Zero-Shot Task Transfer: Supplementary Section

Arghya Pal, Vineeth N Balasubramanian
Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad, INDIA
{cs15resch11001, vineethnb}@iith.ac.in

In this section, we include more details on obtaining the task correlation matrix, Γ, described in Section 3, ablation studies with varying source tasks, as well as additional results and comparisons, which could not be included in the main paper due to space constraints. All notations are as followed in the main paper, unless defined separately. Please find the references, that we will be referring here, are cited at the "References" Section at the end of this supplementary section.

7. More on Task Correlation

Dawid-Skene method: As mentioned in Section 3 and Section 4, we used the well-known Dawid-Skene (DS) method [2][10] to aggregate votes from human users to compute the task correlation matrix Γ. We now describe the DS method.

We assume a total of M annotators providing labels for N items, where each label belongs to one of K classes. DS associates each annotator $m \in M$ with a $K \times K$ confusion matrix Θ^m to measure an annotator's performance. The final label is a weighted sum of annotator's decisions based on their confusion matrices, i.e. Θ^ms where $\{m = 1, \cdots, M\}$. Each entry of the confusion matrix $\theta_{lg} \in \Theta^m$ is the probability of predicting class g when the true class is l. A true label of an item $n \in N$ is y_n and the vector $\mathbf{y}$ denotes true label for all items, i.e. $\mathbf{y} = \{y_1, \cdots, y_N\}$. Let's denote $\xi_{m,n}$ as annotator m's label for item n, i.e. if the annotator labeled the item as $k \in K$, we write $\xi_{m,n} = k$. Let matrix Ξ ($\xi_{m,n} \in \Xi$) denote all labels for all items given by all annotators. The DS method computes the annotators' error tensor $\mathcal{C}$, where each entry $c_{mlg} \in \mathcal{C}$ denotes the probability of annotator m giving label l as label g for item n. The joint likelihood $\mathcal{L}(.)$ of true labels and observed labels Ξ can hence be written as:

$$\mathcal{L}(\mathcal{C}; \mathbf{y}, \Xi) = \prod_{j=1}^{N} \prod_{m=1}^{M} \prod_{g=1}^{K} \left(c_{m y_j g}\right)^{\mathbf{1}_{(\xi_{m,j}=k)}} \tag{6}$$

Maximizing the above likelihood provides us a mechanism to aggregate the votes from the annotators. To this end, we find the maximum likelihood estimate using Expectation Maximization (EM) for the marginal log likelihood below:

$$l(\mathcal{C}) = \log(\sum \mathcal{L}(\mathcal{C}; \mathbf{y}, \Xi)) \tag{7}$$

The E step is given by:

$$\mathbb{E}[\log \mathcal{L}(\mathcal{C}; \mathbf{y}, \Xi)] = \prod_{j=1}^{N} p(y_j = l|\mathcal{C}, \Xi) \log \prod_{m=1}^{M} \prod_{g=1}^{K} \left(c_{mlg}\right)^{\mathbf{1}_{(\xi_{m,j}=k)}} \tag{8}$$

The M step subsequently computes the $\mathcal{C}$ estimate that maximizes the log likelihood:

$$\hat{c}_{mlg} = \frac{\prod_{j=1}^{N} p(y_j = l|\mathcal{C}, \Xi)\mathbf{1}_{(\xi_{m,j}=k)}}{\prod_{k'=1}^{K} \left(\prod_{j=1}^{N} p(y_j = l|\mathcal{C}, \Xi)\mathbf{1}_{(\xi_{m,j}=k')}\right)} \tag{9}$$

$$\hat{p}(y_j) = \frac{\prod_{j=1}^{N} \mathbf{1}_{(\xi_{m,j}=k)}}{N} \tag{10}$$

Once we get annotators' error tensor $\mathcal{C}$ and $p(y_j)$ from Equations 9 and 10, we can estimate:

$$p(y_j = l|\mathcal{C}, \Xi) = \frac{\exp(\sum_{m=1}^{M} \sum_{g=1}^{K} \log \hat{c}_{mlg} \mathbf{1}_{(\xi_{m,j}=k)})}{\sum_{l'=1}^{K} \exp(\sum_{m=1}^{M} \sum_{g=1}^{K} \log \hat{c}_{ml'g} \mathbf{1}_{(\xi_{m,j}=k)})} \tag{11}$$

for all $j \in N$ and $l \in K$. To get the final predicted label, we adopt a winner-takes-all strategy on $p(y_j = l)$ across all $l \in K$. We request readers to refer [10] and [2] for more details.

Implementation: In our experiments, as mentioned before, we considered the Taskonomy dataset [8]. This dataset has 26 vision-related tasks. We are interested in finding the task correlation for each pair of tasks in $\{\tau_1, \cdots, \tau_{26}\}$. Let us assume that we have M annotators. To fit our model in the DS framework, we flatten the task correlation matrix

$\Gamma^{26\times26}$ (described in section 3) in *row-major* order to get item set $N = \{n_1, \cdots, n_{(26\times26)}\}$. For each item $n_k \in N$, the annotator is asked to give task correlation label on a scale of $\{+3, +2, +1, 0, -1\}$. On that scale, a $+3$ denotes *self relation*, $+2$ describes *strong relation*, $+1$ implies *weak relation*, 0 to mention *abstain* and -1 to denote *no relation* between two tasks $\tau_i, \tau_j \in \tau$. After getting annotators' vote, we build matrix Ξ. Subsequently, we find annotators' error tensor $\mathcal{C}$ (Equation 6), likelihood estimation (Equations 7, 8, 9, 10). We get predicted class labels after a *winner-takes-all* in Equation 11. Predicted class labels are the task correlation we wish to get. We get the final task correlational matrix $\Gamma^{26\times26}$, after a de-flattening of y_j^{true} for all $j = 1, \cdots, N$.

Annotators	RANSAC[7]	LR[3]	G3D[9]	TN[8]
3	28%	22%	29%	40%
10	51%	29%	31%	52%
20	90%	82%	92%	42%
30	*88%*	*81%*	*72%*	*64%*
35	88%	82%	75%	61%
40	90%	72%	69%	63%
45	87%	80%	61%	70%
50	90%	82%	72%	50%

Table 6: ***Win rates (%)* of $TTNet_6$ with a varied number of annotators**. We considered the *win rate* (%) on angular error. Columns are state-of-the-art methods and rows are our $TTNet_6$ trained using different Γ_is, where $i = \{3, 10, 20, 30, 35, 40, 45, 50\}$.

Ablation study on different number of annotators: The results in the main paper were performed with 30 annotators. In this section, we studied the robustness of our method when Γ is obtained by varying the number of annotators, M_i, where $i \in \{3, 10, 20, 30, 35, 40, 45, 50\}$. Table 6 shows a *win rate* (%) [8] on the camera pose estimation task using TTNet $_6$ (when 6 source tasks are used). While there are variations, we identified $i = 30$ as the number of annotators where the results are most robust, and used this setting for the rest of our experiments (in the main paper).

8. Ablation Studies on Varying Known Tasks

In this section, we present two ablation studies w.r.t. known tasks, *viz*, (i) number of known tasks and (ii) choice of known tasks. These studies attempt to answer the questions: how many known tasks are sufficient to adapt to zero-shot tasks in the considered setting? Which known tasks are more favorable to transfer to zero-shot tasks? While an exhaustive study is infeasible, we attempt to answer these questions by conducting a study across six different models: TTNet_4, TTNet_6, TTNet_{10}, TTNet_{15}, TTNet_{18}, and TTNet_{20} (where the subscript denotes the number of source tasks considered). We used *win rate* (%) against [8] for each of the zero-shot tasks. Table 7 shows the results of our studies with varying number and choice of known source tasks. Expectedly, a higher number of known tasks provides improved performance. It is observed from the table that our methodology is fairly robust despite changes in choice of source tasks, and that TTNet_6 provides a good balance by having a good performance even with a low number of source tasks. Interestingly, most of the source tasks considered for TTNet_6 (autoencoding, denoising, 2D edges, occlusion edges, vanishing point, and colorization) are tasks that do not require significant annotation, thus providing a model where very little source annotation can help generalize to more complex target tasks on the same domain.

9. Other Results

In this section, we will discuss different results that we did not cover due the page constraint in the main paper.

9.1. Qualitative Results on Cityscapes Dataset

To further study the generalizability of our models, we finetuned TTNet on the Cityscapes dataset [1]. We get source task model parameters (trained on Taskonomy dataset) to train TTNet_6. We then finetuned TTNet_6 on the segmentation model parameters trained on Cityscapes data. (We modified one source task, i.e. autoencoding to segmentation, of our proposed TTNet $_6$, see table 7, 3^{rd} row. All other source tasks are unaltered.) Results of the learned model parameters for four zero-shot tasks, i.e. Surface normal, depth, 2D edge and 3D keypoint, are reported in Figure 7, with comparison to [8] (which is trained explicitly for these tasks). Despite the lack of supervised learning, the figure shows that tt is evident from the qualitative assessment (figure 7) that our model seems to capture more detail.

9.2. More Qualitative Results

We report more qualitative results of: (i) room layout in Figure 8; (ii) surface normal estimation in Figure 9; (iii) depth estimation in Figure 10; and (iv) camera pose estimation in Figure 11.

9.3. Zero-shot to Known Task Transfer: Quantitative Evaluation

In continuation to our discussions in Section 5, we ask ourselves the question: are our regressed model parameters for zero-shot tasks capable of transferring to a known task? To study this, we consider the autoencoder-decoder parameters for a zero-shot task learned through our methodology, and finetune the decoder (fixing the encoder parameters) to a target known task, following the procedure in [8]. Table 8 shows the quantitative results when choosing the source (zero-shot) tasks as surface normal estimation (N) and room layout estimation (L). We compared our TTNet against [8] quantitatively by studying the *win rate* (%) of the two methods against other state-of-the-art methods: Wang *et al.* [6],

	Autoencoding	Object Class	Scene Class	Curvature	Denoising	2D Edges	Occlusion Edges	Ego motion	Cam Pose (fixed)	2D Key Point	3D Key Point	Cam Pose (non-fixed)	Matching	Reshading	Z-Depth	Distance	Normals	Room Layout	2.5D Segmentation	2D Segmentation	Semantic Segmentation	Vanishing Point	Jig-Saw Puzzle	Random Projection	Colorization	*Win Rate (Normal)* (%)	*Win Rate (Room Layout)* (%)	*Win Rate (Depth)* (%)	*Win Rate (Camera Ps.(fixed)* (%)
4	✓	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	79%	62%	71%	71%
	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	71%	58%	61%	59%
	✓	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗	✓	75%	79%	79%	52%
6	✓	✗	✗	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	✓	**88%**	**85%**	**87%**	**89%**
	✓	✗	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	✓	**87%**	**86%**	**86%**	**89%**
	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗	**85%**	**88%**	**86%**	**82%**
10	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	85%	84%	87%	85%
	✓	✓	✗	✓	✓	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓	87%	88%	91%	92%
	✓	✓	✓	✓	✓	✗	✗	✓	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✓	✓	✗	✗	✗	✓	88%	83%	81%	89%
15	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓	✓	88%	85%	91%	93%
	✓	✗	✗	✓	✓	✓	✗	✓	✗	✗	✗	✓	✓	✓	✗	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓	89%	87%	81%	85%
18	✓	✓	✓	✓	✓	✓	✗	✓	✗	✗	✓	✓	✓	✓	✗	✓	✗	✗	✓	✗	✓	✓	✓	✓	✓	93%	91%	97%	91%
	✓	✗	✓	✓	✓	✓	✗	✓	✗	✓	✗	✓	✓	✓	✗	✓	✗	✗	✓	✓	✓	✓	✓	✓	✓	95%	91%	93%	94%
20	✓	✗	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✗	✓	✗	✗	✓	✓	✓	✓	✓	✓	✓	94%	91%	93%	89%

Table 7: **Ablation study of different TTNets with different combination of source tasks.** Row numerals, i.e. 4, 6, 10, 15, 18 and 20, indicate the number of source tasks that have been considered to train the model. A blue box (with ✓) indicates a source task. The last four columns indicate *win rates* (%) for each of the 4 zero-shot tasks considered against [8]. We observe that our methodology is fairly robust and that TTNet_6 gives a fairly good performance even with a low number of source tasks.

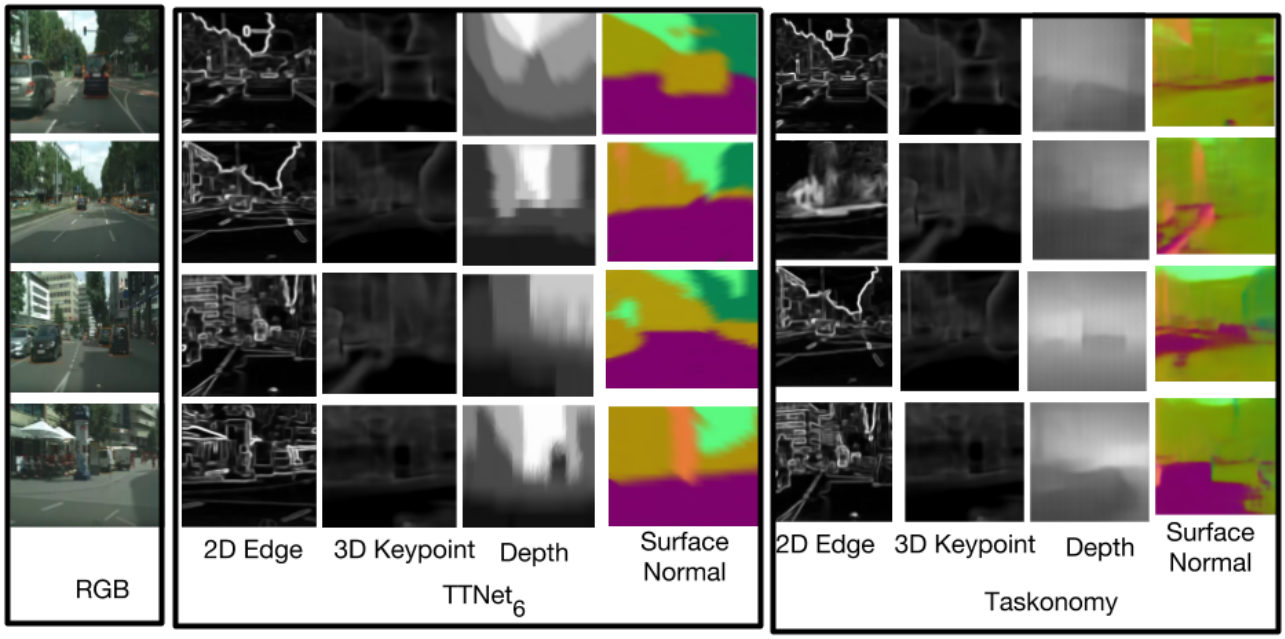

Figure 7: **Results on Cityscapes data.** We finetuned TTNet_6 on the Cityscapes dataset [1], and the surface normal, depth, 2D edge and 3D keypoint results are reported using the model parameters learned by TTNet_6.

G3D [9], and full supervision. However, it is worthy to mention that our parameters are obtained through the proposed zero-shot task transfer, while all other comparative methods are explicitly trained on the dataset for the task.

9.4. Alternate Methods for Task Correlation Computation

In our results so far, we studied the effectiveness of computing the task correlation matrix by aggregation of crowd

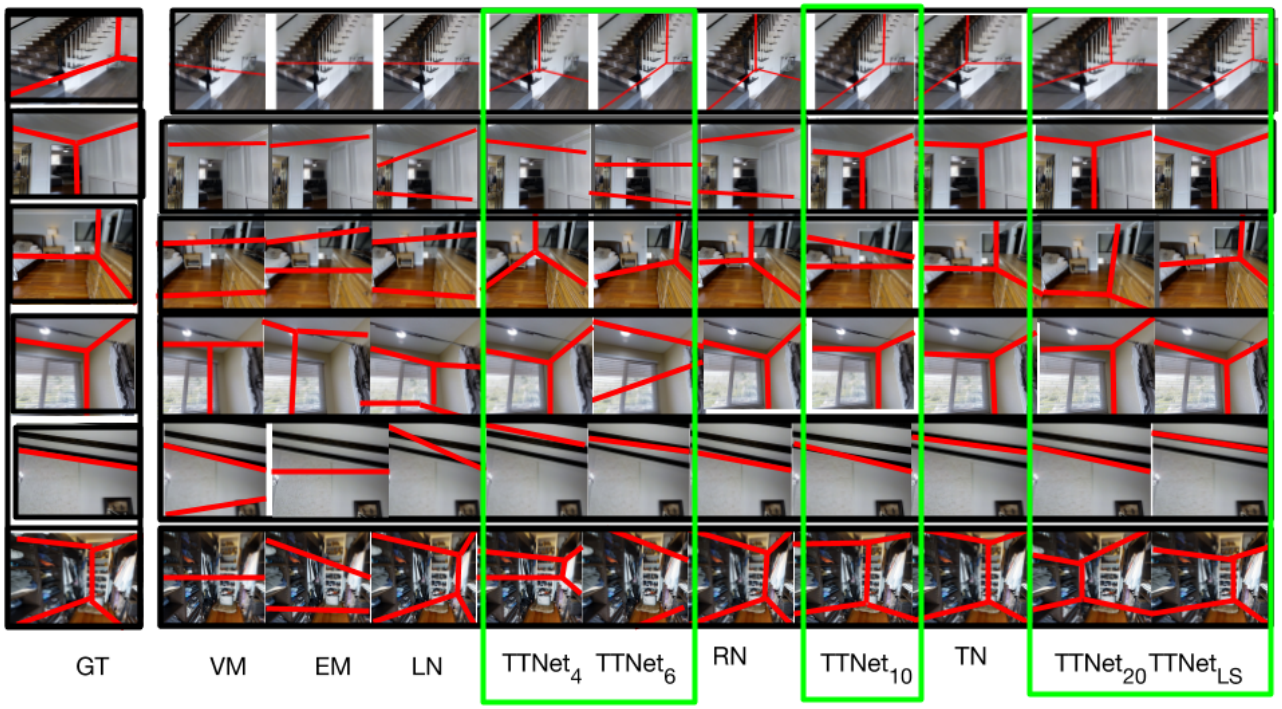

Figure 8: More results of room layout estimation

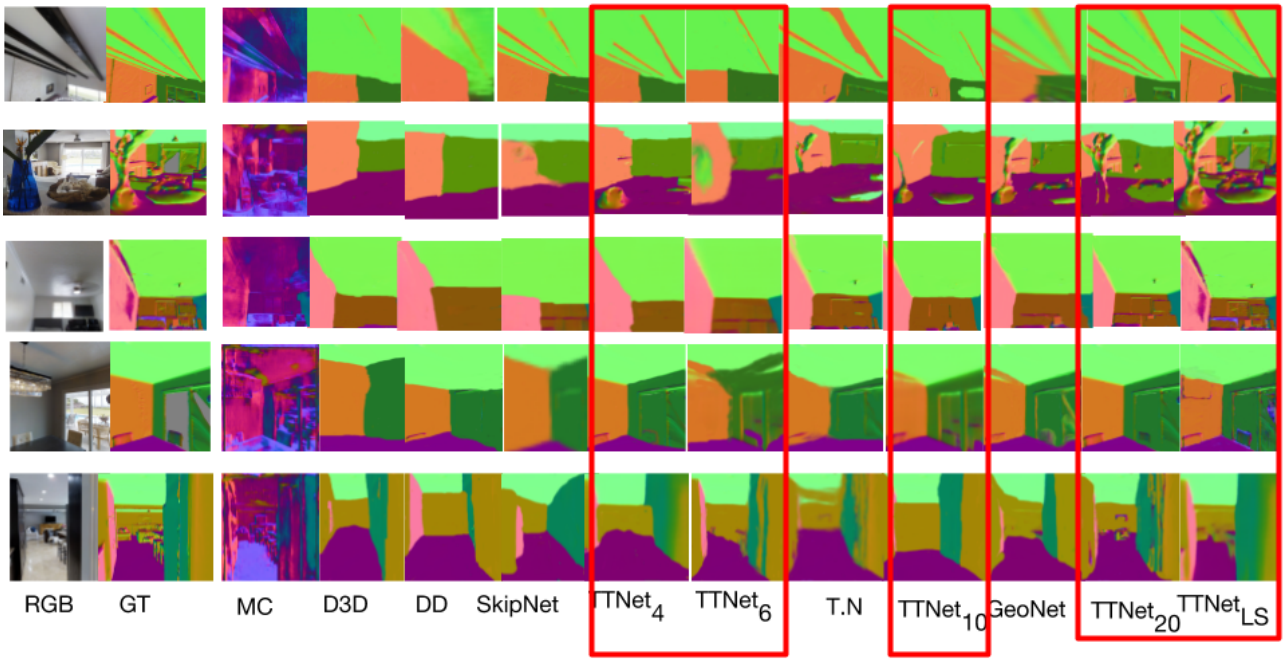

Figure 9: More results of surface normal estimation

votes. In this section, we instead use the task graph obtained in [8] to obtain the task correlation matrix Γ. We call this matrix Γ_{TN}. Figure 12 shows a qualitative comparison of TTNet$_6$ where the Γ_{TN} is obtained from the taskonomy graph, and Γ is based on crowd knowledge. It is evident that our method shows promising results on both cases.

It is worthy to note that although one can use the taskonomy graph to build Γ: (i) the taskonomy graph is model and data specific [8]; while Γ coming from crowd votes does not explicitly assume any model or data and can be easily

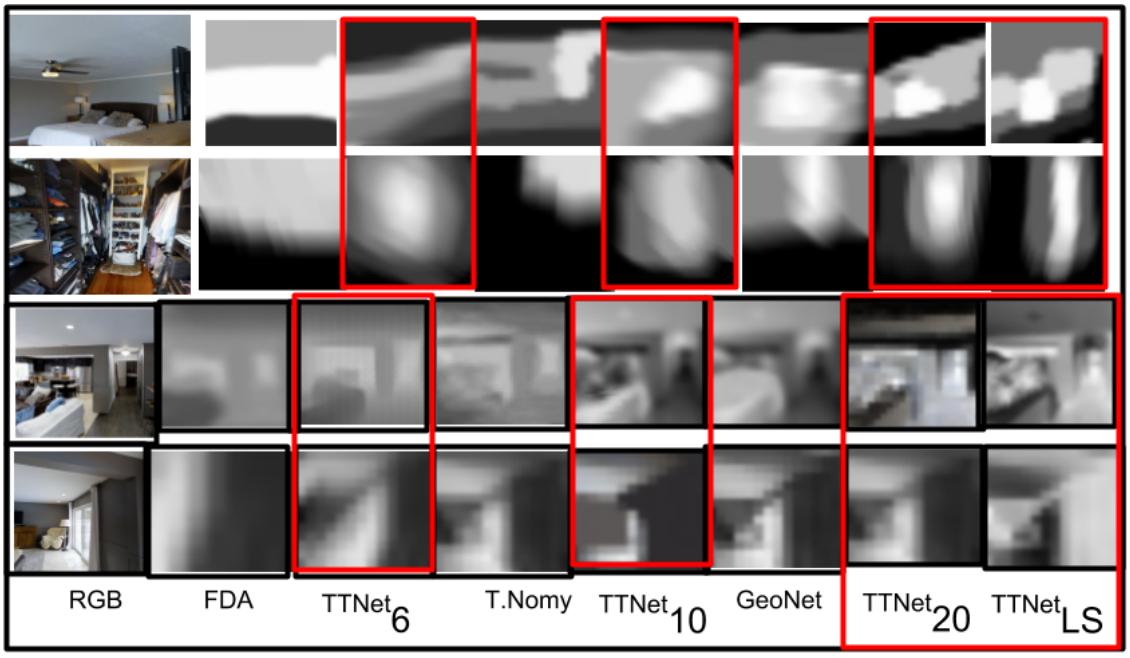

Figure 10: More results of depth estimation

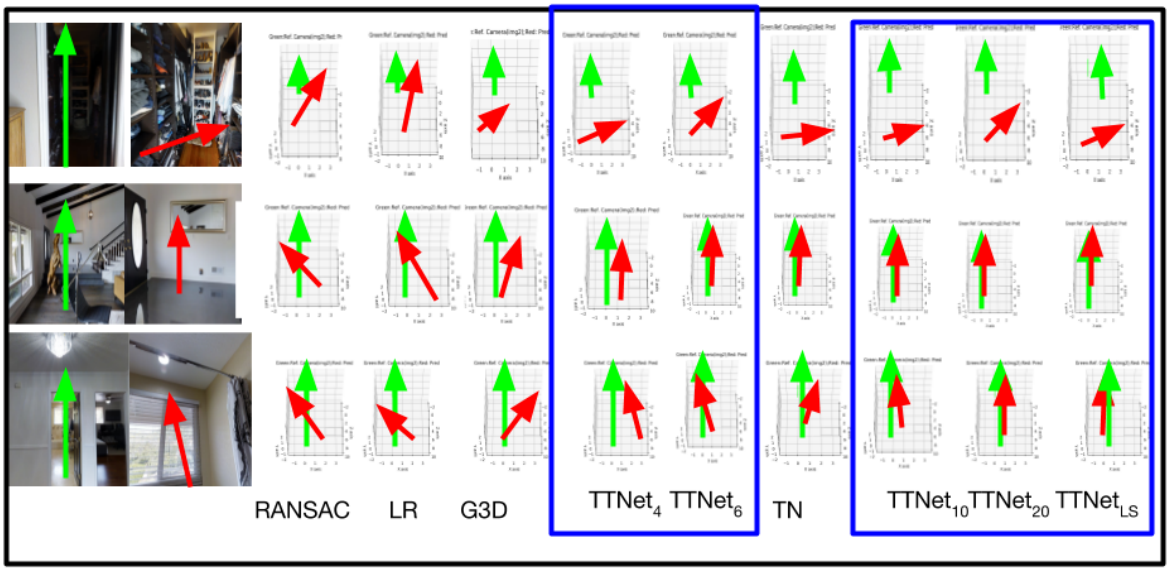

Figure 11: More results of camera pose estimation

obtained; (ii) during the process of building the taskonomy graph, an explicit access to zero-shot task ground truth is unavoidable; while, constructing Γ from crowd votes is possible without accessing any explicit ground truth.

9.5. Evolution of TTNet

Thus far, we showed the final results of our meta-learner after the model is fully trained. We now ask the question - how does the training of the TTNet model progress over training? We used the zero-shot task model parameters from TTNet_6 during its course of training, and Figure 13 shows qualitative results of different epochs of four zero-shot tasks over the training phase. The results show that the model's training progresses gradually over the epochs, and the model obtains promising results in later epochs. For example, in Figure 13(a), finer details such as wall boundaries, sofa, chair and other minute details are learned in later epochs.

Model	TTNet$_6$						Taskonomy					
	Wang		Zamir		Full Sup		Wang		Zamir		Full Sup	
	N	L	N	L	N	L	N	L	N	L	N	L
Depth	85	**87**	81	**97**	**67**	42	**98**	85	**92**	88	60	**46**
2.5 D	**88**	75	**75**	**81**	**89**	35	**88**	**77**	73	88	85	**39**
Curvature	**84**	87	**91**	58	**86**	47	78	**89**	88	**78**	60	**50**

Table 8: **Zero-shot to known task transfer.** We consider the autoencoder-decoder parameters for a zero-shot task learned through our method, and finetune the decoder (fixing the encoder) to a target known task, following the procedure in [8]. Source tasks (zero-shot) are surface normal (N), and, room layout (L). Target tasks are depth, 2.5D segmentation and curvature. *Win rates* (%) of task transfer with respect to self-supervised methods, such as, Wang *et al.* [5], Zamir *et al.* [9] as well as fully supervised setting are shown (all values are in %), with bold face numerals denoting winning entries.

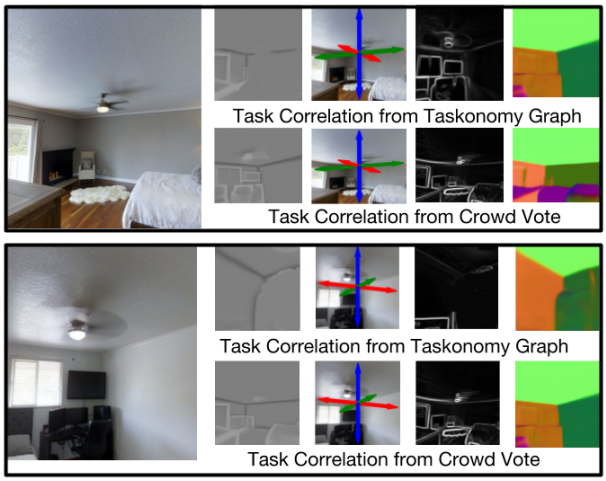

Figure 12: **Qualitative results of TTNet$_6$ when task correlation matrix (Γ) is obtained from task graph computed in [8].** We studied considering the task graph computed in [8] (instead of crowd vote) to build the task correlation matrix Γ_{TN}. First column represents RGB image and, subsequent columns (from 2^{nd} to 4^{th} columns) are zero-shot tasks: curvature, vanishing points, 2D key point and surface normal estimation

9.6. Analyzing Importance of Known Tasks

One can use the latent task basis to estimate this. We followed GO-MTL [4] to compute the task basis. Optimal model parameters of known tasks are mapped to a low-dim vector space R using an autoencoder, before applying GO-MTL. We used ResNet-18 for encoder-decoder, dim of R as 100, and dim of basis as 8. More specifically, optimal model parameters of each known task are mapped to a low-dimensional space R, i.e. $\mathcal{S} : \Theta_{\tau_i} \rightarrow R_i$. $S(.)$ is an autoencoder trained on model parameters of known tasks $\{\Theta^*_{\tau_1}, \cdots, \Theta^*_{\tau_m}\}$, i.e. $\min_J \sum_{i=1}^{m} ||\mathcal{S}(\Theta_{\tau_i}; J) - \Theta^*_{\tau_i}||^2 + \lambda \sum_{(x,y)\in(X_{\tau_i},\mathbf{y}_{\tau_i})} \mathcal{L}\big(\mathcal{D}_{\tilde{\Theta}_{D\tau_i}}(\mathcal{E}_{\tilde{\Theta}_{E\tau_i}}(x)), y\big)$ (similar to Eqn 5, main paper). $\mathcal{S}(.)$ infers latent representation R_{zero} for regressed model parameter of zero-shot task $\Theta_{\tau_{zero}}$. We used ResNet-18 both for encoder-decoder, dimension of R as 100, and the dimension of task basis as 8. We can then have task matrix $\mathbf{W}_{100X26} = \mathbf{L}_{100X8}\mathbf{S}_{8X26}$, comprised of all R_i and R_{zero}.

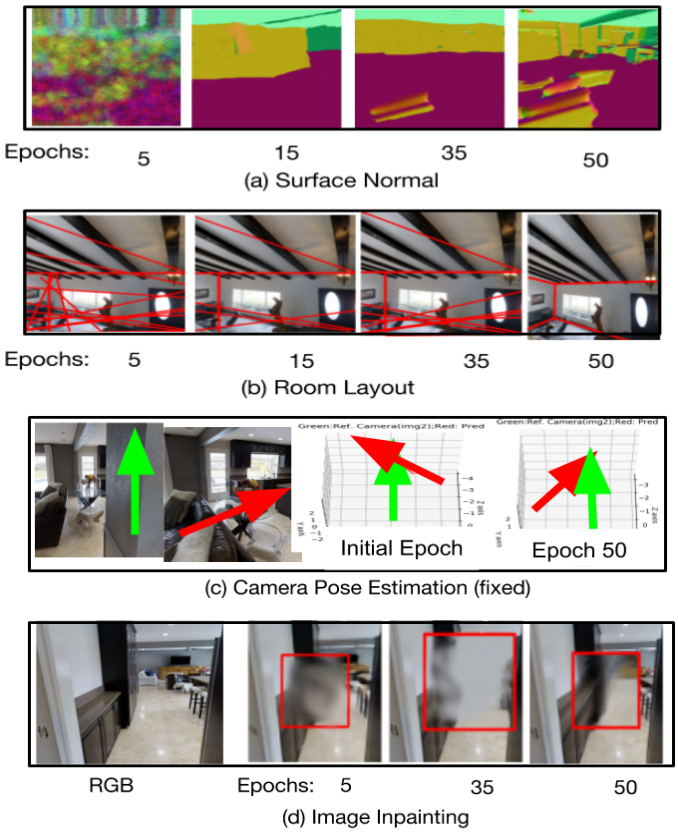

Figure 13: **Zero-shot tasks results during the training of TTNet.** We regressed zero-shot task parameters from TTNet$_6$ during its course of training. The qualitative results show the gradual learning of the model parameters of the epochs.

9.7. Comparison against random baseline

A comparison of Win rates (in %) of TTNet over random baseline (against Taskonomy) on 1K samples is shown in Figure 14. TTNet outperforms the baseline by a significant amount.

Auto(100/12)	Cur(92/14)	Den(88/12)	2DEg(99/24)	Oclu.Eg(94/6)	2DKey(94/13)	3Dkey(89/14)
Resd(99/15)	Z-dpth(92/9)	Dis(91/6.)	Nrml(95/8)	Egom(96/6)	VanPts(91/11)	2DSg(95/14)
2.5 Sg(87/14)	SemSeg(82/9)	Jigsw(84/6)	Layot(82/15)	ObjCls(92/10)	Matchng(97/8)	ScnCls(89/18)
Camera Pose (fxd) (78/3)		Camra Ps(non fxd)(70/2)		In-Ptng(86/4)	Clriztn(96/21)	Clsficatn(86/9)

Figure 14: Winrate (in %) of comparison of TTNet and Taskonomy against a random baseline

References

[1] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 3

[2] A. P. Dawid and A. M. Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied statistics*, pages 20–28, 1979. 1

[3] S. Korman and R. Litman. Latent ransac. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6693–6702, 2018. 2

[4] A. Kumar and H. Daume III. Learning task grouping and overlap in multi-task learning. *arXiv preprint arXiv:1206.6417*, 2012. 6

[5] X. Wang, D. Fouhey, and A. Gupta. Designing deep networks for surface normal estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 539–547, 2015. 6

[6] Y.-X. Wang, D. Ramanan, and M. Hebert. Learning to model the tail. In *NIPS*, 2017. 2

[7] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014. 2

[8] A. R. Zamir, A. Sax, W. Shen, L. Guibas, J. Malik, and S. Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3712–3722, 2018. 1, 2, 3, 4, 6

[9] A. R. Zamir, T. Wekel, P. Agrawal, C. Wei, J. Malik, and S. Savarese. Generic 3d representation via pose estimation and matching. In *European Conference on Computer Vision*, pages 535–553. Springer, 2016. 2, 3, 6

[10] C. Zhu, H. Xu, and S. Yan. Online crowdsourcing. *CoRR*, abs/1512.02393, 2015. 1