Exploring Relational Context for Multi-Task Dense Prediction

David Bruggemann, Menelaos Kanakis, Anton Obukhov, Stamatios Georgoulis, Luc Van Gool
ETH Zurich
{brdavid, kanakism, obukhova, georgous, vangool}@vision.ee.ethz.ch

Abstract

The timeline of computer vision research is marked with advances in learning and utilizing efficient contextual representations. Most of them, however, are targeted at improving model performance on a single downstream task. We consider a multi-task environment for dense prediction tasks, represented by a common backbone and independent task-specific heads. Our goal is to find the most efficient way to refine each task prediction by capturing cross-task contexts dependent on tasks' relations. We explore various attention-based contexts, such as global and local, in the multi-task setting and analyze their behavior when applied to refine each task independently. Empirical findings confirm that different source-target task pairs benefit from different context types. To automate the selection process, we propose an Adaptive Task-Relational Context (ATRC) module, which samples the pool of all available contexts for each task pair using neural architecture search and outputs the optimal configuration for deployment. Our method achieves state-of-the-art performance on two important multi-task benchmarks, namely NYUD-v2 and PASCAL-Context. The proposed ATRC has a low computational toll and can be used as a drop-in refinement module for any supervised multi-task architecture.

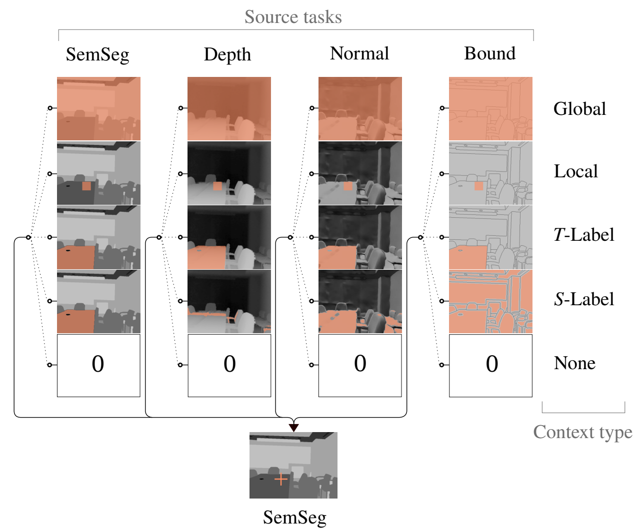


Figure 1. Schematic of the task relational context (orange overlay) for the marked pixel (orange cross) of target task semantic segmentation. Our algorithm selects one distillation context type for each source task (dashed lines represent a switch). Alternatively, the connection can be severed by choosing *none*. The procedure is analogous for all other target tasks.

1. Introduction

The role of context in computer vision is hard to overstate; most notable breakthroughs boil down to a clever extraction [30], learning [26], and utilization [25] of contextual representations. The success of Convolutional Neural Networks (CNN) is largely due to their inherent ability to capture the local context and build very deep [41] contextual hierarchies within the model. Recently, the progressive adoption of the attention mechanism in computer vision [51] has brought forth more flexible context descriptions conditioned on the interdependence of individual pixels, while steadily replacing the traditional convolutional building blocks [11].

Multi-Task Learning (MTL) [6] is concerned with sharing representations between tasks. Motivated by the observation that representations of visual tasks are often highly correlated [56], recent works [50, 44] focusing on multi-task dense prediction have extended context extraction across tasks through soft-gated message passing. Referred to as multi-modal distillation in the literature [50], the idea is to augment the high-level representations of downstream *target tasks* by selectively aggregating complementary features of a set of *source tasks*. The gating function in the distillation thereby learns to focus on useful cross-task information flow.

Despite their effectiveness, current multi-modal distillation schemes [50, 44] suffer from two main limitations: (1) The employed gates only regulate information flow based on the source task feature values. As such, the distillation module fails to capture task interactions fully. (2) Each target pixel exclusively receives information from its source counterpart, *i.e.*, the message passing is restricted locally. Compelled by these drawbacks, we propose a new type of attention-driven multi-modal distillation scheme, based on three key contributions:

1. Increase the expressivity of the cross-task gate by conditioning it on the interdependence of source and target task pixels. Our multi-modal distillation scheme is therefore *relational*.
2. Enable global cross-task message passing by enlarging the receptive field of the distillation scheme. We refer to each pixel's distillation receptive field as its *distillation context*.
3. Customize the distillation context type for each source-target task pair. We formulate five context type candidates (*global*, *local*, *T-label*, *S-label*, *none*) and *adapt* the type automatically with respect to each source-target task pair in a given architecture (see Fig. 1).

Contributions 1 and 2 are addressed by leveraging and adapting the scaled-dot product attention mechanism [45] for multi-modal distillation. For contribution 3, we repurpose modern Neural Architecture Search (NAS) methods to automatically find the optimal context type for each source-target task connection. Overall, we present a novel Adaptive Task-Relational Context (ATRC) module which can be used as a drop-in module for CNNs to refine any dictionary of supervised dense prediction tasks. We show its effectiveness empirically with the architecture shown in Fig. 2: a single neural network for all tasks with a shared backbone of RGB input, multiple task-specific heads, and ATRC distillation modules to refine each task's predictions.

The paper is structured as follows: Sec. 2 provides an overview of related work; Sec. 3.1 introduces the architecture of ATRC; Sec. 3.2 explains the types of relational contexts in consideration; Sec. 3.3 covers the adaptation of the context type through NAS techniques; Sec. 4 provides the empirical study details and verifies the proposed method state-of-the-art performance on several important benchmarks; Sec. 5 concludes the paper.

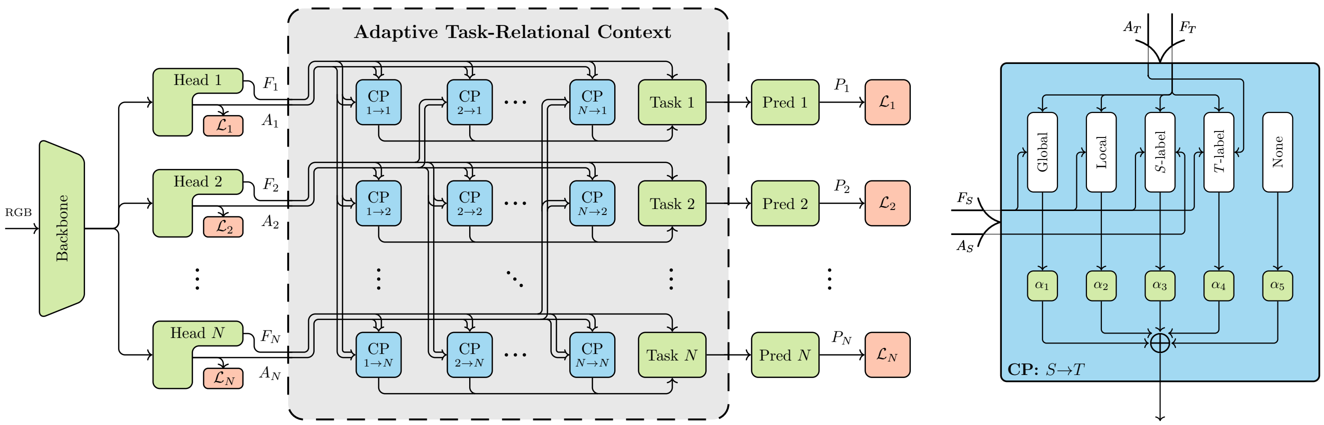
2. Related Work

Multi-Task Learning (MTL) methods employ two main paradigms to learn shared representations: *hard parameter sharing* and *soft parameter sharing*. Hard parameter sharing characterizes architectures which typically share the first hidden representations among the tasks while branching to independent task-specific representations at a later stage. Most approaches split to task-specific heads at a single branch point [24, 22, 10, 39]. However, such naive branching can be sub-optimal, raising interest in mechanisms that allow for finely branched architectures [31, 43, 5]. Our work is complementary to these hard parameters sharing methods, since we introduce a module which refines task-specific features. Soft parameter sharing, in contrast, marks architectures which induce knowledge transfer between separate task-specific networks through feature fusing mechanisms. Feature fusing can be introduced along the entire network depth [35, 15, 28], whereby computational cost is often a limiting factor. Our proposed module can be interpreted as a sophisticated feature fusing mechanism, applied only at a single stage to refine high-level representations.

Several recent MTL works follow a similar strategy: PAP [56] and PSD [57] refine task-specific feature maps through global and local self-attention respectively. The employed attention masks are first refined by propagating affinity patterns across tasks and then applied iteratively on the target task feature maps. In contrast to [56, 57], our approach directly attends to source task features by explicitly modeling pairwise interactions between source and target tasks. More closely related to our work, PAD-Net [50] uses multi-modal distillation to enhance task-specific predictions. Information flow from each source to target task is regulated with a sigmoid-activated gate function. MTI-Net [44] combines the multi-modal distillation module of PAD-Net with a multi-scale refinement scheme to facilitate cross-task talk at multiple scales. However, the gates used in the distillation module of [50, 44] are functions of the source task features only and operate per pixel. Our method, on the other hand, leverages pairwise task similarities to create more expressive gates through the attention mechanism, while also enabling global cross-task message passing.

Attention was originally developed to improve sentence alignment in neural machine translation [2]. In computer vision, variants of scaled dot-product attention [45] in particular have been used to capture global relationships over the entire pixel space [47, 3, 52], locally [36], and even channel-wise [13]. In these approaches, the representation of each target pixel is augmented by aggregating the representations of pixels within the specified context. Each context pixel thereby contributes according to its relation to the target, hence the term *relational context*. Relevant to our work, A^2-Net [9], ACFNet [54], and OCR-Net [53] define their own relational context types by grouping pixels into distinct regions (*e.g.*, object class) and attending to prototypical representations of those regions instead. All of the above mentioned methods focus on attention for a single downstream task and utilize fixed context descriptions. Our work extends these concepts to a multi-task scenario while choosing the optimal relational context type from a pool of candidates for each source-target task pair.

Neural Architecture Search (NAS) automates the process of engineering problem-specific neural network architectures, with the goal of minimizing hand-crafted network design. To this end, seminal works use either reinforcement learning [58, 59] or evolutionary [38, 37] algorithms to sample promising candidate architectures from a large



(a) (b)

Figure 2. **(a)** Overview of a multi-task network with the proposed Adaptive Task-Relational Context (ATRC) module. The main network can have any topology, provided that the head for each task n produces both the features (F_n) for ATRC to refine and the auxiliary prediction (A_n). In our experiments we predict F_n and A_n with the main and auxiliary independent heads respectively. Within ATRC each task is routed as target task to N Context Pooling (CP) blocks (n-th row of CP blocks) and as source task to N CP blocks (n-th column). The outputs of CP blocks are concatenated for each task independently and fed through a projection module ('Task n'). The predictions P_n are obtained after processing ATRC outputs with a final layer ('Pred n'). **(b)** Dissection of a CP block, refining target task (T) features through source task (S) information. During the search stage, the CP block extracts all five contextual representations (white blocks, see Sec. 3.2) and returns a convex combination of them. After search convergence, a single context type is sampled via argmax, *i.e.*, the α_i form a one-hot vector. **Legend**: Green blocks denote modules with learned weights, red blocks denote loss functions. Best viewed in color.

search space. Although effective, architecture search with these methods can be very compute-intensive, prompting researchers to explore differentiable NAS [27, 49, 17]. Instead of a single operation, differentiable NAS uses a convex combination of several operations at a given layer, enabling gradient-based optimization of the search space by training the operation mixing weights. The primary contribution of our work is a novel multi-modal distillation module; we thus utilize existing advances in differentiable NAS [49] and a custom search space to automate the context selection for different source-target task pairs.

3. Adaptive Task-Relational Context

In this section, we describe the proposed Adaptive Task-Relational Context (ATRC) module within a general multi-task learning framework. First, we briefly outline the overall architecture, before dissecting the building blocks of the ATRC module. Finally, we discuss the employed adaptive context type search scheme.

3.1. Architecture Design

Our ATRC module can be incorporated as a refinement stage in any multi-task neural network (*e.g.*, across multiple scales). For transparency we intentionally keep the example configuration simple (see Fig. 2a): The backbone is shared among all tasks; shallow heads are used per task to generate task-specific features F_n and auxiliary predictions A_n, where $n \in \{1, ..., N\}$ indexes the task. In our basic design, we predict F_n and A_n independently, using a 3×3 Conv-BN-ReLU and 1×1 Conv-BN-ReLU-1×1 Conv layer respectively. The role of the A_n is further explained in Sec. 3.2.3.

The ATRC module refines the features F_T of each target task T by attending to the features F_n of every available task $n \in \{1, ..., N\}$ within a separate Context Pooling (CP) block for each source-target task pair. Each row of the cartesian grid of CP blocks in Fig. 2a thus serves to refine one target task T, using information from a different source task S in each column. The self-attention performed in the CP blocks on the diagonal enables the distillation module to additionally capture intra-task relationships. The outputs of all CP blocks within a row are concatenated along the channel dimension, fused with a 1×1 Conv-BN layer, concatenated with the original target task features F_T, and processed with 1×1 Conv-BN-ReLU. Lastly, the refined features are fed through a 1×1 Conv layer ('Pred T' in Fig. 2a) to obtain the final predictions P_T.

3.2. Context Pooling Block

A CP block aims to extract useful features from one source task S to augment one target task T. To this end, each CP block performs at its core a version of scaled dot-product attention, the main component of the widely successful Transformer [45]. Accordingly, the target task feature map F_T and the source task feature map F_S are first transformed to queries q, keys k and values v using 1×1 Conv-

BN-ReLU layers f_*.

$$q = f_q(F_T), \quad k = f_k(F_S), \quad v = f_v(F_S) \tag{1}$$

Throughout this paper, we assume that tensors are flattened along the spatial dimension (including q, k, v). A matrix of attention weights $\mathcal{A}$ is generated based on the pairwise similarity between q and k features. CP block outputs v' are attention-weighted combinations of v features (d_k is the channel dimension of k).

$$v' = \underbrace{\operatorname{softmax}\left(\frac{qk^\intercal}{\sqrt{d_k}}\right)}_{\mathcal{A}} v \tag{2}$$

In the multi-task setting, the attention weights can be interpreted as modeling the likelihood of feature co-occurrence [55] in transformed target (q) and source (k) task maps. The contribution of each source task pixel within the context of the target task pixel is then gated according to the estimated co-occurrence likelihood. Intuitively, co-occurrence might improve the robustness of target task predictions in ambiguous cases, *e.g.*, for T = 'semantic segmentation' and S = 'depth estimation', the context of a pixel of class 'sky' is more likely to consist of many pixels with large depth.

The attention maps in Eq. 2 model pixel interactions globally ('all-to-all'), *i.e.*, the distillation context of each pixel is unconstrained. Depending on the present source and target task combination, this might not be ideal. We therefore introduce four variants of the above attention mechanism in Sec. 3.2.1, 3.2.2, 3.2.3, each characterized by a different context definition. In Sec. 3.3 we describe how we adapt the CP block for different source-target task pairs.

3.2.1 Global Context

In this case, the distillation context of a specific target pixel is simply every pixel of the source task. Naive implementations of this approach lead to a prohibitively large memory footprint, as the complexity of computing the attention weights scales with $\mathcal{O}(L^2)$, where L is the number of pixels.

To circumvent this issue, we utilize a linearization scheme similar to [21]. In particular, we can calculate the attention map for a target pixel i using an arbitrary similarity function $\operatorname{sim}(\cdot)$ with positive domain instead of softmax.

$$v'_i = \frac{\sum_{j=1}^{L} \operatorname{sim}(q_i, k_j)\, v_j}{\sum_{j=1}^{L} \operatorname{sim}(q_i, k_j)} \tag{3}$$

This includes all kernel functions $\operatorname{sim}(q_i, k_j) = \phi(q_i)\phi(k_j)$, which allows us to shift the multiplication order: $\phi(k_j)$ and v_j can be multiplied first and reused for every $\phi(q_i)$, which reduces the overall complexity to $\mathcal{O}(L)$. In this work, we simply choose a linear kernel $\phi(x) = x$, corresponding to cosine similarity. To avoid numerical issues, we replace the ReLU activation functions in f_q and f_k of Eq. 1 with the smooth approximation $\operatorname{softplus}(x) = \log(1 + \exp(x))$.

3.2.2 Local Context

We can constrain the context to encompass only source pixels spatially close to the target pixel [36], mimicking the receptive field of a convolution. With $\mathcal{N}_b(i)$ denoting the 2D spatial neighborhood of target pixel i with extent b (we use $b = 9 \times 9$), the attention formula analogously to Eq. 2 is:

$$v'_i = \sum_{j \in \mathcal{N}_b(i)} \operatorname*{softmax}_{\mathcal{N}_b(i)}\left(\frac{q_i k_j^\intercal}{\sqrt{d_k}}\right) v_j \tag{4}$$

This operation resembles a convolution with a spatially-adaptive filter [42]—the attention map.

3.2.3 Label Context

Both the global and local relational contexts are spatially defined, *i.e.*, distillation is conducted through a spatial attention mask. *Label* context, on the other hand, is defined in label space, meaning that we (1) partition the label space into a set of disjoint label regions, (2) find a prototypical representation for each region, and (3) relate each pixel to each region prototype. This concept has been applied to semantic segmentation in [54, 53]. In this section we generalize it to any dense prediction task and explore its potential for MTL.

Partitioning the label space is straightforward for classification tasks, *i.e.*, the label regions can be equivalent to the classes. For regression tasks, however, we need to discretize the continuous label space. Consequently, we bin the values on a logarithmic scale for depth prediction and cluster predictions on the unit sphere using k-means for surface normal estimation (see Sec. D of the supplementary material for details).

We follow the approach of OCR-Net [53] for supervised learning of the region prototypes for each task n: Specifically, auxiliary prediction heads calculate the spatial maps $A_n \in \mathbb{R}^{L \times R_n}$ (see Fig. 2a), where each entry indicates the degree to which a pixel $l \in \{1, ..., L\}$ belongs to a label region $r \in \{1, ..., R_n\}$. During training, these maps are learned with ground truth supervision using a cross-entropy loss. The resulting maps A_n are normalized using spatial softmax to obtain $\hat{A}_n$, representing the spatial probability density of each label region r. In a multi-task setup, we can then choose to define the label regions in either target or source task label space:

T**-label**. In this approach, label regions are defined in target task (T) space. Source task features are spatially aggregated using target task spatial maps $\hat{A}_T$, yielding the

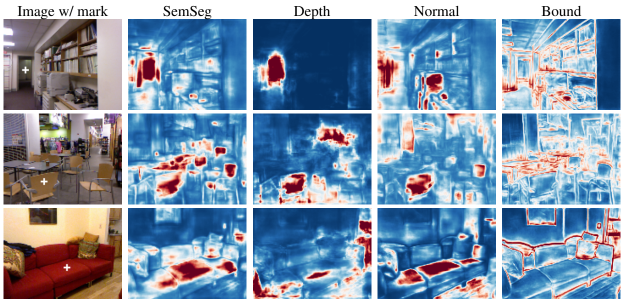


Figure 3. Heatmaps showing label context attention maps relating to the pixel marked with a white cross in the left image, *i.e.* we visualize the corresponding row of $\mathcal{A}$ in Eq. 2. For each target task we visualize the self-attention maps only.

region prototypes $p_S \in \mathbb{R}^{R_T \times C}$, where C is the source task channel dimension.

$$p_S = \hat{A}_T^\intercal F_S \tag{5}$$

p_S is then substituted for F_S in Eq. 1 to obtain k and v.

S**-label.** Alternatively, source task features can also be aggregated via source task (S) spatial maps $\hat{A}_S$, by substituting $\hat{A}_S$ for $\hat{A}_T$ in Eq. 5.

The key difference between the two approaches is best illustrated with an example: Assuming target task semantic segmentation and source task depth estimation, the T-label context groups depth features according to object class and makes each target pixel attend to the prototypical depth features for each object class (*e.g.*, the representative depth feature of all 'car' pixels). Conversely, the S-label context simply groups depth features according to their depth, enabling semantic features to interact with entire depth regions.

We visualize example self-attention maps for a single target pixel (white cross) of a trained label context distillation model in Fig. 3. The maps illustrate that the model learns to focus on context pixels within distinct label regions.

3.3. Automated Context Type Selection

While all presented context types could help improve target task features, some might be more effective than others in specific scenarios. Therefore, CP blocks are designed to tailor their context type (attention mechanism) to the present source-target task pair. In this paper we opt for differentiable NAS techniques to automatically select a single context type for each CP block, by optimizing a supergraph encompassing all options (see Fig. 2b). However, a CP block is not limited to a single context type *per se* and could instead refine predictions given a combination of context types in a static [29, 48] or even dynamic [18] fashion.

Our search space consists of five candidates in each CP block: *global*, *local*, *T-label*, *S-label*, and a *none* operation. The *none* operation simply severs the information flow between two tasks, which can prevent task interference, a common problem in MTL [24, 20]. Operation selection in a CP block j can be formulated as a multiplication of all candidates O_j with a one-hot vector Z_j sampled from the categorical distribution $p_{\alpha_j}(Z_j)$.

$$\tilde{O}_j = Z_j^\intercal O_j \tag{6}$$

Continuous relaxation of the search space (while maintaining this sampling process) is achieved through the Gumbel-Softmax gradient estimator [32, 19], yielding a softened one-hot random variable $\hat{Z}_j$.

$$\hat{Z}_j^{(i)} = \frac{\exp\left(\left(\log \alpha_j^{(i)} + G_j^{(i)}\right)/\lambda\right)}{\sum_{u=1}^{5} \exp\left(\left(\log \alpha_j^{(u)} + G_j^{(u)}\right)/\lambda\right)} \tag{7}$$

$G_j^{(u)} \sim \mathrm{Gumbel}(0, 1)$ is a Gumbel random variable, and λ is the $\mathrm{softmax}$ temperature. In our case, the architecture parameters α are updated in the same round of backpropagation as the network weights (single-level optimization). A more detailed discussion of Gumbel-Softmax for differentiable NAS is provided in [49].

Empirically, samples from α-distributions trained with Gumbel-Softmax exhibit large variance after convergence, leading to unstable evaluation of sampled subgraphs. We thus use a two-pronged strategy to counteract this problem: (1) Similarly to [14], we adopt entropy regularization on $p_{\alpha_j}(Z_j)$ to explicitly control the sampling variance. Instead of the commonly employed candidate operation pretraining, we can simply start the architecture search from scratch with a negative regularization weight to enforce a uniform α-distribution. The weight is gradually increased to a positive value during training to ultimately incentivize low-entropy solutions, which imply a low variance as the architecture is sampled from the supergraph. (2) We stop the architecture sampling process in CP block j completely once p_{α_j} has reached a low-entropy solution. After a defined threshold is surpassed, we fix the block selection procedure in j using argmax. Using this strategy, we obtain high-performing architectures directly during the search stage (see Fig. 4), demonstrating that our search objective is well defined. Nevertheless, for a fair comparison, we still retrain the discovered architectures from scratch—as is common practice [27, 49].

4. Experiments

We briefly review the experimental setup, before presenting empirical studies. Training details are provided in Sec. A of the supplementary material and reference code is available at `https://github.com/brdav/atrc`.

4.1. Setup

Datasets. Experiments are conducted on two widely-used dense prediction datasets: (1) *NYUD-v2* [40], which consists

Distillation module	Resource		SemSeg ↑		Depth ↓		Normal ↓		Bound ↑		Δ_m [%] ↑
	Params (M)	MAdds (G)	mean	std.	mean	std.	mean	std.	mean	std.	
None (single task baseline)	16.09	40.93	38.02	0.14	0.6104	0.0041	20.94	0.08	76.22	0.07	0.00
None (multi-task baseline)	4.52	17.59	36.35	0.26	0.6284	0.0034	21.02	0.06	76.36	0.05	-1.89
Cross-Stitch [35]	4.52	17.59	36.34	0.55	0.6290	0.0051	20.88	0.04	76.38	0.07	-1.75
PAP [56]	4.54	53.04	36.72	0.31	0.6178	0.0065	20.82	0.03	76.42	0.07	-0.95
PSD [57]	4.71	21.10	36.69	0.55	0.6246	0.0036	20.87	0.07	76.42	0.13	-1.30
PAD-Net A [50] / NDDR-CNN [15]	4.59	18.68	36.72	0.31	0.6288	0.0037	20.89	0.02	76.32	0.07	-1.51
PAD-Net B [50]	5.02	25.18	36.70	0.16	0.6264	0.0021	20.85	0.03	76.50	0.06	-1.33
PAD-Net C [50] / MTI-Net [44]	5.50	32.42	36.61	0.15	0.6270	0.0048	20.85	0.03	76.38	0.07	-1.44
Global relational context	4.73	21.43	38.30	0.65	0.6007	0.0073	20.60	0.07	76.26	0.05	1.00
Local relational context	4.73	22.19	36.79	0.29	0.6260	0.0044	20.91	0.06	76.44	0.05	-1.34
T-label relational context	5.06	25.91	38.88	0.31	0.6059	0.0014	20.48	0.05	76.30	0.06	1.33
S-label relational context	5.06	25.91	38.33	0.64	0.6006	0.0019	20.56	0.06	76.26	0.05	1.07
ATRC (ours)	5.06	25.76	38.90	0.43	0.6010	0.0046	20.48	0.02	76.34	0.12	**1.56**

Table 1. Controlled distillation module comparison on NYUD-v2 with a HRNet18 backbone. For all models except the single task baseline, a shared encoder and small task-specific heads are used (Sec. 3.1). We insert the different distillation modules before the final prediction layer.

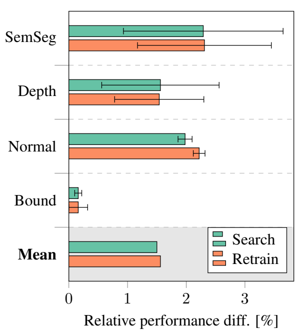


Figure 4. Performance comparison of the models sampled from the supergraph at the end of the context type search *vs*. after retraining. The chart shows mean and std. of the relative performance improvement w.r.t. single task (ST) models: $(M_m - M_{ST})/M_{ST}$ for model m and 'higher = better' metric M, and *vice versa* for 'lower = better'.

of 795 training and 654 testing images of indoor scenes, with annotations for semantic segmentation ('SemSeg'), depth estimation ('Depth'), surface normal estimation ('Normal'), and boundary detection ('Bound'). (2) *PASCAL-Context* [8], a split of the larger PASCAL dataset [12], providing 4998 training and 5105 testing images, labeled for semantic segmentation, human parts segmentation ('PartSeg'), saliency estimation ('Sal'), surface normal estimation, and boundary detection. We use the distilled saliency and surface normal labels of [33].

Backbones. We test our framework using several backbones: HRNetV2-W18-small (HRNet18), HRNetV2-W48 (HRNet48) [46], and ResNet-50 [16].

Metrics. We evaluate 'Semseg' and 'PartSeg' with mean intersection over union, 'Depth' with root mean square error, 'Normal' with mean angular error, 'Sal' with maximum F-measure as in [1], and 'Bound' with the optimal-dataset-scale F-measure of [34]. All experiments in this paper are repeated five times; the mean is reported for every metric (in Table 1 also the standard deviation). To quantify overall multi-task performance for N tasks, we adopt the average per-task performance drop (Δ_m) with respect to single task baselines b for model m [33]: $\Delta_m = \frac{1}{N}\sum_{i=1}^{N}(-1)^{\gamma_i}(M_{m,i} - M_{b,i})/M_{b,i}$. $\gamma_i = 1$ if lower is better for metric M_i and $\gamma_i = 0$ otherwise.

4.2. Distillation Module Benchmarking

In Table 1 we conduct a series of controlled experiments to assess the effectiveness of different distillation modules fairly. Using a HRNet18 backbone, we alter the MTL architecture design described in Sec. 3.1 only by replacing the ATRC module with other distillation modules. For the baselines, no distillation module is used.

As expected, all investigated distillation modules outperform the trivial multi-task baseline in terms of multi-task performance Δ_m. Furthermore, most relational context modules fare significantly better than their alternatives. Excepting local relational context, augmenting the multi-task network with relational context beats the single task baseline while maintaining a far lower computational footprint.

Table 1 also reveals that no single relational context type dominates for every task. This suggests that a more fine-grained context customization for each individual source-target task pair could improve overall performance. Indeed, applying our automated context type selection (Sec. 3.3), ATRC, produces the best result in multi-task performance.

Fig. 5 visualizes the resource cost of the various distillation modules by plotting the multi-task performance *vs*. number of parameters and multiply-add operations (MAdds). The computational overhead of the relational context modules—and most other distillation modules—remains low compared to single task networks. Our ATRC combines the benefits of all the relational context modules by maximizing performance while remaining bounded in terms of resource cost.

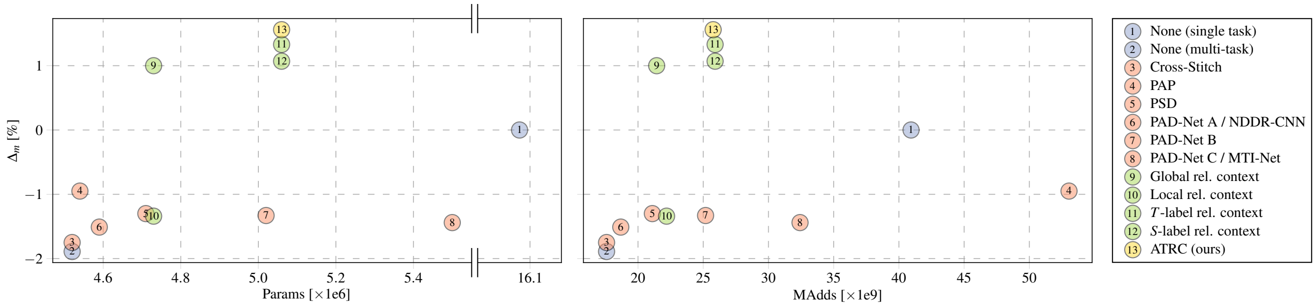


Figure 5. Distillation module resource analysis using an HRNet18 backbone on NYUD-v2. We plot multi-task performance Δ_m *vs*. number of parameters (left) and MAdds (right) for multi-task models with different distillation modules inserted before the final prediction layer.

4.3. Comparison with State-of-the-Art

To validate the proposed ATRC module, we present experimental comparisons with the following baselines across a number of scenarios: separate single task networks, multi-task network (shared backbone; task-specific heads; no distillation) and the state-of-the-art MTI-Net [44]. Tables 2 and 3 display the results obtained on the NYUD-v2 dataset, using HRNet18 and HRNet48 backbones respectively, while Table 4 shows PASCAL-Context results using HRNet18. MTI-Net uses a large-scale decoder head consisting of two separate stages: the Feature Propagation Module (FPM) and a multi-scale multi-modal distillation module (the analog of our ATRC module). To ensure a fair comparison, we apply our method on both the basic architecture described in Sec. 3.1, as well as the backbone complemented with the FPM (+174% and +79% in number of parameters for HRNet18 and HRNet48 respectively).

In all investigated cases, ATRC enhances performance significantly compared to the multi-task baseline. Furthermore, our method combined with the FPM consistently outperforms MTI-Net, even though MTI-Net applies multi-modal distillation on four scales, while we only distill on the largest scale (causing our model to be more parameter efficient, *e.g.*, -22% in Table 2). This implies that task interactions can be adequately captured at a single scale for distillation, provided that the backbone is able to extract and fuse multi-scale information effectively (like HRNet).

Overall, the multi-task approaches are less effective compared to single task baselines on the PASCAL-Context dataset. This finding is in agreement with other works [33, 44] and could be attributed to the larger and more diverse task dictionary. Nevertheless, the ranking order of the multi-task approaches in terms of multi-task performance remains consistent with the results obtained for NYUD-v2.

4.4. Source Task Importance

The simple design of the proposed ATRC module allows us to investigate the importance of each source-target task connection ($\widehat{=}$ CP block) for the final predictions of fitted models. To this end, we adapt permutation feature importance [4] to our setting. We can determine the importance of a CP block by recording the drop in multi-task performance Δ_m when the output of that block is randomly shuffled over the dataset. To get a more reliable estimate, this procedure is repeated multiple times with different permutations. Neglecting feature multicollinearity, the average drop in Δ_m provides an estimate of how strongly the fitted model depends on the inspected source task for the corresponding target task prediction. We use held-out data in this experiment to assess the importance for generalization power.

Fig. 6 visualizes the results for NYUD-v2. The inspection reveals that self-attention remains the most important distillation connection for three out of four tasks. However, depth

Model	FPM	SemSeg ↑	Depth ↓	Normal ↓	Bound ↑	Δ_m [%] ↑
Single task		38.02	0.6104	20.94	76.22	0.00
Multi-task		36.35	0.6284	21.02	76.36	-1.89
MTI-Net [44]	✓	39.89	0.5824	20.57	76.60	2.94
ATRC (ours)		38.90	0.6010	20.48	76.34	1.56
	✓	40.80	0.5826	20.51	76.50	**3.57**

Table 2. NYUD-v2 performance comparison, using a HRNet18 backbone. FPM = Feature Propagation Module [44].

Model	FPM	SemSeg ↑	Depth ↓	Normal ↓	Bound ↑	Δ_m [%] ↑
Single task		45.87	0.5397	20.09	77.34	0.00
Multi-task		41.96	0.5543	20.36	77.62	-3.05
MTI-Net [44]	✓	45.97	0.5365	20.27	77.86	0.15
ATRC (ours)		46.27	0.5495	20.20	77.60	-0.28
	✓	46.33	0.5363	20.18	77.94	**0.49**

Table 3. NYUD-v2 performance comparison, using a HRNet48 backbone. FPM = Feature Propagation Module [44].

Model	FPM	SemSeg ↑	PartSeg ↑	Sal ↑	Normal ↓	Bound ↑	Δ_m [%] ↑
Single task		62.23	61.66	85.08	13.69	73.06	**0.00**
Multi-task		51.48	57.23	83.43	14.10	69.76	-6.77
MTI-Net [44]	✓	61.70	60.18	84.78	14.23	70.80	-2.12
ATRC (ours)		57.89	57.33	83.77	13.99	69.74	-4.45
	✓	62.69	59.42	84.70	14.20	70.96	-1.98

Table 4. PASCAL-Context performance comparison, using a HR-Net18 backbone. FPM = Feature Propagation Module [44].

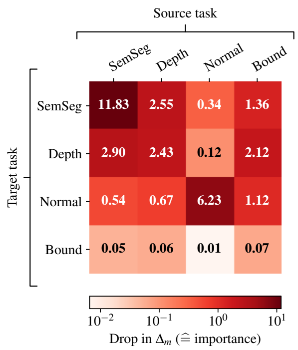


Figure 6. Source task importance; measured by permutation testing of fitted ATRC models on NYUD-v2. The contribution of a source task in the distillation is gauged by the drop in multi-task performance Δ_m as the output of the corresponding source-target task distillation is randomly permuted. The values shown in the matrix are mean percentage drops in Δ_m.

estimation seems to rely more strongly on semantic segmentation source features, corroborating empirical evidence in the literature that depth estimation can be improved significantly using semantic predictions [50]. Overall, boundary detection profits little from multi-modal distillation according to this analysis, which is consistent with the lack of noteworthy performance gain for this task in Table 1. We hypothesize that this could be due to the large discrepancy between the loss (we follow others [33, 44, 20] and use balanced cross entropy) and metric for this task. A more tailored loss function such as [23] might help in this case.

Source task importance scores are linearly correlated with the search algorithm reliability—albeit weakly (Pearson correlation coefficient of 0.43). Notably, we observe 100% reliability for the three most important source-target task connections of Fig. 6. This suggests that the search algorithm is more consistent for important decisions. We quantify search algorithm reliability using percentage agreement in candidate selection between all search run pairs (does not account for chance agreement, see Sec. F).

4.5. Complementary Methods

To demonstrate its flexibility, we combine our ATRC module with (1) the contextual Atrous Spatial Pyramid Pooling (ASPP) module of [7] and (2) automatic backbone branching via Branched Multi Task Architecture Search (BMTAS) [5]. For these experiments, we use a dilated ResNet-50 backbone (output stride 16) with a skip connection at stride 4, and fully convolutional task-specific heads.

ASPP is a popular multi-scale context aggregation module leveraging dilated convolutions. We insert a separate ASPP module before each task-specific head. Table 5 shows that ATRC also improves the performance of the ASPP-augmented network, indicating that the two context aggregation stages are complementary to some extent. Interestingly, the proportions of selected relational context types in the ATRC search change drastically with ASPP, as illustrated in Fig. 7: The proportion of local context rises from 0% (w/o ASPP) to 41.6% (w/ ASPP), demonstrating that ATRC adapts the context types given the nature of different backbones (*e.g.*, the enhanced receptive field of ASPP is better complemented with local information).

Model	ATRC	SemSeg ↑	PartSeg ↑	Sal ↑	Normal ↓	Bound ↑	Δ_m [%] ↑
Single task		56.65	62.67	80.62	14.66	74.00	0.00
Multi-task		50.78	59.37	78.99	15.16	71.18	-4.97
	✓	62.99	59.79	82.25	14.67	71.20	**0.95**
ASPP [7]		62.70	59.98	83.81	14.34	71.28	1.77
	✓	63.60	60.23	83.91	14.30	70.86	**2.13**
BMTAS [5]		56.37	62.54	79.91	14.60	72.83	-0.55
	✓	67.67	62.93	82.29	14.24	72.42	**4.53**

Table 5. PASCAL-Context performance of ASPP [7] and BMTAS [5] when supplemented with our ATRC. For ASPP, we insert an ASPP module at the beginning of each task-specific head. For BMTAS, we use their method to find a branched backbone (instead of fully shared). ATRC is complementary to both approaches. Experiments are based on a dilated ResNet-50 backbone.

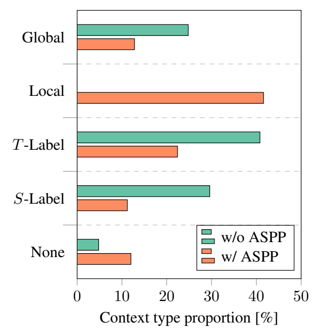


Figure 7. Proportions of selected context types over five search runs, for architectures without and with an ASPP module [7] inserted before the ATRC module. The change in proportion of the local context indicates that ATRC adapts to better complement the new backbone. This experiment was conducted on the PASCAL-Context dataset using a ResNet-50 based architecture.

Branched networks are a hard parameter sharing MTL strategy and, as such, complementary to multi-modal distillation (see Sec. 2). We show this by applying our method in combination with a branched backbone configuration, determined through the NAS-based BMTAS. The results in Table 5 demonstrate that ATRC improves performance also for branched multi-task networks.

5. Conclusion

We presented ATRC, a novel multi-modal distillation module which exploits inter- and intra-task relationships to refine pixel-wise predictions. The proposed approach leverages scaled dot-product attention to enrich the features of a target task through contextual source task features, while explicitly factoring in tasks' relations. We formulate four relational context types for multi-modal distillation (*global*, *local*, *T-label*, and *S-label* context) and detail an algorithm which customizes the context type for every given source-target task pair. Experimental analyses on NYUD-v2 and PASCAL-Context benchmarks indicate that our ATRC module outperforms comparable multi-modal distillation modules established in the literature. Overall, the presented framework shows great promise for multi-task dense prediction and opens the door for future research in customized task-relational context descriptions.

References

[1] Radhakrishna Achanta, Sheila Hemami, Francisco Estrada, and Sabine Susstrunk. Frequency-tuned salient region detection. In *CVPR*, 2009.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

[3] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *ICCV*, 2019.

[4] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[5] David Bruggemann, Menelaos Kanakis, Stamatios Georgoulis, and Luc Van Gool. Automated search for resource-efficient branched multi-task networks. In *BMVC*, 2020.

[6] Richard Caruana. Multitask learning: A knowledge-based source of inductive bias. In *ICML*, 1993.

[7] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

[8] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *CVPR*, 2014.

[9] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. a^2-nets: Double attention networks. In *NeurIPS*, 2018.

[10] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *ICML*, 2018.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[12] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010.

[13] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *CVPR*, 2019.

[14] Yuan Gao, Haoping Bai, Zequn Jie, Jiayi Ma, Kui Jia, and Wei Liu. Mtl-nas: Task-agnostic neural architecture search towards general-purpose multi-task learning. In *CVPR*, 2020.

[15] Yuan Gao, Jiayi Ma, Mingbo Zhao, Wei Liu, and Alan L Yuille. Nddr-cnn: Layerwise feature fusing in multi-task cnns by neural discriminative dimensionality reduction. In *CVPR*, 2019.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[17] Shoukang Hu, Sirui Xie, Hehui Zheng, Chunxiao Liu, Jianping Shi, Xunying Liu, and Dahua Lin. Dsnas: Direct neural architecture search without parameter retraining. In *CVPR*, 2020.

[18] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[19] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017.

[20] Menelaos Kanakis, David Bruggemann, Suman Saha, Stamatios Georgoulis, Anton Obukhov, and Luc Van Gool. Reparameterizing convolutions for incremental multi-task learning without task interference. In *ECCV*, 2020.

[21] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *ICML*, 2020.

[22] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, 2018.

[23] Iasonas Kokkinos. Pushing the boundaries of boundary detection using deep learning. In *ICLR*, 2016.

[24] Iasonas Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *CVPR*, 2017.

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[26] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[27] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *ICLR*, 2019.

[28] Shikun Liu, Edward Johns, and Andrew J Davison. End-to-end multi-task learning with attention. In *CVPR*, 2019.

[29] Yong Liu and Xin Yao. Ensemble learning via negative correlation. *Neural networks*, 12(10):1399–1404, 1999.

[30] David G Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999.

[31] Yongxi Lu, Abhishek Kumar, Shuangfei Zhai, Yu Cheng, Tara Javidi, and Rogerio Feris. Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification. In *CVPR*, 2017.

[32] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *ICLR*, 2017.

[33] Kevis-Kokitsi Maninis, Ilija Radosavovic, and Iasonas Kokkinos. Attentive single-tasking of multiple tasks. In *CVPR*, 2019.

[34] David R Martin, Charless C Fowlkes, and Jitendra Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *TPAMI*, 26(5):530–549, 2004.

[35] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *CVPR*, 2016.

[36] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. In *NeurIPS*, 2019.

[37] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *AAAI*, 2019.

[38] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V Le, and Alexey Kurakin. Large-scale evolution of image classifiers. In *ICML*, 2017.

[39] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *NeurIPS*, 2018.

[40] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[42] Hang Su, Varun Jampani, Deqing Sun, Orazio Gallo, Erik Learned-Miller, and Jan Kautz. Pixel-adaptive convolutional neural networks. In *CVPR*, 2019.

[43] Simon Vandenhende, Stamatios Georgoulis, Bert De Brabandere, and Luc Van Gool. Branched multi-task networks: deciding what layers to share. In *BMVC*, 2020.

[44] Simon Vandenhende, Stamatios Georgoulis, and Luc Van Gool. Mti-net: Multi-scale task interaction networks for multi-task learning. In *ECCV*, 2020.

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[46] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *TPAMI*, 2020.

[47] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018.

[48] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.

[49] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. Snas: Stochastic neural architecture search. In *ICLR*, 2019.

[50] Dan Xu, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Padnet: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing. In *CVPR*, 2018.

[51] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015.

[52] Minghao Yin, Zhuliang Yao, Yue Cao, Xiu Li, Zheng Zhang, Stephen Lin, and Han Hu. Disentangled non-local neural networks. In *ECCV*, 2020.

[53] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. In *ECCV*, 2020.

[54] Fan Zhang, Yanqin Chen, Zhihang Li, Zhibin Hong, Jingtuo Liu, Feifei Ma, Junyu Han, and Errui Ding. Acfnet: Attentional class feature network for semantic segmentation. In *ICCV*, 2019.

[55] Hang Zhang, Han Zhang, Chenguang Wang, and Junyuan Xie. Co-occurrent features in semantic segmentation. In *CVPR*, 2019.

[56] Zhenyu Zhang, Zhen Cui, Chunyan Xu, Yan Yan, Nicu Sebe, and Jian Yang. Pattern-affinitive propagation across depth, surface normal and semantic segmentation. In *CVPR*, 2019.

[57] Ling Zhou, Zhen Cui, Chunyan Xu, Zhenyu Zhang, Chaoqun Wang, Tong Zhang, and Jian Yang. Pattern-structure diffusion for multi-task learning. In *CVPR*, 2020.

[58] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In *ICLR*, 2017.

[59] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *CVPR*, 2018.