End-to-End Multi-Task Learning with Attention

Shikun Liu Edward Johns Andrew J. Davison
Department of Computing, Imperial College London
{shikun.liu17, e.johns, a.davison}@imperial.ac.uk

Abstract

We propose a novel multi-task learning architecture, which allows learning of task-specific feature-level attention. Our design, the Multi-Task Attention Network (MTAN), consists of a single shared network containing a global feature pool, together with a soft-attention module for each task. These modules allow for learning of task-specific features from the global features, whilst simultaneously allowing for features to be shared across different tasks. The architecture can be trained end-to-end and can be built upon any feed-forward neural network, is simple to implement, and is parameter efficient. We evaluate our approach on a variety of datasets, across both image-to-image predictions and image classification tasks. We show that our architecture is state-of-the-art in multi-task learning compared to existing methods, and is also less sensitive to various weighting schemes in the multi-task loss function. Code is available at https://github.com/lorenmt/mtan.

1. Introduction

Convolutional Neural Networks (CNNs) have seen great success in a range of computer vision tasks, including image classification [11], semantic segmentation [1], and style transfer [13]. However, these networks are typically designed to achieve only one particular task. For more complete vision systems in real-world applications, a network which can perform multiple tasks simultaneously is far more desirable than building a set of independent networks, one for each task. This is more efficient not only in terms of memory and inference speed, but also in terms of data, since related tasks may share informative visual features.

This type of learning is called Multi-Task Learning (MTL) [20, 14, 6], and in this paper we present a novel architecture for MTL based on feature-level attention masks, which add greater flexibility to share complementary features. Compared to standard single-task learning, training multiple tasks whilst successfully learning a shared representation poses two key challenges:

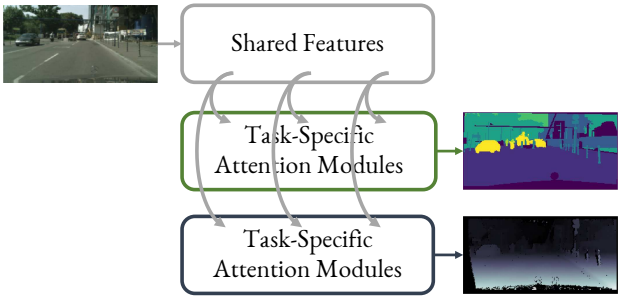


Figure 1: Overview of our proposal MTAN. The shared network takes input data and learns task-shared features, whilst each attention network learns task-specific features, by applying attention modules to the shared network.

i) **Network Architecture (how to share)**: A multi-task learning architecture should express both *task-shared* and *task-specific* features. In this way, the network is encouraged to learn a generalisable representation (to avoid over-fitting), whilst also providing the ability to learn features tailored to each task (to avoid under-fitting).

ii) **Loss Function (how to balance tasks)**: A multi-task loss function, which weights the relative contributions of each task, should enable learning of all tasks with equal importance, without allowing easier tasks to dominate. Manual tuning of loss weights is tedious, and it is preferable to automatically learn the weights, or design a network which is robust to different weights.

However, most prior MTL approaches focus on only one of these two challenges, whilst maintaining a standard implementation of the other. In this paper, we introduce a unified approach which addresses both challenges cohesively, by designing a novel network which (i) enables both task-shared and task-specific features to be learned automatically, and consequently (ii) learns an inherent robustness to the choice of loss weighting scheme.

The proposed network, which we call the Multi-Task Attention Network (MTAN) (see Figure 1), is composed of a single shared network, which learns a global feature pool containing features across all tasks. Then for each task,

rather than learning directly from the shared feature pool, a soft attention mask is applied at each convolution block in the shared network. In this way, each attention mask automatically determines the importance of the shared features for the respective task, allowing learning of both task-shared and task-specific features in a self-supervised, end-to-end manner. This flexibility enables much more expressive combinations of features to be learned for generalisation across tasks, whilst still allowing for discriminative features to be tailored for each individual task. Furthermore, automatically choosing which features to share and which to be task specific allows for a highly efficient architecture with far fewer parameters than multi-task architectures which have explicit separation of tasks [26, 20].

MTAN can be built on any feed-forward neural network depending on the type of tasks. We first evaluate MTAN with SegNet [1], an encoder-decoder network on the tasks of semantic segmentation and depth estimation on the outdoor CityScapes dataset [4], and then with an additional task of surface normal prediction on the more challenging indoor dataset NYUv2 [21]. We also test our approach with a different backbone architecture, Wide Residual Network [31], on the recently proposed Visual Decathlon Challenge [23], to solve 10 individual image classification tasks. Results show that MTAN outperforms several baselines and is competitive with the state-of-the-art for multi-task learning, whilst being more parameter efficient and therefore scaling more gracefully with the number of tasks. Furthermore, our method shows greater robustness to the choice of weighting scheme in the loss function compared to baselines. As part of our evaluation of this robustness, we also propose a novel weighting scheme, Dynamic Weight Average (DWA), which adapts the task weighting over time by considering the rate of change of the loss for each task.

2. Related Work

The term Multi-Task Learning (MTL) has been broadly used in machine learning [2, 8, 6, 17], with similarities to transfer learning [22, 18] and continual learning [29]. In computer vision, multi-task learning has been used to for learning similar tasks such as image classification in multiple domains [23], pose estimation and action recognition [9], and dense prediction of depth, surface normals, and semantic classes [20, 7]. In this paper, we consider two important aspects of multi-task learning: how can a good multi-task network architecture be designed, and how to balance feature sharing in multi-task learning across all tasks?

Most multi-task learning network architectures for computer vision are designed based on existing CNN architectures. For example, Cross-Stitch Networks [20] contain one standard feed-forward network per task, with cross-stitch units to allow features to be shared across tasks. The self-supervised approach of [6], based on the ResNet101 architecture [30], learns a regularised combination of features from different layers of a single shared network. UberNet [16] proposes an image pyramid approach to process images across multiple resolutions, where for each resolution, additional task-specific layers are formed top of the shared VGG-Net [27]. The Progressive Networks [26] uses a sequence of incrementally-trained networks to transfer knowledge between tasks. However, architectures such as Cross-Stitch Networks and Progressive Networks require a large number of network parameters, and scale linearly with the number of tasks. In contrast, our model requires only a rough 10% increase in parameters for per learning task.

On the balancing of feature sharing in multi-task learning, there is extensive experimental analysis in [20, 14], with both papers arguing that different amounts of sharing and weighting tend to work best for different tasks. One example of weighting tasks appropriately is with the use of weight uncertainty [14], which modifies the loss functions in multi-task learning using task uncertainty. Another method is that of GradNorm [3], which manipulates gradient norms over time to control the training dynamics. As an alternative to using task losses to determine task difficulties, Dynamic Task Prioritisation [10] encourages prioritisation of difficult tasks directly using performance metrics such as accuracy and precision.

3. Multi-Task Attention Network

We now introduce our novel multi-task learning architecture, the Multi-Task Attention Network (MTAN). Whilst the architecture can be incorporated into any feed-forward network, in the following section we demonstrate how to build MTAN upon an encoder-decoder network, SegNet [1]. This example configuration allows for image-to-image dense pixel-level prediction, such as semantic segmentation, depth estimation, and surface normal prediction.

3.1. Architecture Design

MTAN consists of two components: a single shared network, and K task-specific attention networks. The shared network can be designed based on the particular task, whilst each task-specific network consists of a set of attention modules, which link with the shared network. Each attention module applies a soft attention mask to a particular layer of the shared network, to learn task-specific features. As such, the attention masks can be considered as feature selectors from the shared network, which are automatically learned in an end-to-end manner, whilst the shared network learns a compact global feature pool across all tasks.

Figure 2 shows a detailed visualisation of our network based on VGG-16 [27], illustrating the encoder half of SegNet. The decoder half of SegNet is then symmetric to VGG-16. As shown, each attention module learns a soft attention mask, which itself is dependent on the features in the shared

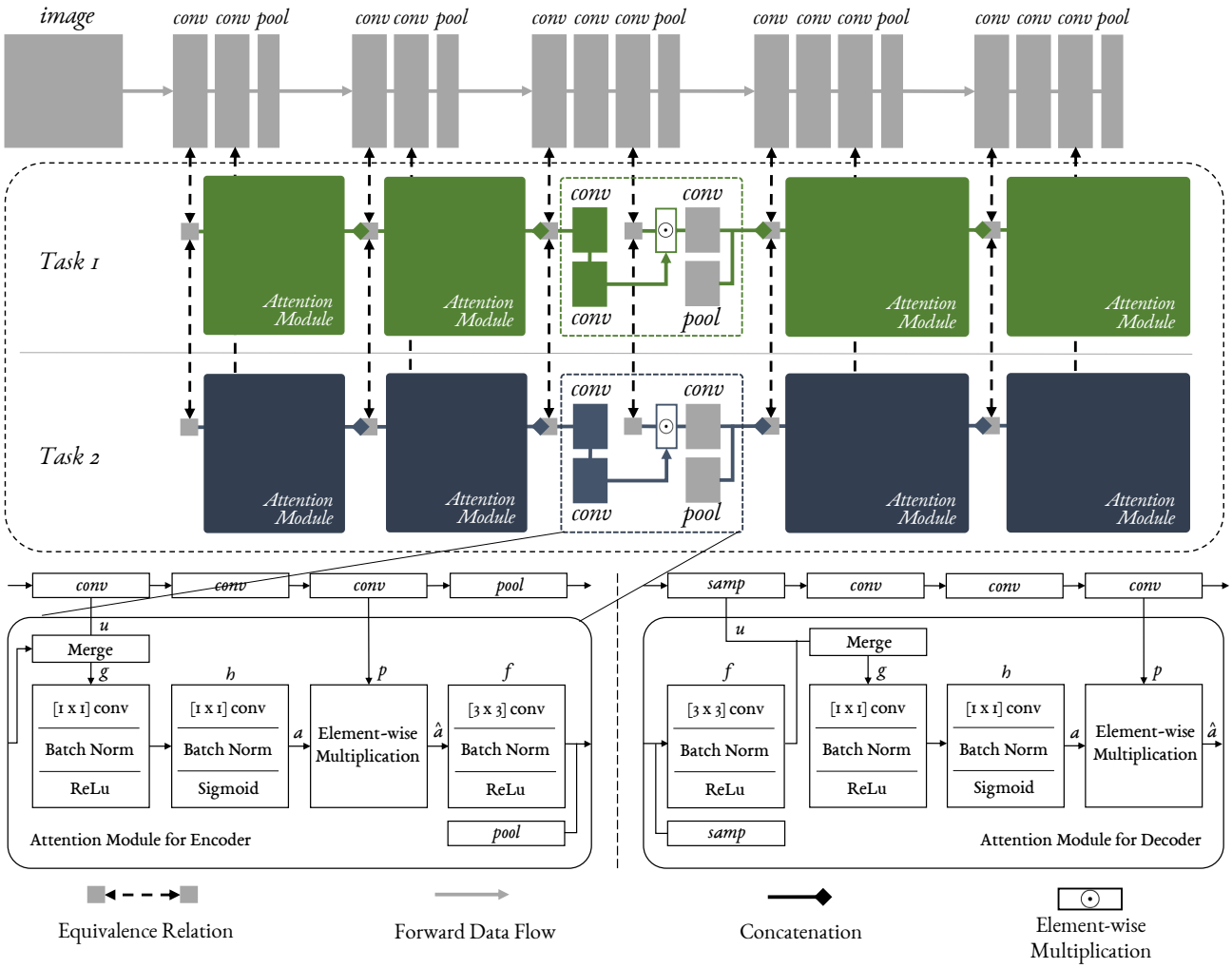


Figure 2: Visualisation of MTAN based on VGG-16, showing the encoder half of SegNet (with the decoder half being symmetrical to the encoder). Task one (green) and task two (blue) have their own set of attention modules, which link with the shared network (grey). The middle attention module has its structure exposed for visualisation, which is further expanded in the bottom section of the figure, showing both the encoder and decoder versions of the module. All attention modules have the same design, although their weights are individually learned.

network at the corresponding layer. Therefore, the features in the shared network, and the soft attention masks, can be learned jointly to maximise the generalisation of the shared features across multiple tasks, whilst simultaneously maximising the task-specific performance due to the attention masks.

3.2. Task Specific Attention Module

The attention module is designed to allow the task-specific network to learn task-related features, by applying a soft attention mask to the features in the shared network, with one attention mask per task per feature channel. We denote the shared features in the j^{th} block of the shared network as $p^{(j)}$, and the learned attention mask in this layer for task i as $a_i^{(j)}$. The task-specific features $\hat{a}_i^{(j)}$ in this layer, are then computed by element-wise multiplication of the attention masks with the shared features:

$$\hat{a}_i^{(j)} = a_i^{(j)} \odot p^{(j)} \,, \tag{1}$$

where $\odot$ denotes element-wise multiplication.

As shown in Figure 2, the first attention module in the encoder takes as input only features in the shared network. But for subsequent attention modules in block j, the input is formed by a concatenation of the shared features $u^{(j)}$, and the task-specific features from the previous layer $\hat{a}_i^{(j-1)}$:

$$a_i^{(j)} = h_i^{(j)}\left(g_i^{(j)}\left(\left[u^{(j)}; f^{(j)}\left(\hat{a}_i^{(j-1)}\right)\right]\right)\right), \; j \geq 2 \tag{2}$$

Here, $f^{(j)}, g_i^{(j)}, h_i^{(j)}$ are convolutional layers with batch normalisation, following a non-linear activation. Both $g_i^{(j)}$ and $h_i^{(j)}$ are composed of $[1 \times 1]$ kernels presenting the i^{th} task-specific attention mask in block j. $f^{(j)}$ is composed of $[3 \times 3]$ kernels representing a shared feature extractor for passing to another attention module, following by a pooling or sampling layer to match the corresponding resolution.

The attention mask, following a sigmoid activation to ensure $a_i^{(j)} \in [0, 1]$, is learned in a self-supervised fashion with back-propagation. If $a_i^{(j)} \to 1$ such that the mask becomes an identity map, the attended feature maps are equivalent to global feature maps and the tasks share all the features. Therefore, we expect the performance to be no worse than that of a shared multi-task network, which splits into individual tasks only at the end of the network, and we show results demonstrating this in Section 4.

3.3. The Model Objective

In general multi-task learning with K tasks, input $\mathbf{X}$ and task-specific labels $\mathbf{Y}_i, i = 1, 2, \cdots, K$, the loss function is defined as,

$$\mathcal{L}_{tot}(\mathbf{X}, \mathbf{Y}_{1:K}) = \sum_{i=1}^{K} \lambda_i \mathcal{L}_i(\mathbf{X}, \mathbf{Y}_i). \tag{3}$$

This is the linear combination of task-specific losses $\mathcal{L}_i$ with task weightings λ_i. In our experiments, we study the effect of different weighting schemes on various multi-task learning approaches.

For image-to-image prediction tasks, we consider each mapping from input data $\mathbf{X}$ to a set of labels $\mathbf{Y}_i$ as one task with total three tasks for evaluation. In each loss function, $\hat{\mathbf{Y}}$ represents the network's prediction, and $\mathbf{Y}$ represents the ground-truth label.

- For semantic segmentation, we apply a pixel-wise cross-entropy loss for each predicted class label from a depth-softmax classifier.

$$\mathcal{L}_1(\mathbf{X}, \mathbf{Y}_1) = -\frac{1}{pq} \sum_{p,q} \mathbf{Y}_1(p, q) \log \hat{\mathbf{Y}}_1(p, q). \tag{4}$$

- For depth estimation, we apply an L_1 norm comparing the predicted and ground-truth depth. We use true depth for the NYUv2 indoor scene dataset, and inverse depth in CityScapes outdoor scene dataset as standard, which can more easily represent points at infinite distances, such as the sky:

$$\mathcal{L}_2(\mathbf{X}, \mathbf{Y}_2) = \frac{1}{pq} \sum_{p,q} |\mathbf{Y}_2(p, q) - \hat{\mathbf{Y}}_2(p, q)|. \tag{5}$$

- For surface normals (only available in NYUv2), we apply an element-wise dot product at each normalised pixel with the ground-truth map:

$$\mathcal{L}_3(\mathbf{X}, \mathbf{Y}_3) = -\frac{1}{pq} \sum_{p,q} \mathbf{Y}_3(p, q) \cdot \hat{\mathbf{Y}}_3(p, q). \tag{6}$$

For image classification tasks, we consider each dataset as one task for which each dataset represents each individual classification task for one domain. We apply standard cross-entropy loss for all classification tasks.

4. Experiments

In this section, we evaluate our proposed method on two types of tasks: one-to-many predictions for image-to-image regression tasks in Section 4.1 and many-to-many predictions for image classification tasks (Visual Decathlon Challenge) in Section 4.2.

4.1. Image-to-Image Prediction (One-to-Many)

In this section, we evaluate MTAN built upon SegNet [1] on image-to-image prediction tasks. We first introduce the datasets used for validation in Section 4.1.1, and several baselines for comparison in Section 4.1.2. In Section 4.1.3, we introduce a novel adaptive weighting method, and in Section 4.1.4 we show the effectiveness of MTAN with various weighting methods compared with single and multi-task baseline methods. We explore how the performance of our method scales with task complexity in Section 4.1.5 and we show visualisations of the learned attention masks in Section 4.1.6.

4.1.1 Datasets

CityScapes. The CityScapes dataset [4] consists of high resolution street-view images. We use this dataset for two tasks: semantic segmentation and depth estimation. To speed up training, all training and validation images were resized to $[128 \times 256]$. The dataset contains 19 classes for pixel-wise semantic segmentation, together with ground-truth inverse depth labels. We pair the depth estimation task with three levels of semantic segmentation using 2, 7 or 19 classes (excluding the void group in 7 and 19 classes). Labels for the 19 classes and the coarser 7 categories are defined as in the original CityScapes dataset. We then further create a 2-class dataset with only background and foreground objects. The details of these segmentation classes are presented in Table 1. We perform multi-task learning for 7-class CityScapes dataset in Section 4.1.4. We compare the 2/7/19-class results in Section 4.1.5, with visualisation of these attention maps in Section 4.1.6.

NYUv2. The NYUv2 dataset [21] is consisted with RGB-D indoor scene images. We evaluate performances

on three learning tasks: 13-class semantic segmentation defined in [5], true depth data which is recorded by depth cameras from Microsoft Kinect, and surface normals which are provided in [7]. To speed up training, all training and validation images were resized to $[288 \times 384]$ resolution.

Compared to CityScapes, NYUv2 contains images of indoor scenes, which are much more complex since the viewpoints can vary significantly, changable lighting conditions are present, and the appearance for each object class shifts widely in texture and shape. We evaluate performance on different datasets, together with different numbers of tasks, and further with different class complexities, in order to attain a comprehensive understanding on how our proposed method behaves and scales under a range of scenarios.

2-class	7-class	19-class
background	void	void
	flat	road, sidewalk
	construction	building, wall, fence
	object	pole, traffic light, traffic sign
	nature	vegetation, terrain
	sky	sky
foreground	human	person, rider
	vehicle	carm truck, bus, caravan, trailer, train, motorcycle

Table 1: Three levels of semantic classes for the CityScapes data used in our experiments.

4.1.2 Baselines

Most image-to-image multi-task learning architectures are designed based on specific feed-forward neural networks, or implemented on varying network architectures, and thus they are typically not directly comparable based on published results. Our method is general and can be applied to any feed-forward neural network, and so for a fair comparison, we implemented 5 different network architectures (2 single-task + 3 multi-task) based on SegNet [1], which we consider as baselines:

- **Single-Task, One Task:** The vanilla SegNet for single task learning.
- **Single-Task, STAN:** A Single-Task Attention Network, where we directly apply our proposed MTAN whilst only performing a single task.
- **Multi-Task, Split (Wide, Deep):** The standard multi-task learning, which splits at the last layer for the final prediction for each specific task. We introduce two verions of **Split**: **Wide**, where we adjusted the number of convolutional filters, and **Deep**, where we adjusted the number of convolutional layers, until **Split** had at least as many parameters as MTAN.
- **Multi-Task, Dense:** A shared network together with task-specific networks, where each task-specific network receives all features from the shared network, without any attention modules.
- **Multi-Task, Cross-Stitch:** The Cross-Stitch Network [20], a previously proposed adaptive multi-task learning approach, which we implemented on SegNet.

Note that all the baselines were designed to have at least as many parameters than our proposed MTAN, and were tested to validate that our proposed method's better performance is due to the attention modules, rather than simply due to the increase in network parameters.

4.1.3 Dynamic Weight Average

For most multi-task learning networks, training multiple tasks is difficult without finding the correct balance between those tasks, and recent approaches have attempted to address this issue [3, 14]. To test our method across a range of weighting schemes, we propose a simple yet effective adaptive weighting method, named Dynamic Weight Average (DWA). Inspired by GradNorm [3], this learns to average task weighting over time by considering the rate of change of loss for each task. But whilst GradNorm requires access to the network's internal gradients, our DWA proposal only requires the numerical task loss, and therefore its implementation is far simpler.

With DWA, we define the weighting λ_k for task k as:

$$\lambda_k(t) := \frac{K \exp(w_k(t-1)/T)}{\sum_i \exp(w_i(t-1)/T)}, w_k(t-1) = \frac{\mathcal{L}_k(t-1)}{\mathcal{L}_k(t-2)}, \tag{7}$$

Here, $w_k(\cdot)$ calculates the relative descending rate in the range $(0, +\infty)$, t is an iteration index, and T represents a temperature which controls the softness of task weighting, similar to [12]. A large T results in a more even distribution between different tasks. If T is large enough, we have $\lambda_i \approx 1$, and tasks are weighted equally. Finally, the softmax operator, which is multiplied by K, ensures that $\sum_i \lambda_i(t) = K$.

In our implementation, the loss value $\mathcal{L}_k(t)$ is calculated as the average loss in each epoch over several iterations. Doing so reduces the uncertainty from stochastic gradient descent and random training data selection. For $t = 1, 2$, we initialise $w_k(t) = 1$, but any non-balanced initialisation based on prior knowledge could also be introduced.

4.1.4 Results on Image-to-Image Predictions

We now evaluate the performance of our proposed MTAN method in image-to-image multi-task learning, based on the SegNet architecture. Using the 7-class version of the

CityScapes dataset and 13-class version of NYUv2 dataset, we compare all the baselines introduced in Section 4.1.2.

Training. For each network architecture, we ran experiments with three types of weighting methods: equal weighting, weight uncertainty [14], and our proposed DWA (with hyper-parameter temperature $T = 2$, found empirically to be optimum across all architectures). We did not include GradNorm [3] because it requires a manual choice of subset network weights across all baselines, based on their specific architectures, which distracts from a fair evaluation of the architectures themselves. We trained all the models with ADAM optimiser [15] using a learning rate of 10^{-4}, with a batch size of 2 for NYUv2 dataset and 8 for CityScapes dataset. During training, we halve the learning rate at 40k iterations, for a total of 80k iterations.

Results. Table 2 and 3 shows experimental results for CityScales and NYUv2 datasets across all architectures, and across all loss function weighting schemes. Results also show the number of network parameters for each architecture. Our MTAN method performs similarly to our baseline Dense in the CityScapes dataset, whilst only having less than half the number of parameters, and outperforms all other baselines. For the more challenging NYUv2 dataset, our method outperforms all baselines across all weighting methods and all learning tasks.

#P.	Architecture	Weighting	Segmentation (Higher Better) mIoU	Segmentation (Higher Better) Pix Acc	Depth (Lower Better) Abs Err	Depth (Lower Better) Rel Err
2	One Task	n.a.	51.09	90.69	0.0158	34.17
3.04	STAN	n.a.	51.90	90.87	0.0145	27.46
1.75	Split, Wide	Equal Weights	50.17	90.63	0.0167	44.73
		Uncert. Weights [14]	**51.21**	**90.72**	**0.0158**	44.01
		DWA, $T = 2$	50.39	90.45	0.0164	**43.93**
2	Split, Deep	Equal Weights	**49.85**	88.69	0.0180	43.86
		Uncert. Weights [14]	48.12	88.68	**0.0169**	**39.73**
		DWA, $T = 2$	49.67	**88.81**	0.0182	46.63
3.63	Dense	Equal Weights	**51.91**	90.89	0.0138	27.21
		Uncert. Weights [14]	51.89	**91.22**	**0.0134**	**25.36**
		DWA, $T = 2$	51.78	90.88	0.0137	26.67
≈2	Cross-Stitch [20]	Equal Weights	50.08	90.33	0.0154	34.49
		Uncert. Weights [14]	50.31	90.43	**0.0152**	**31.36**
		DWA, $T = 2$	**50.33**	**90.55**	0.0153	33.37
1.65	MTAN (Ours)	Equal Weights	53.04	**91.11**	**0.0144**	**33.63**
		Uncert. Weights [14]	**53.86**	91.10	0.0144	35.72
		DWA, $T = 2$	53.29	91.09	0.0144	34.14

Table 2: 7-class semantic segmentation and depth estimation results on CityScapes validation dataset. #P shows the number of network parameters, and the best performing combination of multi-task architecture and weighting is highlighted in bold. The top validation scores for each task are annotated with boxes.

In particular, our method has two key advantages. First, due to the efficiency of having a single shared feature pool with attention masks automatically learning which features to share, our method outperforms other methods without requiring extra parameters (column #P), and even with significantly fewer parameters in some cases.

Second, our method maintains high performance across different loss function weighting schemes, and is more robust to the choice of weighting scheme than other methods, avoiding the need for cumbersome tweaking of loss weights. We illustrate the robustness of our method to the weighting schemes with a comparison to the Cross-Stitch Network [20], by plotting learning curves in Figure 3 with respect to the performance of three learning tasks in NYUv2 dataset. We can clearly see that our network follows similar learning trends across various weighting schemes, compared to the Cross-Stitch Network which produces notably different behaviour across the different schemes.

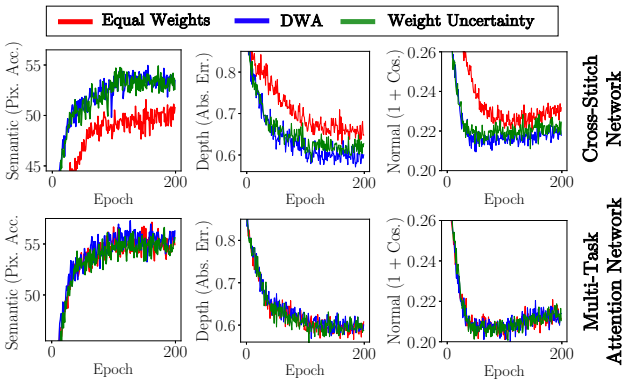


Figure 3: Validation performance curves on the NYUv2 dataset, across all three tasks (semantics, depth, normals, from left to right), showing robustness to loss function weighting schemes on the Cross-Stitch Network [20] (top) and our Multi-task Attention Network (bottom).

Figure 4 then shows qualitative results on the CityScapes validation dataset. We can see the advantage of our multi-task learning approach over vanilla single-task learning, where the edges of objects are clearly more pronounced.

4.1.5 Effect of Task Complexity

For further introspection into the benefits of multi-task learning, we evaluated our implementations on CityScapes across different numbers of semantic classes, with the depth labels the same across all experiments. We trained the networks with the same settings as in Section 4.1.4, with an additional multi-task baseline **Split** (the standard version), which we found to perform better than the other modified versions. All networks are trained with equal weighting.

Table 4 (left) shows the validation performance improvement across all multi-task implementations and the single-task STAN implementation, plotted relative to the performance of the vanilla single-task learning on the CityScapes dataset. Interestingly, for only a 2-class setup, the single-task attention network (STAN) performs better than all

Type	#P.	Architecture	Weighting	Segmentation (Higher Better) mIoU	Segmentation (Higher Better) Pix Acc	Depth (Lower Better) Abs Err	Depth (Lower Better) Rel Err	Surface Normal: Angle Distance (Lower Better) Mean	Surface Normal: Angle Distance (Lower Better) Median	Surface Normal: Within t° (Higher Better) 11.25	Surface Normal: Within t° (Higher Better) 22.5	Surface Normal: Within t° (Higher Better) 30
Single Task	3	One Task	n.a.	15.10	51.54	0.7508	0.3266	31.76	25.51	22.12	45.33	57.13
	4.56	STAN	n.a.	15.73	52.89	0.6935	0.2891	32.09	26.32	21.49	44.38	56.51
Multi Task	1.75	Split, Wide	Equal Weights	15.89	51.19	0.6494	0.2804	33.69	28.91	18.54	39.91	52.02
			Uncert. Weights [14]	15.86	51.12	**0.6040**	0.2570	**32.33**	**26.62**	**21.68**	**43.59**	**55.36**
			DWA, $T = 2$	**16.92**	**53.72**	0.6125	**0.2546**	32.34	27.10	20.69	42.73	54.74
	2	Split, Deep	Equal Weights	13.03	41.47	0.7836	0.3326	38.28	36.55	9.50	27.11	39.63
			Uncert. Weights [14]	**14.53**	43.69	0.7705	0.3340	**35.14**	**32.13**	**14.69**	**34.52**	**46.94**
			DWA, $T = 2$	13.63	**44.41**	**0.7581**	**0.3227**	36.41	34.12	12.82	31.12	43.48
	4.95	Dense	Equal Weights	16.06	52.73	0.6488	0.2871	33.58	28.01	20.07	41.50	53.35
			Uncert. Weights [14]	**16.48**	**54.40**	0.6282	0.2761	**31.68**	**25.68**	**21.73**	**44.58**	**56.65**
			DWA, $T = 2$	16.15	54.35	**0.6059**	**0.2593**	32.44	27.40	20.53	42.76	54.27
	≈3	Cross-Stitch [20]	Equal Weights	14.71	50.23	0.6481	0.2871	33.56	28.58	20.08	40.54	51.97
			Uncert. Weights [14]	15.69	52.60	0.6277	0.2702	32.69	27.26	21.63	42.84	54.45
			DWA, $T = 2$	**16.11**	**53.19**	**0.5922**	**0.2611**	**32.34**	**26.91**	**21.81**	**43.14**	**54.92**
	1.77	MTAN (Ours)	Equal Weights	**17.72**	55.32	**0.5906**	0.2577	31.44	**25.37**	**23.17**	45.65	57.48
			Uncert. Weights [14]	17.67	**55.61**	0.5927	0.2592	**31.25**	25.57	22.99	**45.83**	**57.67**
			DWA, $T = 2$	17.15	54.97	0.5956	**0.2569**	31.60	25.46	22.48	44.86	57.24

Table 3: 13-class semantic segmentation, depth estimation, and surface normal prediction results on the NYUv2 validation dataset. #P shows the number of network parameters, and the best performing combination of multi-task architecture and weighting is highlighted in bold. The top validation scores for each task are annotated with boxes.

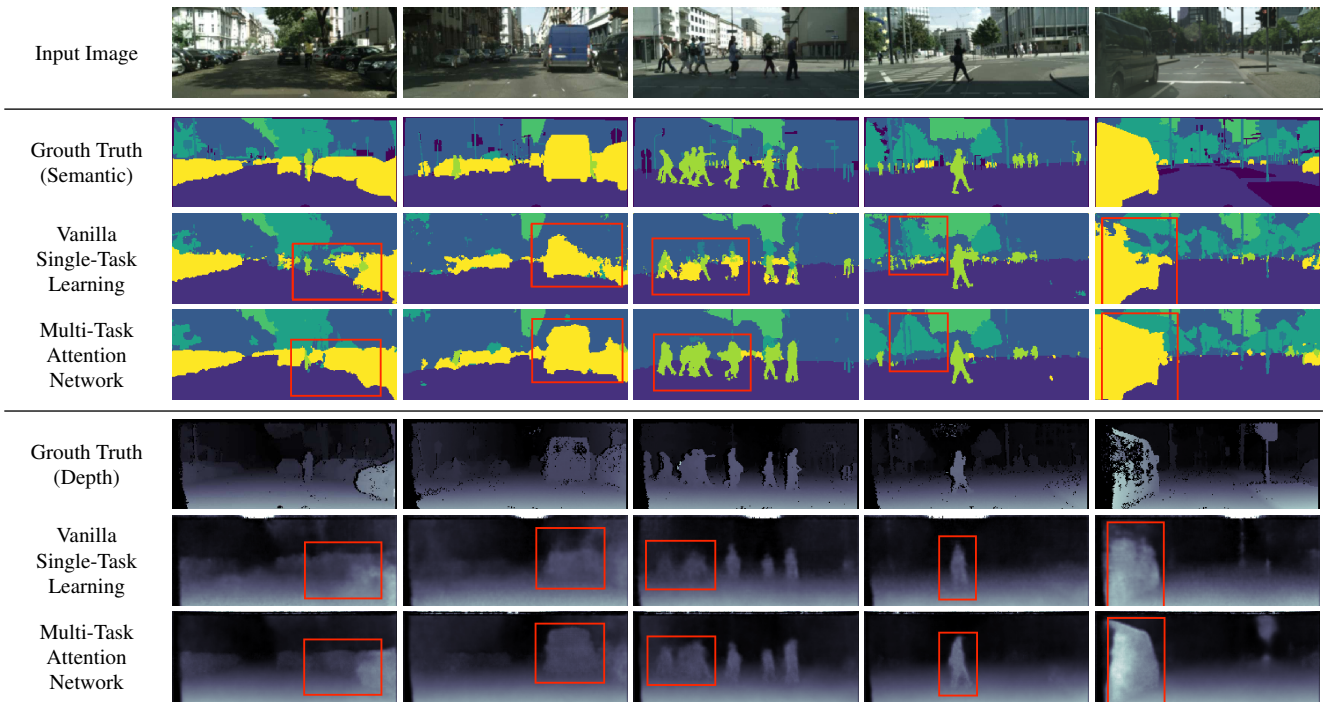
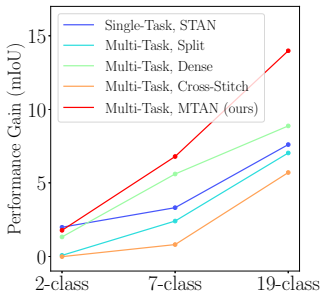


Figure 4: CityScapes validation results on 7-class semantic labelling and depth estimation, trained with equal weighting. The original images are cropped to avoid invalid points for better visualisation. The red boxes are regions of interest, showing the effectiveness of the results provided from our method and single task method.

multi-task methods since it is able to fully utilise network parameters in a simple manner for the simple task. However, for greater task complexity, the multi-task methods encourage the sharing of features for a more efficient use of available network parameters, which then leads to better results. We also observe that, whilst the relative performance gain increases for all implementations as the task complexity increases, our MTAN method increases at a greater rate.



Method	#P.	ImNet.	Airc.	C100	DPed	DTD	GTSR	Flwr	Oglt	SVHN	UCF	Mean	Score
Scratch [23]	10	59.87	57.10	75.73	91.20	37.77	96.55	56.3	88.74	96.63	43.27	70.32	1625
Finetune [23]	10	59.87	60.34	82.12	92.82	55.53	97.53	81.41	87.69	96.55	51.20	76.51	2500
Feature [23]	1	59.67	23.31	63.11	80.33	45.37	68.16	73.69	58.79	43.54	26.8	54.28	544
Res. Adapt.[23]	2	59.67	56.68	81.20	93.88	50.85	97.05	66.24	89.62	96.13	47.45	73.88	2118
DAN [25]	2.17	57.74	64.12	80.07	91.30	56.54	98.46	86.05	89.67	96.77	49.38	77.01	2851
Piggyback [19]	1.28	57.69	65.29	79.87	96.99	57.45	97.27	79.09	87.63	97.24	47.48	76.60	2838
Parallel SVD [24]	1.5	60.32	66.04	81.86	94.23	57.82	99.24	85.74	89.25	96.62	52.50	78.36	3398
MTAN (Ours)	1.74	63.90	61.81	81.59	91.63	56.44	98.80	81.04	89.83	96.88	50.63	77.25	2941

Table 4: Left: CityScapes performance gain in percentage for all implementations compared with the vanilla single-task method. Right: Top-1 classification accuracy on the Visual Decathlon Challenge online test set. #P is the number of parameters as a factor of a single-task implementation. The upper part of table presents results from single task learning baselines; lower part of table presents results from multi-task learning baselines.

4.1.6 Attention Masks as Feature Selectors

To understand the role of the proposed attention modules, in Figure 5 we visualise the first layer attention masks learned with our network based on CityScapes dataset. We can see a clear difference in attention masks between the two tasks, with each mask working as a feature selector to mask out uninformative parts of the shared features, and focus on parts which are useful for each task. Notably, the depth masks have a much higher contrast than the semantic masks, suggesting that whilst all shared features are generally useful for the semantic task, the depth task benefits more from extraction of task-specific features.

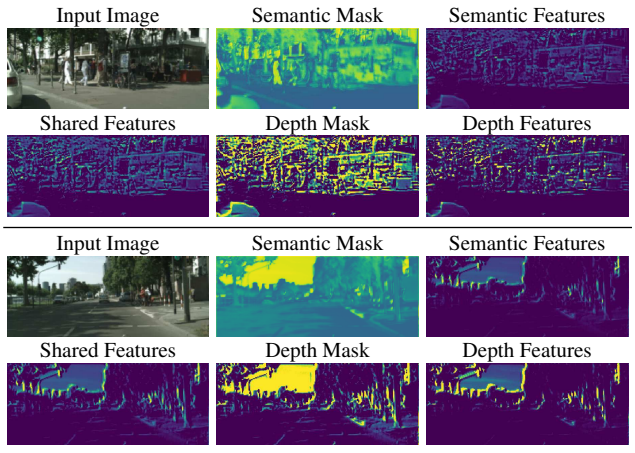


Figure 5: Visualisation of the first layer of 7-class semantic and depth attention features of our proposed network. The colours for each image are rescaled to fit the data.

4.2. Visual Decathlon Challenge (Many-to-Many)

Finally, we evaluate our approach on the recently introduced Visual Decathlon Challenge, consisting of 10 individual image classification tasks (many-to-many predictions). Evaluation on this challenge reports per-task accuracies, and assigns a cumulative score with a maximum value of 10,000 (1,000 per task) based on these accuracies. The complete details about the challenge settings, evaluation, and datasets used, can be found at http://www.robots.ox.ac.uk/~vgg/decathlon/.

Table 4 (right) shows results for the online test set of the challenge. As consistent with the prior works, we apply MTAN built on Wide Residual Network [31] with a depth of 28, widening factor of 4, and a stride of 2 in the first convolutional layer of each block. We train our model using a batch size of 100, learning rate of 0.1 with SGD, and weight decay of $5 \cdot 10^{-5}$ for all 10 classification tasks. We halve the learning rate every 50 epochs for a total of 300 epochs. Then, we fine-tune 9 classification tasks (all except ImageNet) with a learning rate 0.01 until convergence. The results show that our approach surpasses most of the baselines and is competitive with the current state-of-the-art, without the need for complicated regularisation strategies such as applying DropOut [28], regrouping datasets by size, or adaptive weight decay for each dataset, as required.

5. Conclusions

In this work, we have presented a new method for multi-task learning, the Multi-Task Attention Network (MTAN). The network architecture consists of a global feature pool, together with task-specific attention modules for each task, which allows for automatic learning of both task-shared and task-specific features in an end-to-end manner. Experiments on the NYUv2 and CityScapes datasets with multiple dense-prediction tasks, and on the Visual Decathlon Challenge with multiple image classification tasks, show that our method outperforms or is competitive with other methods, whilst also showing robustness to the particular task weighting schemes used in the loss function. Due to our method's ability to share weights through attention masks, our method achieves this state-of-the-art performance whilst also being highly parameter efficient.

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[2] Rich Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer, 1998.

[3] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International Conference on Machine Learning*, pages 793–802, 2018.

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016.

[5] Camille Couprie, Clément Farabet, Laurent Najman, and Yann Lecun. Indoor semantic segmentation using depth information. In *International Conference on Learning Representations (ICLR2013), April 2013*, 2013.

[6] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[7] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2650–2658, 2015.

[8] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 109–117. ACM, 2004.

[9] Georgia Gkioxari, Bharath Hariharan, Ross Girshick, and Jitendra Malik. R-cnns for pose estimation and action detection. *arXiv preprint arXiv:1406.5212*, 2014.

[10] Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei. Dynamic task prioritization for multitask learning. In *European Conference on Computer Vision*, pages 282–299. Springer, 2018.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[12] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[13] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016.

[14] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7482–7491, 2018.

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[16] Iasonas Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[17] Abhishek Kumar and Hal Daumé III. Learning task grouping and overlap in multi-task learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1723–1730. Omnipress, 2012.

[18] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer feature learning with joint distribution adaptation. In *Proceedings of the IEEE international conference on computer vision*, pages 2200–2207, 2013.

[19] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 67–82, 2018.

[20] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3994–4003, 2016.

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

[22] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[23] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems*, pages 506–516, 2017.

[24] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Efficient parametrization of multi-domain deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8119–8127, 2018.

[25] Amir Rosenfeld and John K Tsotsos. Incremental learning through deep adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[26] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[27] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[28] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[29] Sebastian Thrun and Lorien Pratt. *Learning to learn*. Springer Science & Business Media, 2012.

[30] Fei Wang, Mengqing Jiang, Chen Qian, Shuo Yang, Cheng Li, Honggang Zhang, Xiaogang Wang, and Xiaoou Tang.

Residual attention network for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164, 2017.

[31] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 87.1–87.12. BMVA Press, September 2016.