Sharing Decoders: Network Fission for Multi-task Pixel Prediction

Steven Hickson
Google/Georgia Tech
shickson@google.com

Karthik Raveendran
Google

Irfan Essa
Google/Georgia Tech

Abstract

We examine the benefits of splitting encoder-decoders for multitask learning and showcase results on three tasks (semantics, surface normals, and depth) while adding very few FLOPS per task. Current hard parameter sharing methods for multi-task pixel-wise labeling use one shared encoder with separate decoders for each task. We generalize this notion and term the splitting of encoder-decoder architectures at different points as fission. Our ablation studies on fission show that sharing most of the decoder layers in multi-task encoder-decoder networks results in improvement while adding far fewer parameters per task. Our proposed method trains faster, uses less memory, results in better accuracy, and uses significantly fewer floating point operations (FLOPS) than conventional multi-task methods, with additional tasks only requiring 0.017% more FLOPS than the single-task network. We show results with a real-time model on a Pixel phone with released source code.

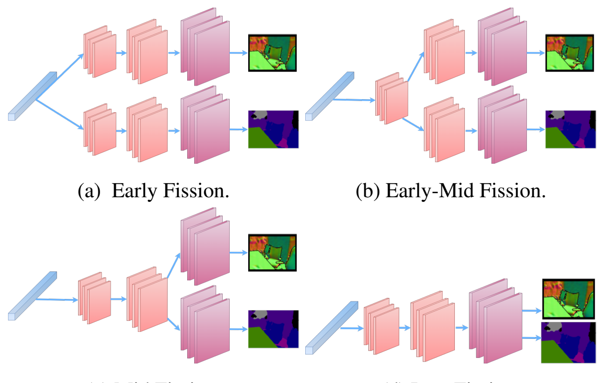
1. Introduction

Multi-task learning aims to jointly learn multiple tasks and helps with generalization, improving results given an inductive bias[34]. Deep learning methods have made great improvements in multi-task pixel-wise prediction tasks through the use of architectural improvements like encoder-decoders[12]. Early work by Caruana[4] shows that hard parameter sharing with backprop nets "discovers task relatedness without the need of supervisory signals"; however, this hasn't been extended fully to encoder-decoders. While there have been several studies on combining different modalities (i.e. fusion), there has been relatively little work that examines the properties of the task-specific decoders. Motivated by this, we undertake an exploration of hard parameter sharing of encoder-decoders with several outputs, which we call *network fission*. We consider (a) early fission (splitting the network at the encoder with separate decoders, Figure 1a), (b) late fission (splitting the network only at the very end and sharing the entire encoder-decoder Figure 1d), and (c) mid fission (sharing most of the encoder-decoder and splitting at one of the intermediate decoder blocks, Figures 1b and 1c).

Initial work for semantic labeling and depth prediction focused on using fully convolutional networks (FCNs) [24] but later decoder-based architectures were proposed [30, 2] to overcome the limitations of FCNs. Encoder-Decoder architectures are effective and established methods for pixel-wise prediction tasks. When doing multitask pixel-wise predictions, hard parameter sharing is a popular method and it is common to take the features at the end of the encoder and then split to give each task its own decoder. This is the method used by many [19, 18, 23, 11, 15], which we refer to as early fission.

There are several recent soft parameter sharing methods[27, 34, 33] that have a model for each task and learn parameters between single task networks to combine, stitch, and/or regularize them; however these can be difficult to learn and are prone to overfitting. They also aren't conducive to small models needed for mobile devices. In terms of hard parameter sharing, most methods only use early fission (splitting at the encoder bottleneck with separate decoders for each task) as shown in Figure 1a. We do ablation studies on different fission methods and find that mid fission (sharing all of the decoder blocks until the last decoder layer) outperforms other multi-task methods and requires significantly less FLOPS. We evaluate this on 3 different architectures, including 1 server model, and 2 lightweight mobile models. We evaluate our method on the tasks of depth/disparity, surface normals, and semantic labels.

The main goal of this work is to examine the benefits of splitting encoder-decoders in different places for multi-task learning and showcase improvements using new frameworks. In the past, researchers explored combining data with different input fusion schemes; similarly, we should also examine different output fission schemes. For comparison, an example of early fusion [3, 7] is where RGB and depth are stacked as 4 channels and input into the same set of convolutions. Late fusion [39, 5] for the same task puts each input modality into a set of convolutions and joins the outputs at the end. Work later proposed mid-fusion [14] where each input modality has some en-


(a) Early Fission. (b) Early-Mid Fission.

(c) Mid Fission. (d) Late Fission.

Figure 1: The possible fissions for the eASPP decoder part of the architecture[36]. The blue block is the encoder bottleneck, the orange blocks are the first two decoder blocks, which include convolutional, batchnorm, and relu layers with 256 output channels and skip connections. The purple blocks are the last decoder block which has convolutional transpose, batchnorm, and relu layers.

coding, but then those features are concatenated and further convolutions are done before an embedding is reached. We can make the same deductions for different outputs and explore early, mid, and late fission.

An easy mapping can be made from these modality fusion schemes to our proposed modality fission schemes. Early fission as shown in Figure 1a is the current method of splitting at the encoder bottleneck and giving each task its own decoder. Late fission as shown in Figure 1d would be both the encoder and decoder fully shared with only one output, computing a loss differently on different channel subsets of the output. For the commonly used 3-stage decoder blocks, there are two potential splits that qualify as mid fission. Figure 1b shows what we call early-mid fission: sharing all of the encoder and the first decoder block (or more blocks for larger decoders). Figure 1c shows what we call mid fission: sharing all of the encoder and most of the decoder but allowing the last set of layers to learn features separately for each output modality. The proposed mid fission has not been fully explored and analyzed before and many multi-task methods would benefit from this architectural change.

Our **main contributions** are **(1)** Comprehensive ablation studies comparing fission schemes with several modalities on 3 different challenging datasets, **(2)** A shared decoder architecture we call mid fission that has fewer parameters, outperforms single task training, and outperforms current multi-task methods with significantly less FLOPS (only 0.0167% more FLOPS required per task), and **(3)** A real-time demo of 3 tasks simultaneously with the model and code[1] released.

2. Related Work

Multi-task learning has a long history but most of the paper will focus on recent techniques. Our approach is inspired by early work by Caruana et. al[4] who discuss using hard parameter sharing neural networks to predict several tasks. This methodology, though being around since 1993, has not been leveraged fully with current encoder-decoder networks. There are many datasets used to leverage pixel-wise tasks (semantic labeling, depth, normals, HHA, curvature, etc), both synthetic like Scenenet [26] and real such as Scannet [8], NYUDv2 [29], and Matterport [6]. Availability of RGB-D data has resulted on its use to predict semantic labels [13]. Different data source fusions have been developed (early, mid, and late) to determine where in a network to ”join” the information. The inverse, using only RGB to predict semantic labels, depth, normals, etc., is also a topic of interest [9]. We briefly cover these efforts here.

Eigen et. al [9] use a common multi-scale FCN architecture to predict depth, surface normals, and semantic labels but not jointly. They only share the architecture layout but train separately. This was one of the first works on FCNs for joint tasks. Gupta et. al[13] later work on NYUDv2 [29] and introduce an encoding called HHA (horizontal disparity, height above ground, and angle). They then use that encoding to train a CNN whose features are combined with features from a CNN trained on RGB in order to do object detection and semantic segmentation (late fusion).

Other methods chose to combine networks or learn shared parameters with soft parameter sharing. Ruder et. al [34] discuss multi-task learning using a method they call sluice networks where parameters control which sub-spaces are shared between main and auxiliary tasks. There is a shared input layer and task-specific output layers and all other layers are shared with a parameter to control what is used for each prediction. They show small networks for NLP tasks as opposed to pixel-wise predictions which involve larger, more complex architectures with an encoder-decoder structure. Misra et. al [27] introduce cross-stitch networks with a similar method of [34] using a parameter to control the flow between the layers of the networks. They try to learn how to stitch together two networks with a shared representation as a linear combination of activations. They explore when to split a network for two separate tasks but only from a FCN style network based on Alexnet [20]. This is different from more current methods which use those architectures as an encoder and have a separate decoder per task. Jafari et. al [17] use separately trained networks for depth and semantics as input to a joint refinement network with the hope of mutually improving both results using cross-modality influences. Many of these soft parameter sharing methods can be transformed into others by specifically setting certain parameters. However, they can be difficult to learn and require a full network trained on each individual task which is a large amount of FLOPS. As opposed to learning tasks together, Taskonomy [38] attempts to learn task-relatedness by transferring learning between modalities instead of doing multi-task learning.

Other hard parameter sharing work has focused on using several different modalities as both inputs and outputs or as intermediate tasks. Kuga et. al [21] use several different modalities as both inputs and outputs with each having it's own encoder and decoder but sharing skip connections and a shared latent representation. Xu et. al [37] use depth, surface normals, contours, and semantics as a set of intermediate tasks, which are then used via a multi-modal input into a distillation model. In this method, there is one shared encoder with each auxiliary task having a separate decoder. The results are then put into the multi-modal distillation module for final task prediction, where each task has it's own decoder. Maninis et. al [25] use a shared encoder with learned soft attention modules to train one network to do several tasks, one at a time, only focusing on the encoder.

Most work has focused on what we call the early fission model, where one shared encoder is used with separate decoders for each task. Each of these then requires clever methods in order to improve results. Initial works on many scene tasks including geometry and semantic labels [19] discuss task-interference and find that jointly learning can impede accuracy. [19] jointly handle many tasks end-to-end with a network architecture relying on task specific responses with many fusion layers with skip pooling across different resolutions to deal with different tasks. Their method also use memory-efficient back-propogation to handle training for many different tasks. Kendall et. al [18] use a shared encoder with separate decoders for each task and focus on using task-uncertainty to improve the training for each task. Liu et. al [23] propose an end-to-end multi-task attention network using one global feature pool and a soft attention module per task. Using this, they hope to learn task-shared and task-specific features in an automatic manner. Gao et. al [11] propose a fused network model by combining multiple single-task networks using discriminative dimensionality reduction with several 1x1 convolutions. This method however, is high in parameters as it combines several networks trained for separate tasks just like other soft parameter sharing methods. Hickson et. al [15] use a shared encoder with two separate decoders to predict semantic labels and surface normals with a real-time mobile model. They also propose a method to clean up ground-truth surface normals for better results which we utilize.

We go beyond previous hard parameter sharing encoder-decoder work and try to analyze further output network fission schemes to allow for the best split between task-shared features and task-independent features efficiently.

3. Studying the different fission schemes

The backbone architecture and hyperparameters remain the same for all ablations in order to have a valid comparison. The server model is a variant of AdapNet++, while the two mobile models are a variant of Mobilenet[35] with a unet decoder for the first and MobilenetV3[16] with a u-net style decoder with self-attention for the second. We explain metrics, datasets, architecture and hyperparameters in the Supplementary section. Here we will go over the losses and each method (early, late, and mid fission).

For the backbone of our server ablation studies, we adopted the Adapnet++ encoder-decoder architecture from [36]. This architecture was chosen because it has state of the art results on several datasets despite requiring fewer parameters. Importantly, it does not include additional modality inputs as we are exploring RGB-to-many-task prediction. For training, we use two auxiliary losses after each of the first two decoder blocks where a fully connected layer predicts a task. We weight each of these auxiliary losses when computing the total loss, with $\alpha_1 = 0.6$ and $\alpha_2 = 0.5$ while $\alpha_0 = 1$ (the final layer). An ablation study on the auxiliary losses with multiple tasks is shown in the Supplementary section.

3.1. Losses

For semantic labeling, we use the softmax cross entropy loss with auxiliary losses in Equation 1, where y is the one-hot encoded ground truth pixel-wise label, σ is the softmax function, z_0 is the final output of the task decoder, z_1 and z_2 are auxiliary branches of the decoder as in [36]. ϵ is 1^{-10} for stability.

$$L_{\text{semantics}}(y,z) = -\sum_{a=0}^{3}\alpha_a \sum_{i=0}^{n} y^{(i)} \log(\sigma(z_{\text{a}}^{(i)}) + \epsilon). \quad (1)$$

For surface normal prediction, we use the cosine similarity with auxiliary losses in Equation 2. In this equation, $\hat{y}$ is the pixel-wise ground truth surface normals normalized to a unit vector where each normal is clipped between -1.0 and 1.0 to prevent numerical errors. $\hat{z_0}$ are the final outputs of the task decoder and $\hat{z}_1$, $\hat{z}_2$ and auxiliary branches all normalized and clipped the same as $\hat{y}$.

$$L_{\text{normals}}(y,z) = 1 - \sum_{a=0}^{3}\alpha_1(1 - \sum_{i=0}^{3} \hat{y}^{(i)}.\hat{z}_a^{(i)}) \quad (2)$$

For depth prediction, we use the L1 loss with auxiliary losses in Equation 3 where y is the ground truth pixel-wise depth divided by 1000.0 so as to be in meters. z_0, z_1, z_2 are the final outputs of the task decoder, the first auxiliary branch, and the second auxiliary branch respectively.

$$L_{\text{depth}}(y,z) = \sum_{a=0}^{3}\alpha_a|y - z_{\text{a}}|. \quad (3)$$

For disparity (inverse depth) prediction, we use a set of scale and shift invariant loss, smoothness loss, and gradient loss as detailed by [31]. We sum this with 0.25 of the depth prediction loss to prevent large predictions that can cause errors in training and in inference when quantized. More details on this in the Supplementary section.

Now we discuss the training used for all the proposed early, mid, and late fission schemes.

3.2. Early Fission

As discussed in Section 2, early fission is a shared encoder with separate decoders for each task shown in Figure 1a. This method has been widely used in the literature. The bottleneck from the encoder connects to two separate decoders with their own skip connections and deconvolutional layers. The loss is then computed via Equation 4 where L_T is the total loss. For each task 1 to n there is an individual loss L_i and a loss weight λ_i. In much work, there is no loss balancing, meaning all λ is set to 1.0, in others it's set manually based off of loss convergence properties[15], and others learn it[18]. In our experiments, when we balance the losses, we set $\lambda_{\text{semantics}}$ to 1.0, λ_{normals} to 10.0 and λ_{depth} to 0.5 based off of the loss values after 10K iterations when the individual single-task network starts to converge. We do this for all 3 architectures.

$$L_T = \sum_{i=1}^{n} \lambda_i L_i(y_i, z_i). \tag{4}$$

3.3. Late Fission

In late fission, we explore what happens if the entire decoder and encoder are shared as shown in Figure 1d. The output of this is treated as a concatenated set of task-level outputs with the loss computed via Equation 5. Here there are several loss equations L_i and several labels y_i but only one output z, which is partitioned into n splits of size τ_i depending on the output size. In our case, $\tau_{\text{semantics}}$ is the number of semantic labels plus one (for the background class), τ_{normals} is 3, and τ_{depth} is 1. λ is the same as in early fission.

$$L_T = \sum_{i=1,k=0}^{n,k+=\tau_i} \lambda_i L_i(y_i, z[k:(k+\tau_i)]). \tag{5}$$

3.4. Early-Mid and Mid Fission

In our proposed shared decoder architecture, we use mid-fission as shown in Figures 1b and 1c. The loss for this is calculated the same as in Section 3.2. The only difference is where we split the decoder. We found both early-mid and mid fission to have superior results, shown in Table 1.

3.5. Relationship to State-of-the-art Methods

Our method is a type of hard parameter sharing, meaning sharing hidden layers between tasks followed by task-specific output layers[34]. Here the task-specific and task-shared features are learned in the hidden layers but everything else is fixed. This has the benefit of being simple to learn and reducing overfitting as discussed in[4, 33] with the detriment of not being robust to loosely/non related tasks.

Soft parameter sharing is where each task has it's own model with it's own parameters where the parameters are encourage in some way to be related or similar. One example of this that we compare against are Cross-stitch networks[27], which model shared representations as a linear combination of input activation maps. NDDR[11] can generalize from cross-stitch networks and utilize a very similar idea. Sluice networks[34] can generalize to a specific case of cross-stitch networks as well and are closely related to both cross-stitch and NDDR.

These types of soft parameter sharing methods require a full network architecture for each task, with separate parameters computing specific task-shared features between the two networks. This means they are more computationally expensive then some of the smaller hard parameter sharing methods such as ours. They can also be harder to learn and to implement for many multi-task problems as they require training several single task networks and parameters to combine them and are prone to overfitting. They are also not suited well to small models for real-time on resource constrained devices such as phones. Our method yields models only slightly larger than a single task encoder-decoder architecture. A well-designed hard parameter sharing architecture for related tasks can outperform these other soft parameter sharing methods such as [27, 11, 34] as shown in Table 4 where our server architecture outperforms all 3 of these methods on NYU40, while only requiring 0.017% more FLOPS than the single task network.

4. Results

We evaluate our method on several different datasets which are discussed at length in the Supplementary section. Our ablations studies are done with the Adapnet++ server architecture on the Scenenet RGBD dataset [26] because it is a synthetic dataset with many samples, clean depth/semantic images, and low noise/label error. The decision to evaluate on this data was done to avoid spurious errors caused by noise and labeling errors. For real world evaluation, we use both the NYUDv2 [29] and Scannetv2 [8] datasets to verify our results. For all datasets, we evaluate with the dataset image resolution despite our training resolution. For depth/disparity, we use the evaluation metrics from [22], which are the percent of pixels under log maximum relative depth error of $1.25^{\{1,2,3\}}$. For surface normals we use the same metrics as [10] which are the percent of pixels with angle error less than $11.25°$, $22.5°$, and $30°$ as well as mean angle error (MAE). Equations for all of these are included in the Supplementary section for reference. For semantic labels, we use the mean class intersection over union score as well as mean pixel accuracy. Evaluations of these for a single modality/task baseline network are in the Supplementary section.

For comparing multi-task to single-task, we introduce a metric called % improvement ($\%\tau$), which shows the average improvement each task had over the single task version. This is shown in Equation 6 where for each task i: s_i is the metric for the single task method and m_i is the metric for the multi-task method. The result is multiplied by 100 after subtracting 1 to convert it into percent better (positive) or worse (negative) than the single task results. For surface normals, we use $\% < 11.25°$ for m, s_{normal}, for depth we use $\%$ under relative depth error of 1.25 for m, s_{depth}, and for semantics, we use the mean intersection over union (mIoU) for $m, s_{\mathrm{semnatics}}$,

$$\%\tau = \frac{100}{n}(\sum_{i=0}^{n} \frac{m_i}{s_i} - 1). \quad (6)$$

We also use a variant of this that accounts for the increase of the model size over a single task. We call this percent improvement per FLOP increase percentage (PIPFIP). It is simply ($\%\tau$) divided by the percent increase in model size given by multi-task model size f_m and the single task network size (we use semantics as the single task) f_l as seen in Equation 7. We set negative values to 0 as there is no improvement then

$$\text{PIPFIP} = \frac{\%\tau}{100(\frac{f_m}{f_l} - 1)}. \quad (7)$$

4.1. Fission-scheme Ablation Study

To test the three proposed output fission schemes in Sections 3.2, 3.3, and 3.4, we do an ablation study on the Scenenet synthetic dataset shown in Table 1. We use the same encoder structure for all 4 decoder types with the two tasks chosen being surface normals and semantic labels. We use two tasks for this ablation study in order to examine the effect of two tasks vs three tasks, which we show later. Here we focus on semantic labels and surface normals. Results with depth & normals and depth & semantic labels are shown in the Supplementary section.

For this ablation study, we used loss balancing of $\lambda_{\text{normals}} = 10$ and $\lambda_{\text{semantics}} = 1$ as this was found to be beneficial for all three methods in the loss balancing ablation study found in the Supplementary section. We test all four output fission methods with three different initializations: a) the decoder trained from scratch and encoder initialized with ImageNet weights, b) initializing with the single task network trained on semantic labels, and c) initializing with the single task network trained on normals We tested with different initializations since many other methods such as [36] utilize this for multi-task learning.

Interestingly enough, when trained from scratch, all 4 methods outperform the single task method on surface normals, however, mid fission underperforms on semantic label prediction. Late fission actually is competitive with early fission in these two tasks despite having far fewer task-specific parameters which implies that these tasks are either very interrelated (i.e. they have many efficient task-shared parameters) or that one of the tasks only requires a few parameters. We can rule out the few parameter hypothesis given much literature on the number of parameters used for semantic label prediction as in [11] and the channel multiplier ablation study in [15] for surface normals.

When initialized with the single task trained on semantic labels, the surface normal prediction quality suffers greatly and semantic label prediction also decreases for everything but early fission, which is counter-intuitive. Our theory is that a network trained on surface normals makes for a good initialization for multi-task learning; we see this throughout our ablation studies. When initializing with surface normals, surface normal prediction of all 4 methods greatly increases while semantic label prediction for early and late fission decrease slightly, which is expected given we are emphasizing the surface normal loss. However, our proposed early-mid fission and mid fission schemes greatly increase semantic label prediction as well with this initialization due to task-shared features. It outperforms all other modes of early and late fission in both surface normal and semantic label prediction while also outperforming individual task predictions. Early-mid fission is almost as many flops as early fission. However, the mid fission method only requires 68.3% of the FLOPS of early fission as shown in the bottom of Table 1 yet it outperforms early fission which is used by most current multi-task pixel-wise labeling methods. It might seem non-intuitive that mid fission is fewer FLOPS than late fission but this is due to the number of output channels of the final 1x1 convolutions. For example, $17x17$ (14 semantic labels + 3 normals) requires more operations than $14x14 + 3x3$. Due to mid fission's superior performance and minimum FLOPs, we select it for the further ablation studies.

In the Supplementary section, we show results for mid fission for a model trained jointly on several different combinations of tasks with different initializations: from scratch and from each single-task network. For mid fission, initializing from normals seems to always improve all the tasks. This makes some sense given that surface normals are highly correlated to other tasks as suggested in [38]. To check how much our results would vary in Table 1, we ran a variability study by training our mid fission network 3 times to see if they converge to similar metrics. This is shown in the Supplementary section and metrics only shift +/-0.1% at most verifying these results are meaningful and not just noise. Mid fission outperforms the single-task semantic la-

Method	Normals				Semantics	Imp	FLOPS	PIPFIP
	% < 11.25	% < 11.25	% < 30	MAE	mIoU	%	billion	%
Individually Trained Baselines								
Normals	83	91.5	93.8	8.3	-	-	38.43	-
Semantics	-	-	-	-	50.3	-	38.57	-
Trained From Scratch								
Early	84.9	92.7	94.9	7.6	51	1.84	56.56	0.04
Early-Mid	84.9	92.6	94.7	7.7	50.8	1.64	52.96	0.04
Mid	83.9	92.2	94.5	8	49.8	0.05	**38.58**	1.74
Late	85	92.6	94.8	7.6	50.9	1.8	38.64	9.92
Initialized from Labels								
Early	80.1	91.1	93.8	9.3	51	-1.05	56.56	0
Early-Mid	83.7	92.3	94.5	8	49.8	2.31	52.96	0.06
Mid	78.2	89.8	92.9	9.8	49	-4.2	**38.58**	0
Late	81.1	91.1	93.8	8.9	50.1	-1.34	38.64	0
Initialized from Normals								
Early	85.1	92.7	94.8	7.5	48.5	-0.52	56.56	0
Early-Mid	**86.6**	**93.5**	**95.4**	**6.9**	**51.6**	**3.58**	52.96	0.10
Mid	**86.6**	**93.4**	**95.3**	**7**	**51.3**	3.16	**38.58**	**121.99**
Late	85.6	92.9	95	7.3	50	1.27	38.64	7.00

Table 1: Joint Normals and Semantics results with different fission methods when the loss is balanced ($\lambda_{\text{normals}} = 10.0$ and $\lambda_{\text{semantics}} = 1.0$) and auxiliary loses are used ($\alpha_1 = 0.6, \alpha_2 = 0.5$ for both tasks) with different initialization methods. Imp is the percent improvement metric and PIPFIP is percent improvement per flop increase percentage.

beling network by 1% mIOU, which is significantly more than the variance due to training.

4.2. All 3 Modality Results

Using what was learned in the aforementioned ablation studies, we test the different fission methods with all 3 tasks, depth prediction, surface normal prediction, and semantic labeling.

In Table 2, we do a similar ablation study as in Table 1 but now with all 3 tasks. Note here that early, late, and mid fission all improve on normal prediction when balancing losses. However, late fission degrades on depth and semantic prediction tasks while early and early-mid fission degrade on depth prediction. Mid fission increases across the board as we have seen in our other ablation studies. The results are not as good as the two task results due to the difficulties of predicting 3 different tasks. This makes some amount of sense given depth is an absolute distance of a pixel from camera viewpoint and that is not necessarily a good indicator of the normal or label. This is the reason we select surface normals and semantic labels as the two main tasks on our real dataset experiments. Different losses on the depth such as reverse huber, scale-shift invariant depth, or cross-task consistency could help improve both, which we use in the mobile model in Section 4.4.

Note that we also initialize the early-mid and mid fission with the single task trained on surface normals but keep standard initialization for early and late fission as that was deemed best in Section 4.1. Loss balancing ($\lambda_{\text{depth}} = 0.5$, $\lambda_{\text{normals}} = 10$, $\lambda_{\text{semantics}} = 1$) is used in the last 3 columns and for all columns, all auxiliary losses are used. Note that while surface normal prediction and semantic labeling prediction are still outperforming the single task-networks, depth prediction is equal in percent of pixels under 1.25^2 and 1.25^3 but slightly worse in percent of pixels under 1.25. Our hypothesis is that the number of encoder channels would have to be increased to generate more features relevant to the depth task; it must have more task-specific features and less task-shared features. We found this to be true in our ablation studies with our results using depth. The mid fission strategy still outperforms others even though early fission is widely used in the literature for multi-task prediction. Note that here mid fission only requires 51.9% of the FLOPS of early fission since adding a new task only adds a small number of new decoder parameters with one last block for each new task (0.017% FLOPS). Mid fission is also the only method with a positive $\%$ improvement and PIPFIP score.

4.3. Real-world Results

We also test our proposed method on two real-world datasets, NYUDv2[29] and Scannet[8]. We chose to train and evaluate on the tasks of surface normals and semantic labels as they seem to have the greatest correlation and the most task-shared features as discovered by the ablation studies. Table 3 show multi-task results on the 13 class label subset. This dataset has very few training images so we initialize with our model with our network trained on

Method	Normals % $<$			Depth % $<$			Semantics	Imp	FLOPS	PIPFIP
	11.25	22.5	30	1.25	1.25^2	1.25^3	mIoU	%	billion	%
Single Task										
Normals	83	91.5	93.8	-	-	-	-	-	38.43	0
Depth	-	-	-	86.1	95.3	97.5	-	-	38.40	0
Semantics	-	-	-	-	-	50.3	-	-	38.57	0
No Loss Balancing										
Early	76.2	89.5	92.7	80.2	89.1	92	49.7	-5.38	74.54	0
E-Mid	81.3	91.4	94.0	78.3	87.3	90.4	50.0	-3.89	67.35	0
Mid	70.8	86.5	90.8	77.1	87.5	90.2	49.0	-9.21	**38.58**	0
Late	73.3	88.1	91.9	81.8	90.7	93.4	**50.5**	-5.39	38.66	0
Balancing $\lambda_d = 0.5$,$\lambda_n = 10$										
Early	84.6	92.5	94.7	78.7	88.9	91.7	**50.6**	-1.99	74.54	0
E-Mid	**86.2**	**93.3**	**95.2**	77.2	89.1	92.9	**50.6**	-1.86	67.35	0
Mid	85	**93.1**	**95.1**	**84.8**	**95.3**	**97.5**	**50.4**	**0.86**	**38.58**	**33.32**
Late	84.2	92.4	94.6	78.9	90.8	93.7	49.4	-2.87	38.66	0

Table 2: Joint 3 task results with different fission methods both with no loss balancing and when the loss is balanced ($\lambda_{\text{semantics}} = 1$, $\lambda_{\text{depth}} = 0.5$ and $\lambda_{\text{normals}} = 10$). N is the single task normals baseline, D the depth, and S the semantics.

	Normals % $<$				Semantics
Method	11.25	22.5	30	mAE	mIoU
[32] from [23]	21.8	43.1	54.9	32.3	16.1
[23]	23.2	45.7	57.5	31.1	17.7
Ours	**50.1**	**70.8**	**78.1**	**20.6**	**42.1**

Table 3: NYUDv2 13 class results with our proposed mid fission method compared to state of the art training on NYUDv2 13 class.

Scannetv2. We also show results on the 40 class label subset of NYUDv2 in Table 4 with many other state-of-the-art multitask and single task methods. Note that some train on normals from [9] quantized to RGB images. This creates several degrees of normal error, resulting in $\% < 11.25$ being much worse but the rest of the metrics being better. We do not recommend this as it is not indicative of actual surface normals but show our method quantized as well to evaluate properly.

Finally, we also train and evaluate on the Scannetv2 dataset in Table 5. We separate a subset of 10% of the scenes in the training set as validation set since there is no given validation split. We evaluated on this due to the large number of experiments done and GPU limitations for training. Many of the other methods such as [36] use different initialization to improve results. It is common to also crop sections of the test image with flips to evaluate several portions of the test image at full resolution in order to produce the best results. We avoided these techniques as they are not indicative of which architecture is most promising. Therefore, we evaluate by simply resizing the test image and running it through the network once.

	Normals % $<$				Semantics
Method	11.25	22.5	30	mAE	mIoU
Misra[27] from [11]*	48.6	76	86.5	15.2	34.8
Gao et al.[11]*	53.5	79.5	88.8	13.9	36.2
Ruder[34] from [11]*	49.7	77.1	88.0	14.8	34.9
Ours quantized*	40.9	82.7	90.5	15.2	38.5
Misra et al.[27]	39	54.4	60.2	34.1	19.3
Mousavian et al.[28]	N/A	N/A	N/A	N/A	39.2
Kokkinos et al.[19]	35.3	65.9	76.9	21.4	N/A
Xu et al.[37]	N/A	N/A	N/A	N/A	33.1
Hickson et al.[15]	59.5	72.2	77.3	19.7	N/A
Ours	**60.2**	**79.1**	**86.1**	**15.2**	**40.6**

Table 4: NYUDv2 40 class results with our proposed mid fission method compared to state of the art (fine-tuned). * Indicates the normals are from [9] that were then quantized to RGB images.

Other reported results are shown in the first half of the table for fairness followed by our trained results. These include our results for the backbone architecture [36], the standard early fission method, and our proposed mid fission method. Our proposed method is only slightly larger (0.017% more FLOPS) than the single-task architecture and 68.2% of the FLOPS of the standard early-fission method. We did not compare with methods that use several input sources (RGB+D,RGB+HHA) as we are evaluating on the single-to-many multi-task labeling domain.

Our proposed method considerably outperforms the competitive Adapnet++ baseline[36]. Figure 2 shows some qualitative results from the Scannetv2 validation set showing our impressive results.

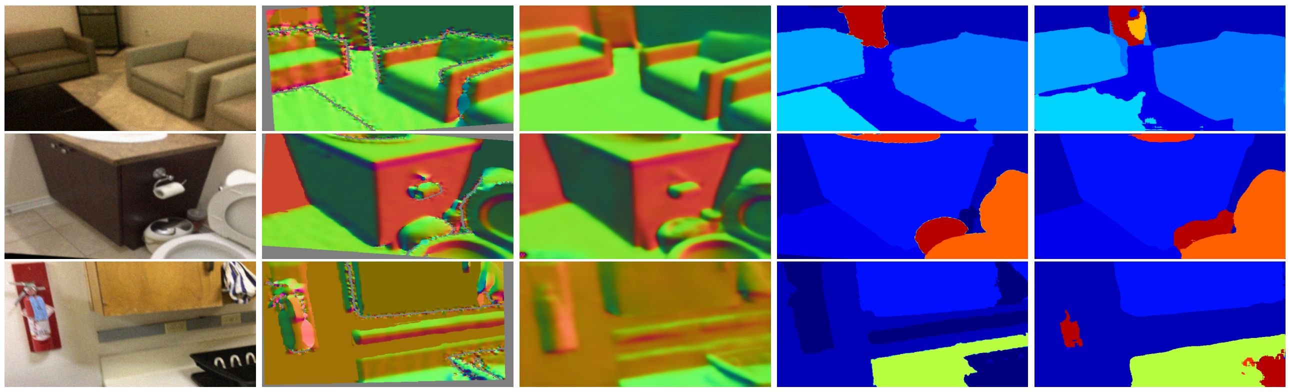


Figure 2: Illustration of the Scannetv2 data, together with the predictions of our model on them. Columns, from left to right: RGB image, ground truth surface normals, our normal predictions, ground truth semantic labels, then our semantic predictions.

	Normals % <				Semantics
Method	11.25	22.5	30	mAE	mIoU
[2]	N/A	N/A	N/A	N/A	27.5
Adapnet	N/A	N/A	N/A	N/A	47.3
DeepLabv3	N/A	N/A	N/A	N/A	50.1
[36]	N/A	N/A	N/A	N/A	50.3
[15]	50.1	63.2	68.2	28.8	N/A
Ours [36]	N/A	N/A	N/A	N/A	56.2
Ours Early	50.8	72.9	80.3	18.6	56
Ours Mid	**57.8**	**77.4**	**83.8**	**16.3**	**62.5**

Table 5: Scannetv2 validation set results. with our proposed mid fission method compared to state of the art.

4.4. Real-time Mobile Model Results

Multi-task networks are even more important to resource constrained devices such as robots, mobile phones, and laptops. Due to this, we tested the same method of splitting the decoder in several places on two separate mobile device specific architectures with results shown in Table 6. We tested on 3 tasks, surface normals, semantic labels, and disparity. We did 3 tasks instead of just 2 as in the server model to show the potential of this method for mobile inference. The first architecture, Mobilenetv1-unet is a variant of the architecture used in [15] with receptive field enlarged using 5x5 convolution kernels and prelu layers. The second architecture Mobilenet-V3 w/attention uses [16] as the encoder followed by a rASPP layer and two self attention blocks in the decoder. We have included model size, latency, and runtime memory on a Pixel 3XL. The Mobilenetv3 model is smaller so it has lower latency and worse results. What is important to note that for both of these, the same findings we found in the server model hold true. Mid-fission is substantially faster and has smaller model size all while having comparable or even better accuracy than the early fission approach. In Table 6, latency is in milliseconds on a Pixel 3XL, model size is the size (MB) of the tflite model with float16 quantization, and memory is runtime memory MB. An average of 50 runs is used for these metrics.

Metrics	MobilenetV1-Unet			MobilenetV3-Attention		
	Early	Mid	Late	Early	Mid	Late
Normals % <						
11.25	**49.98**	46.67	49.16	40.46	**43.99**	39.43
22.5	66.43	**67.61**	**67.87**	66.5	**68.**	65.24
30	74.16	**77.76**	75.75	76.6	**77.51**	75.07
Disparity % <						
1.25	**96.12**	95.5	91.22	**91.98**	79.52	30.21
1.25^2	99.2	**99.63**	98.56	**99.82**	98.51	74.66
1.25^3	**99.97**	**99.99**	99.68	**100**	99.76	96.4
Semantic Accuracy	81.4	**84.94**	80.12	60.67	**67.9**	60.45
Latency (ms)	30.1	**18.78**	31.94	11.59	**9.61**	23.9
Model size (MB)	1.8	**1.3**	**1.3**	2	**1.7**	**1.7**
Memory (MB)	157.1	110.9	**104.3**	72.5	**69.1**	71.6

Table 6: Fission methods on 2 mobile architectures on Scannetv2 with NYU13 semantic labels and losses as described in Section 3.1

5. Conclusion

We explore fission methods for multi-task pixel-wise prediction tasks. We show ablation studies on different fission methods and find that our proposed mid-fission method outperforms standard early fission methods with only 0.017% more FLOPS per task. We evaluate this on several tasks on several datasets producing SoTA results of multiple modalities on multiple datasets. We also evaluate this on mobile architectures to demonstrate its use for resource constrained devices such as mobile phones and robots. We open source the model and example code[1]. We hope this can help multi-task pixel-wise prediction methods achieve better results using this simple architecture change.

References

[1] Paper code. https://github.com/StevenHickson/Adapnet-Shape.

[2] 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.

[3] Cesar Cadena and Jana Košecká. Semantic segmentation with heterogeneous sensor coverages. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2639–2645. IEEE, 2014.

[4] Richard Caruana. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48. Morgan Kaufmann, 1993.

[5] Daniel Castro, Steven Hickson, Vinay Bettadapura, Edison Thomaz, Gregory Abowd, Henrik Christensen, and Irfan Essa. Predicting daily activities from egocentric images using deep learning. In *proceedings of the 2015 ACM International symposium on Wearable Computers*, pages 75–82. ACM, 2015.

[6] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017.

[7] Camille Couprie, Clément Farabet, Laurent Najman, and Yann LeCun. Indoor semantic segmentation using depth information: 1st international conference on learning representations, iclr 2013. In *1st International Conference on Learning Representations, ICLR 2013*, 2013.

[8] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas A Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, volume 2, page 10, 2017.

[9] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[10] David F Fouhey, Abhinav Gupta, and Martial Hebert. Data-driven 3d primitives for single image understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3392–3399, 2013.

[11] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[12] Qichuan Geng, Zhong Zhou, and Xiaochun Cao. Survey of recent progress in semantic image segmentation with cnns. *Science China Information Sciences*, 61(5):051101, 2018.

[13] Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European conference on computer vision*, pages 345–360. Springer, 2014.

[14] Caner Hazirbas, Lingni Ma, Csaba Domokos, and Daniel Cremers. Fusenet: Incorporating depth into semantic segmentation via fusion-based cnn architecture. In *Asian conference on computer vision*, pages 213–228. Springer, 2016.

[15] Steven Hickson, Karthik Raveendran, Alireza Fathi, Kevin Murphy, and Irfan Essa. Floors are flat: Leveraging semantics for real-time surface normal prediction. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

[16] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1314–1324, 2019.

[17] Omid Hosseini Jafari, Oliver Groth, Alexander Kirillov, Michael Ying Yang, and Carsten Rother. Analyzing modular cnn architectures for joint depth prediction and semantic segmentation. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4620–4627. IEEE, 2017.

[18] 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.

[19] Iasonas Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *CVPR*, volume 2, page 8, 2017.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[21] Ryohei Kuga, Asako Kanezaki, Masaki Samejima, Yusuke Sugano, and Yasuyuki Matsushita. Multi-task learning using multi-modal encoder-decoder networks with shared skip connections. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 403–411, 2017.

[22] Lubor Ladicky, Jianbo Shi, and Marc Pollefeys. Pulling things out of perspective. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 89–96, 2014.

[23] Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-end multi-task learning with attention. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[24] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440. IEEE, 2014.

[25] Kevis-Kokitsi Maninis, Ilija Radosavovic, and Iasonas Kokkinos. Attentive single-tasking of multiple tasks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[26] John McCormac, Ankur Handa, Stefan Leutenegger, and Andrew J Davison. Scenenet rgb-d: Can 5m synthetic images beat generic imagenet pre-training on indoor segmentation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, volume 4, 2017.

[27] 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.

[28] Arsalan Mousavian, Hamed Pirsiavash, and Jana Košecká. Joint semantic segmentation and depth estimation with deep convolutional networks. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 611–619. IEEE, 2016.

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

[30] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528, 2015.

[31] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *arXiv preprint arXiv:1907.01341*, 2019.

[32] Zhongzheng Ren and Yong Jae Lee. Cross-domain self-supervised multi-task feature learning using synthetic imagery. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[33] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

[34] Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. Learning what to share between loosely related tasks. *arXiv preprint arXiv:1705.08142*, 2017.

[35] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

[36] Abhinav Valada, Rohit Mohan, and Wolfram Burgard. Self-supervised model adaptation for multimodal semantic segmentation. *International Journal of Computer Vision (IJCV)*, jul 2019. Special Issue: Deep Learning for Robotic Vision.

[37] Dan Xu, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 675–684, 2018.

[38] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3712–3722, 2018.

[39] Richard Zhang, Stefan A Candra, Kai Vetter, and Avideh Zakhor. Sensor fusion for semantic segmentation of urban scenes. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1850–1857. IEEE, 2015.