectures in the multi-MNIST experiment, marked with small markers, are as follows:

Multi-branch architectures: For the MultiMNIST experiments, we use an architecture based on LeNet. In our experiments we follow Sener & Koltun (2018) and use the two 5x5 convolutional layers and the first fully-connected layer as the shared BU backbone module and the two other fully-connected layers as the branch module. Two other possible design choices are to divide the network into shared layers and branches, after the first two convolutional layers (corresponding to the right blue circle) or after the second FC layer (corresponding to the left blue circle).

Channel-Modulation architectures: The channel modulation architecture consists of a BU module and a single branch. Here, three trainable FC layers (EMB module) create the task embeddings with an appropriate channel dimensions. In our experiments we used the original LeNet, based on two convolutional layers which correspond to featuremaps with 10 and 20 channels. The architectures that correspond to the two small cyan stars were obtained by increasing the number of channels in the convolutional layers to (15, 25) and to (20, 30).

Our Control Network: In the experiments we used a TD stream which is a replica of the BU stream in terms of channel dimensions (1, 10 and 20 channels in its feature-maps). This is a reasonable design choice since, given any BU architecture, it fixes our BU-TD-BU architecture to a standard structure and prevents an extensive neural architecture search, which might be application dependent. However, for demonstrating our advantages in term of accuracy vs. model size we created cheaper architectures which use a reduced number of channels along the layers in the TD stream. The three left red stars in figure 1a correspond to networks with 1, 4 and 6 channels in their TD stream. The right red star corresponds to a wider network, with (15, 25) channels in the BU convolutional layers and 10 channels in its TD convolutional layers.

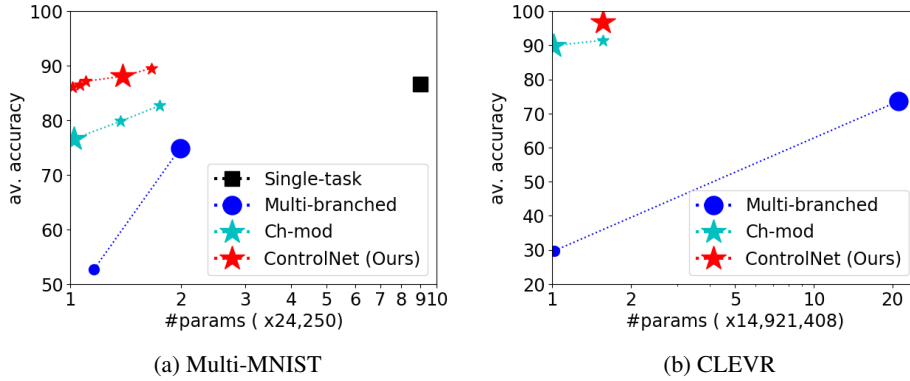


Figure 1: **Comparison of four architectures by average accuracy vs. model size.** Our approach shows better accuracy compared with alternatives with the same model size. top-left is better. (a) Multi-MNIST (b) CLEVR

The additional architectures in the CLEVR experiment, marked with small markers in Figure 1b were similarly designed. The figures show that our family of architectures corresponds to the highest (red) curve in the plot (top-left is better), indicating higher performance than alternatives for a similar number of parameters.

4 ADDITIONAL RESULTS

In this section, we present the experimental results not included in the main text.

4.1 MULTI-MNIST EXPERIMENT

Table 3 shows the mean accuracy and the standard deviation of 5 independent trainings and evaluations for each of the results in the Multi-MNIST experiment, higher is better. Our architecture achieves consistently better accuracies per task than any of the other alternatives. Comparing to the single task baseline we achieve better accuracies while using much less parameters (the third column shows the ratio between the number of parameters and a standard LeNet architecture). Scaling the number of tasks in our architecture costs almost no additional resources. On the 4-tasks and 9-tasks experiments our architecture uses less parameters than the uniform scaling approach with a large accuracy gap. The channel modulation and the task routing approaches achieve better accuracies than the uniform scaling approach, but their results are significantly lower than ours. Further optimizing our architecture in terms of model size can be done, but outside the scope of the current work.

Table 3: Performance (mean \pm std of 5 repetitions) on Multi-MNIST, higher is better. Our architecture achieves significantly better results than any of the other alternatives.

Tasks	ALG	#P	"TL" acc	"L" acc	"BL" acc	"TM" acc	"C" acc	"BM" acc	"TR" acc	"R" acc	"BR" acc
2	Single task	x2	96.99 \pm 0.1								95.93 \pm 0.1
	Uniform sca	x1.12	95.86 \pm 0.1								94.75 \pm 0.2
	MOO	x1.12	96.25 \pm 0.3								95.38 \pm 0.2
	ch-mod	x1.002	96.53 \pm 0.1								95.21 \pm 0.1
	task-routing	x1.002	95.67 \pm 0.2								94.57 \pm 0.2
	Ours	x1.29	97.16 \pm 0.1								96.19 \pm 0.1
4	Single task	x4	95.73 \pm 0.1		94.81 \pm 0.1				93.11 \pm 0.2		92.97 \pm 0.1
	Uniform sca	x1.37	92.86 \pm 0.3		91.80 \pm 0.3				88.88 \pm 0.7		89.30 \pm 0.6
	MOO	x1.37	92.96 \pm 0.2		92.08 \pm 0.3				89.87 \pm 0.3		90.04 \pm 0.5
	ch-mod	x1.007	93.29 \pm 0.2		92.11 \pm 0.3				90.09 \pm 0.2		90.05 \pm 0.4
	task-routing	x1.007	93.66 \pm 0.2		92.86 \pm 0.1				90.73 \pm 0.2		91.10 \pm 0.1
	Ours	x1.32	95.76 \pm 0.1		95.11 \pm 0.1				93.81 \pm 0.3		93.89 \pm 0.3
9	Single task	x9	93.44 \pm 0.1	83.05 \pm 0.2	89.61 \pm 0.4	88.19 \pm 0.5	75.01 \pm 0.2	86.57 \pm 0.3	92.54 \pm 0.5	81.05 \pm 0.2	90.10 \pm 0.3
	Uniform sca	x1.99	83.87 \pm 0.9	68.63 \pm 1.0	78.81 \pm 0.2	77.76 \pm 0.3	63.14 \pm 0.6	76.11 \pm 0.3	81.49 \pm 0.6	64.98 \pm 0.7	78.44 \pm 0.8
	MOO	x1.99	81.47 \pm 1.3	70.59 \pm 0.6	78.02 \pm 1.1	76.83 \pm 1.1	66.04 \pm 0.7	75.98 \pm 0.3	80.55 \pm 0.6	68.95 \pm 0.5	77.78 \pm 0.7
	ch-mod	x1.015	86.23 \pm 0.5	69.99 \pm 0.4	81.27 \pm 0.9	79.90 \pm 0.5	61.70 \pm 0.9	75.69 \pm 1.0	84.56 \pm 0.4	68.23 \pm 0.7	81.43 \pm 0.5
	task-routing	x1.015	88.26 \pm 0.6	74.99 \pm 0.3	84.52 \pm 0.7	82.88 \pm 0.5	67.76 \pm 0.9	80.75 \pm 0.6	87.06 \pm 0.3	74.13 \pm 0.5	84.35 \pm 0.7
	Ours	x1.39	93.90 \pm 0.2	84.18 \pm 0.3	91.20 \pm 0.4	90.21 \pm 0.4	77.19 \pm 0.3	88.14 \pm 0.2	93.55 \pm 0.2	82.96 \pm 0.1	91.29 \pm 0.4

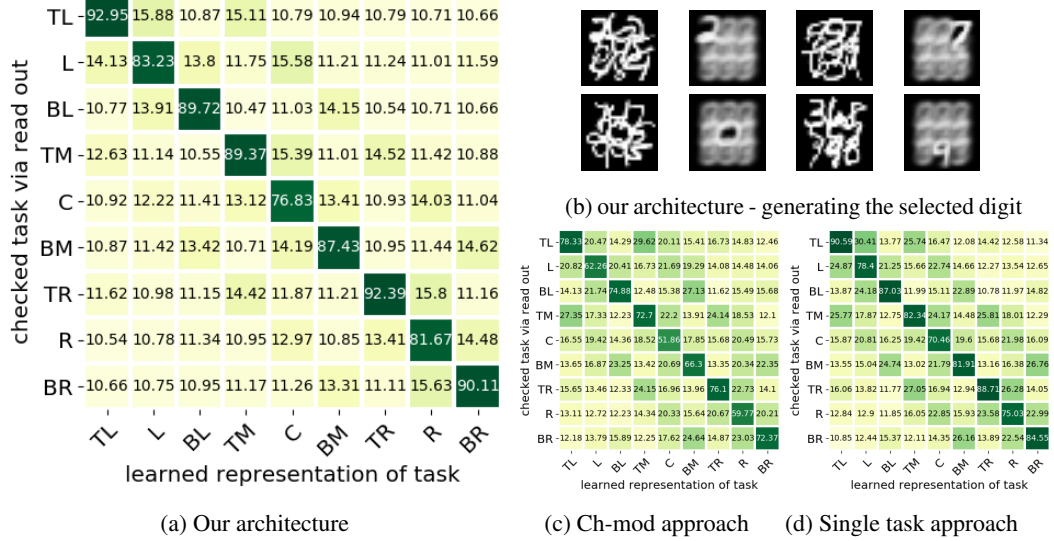


Figure 2: **Read out experiment results - 9 tasks.** The representation at the top of BU2 in our architecture (a) is task selective and shows almost chance-level accuracies when predicting other tasks (b) generates only the corresponding digit. (c, d) Other architectures, on the other hand, demonstrate less specificity to the selected task.

4.1.1 TASK SPECIFICITY FOR THE 9-CLASSES EXPERIMENT

Following by the task-selectivity properties found in the 4-classes experiment (see main text) we performed a similar experiment on the 9-classes Multi-MNIST images. Briefly, the representation at the top of BU2 is task-dependent. Measuring its specificity is obtained by reading out each and every of the tasks representation (accumulated with a pre-trained frozen architecture) and predicting the digits for all of the locations, each with its own trainable branch. A specificity measure of our architecture is summarized in figure 2a. The figure shows that the learned representation of each task is highly adjusted to the instructed task and shows almost chance-level accuracies on the other branches. Similarly, a generative branch (two linear layers with a ReLU between) that predicts an image-size segmentation-map from the frozen representation (trained with respect to the original images with a binary cross entropy loss) generates the corresponding digit only. Examples of interests are demonstrated in figure 2b for various tasks. Performing the digit prediction experiment on the shared representation at the top of BU1 network achieves accuracies in the range of (17%, 24%) for all of the branches (BU1 is not task-selective). This "average-representation" is then fed to the TD network to be conditioned on a specific task.

We compared the task-selectivity results of our architecture to other methods and performed the same experiment on the single task and channel-modulation architectures. The results are presented in figures 2c and 2d. The results show that the task-dependent representations in these architectures are less task-selective, showing that early spatial and image-dependent modifications of the feature-maps along the recognition network sharpen task selectivity which is likely to be beneficial in the scope of multi-task learning.

4.1.2 ADDITIONAL ABLATIONS AND COMPARISONS

Number of channels in the TD stream. Table 4a compares the results accuracies of our proposed architecture (first line, where the TD stream is a replica of the BU stream which has 1, 10 and 20 channels in its feature-maps) with cheaper architectures which use a reduced number of channels along the layers in the TD stream. Our experiments show a trend line (accuracies decrease when the number of channels in the TD stream decreases) and illustrate how optimizing the number of channels along the TD stream in terms of efficiency-accuracy trade-off can be obtained.

Table 4: Ablations on Multi-MNIST

(a) number of channels 9-classes experiment			(b) connectivity type 2-classes experiment		
#ch	#P	av. acc.	td	bu2	av. acc
dup	x1.39	88.07	+	+	96.10
6	x1.10	87.13	+	x	96.67
4	x1.06	86.52	x	+	96.23
1	x1.01	86.03	x	x	95.98

(c) lateral importance 9-classes experiment		(d) several branches 4-classes experiment		
	av. acc	branches	#P	av. acc.
all	88.07	1	x1.32	94.64
lateral 3 only	80.91	4	x	94.78
lateral 2 only	87.40	ext, 1	x	95.01
lateral 1 only	86.27	ext, 4	x	95.04

Connectivity types. Our architecture uses two sets of lateral connections; the first set passes information from the BU1 stream to the TD stream, and the second passes information from the TD stream to the BU2 stream. Table 4b compares the results (mean of 5 repetitions) of our proposed architecture when using different connectivity types to the TD stream (first column) and to the BU2 stream (second column). Here + is an addition connectivity and \times is a multiplication connectivity. The table shows higher accuracy when using additive connectivity along the TD stream and multiplicative connectivity along the BU2 stream.

Contribution of the lateral connections. Our architecture in the Multi-MNIST case modifies the recognition network using 3 lateral connections. Table 4c shows the resulting accuracy when using only one lateral connection at a time. Using all 3 lateral connections yields better accuracy than using any of them separately. Interestingly, using only the highest level lateral connection results in low accuracy, suggesting that controlling the units in the first featuremaps of the network according to the task is beneficial to the recognition process.

Number of branches. Our architecture uses a single branch only. Table 4d shows that using four branches, one for each task, further improves the results by 0.24 points. This might be explained by low-capacity of the branch. Consistent with this possibility, extending the branch capacity (using a FC layer with channel size 80 instead of 50) eliminates this gap.

4.2 CUB200 EXPERIMENT

Table 5 shows the mean accuracy of 5 independent trainings and evaluations phases for each of the results in the CUB200 experiment.

4.3 ADDITIONAL QUALITATIVE EXAMPLES

To examine the use of the TD channel for interpretability, we trained The network with an auxiliary localization cross-entropy loss in the last layer of the TD stream (details in section 1.2 and in main text). Figures (3-6) present several examples of interest from the validation set of CUB200 and CLEVR (1645 tasks) not included in the main text.

Table 5: Performance on CUB200, higher is better. Our architecture is scalable with the number of tasks and outperforms other methods. All models trained for 200 epochs with lr 1e-4 using a resnet-18 backbone.

	Single task	Uniform Scaling	MOO	ch-mod	ch-mod + loc	tsk-rout + loc	Ours + loc
wing	78.28	81.68	81.67	82.44	82.65	82.98	84.60
uppertail	75.45	79.67	80.00	82.21	82.09	81.58	82.73
throat	74.49	77.22	77.54	79.29	79.65	79.22	81.54
nape	70.73	75.29	75.44	77.17	77.57	76.92	80.06
leg	62.22	64.57	65.75	69.31	68.37	69.46	68.55
eye	89.65	90.23	90.62	91.63	92.08	92.28	90.96
back	75.25	79.92	80.10	81.89	82.05	82.58	82.99
breast	75.62	78.93	78.89	80.47	80.78	80.58	83.14
forehead	70.78	74.16	74.13	76.95	77.42	76.68	78.55
belly	77.61	80.57	80.72	82.46	82.31	82.01	83.96
crown	72.05	75.11	75.23	77.98	78.05	78.11	79.48
bill	69.95	72.48	72.95	76.63	75.83	78.06	74.07
mean	74.34	77.49	77.75	79.87	79.91	80.04	80.89

REFERENCES

- Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- Runtao Liu, Chenxi Liu, Yutong Bai, and Alan L Yuille. Clevr-ref+: Diagnosing visual reasoning with referring expressions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4185–4194, 2019.
- Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.
- Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in neural information processing systems*, pp. 3856–3866, 2017.
- Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, pp. 527–538, 2018.
- Gjorgji Strezoski, Nanne van Noord, and Marcel Worring. Many task learning with task routing. *arXiv preprint arXiv:1903.12117*, 2019.
- P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

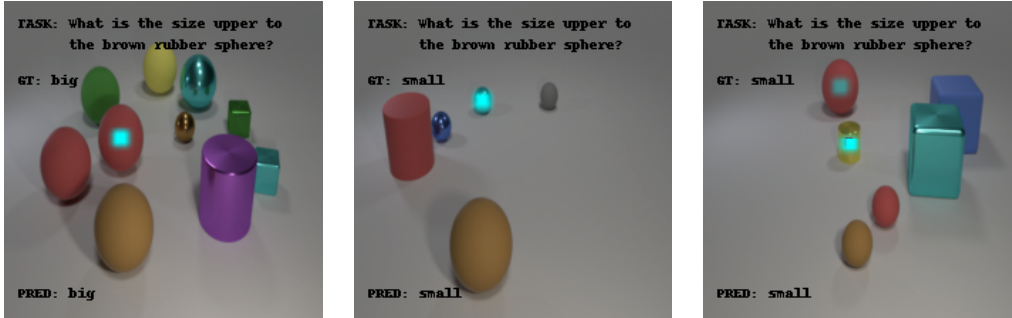


Figure 3: Qualitative examples illustrating the identification of the relevant regions that most affected the network prediction. In all images the target part (the task, shown in the upper part of each image), is precisely localized and the prediction (shown in the lower part of each image) follows the ground truth. Best viewed in color and zooming in.



Figure 4: Error cases. Left 2 images are in fact correct results, counted as failure cases due to annotations errors. Our network successfully localized the target part and correctly predicts its color. Right 2 images demonstrate bad localization examples. Ground truth classes were still predicted, with a very high score, possibly due to the correlated nature of the tasks. Best viewed in color and zooming in.

What is the size of the object above the brown rubber sphere?



What is the color of the object to the right of the cylinder?

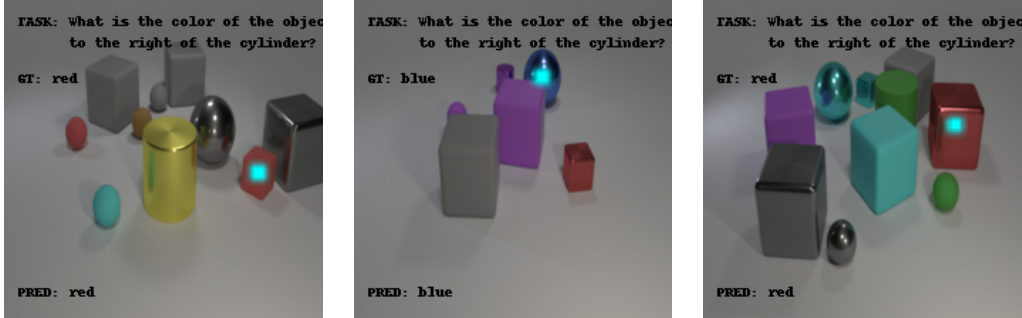
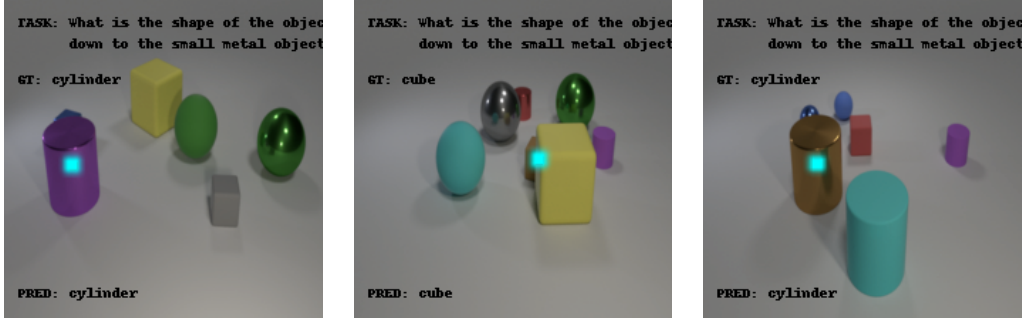


Figure 5: CLEVR, 1645 tasks. First row: What is the size of the object above the brown rubber sphere?, Second row: What is the color of the object to the right of the cylinder? The network successfully localizes the target objects and correctly predicts their size/color. Best viewed in color and zooming in.

What is the shape of the object below the small metal object?



What is the shape of the object below the small metal object? - Error cases

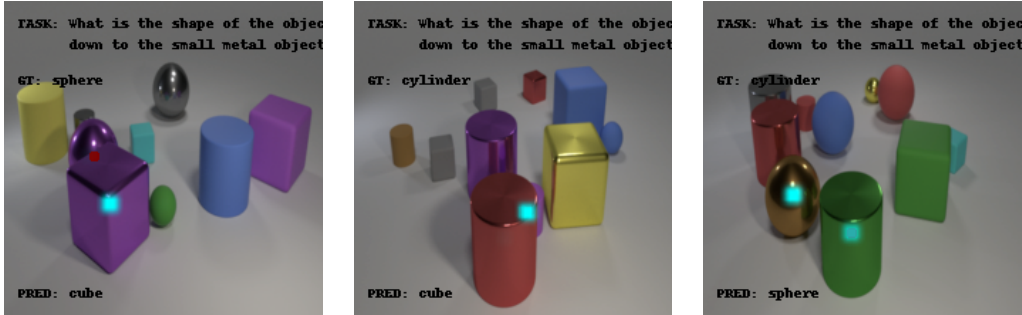


Figure 6: CLEVR, 1645 tasks. What is the shape of the object below the small metal object? First row: occlusion cases. The network successfully predicts the shape although the occlusion. Second row: error cases. Best viewed in color and zooming in.