Zero-Shot Task Transfer

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

Abstract

In this work, we present a novel meta-learning algorithm that regresses model parameters for novel tasks for which no ground truth is available (zero-shot tasks). In order to adapt to novel zero-shot tasks, our meta-learner learns from the model parameters of known tasks (with ground truth) and the correlation of known tasks to zero-shot tasks. Such intuition finds its foothold in cognitive science, where a subject (human baby) can adapt to a novel concept (depth understanding) by correlating it with old concepts (hand movement or self-motion), without receiving an explicit supervision. We evaluated our model on the Taskonomy dataset, with four tasks as zero-shot: surface normal, room layout, depth and camera pose estimation. These tasks were chosen based on the data acquisition complexity and the complexity associated with the learning process using a deep network. Our proposed methodology outperforms state-of-the-art models (which use ground truth) on each of our zero-shot tasks, showing promise on zero-shot task transfer. We also conducted extensive experiments to study the various choices of our methodology, as well as showed how the proposed method can also be used in transfer learning. To the best of our knowledge, this is the first such effort on zero-shot learning in the task space.

1. Introduction

The major driving force behind modern computer vision, machine learning, and deep neural network models is the availability of large amounts of curated labeled data. Deep models have shown state-of-the-art performances on different vision tasks. Effective models that work in practice entail a requirement of very large labeled data due to their large parameter spaces. Expecting availability of large-scale hand-annotated datasets for every vision task is not practical. Some tasks require extensive domain expertise, long hours of human labor, expensive data collection sensors - which collectively make the overall process very expensive. Even when data annotation is carried out using crowdsourcing (e.g. Amazon Mechanical Turk), additional effort is required to measure the correctness (or goodness) of the obtained labels. Due to this, many vision tasks are considered expensive [43], and practitioners either avoid such tasks or continue with lesser amounts of data that can lead to poorly performing models. We seek to address this problem in this work, viz., to build an alternative approach that can obtain model parameters for tasks without any labeled data. Extending the definition of zero-shot learning from basic recognition settings, we call our work *Zero-Shot Task Transfer*.

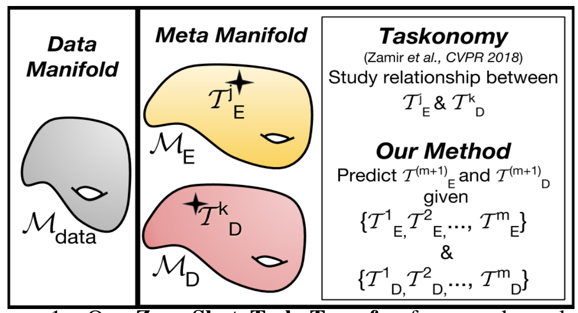

Figure 1: Our **Zero-Shot Task Transfer** framework explores the meta-manifold of Encoder-Decoder parameters of m known tasks (tasks for which exemplar ground truth is available), i.e. $\{\tau_E^1, \tau_E^2, \cdots, \tau_E^m\}$ and $\{\tau_D^1, \tau_D^2, \cdots, \tau_D^m\}$, to regress Encoder-Decoder parameters of zero-shot tasks, i.e. $\tau_E^{(m+1)}$ and $\tau_D^{(m+1)}$ (for which no ground truth is available). We compare our objective with that of Taskonomy [42], where the main objective is to find an optimum path from a source task and target task, to delineate the difference.

Cognitive studies show results where a subject (human baby) can adapt to a novel concept (e.g. depth understanding) by correlating it with known concepts (hand movement or self-motion), without receiving an explicit supervision [15]. In similar spirit, we present our meta-learning algorithm that computes Encoder-Decoder parameters for novel tasks for which no ground truth is available (called *zero-shot tasks*). In order to adapt to a zero-shot task, our meta-learner learns from the Encoder-Decoder parameters of known tasks (with ground truth) and their task correlation to the novel task. Formally, given the knowledge of

m known tasks $\{\tau^1, \cdots, \tau^m\}$, a meta-learner $\mathcal{F}(.)$ can be used to extrapolate parameters for $\tau^{(m+1)}$, a novel task. We are dropping the Encoder-Decoder subscripts, i.e. τ_E^i and τ_D^i for any i^{th} task τ^i, for the sake of simplicity of discussion.

However, with no knowledge of relationships between the tasks, it may not be plausible to learn a meta-learner, as its output could map to any point on the meta-manifold (see Figure 1). We hence consider the task correlation between known tasks and a novel task as an additional input to our framework. There could be different notions on how task correlation is obtained. In this work, we use the approach of wisdom-of-crowd for this purpose. Many vision [30] and non-vision machine learning applications [32], [38] encode such crowd wisdom in their learning methods. Harvesting task correlation knowledge from the crowd is fast, cheap, and brings domain knowledge. High-fidelity aggregation of crowd votes is used to integrate the task correlation between known and zero-shot tasks in our model. We however note that our framework can admit any other source of task correlation beyond crowdsourcing. (We show our results with other sources in the supplementary section.)

Our broad idea of leveraging task correlation can be found similar to the recently proposed idea of Taskonomy [42], but our method and objectives are different in many ways (see Figure 1): (i) Taskonomy studies task correlation to find a way to transfer one task model to another, while our method extrapolates to a zero-shot task, for which no labeled data is available; (ii) To adapt to a new task, Taskonomy requires a considerable amount of target labeled data, while our work does not require any target labeled data (which is, in fact, our objective); (iii) Taskonomy obtains a task transfer graph based on the representations learned by neural networks; while in this work, we leverage task correlation to learn new tasks; and (iv) Lastly, our method can be used to learn multiple novel tasks simultaneously. As stated earlier, though we use crowdsourced task correlation, any other compact notion of task correlation can easily be encoded in our methodology. More precisely, our proposal in this work is not to learn an optimal task relation, but to extrapolate to zero-shot tasks.

Our contributions can be summarized as follows:

- We propose a novel methodology to infer zero-shot task parameters that be used to solve vision tasks with no labeled data.
- The methodology can scale to solving multiple zero-shot tasks simultaneously, as shown in our experiments. Our methodology provides near state-of-the-art results by considering a smaller set of known tasks, and outperforms state-of-the-art models (learned with ground truth) when using all the known tasks, although trained with no labeled data.
- We also show how our method can be used in a transfer learning setting, as well as conduct various studies to study the effectiveness of the proposed method.

2. Related Work

We divide our discussion of related work into subsections that capture earlier efforts that are related to ours from different perspectives.

Transfer Learning: Reusing supervision is the core component of transfer learning, where an already learned model of a task is finetuned to a target task. From the early experimentation on CNN features [41], it was clear that initial layers of deep networks learn similar kind of filters, can can hence be shared across tasks. Methods such as in [3], [23] augment generation of samples by transferring knowledge from one category to another. Recent efforts have shown the capability to transfer knowledge from model of one task to a completely new task [34][33]. Zamir *et al.* [42] extended this idea and built a task graph for 26 vision tasks to facilitate task transfer. However, unlike our work, [42] cannot be generalized to a novel task without accessing the ground truth.

Multi-task Learning: Multi-task learning learns multiple tasks simultaneously with a view of task generalization. Some methods in multi-task learning assume a prior and then iterate to learn a joint space of tasks [7][19], while other methods [26][19] do not use a prior but learn a joint space of tasks during the process of learning. Distributed multi-task learning methods [25] address the same objective when tasks are distributed across a network. However, unlike our method, a binding thread for all these methods is that there is an explicit need of having labeled data for all tasks in the setup. These methods can not solve a zero-shot target task without labeled samples.

Domain Adaptation: The main focus of domain adaptation is to transfer domain knowledge from a data-rich domain to a domain with limited data [27][9]. Learning domain-invariant features requires domain alignment. Such matching is done either by mid-level features of a CNN [13], using an autoencoder [13], by clustering [36], or more recently, by using generative adversarial networks [24]. In some recent efforts [35][6], source and target domain discrepancy is learned in an unsupervised manner. However, a considerable amount of labeled data from both domains is still unavoidable. In our methodology, we propose a generalizable framework that can learn models for a novel task from the knowledge of available tasks and their correlation with novel tasks.

Meta-Learning: Earlier efforts on meta-learning (with other objectives) assume that task parameters lie on a low-dimensional subspace [2], share a common probabilistic prior [22], etc. Unfortunately, these efforts are targeted only to achieve knowledge transfer among known tasks and

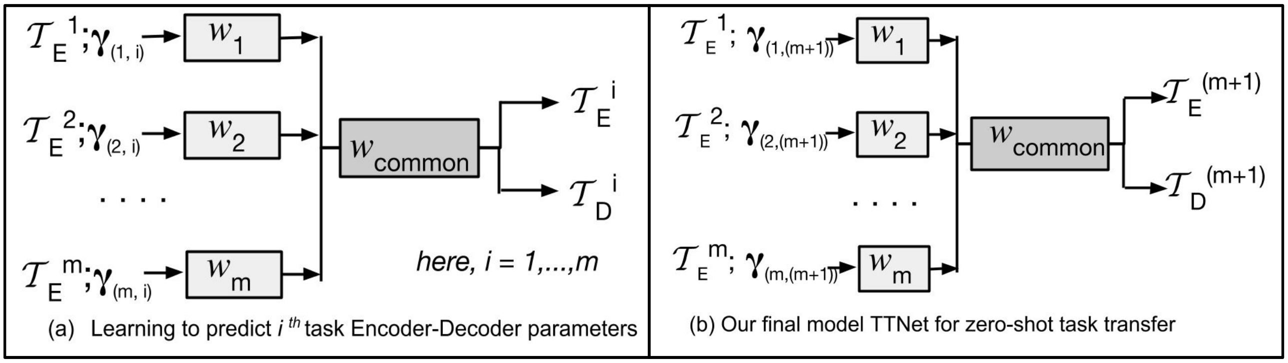

Figure 2: **Overview of our work.** Figure (a) represents the training phase of TTNet, where it learns a correspondence between task correlation $\gamma_{(i,j)}$ of τ^i and τ^j, and the physical distance of τ^i and τ^j on the meta-manifold, given the Encoder-Decoder parameters of m-known tasks and the task correlation The proposed TTNet gives Encoder-Decoder parameters of known tasks; (b) Once TTNet learns a correspondence between task correlation and the physical distance on the manifold, it can regress zero-shot task given the task correlation of zero-shot task and m-known tasks. Please see Section 3 for a more insight.

tasks with limited data. Recent meta-learning approaches consider all task parameters as input signals to learn a meta manifold that helps few-shot learning [28], [37], transfer learning [33] and domain adaptation [13]. A recent approach introduces learning a meta model in a model-agnostic manner [12][17] such that it can be applied to a variety of learning problems. Unfortunately, all these methods depend on the availability of a certain amount of labeled data in target domain to learn the transfer function, and cannot be scaled to novel tasks with no labeled data. Besides, the meta manifold learned by these methods are not explicit enough to extrapolate parameters of zero-shot tasks. Our method relaxes the need for ground truth for zero-shot tasks, by leveraging task correlation among known tasks and novel zero-shot tasks. To the best of our knowledge, this is the *first* such work that involves regressing model parameters of novel tasks without using any ground truth information for the task.

Learning with Weak Supervision: Task correlation is used as a form of weak supervision in our methodology. Recent methods such as [32][38] proposed generative models that use a fixed number of user-defined weak supervision to programatically generate synthetic labels for data in near-constant time. Alfonseca *et al.* [1] use heuristics for weak supervision to acccomplish hierachical topic modeling. Broadly, such weak supervision is harvested from knowledge bases, domain heuristics, ontologies, rules-of-thumb, decisions of weak classifiers or obtained using crowdsourcing. Structure learning [4] also exploits the use of distant supervision signals for generating labels. Such methods use factor graph to learn a high fidelity aggregation of crowd votes. Similar to this, [30] uses weak supervision signals inside the framework of a generative adversarial network. However, none of them operate in a zero-shot setting. We also found related work zero-shot task generalization in the context of reinforcement learning (RL) [29], or in lifelong learning [16]. An agent is validated based on its performance on unseen instructions or a longer instructions. We find that the interpretation of task, and primary objectives, are very different from our present study.

3. Methodology

The primary objective of our methodology is to learn a meta-learning algorithm that regresses nearly optimum parameters of a novel task for which no ground truth (data or labels) is available. To this end, our meta-learner seeks to learn from the model parameters of known tasks (with ground truth) to adapt to a novel zero-shot task. Formally, let us consider K tasks to accomplish, i.e. $\mathcal{T} = \{\tau_1, \cdots, \tau_K\}$, each of whose model parameters lie on a meta-manifold $\mathcal{M}_\theta$ of task model parameters. We have ground-truth available for first m tasks, i.e. $\{\tau_1, \cdots, \tau_m\}$, and we know their corresponding model parameters $\{(\theta_{\tau_i}) : i = 1, \cdots, m\}$ on $\mathcal{M}_\theta$. Complementarily, we have no knowledge of the ground truth for the zero-shot tasks $\{\tau_{(m+1)}, \cdots, \tau_K\}$. (We call the tasks $\{\tau_1, \cdots, \tau_m\}$ as known tasks, and the rest $\{\tau_{(m+1)}, \cdots, \tau_K\}$ as zero-shot tasks for convenience.) Our aim is to build a meta-learning function $\mathcal{F}(\cdot)$ that can regress the unknown zero-shot model parameters $\{(\theta_{\tau_j}) : j = (m+1), \cdots, K\}$ from the knowledge of known model parameters (see Figure 2 (b), i.e.:

$$\mathcal{F}(\theta_{\tau_1}, \cdots, \theta_{\tau_m}) = \theta_{\tau_j}, \quad j = m+1, \cdots, K \quad (1)$$

However, with no knowledge of relationships between the tasks, it may not be plausible to learn $\mathcal{F}(\cdot)$ as it can map to any point on $\mathcal{M}_\theta$. We hence introduce a task correlation matrix, Γ, where each entry $\gamma_{i,j} \in \Gamma$ captures the task correlation between two tasks $\tau_i, \tau_j \in \mathcal{T}$. Equation 1 hence now becomes:

$$\mathcal{F}(\theta_{\tau_1}, \cdots, \theta_{\tau_m}, \Gamma) = \theta_{\tau_j}, \quad j = m+1, \cdots, K \quad (2)$$

The function $\mathcal{F}(.)$ is itself parameterized by W. We design our objective function to compute an optimum value for W as follows:

$$\min_W \sum_{i=1}^{m} ||\mathcal{F}((\theta_{\tau_1}, \gamma_{1,i}), \cdots, (\theta_{\tau_m}, \gamma_{m,i})); W) - \theta^*_{\tau_i}||^2 \tag{3}$$

Similar to [42], without any loss of generality, we assume that all task parameters are learned as an autoencoder. Hence, our previously mentioned task parameters θ_{τ_i} can be described in terms of an encoder, i.e. $\theta_{E_{\tau_i}}$, and a decoder, i.e. $\theta_{D_{\tau_i}}$. We observed that considering only encoder parameters $\theta_{E\tau_i}$ in Equation 3 is sufficient to regress zero-shot encoders and decoders for tasks $\{\tau_{(m+1)}, \cdots, \tau_K\}$. Based on this observation, we rewrite our objective as (we show how our methodology works with other inputs in later sections of the paper):

$$\min_W \sum_{i=1}^{m} ||\mathcal{F}\Big((\theta_{E_{\tau_1}}, \gamma_{1,i}), \cdots, (\theta_{E_{\tau_1}}, \gamma_{m,i}); W\Big) - (\theta^*_{E_{\tau_i}}, \theta^*_{D_{\tau_i}})||^2 \tag{4}$$

where $\theta^*_{E_{\tau_i}}$ and $\theta^*_{D_{\tau_i}}$ and the learned model parameters of a known task $\tau_i \in \mathcal{T}$. This alone is, however, insufficient. The model parameters thus obtained not only should minimize the above loss function on the meta-manifold $\mathcal{M}_\theta$, but should also have low loss on the original data manifold (ground truth of known tasks).

Let $\mathcal{D}_{\theta_{\mathcal{D}_{\tau_i}}}(.)$ denote the data decoder parametrized by $\theta_{D\tau_i}$, and $\mathcal{E}_{\theta_{\mathcal{E}_{\tau_i}}}(.)$ denote the data encoder parametrized by $\theta_{E\tau_i}$. We now add a *data model consistency loss* to Equation 4 to ensure that our regressed encoder and decoder parameters perform well on both the meta-manifold network as well as the original data network:

$$\begin{aligned} \min_W \sum_{i=1}^{m} ||\mathcal{F}\Big((\theta_{E_{\tau_1}}, \gamma_{1,i}), \cdots, (\theta_{E_{\tau_1}}, \gamma_{m,i}); W\Big) \\ - (\theta^*_{E_{\tau_i}}, \theta^*_{D_{\tau_i}})||^2 \\ + \lambda \sum_{\substack{x \in X_{\tau_i} \\ y \in \mathbf{y}_{\tau_i}}} \mathcal{L} \quad \Big(\mathcal{D}_{\tilde{\theta}_{D_{\tau_i}}}(\mathcal{E}_{\tilde{\theta}_{E_{\tau_i}}}(x)), y\Big) \end{aligned} \tag{5}$$

where $\mathcal{L}(\cdot)$ is an appropriate loss function (mean-squared error, cross-entropy or similar) defined for the task τ_i.

Network: To accomplish the aforementioned objective in equation 5, we design $\mathcal{F}(.)$ as a network of m branches, each with parameters $\{W_1, \cdots, W_m\}$ respectively. These are not coupled in the initial layers but are later combined in a W_{common} block that regresses encoder and decoder parameters. Dividing $\mathcal{F}(.)$ into two parts, W_is and W_{common}, is driven by the intuition discussed in [41], that initial layers of $\mathcal{F}(.)$ transform the individual task model parameters into a suitable representation space, and later layers parametrized by W_{common} capture the relationships between tasks and contribute to regressing the encoder and decoder parameters. For simplicity, we refer W to mean $\{W_1, \cdots, W_m\}$ and W_{common}. More specifics of the architecture of our model, TTNet, are discussed as part of our implementation details in Section 4.

Learning Task Correlation: Our methodology admits any source of obtaining task correlation, including through other work such as [42]. In this work, we obtain the task correlation matrix, Γ, using crowdsourcing. We will discuss this in more detail in Section 4.2.

Input: To train our meta network $\mathcal{F}(.)$, we need a batch of model parameters for each known task $\tau_1, \cdots, \tau_m$. This process is similar to the way a batch of data samples are used to train a standard data network. To obtain a batch of p model parameters for each task, we closely follow the procedure described in [40]. This process is as follows. In order to obtain one model parameter set $\Theta^*_{\tau_i}$, for a known task τ_i, we train a base learner (autoencoder), defined by $\mathcal{D}(\mathcal{E}(x; \theta_{E_{\tau_i}}); \theta_{D_{\tau_i}})$. This is achieved by optimizing the base learner on a subset (of size l) of data $\mathbf{x} \in X_{\tau_i}$ and corresponding labels $y \in \mathbf{y}_{\tau_i}$ with an appropriate loss function for the known task (mean-square error, cross-entropy or the like, based on the task). Hence, we learn one $\Theta^{*1}_{\tau_i} = \{\theta^{*1}_{E\tau_i}, \theta^{*1}_{D\tau_i}\}$. Similarly, p subsets of labeled data are obtained using a sampling-with-replacement strategy from the dataset $(X_{\tau_i}, \mathbf{y}_{\tau_i})$ corresponding to τ_j. Following this, we obtain a set of p optimal model parameters (one for each of p subsets sampled), i.e. $\Theta^*_{\tau_j} = \{\Theta^{*1}_{\tau_j}, \cdots, \Theta^{*p}_{\tau_j}\}$, for task τ_j. A similar process is followed to obtain p "optimal" model parameters for each known task $\{\Theta^*_{\tau_1}, \cdots, \Theta^*_{\tau_m}\}$. These model parameters (a total of $p \times m$ across all known tasks) serve as the input to our meta network $\mathcal{F}(.)$.

Training: The meta network $\mathcal{F}(.)$ is trained on the objective function in Eqn 5 in two modes: a *self mode* and a *transfer mode* for each task. Given a known task τ_i, training in *self mode* implies updation of weights W_i and W_{common} alone. On the other hand, training in *transfer mode* implies updation of weights $W_{\neg i}$ (all $W_{j \neq i}, j = 1, \cdots, m$) and W_{common} of $\mathcal{F}(.)$. *Self mode* is similar to training a standard autoencoder, where $\mathcal{F}(.)$ leanrs to projects the model parameters θ_{τ_j} near the given model parameter (learned from ground truth) $\theta^*_{\tau_j}$. In *transfer mode*, a set of model parameters of tasks (other than τ_j) attempt to map the position of learned θ_{τ_j}, near the given model parameter θ_{τ_j} on the meta manifold. We note that the transfer mode is essential in being able to regress model parameters of a task, given model parameters of other tasks. At inference time (for zero-shot task transfer), $\mathcal{F}(.)$ operates in transfer mode.

Regressing Zero-Shot Task Parameters: Once we learn the optimal parameters W^* for $\mathcal{F}(.)$ using Algorithm 1, we use this to regress zero-shot task parameters, i.e. $\mathcal{F}_{W^*}\big((\theta_{E\tau_1}, \gamma_{1,j}), \cdots, (\theta_{E\tau_m}, \gamma_{m,j})\big)$ for all $j = (m+1), \cdots, T$. (We note that the implementation of Algorithm

1 was found to be independent of the ordering of the tasks, $\tau_1, \cdots, \tau_m$.)

Algorithm 1: Training our meta network, TTNet

Input: Number of epochs - Num_Epochs; Number of iterations needed for self mode/transfer mode - k; Optimal model parameters $\{\Theta^*_{\tau_1}, \cdots, \Theta^*_{\tau_m}\}$ of known tasks; Task correlation matrix $\Gamma = \{\gamma_{ij}\}$ ($m \times T$ matrix)

Output: Trained TTNet model, $\mathcal{F}(.)$

for *Num_Epochs* **do**

  for $j = 1, \cdots, m$ **do**

    for k *steps* **do**

      `/* Self mode */`

      Update weights W_i, W_{common} of $\mathcal{F}(.)$ by optimizing:

$$||\mathcal{F}\big((\theta_{E_{\tau_1}}, \gamma_{1,i}), \cdots, (\theta_{E_{\tau_1}}, \gamma_{m,i});$$

$$\mathrm{W}_i, W_{common}\big) - (\theta^*_{E_{\tau_i}}, \theta^*_{D_{\tau_i}})||^2 \quad + \lambda \sum_{\substack{x \in X_{\tau_i} \\ y \in \mathbf{y}_{\tau_i}}} \mathcal{L} \quad (\mathcal{D}_{\tilde{\theta}_{D_{\tau_i}}}(\mathcal{E}_{\tilde{\theta}_{E_{\tau_i}}}(x)), y)$$

    end

    for k *steps* **do**

      `/* Transfer mode */`

      Update weights $W_{\neg i}, W_{common}$ of $\mathcal{F}(.)$ by optimizing:

$$||\mathcal{F}\big((\theta_{E_{\tau_1}}, \gamma_{1,i}), \cdots, (\theta_{E_{\tau_1}}, \gamma_{m,i});$$

$$\mathrm{W}_{\neg i}, W_{common}\big) - (\theta^*_{E_{\tau_i}}, \theta^*_{D_{\tau_i}})||^2 \quad + \lambda \sum_{\substack{x \in X_{\tau_i} \\ y \in \mathbf{y}_{\tau_i}}} \mathcal{L} \quad (\mathcal{D}_{\tilde{\theta}_{D_{\tau_i}}}(\mathcal{E}_{\tilde{\theta}_{E_{\tau_i}}}(x)), y)$$

    end

  end

end

---

4. Results

To evaluate our proposed framework, we consider the vision tasks defined in [42]. (Whether this is an exhaustive list of vision tasks is arguable, but they are sufficient to support our proof of concept.) In this section, we consider four of the tasks as unknown or zero-shot: surface normal, depth estimation, room layout, and camera-pose estimation. We have curated this list based on the data acquisition complexity and the complexity associated with the learning process using a deep network. Surface normal, depth estimation and room layout estimation tasks are monocular tasks but involve expensive sensors to get labeled data points. Camera pose estimation requires multiple images (two or more) to infer six degrees-of-freedom and is generally considered a difficult task. We have three different TTNet s to accomplish them; (1) TTNet_6, TTNet_{10} and TTNet_{20} those consider 6, 10 and 20 vision tasks as known tasks. In addition, we have another model TTNet_{LS} (having 20 known tasks) in which, the regressed parameters are finetuned on a small amount, (20%), of data for the zero-shot tasks. (This provides low supervision to TTNet and hence, the name TTNet_{LS}.) Studies on other sets of tasks as zero-shot tasks are presented in Section 5. We also performed an ablation study on permuting the source tasks differently, which is presented in the supplementary section.

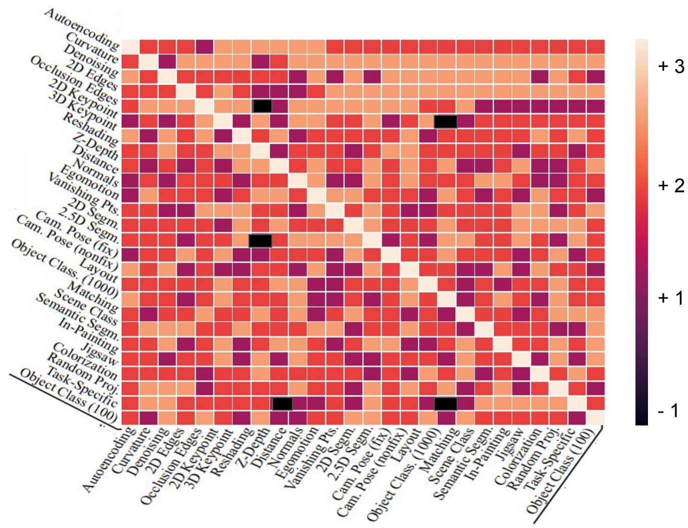

Figure 3: **Task correlation matrix.** (Please zoom the image to see the details) We get the task correlation matrix Γ after receiving votes from 30 annotators. We use this Γ to build our meta-learner TTNet.

4.1. Dataset

We evaluated TTNet on the Taskonomy dataset [42], a publicly available dataset comprised of more than 150K RGB data samples of indoor scenes. It provides the ground truths of 26 tasks given the same RGB images, which is the main reason for considering this dataset. We considered 120K images for training, 16K images for validation, and, 17K images for testing as described in [42].

4.2. Implementation Details

Network Architecture: Following Section 3, each data network is considered an Encoder-Decoder network, and closely follows the model architecture of [42]. The encoder is a fully convolutional ResNet 50 model without pooling, and the decoder comprises of 15 fully convolutional layers for all pixel-to-pixel tasks, e.g. normal estimation, and for low dimensional tasks, e.g. vanishing points, it consists of 2-3 FC layers. To make input samples for TTNet, we created 5000 samples of the model parameters for each task, each of which is obtained by training the model on 1k data points sampled (with replacement) from the Taskonomy dataset. These data networks were trained with mini-batch Stochastic Gradient Descent (SGD) using a batch size of 32, learning rate of 0.001, momentum factor of 0.5 and Adam as an optimizer.

TTNet: We closely followed "classification network" of [12] to build TTNet's architecture. The TTNet is shown in Figure 2 (b). The TTNet initially has m branches, that depends on the model under consideration (TTNet$_m$: $m \in \{6, 10, 20\}$). Each of the m branches is comprised of 15 fully convolutional (FCONV) layers followed by 14 fully connected layers. The m branches are then merged to form a common layer followed by 15 FCONV layers, trained with mini-batch SGD, batch size of 32, learning rate of 0.0001, momentum factor of 0.5 and Adam as an optimizer.

Task correlation: Crowds are asked to response for each pair of tasks (known and zero) on a scale of $+2$ (strong correlation) to -1 (no correlation), while $+3$ is reserved to denote self relation. Votes are aggregated using using Dawid-skene (DS) algorithm. We reported TTNet results based on the task correlation matrix Γ, shown in Figure 3, by aggregating votes of 30 annotators. More details of the Dawid-skene methodology and ablation study on different number of annotators are deferred to the supplementary section.

Method	Mean- (↓)	Medn- (↓)	RMSE- (↓)	11.25- (↑)	22.5- (↑)	30- (↑)
MC[11]	30.30	35.30	-	30.29	57.17	68.29
DD [39]	25.71	20.81	31.01	38.12	59.18	67.21
DD[39]	21.10	15.61	-	44.39	64.48	66.21
Skip [5]	20.21	12.19	28.20	47.90	70.00	78.23
TN[42]	19.90	11.93	23.13	48.03	70.02	78.88
TTNet$_6$	19.22	12.01	26.13	48.02	71.11	78.29
Geo [31]	19.00	11.80	26.90	48.04	72.27	79.68
TTNet$_{10}$	19.81	11.09	22.37	48.83	71.61	79.00
TTNet$_{20}$	19.27	11.91	26.44	48.81	71.97	79.72
TTNet$_{LS}$	**15.10**	**9.29**	**24.31**	**56.11**	**75.19**	**84.71**

Table 1: **Surface Normal Estimation.** Mean, median and RMSE refer to the difference between the model's predicted surface normal and ground truth surface normal (a lower value is better, ↓). Other 3 are the number of pixels within 11.25, 22.5 and 30 thresholds within ground truth's predicted pixels (a higher number is better, ↑). $-$ indicates those values cannot be obtained by the corresponding method.

4.3. Comparison with State-of-the-Art Models

We show both qualitative and quantitative results for our TTNet, trained using the aforementioned methodology, on each of the four identified zero-shot tasks against state-of-the-art models for each respective task below. We note that the same TTNet is validated against all tasks.

4.3.1 Qualitative Results

Surface Normal Estimation: TTNet is compared against: Multi-scale CNN (MC) [11], Deep3D (DD) [39], Deep Network for surface normal estimation (DD) [39], SkipNet (Skip) [5], GeoNet (Geo) [31] and Taskonomy (TN) [42]. The results are shown in Figure 4(a), where the red boxes correspond to our models trained under different settings (as described at the beginning of Section 4). As we increase the number of source tasks, our TTNet shows improved results. TTNet$_{LS}$ captures finer details (see edges of chandelier) which is not visible in other results.

Metd	VM- [14]	EM- [44]	LN- [45]	TT-Net$_6$	RN- [20]	TN- [42]	TT-Nt$_{10}$	TT-Nt$_{20}$	TT-Nt$_{LS}$
Key.	15.5	11.2	7.64	7.5	6.3	6.2	6.00	5.82	**5.52**
Pixel	24.3	16.7	10.6	8.1	8.0	8.0	7.72	7.10	**6.81**

Table 2: **Room Layout.** Both TTNet$_{20}$ and TTNet$_{LS}$ outperformed state-of-the-art models on keypoint and pixel error.

Room Layout Estimation: We followed layout types in [20], and our TTNet's results are compared against: Volumetric (VM) [14], Edge-map (EM) [44], LayoutNet (LN) [45], RoomNet (RN) [20], and Taskonomy (TN) [42]. The green boxes in Figure 4(b) indicate TTNet results; the red edges indicate the predicted room edges. Each model infers room corner points and joins them with straight lines. We report two complex cases in Figure 4 (b): (1) lot of occlusions, and (2) multiple edges such as roof-top, door, etc.

Method	RMSE(lin)	RMSE(log)	ARD	SRD
FDA [21]	0.877	0.283	0.214	0.204
TTN_6	0.745	0.262	0.220	0.210
TN [42]	0.591	0.231	0.242	0.206
TTN_{10}	0.575	0.172	0.236	0.179
Geonet[31]	0.591	0.205	0.149	0.118
$TTNet_{20}$	0.597	0.204	0.140	0.106
$TTNet_{LS}$	**0.572**	**0.193**	**0.139**	**0.096**

Table 3: **Depth estimation:** TTNet$_{20}$ and TTNet$_{LS}$ outperform all other methods studied.

Depth Estimation: Depth is computed from a single image. We compared our TTNet against: FDA [21], Taskonomy [42], and GeoNet [31]. The red bounding boxes show our result. It can be observed from Figure 4(c) that TTNet$_{10}$ outperforms [42]; and TTNet$_{20}$ and TTNet$_{LS}$ outperform all other methods studied.

Camera Pose Estimation (fixed): Camera pose is estimated from two images captured from two different geometric points of same view. A fixed camera pose estimation predicts any five of the 6-degrees of freedom: yaw, pitch, roll and x,y,z translation. In Figure 4(d), two different geometric camera angle translations: (1) perspective, and (2) translation in y and z coordinate are shown for TTNet and RANSAC [10], Latent RANSAC [18], Generic3D pose [43] and Taskonomy [42]. First image is the reference frame of the camera (green arrow), and second image (red arrow) is taken after a geometric translation.

Method	RANSAC[41]	LR[18]	G3D[43]	TN[42]
$TTNet_6$	88%	81%	72%	64%
$TTNet_{10}$	90%	82%	79%	82%
$TTNet_{20}$	90%	82%	92%	80%
$TTNet_{LS}$	**96%**	**88%**	**96%**	**87%**

Table 4: **Camera Pose Estimation (fixed)**. We have considered *win rate* (%) on angular error. Columns are state-of-the-art methods and rows are our four TTNet models.

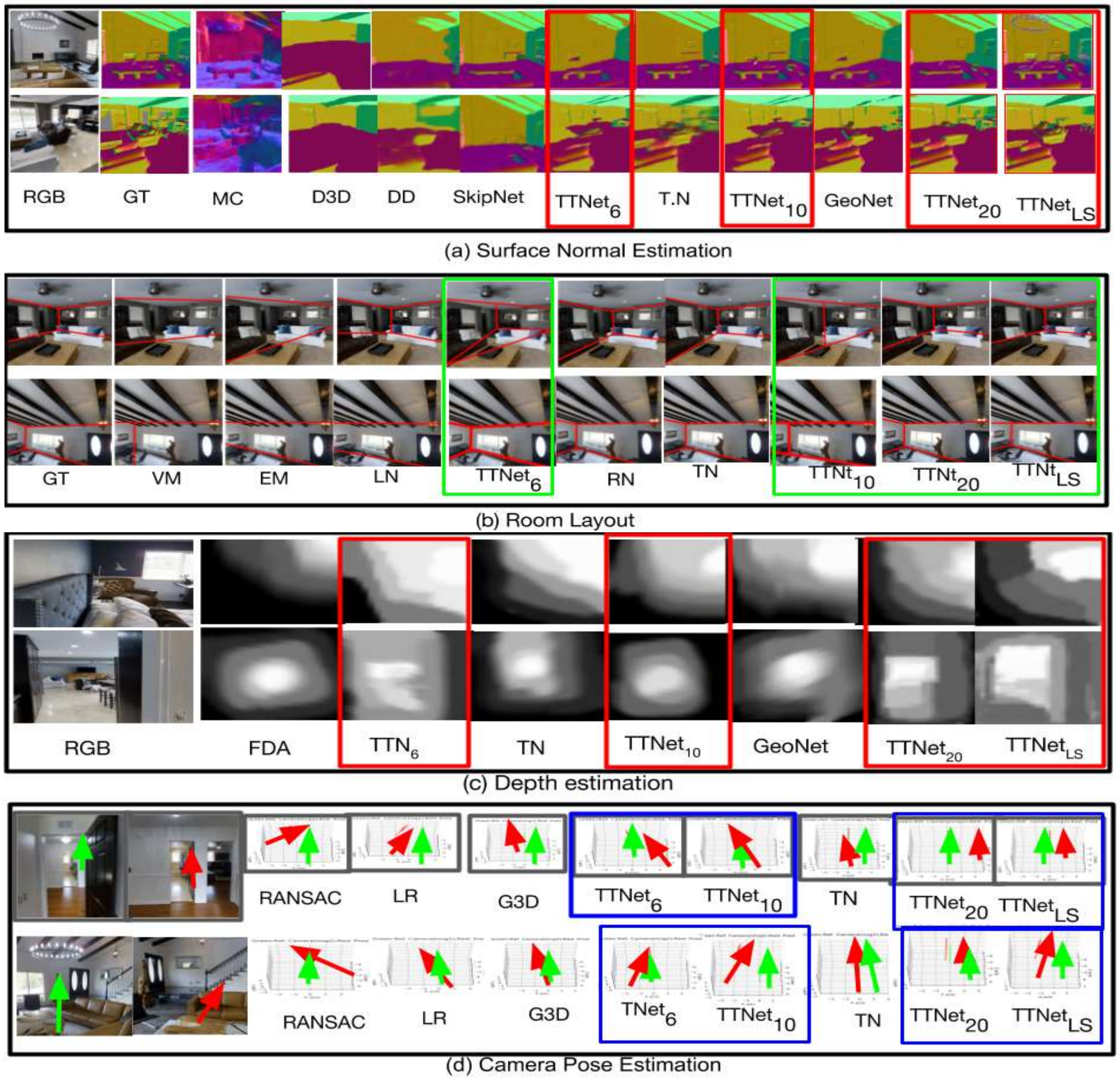

Figure 4: **Qualitative comparison (Best viewed in color):** TTNet models compared against other state-of-the-art models, see Section 4.3.1 for details. (a) ***Surface Normal Estimation:*** Red boxes indicate results of our TTNet models; (b) ***Room Layout:*** Red edges indicate the predicted room edges; green boxes indicate our TTNet model results; (c) ***Depth Estimation:*** Red bounding boxes show our results; (d) ***Camera Pose Esimation:*** First image is the reference frame of the camera, i.e. green arrow. The second image, with red arrow, is taken after a geometric translation w.r.t first image. Blue rectangles show our results.

4.3.2 Quantitative Results

Surface Normal Estimation: We evaluated our method based on the evaluation criteria described in [31], [5]. The results are presented in Table 1. Our TTNet$_6$ is comparable to state-of-the-art Taskonomy [42] and GeoNet [31]. Our TTNet$_{10}$, TTNet$_{20}$, and TTNet$_{LS}$ outperforms all state-of-the-art models.

Room Layout Estimation: Evaluation criteria are: (1) *Keypoint error:* a global measurement avaraged on Euclidean distance between model's predicted keypoint and the ground truth; and (2) *Pixel error:* a local measurement that estimates pixelwise error between the predicted surface labels and ground truth labels. Table 2 presents the results.

Depth Estimation: Following [21] evaluation criteria are: RMSE (lin) = $\frac{1}{N}(\sum_X(d_X - d_X^*)^2)^{\frac{1}{2}}$; RMSE(log) = $\frac{1}{N}(\sum_X(\log d_X - \log d_X^*)^2)^{\frac{1}{2}}$; Absolute relative distance = $\frac{1}{N}\sum_X \frac{|d_X - d_X^*|}{d_X}$; Squared absolute distance = $\frac{1}{N}\sum_X \left(\frac{|d_X - d_X^*|}{d_X}\right)^2$. d_X^* is ground truth depth, d_X is estimated depth, and N is the total number of pixels in all images in the test set.

Camera Pose Estimation (fixed): We adopted the *win rate* (%) evaluation criteria [42] that counts the proportion of images for which a baseline is outperformed. Table 4 shows the win rate of TTNet models on angular error with respect to state-of-the-art models: RANSAC [41], LRANSAC [18], G3D and Taskonomy [42].

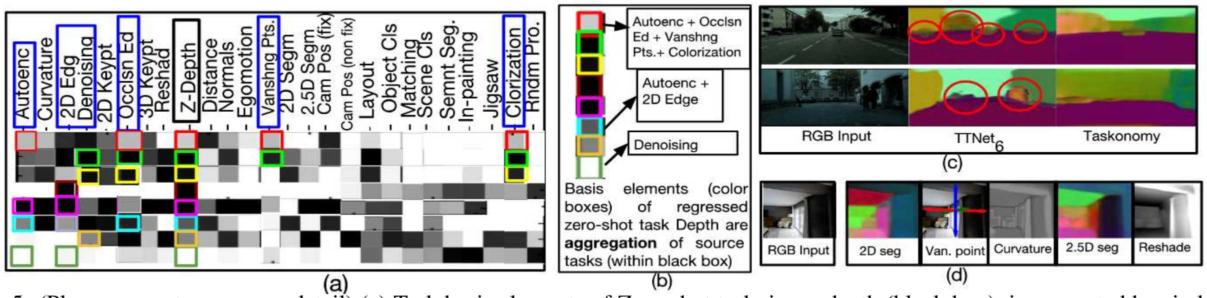

Figure 5: (Please zoom to see more detail) (a) Task basis elements of Zero-shot task, i.e. z-depth (black box), is supported by six known tasks (blue box), which is found using GO-MTL [19]; (b) Zero-shot task basis is comprised of the elements of know tasks basis elements; (c) Surface normal estimation on Cityscapes: Red circles highlight details (car, tree, human) by TTNet; (d) Different choice zero shot tasks: 2D segmentation, Vanishing point estimation, Curvature estimation, 2.5D segmentation and reshading, other than Section 4.

5. Discussion and Analysis

How important is a source task while regressing the zero-shot task?: To study which source task plays the most important role when regressing a zero-shot task, we computed the task basis of source tasks using the GO-MTL approach [19]. The results are shown in Figure 5(a). We consider a source task important if it shares maximum number of basis elements with zero-shot tasks. For e.g., source task "autoencoding" in Fig 5(a) with a blue box shares four basis elements with zero-shot task "z-depth" (discussed further in the Supplementary Section).

Why Zero-shot Task Parameters Performs Better than Supervised Training? When tasks are related (which is the setting in our work), learning from similar tasks can by itself provide good performance. From Figure 5 (b), we can see that, the basis vector of zero-shot task "Z-depth" is composed of latent elements from several source tasks.

Zero-shot to Known Task Transfer: We finetune the decoder to a target known task (encoder remains the same as of zero-shot task). Figure 5(c) shows promising results. Quantitatively, we compared *win rate* of TTNet against [42] w.r.t other state-of-the-art methods: Wang *et al.* [40], G3D [43], and full supervision. Owing to space constraints, these results are presented in the Supplementary section.

Choice of Zero-shot Tasks: A different set of zero tasks than those considered in Sec 4 is studied in Fig 5(d) which shows promising results even for our weakest model, TTNet_6. More results of TTNet_{10}, TTNet_{20}, TTNet_{LS} are included in the Supplementary section.

Performance on Other Datasets: To further study the generalizability of our models, we finetuned TTNet on the Cityscapes dataset [8], and the surface normal results are reported in Figure 5(c), with comparison to [42]. Our model captures more detail.

Object detection on COCO-Stuff dataset: TTNet_6 is finetuned on the COCO-stuff dataset to do object detection, shown in Figure 6. TTNet_6 performs fairly well.

Optimal Number of Known Tasks: In this work, we have reported results of TTNet with 6, 10 and 20 known tasks. We studied the question - how many tasks are sufficient to adapt to zero-shot tasks in the considered setting, and the results are reported in Table 5. Expectedly, a higher number of known tasks provided improved performance.

Method	AP{50:95}	AP{50}	AP{75}	AP{sml}	AP{med}	AP{lrg}
CoupleNet	34.4	54.8	37.2	13.4	8.1	50.8
TTNet{6}	29.9	51.9	34.6	10.8	32.8	45
YOLOv2	21.6	44	19.2	5	22.4	35.5

Figure 6: **Object Detection using TTNet_6**: TTNet_6 is finetuned on the COCO-stuff dataset.

Zero-shot Task Transfer using Taskonomy Graph: We also conducted experiments on using our methodology by using the task correlations obtained from the results of [42] directly. We present these, as well as other results, including the evolution of our TTNet model over the epochs of training, in the Supplementary section.

Model	TT_4	TT_6	TT_8	TT_{10}	TT_{15}	TT_{20}	TT_{LS}
Wang[40]	81	84	84	88	88	91	97
Zamir[43]	73	75	81	82	86	87	90
TN[42]	62	65	84	85	84	89	94

Table 5: *Win rate* (%) of surface normal estimation of TTNet models with varying number of known tasks against: [40], [43], [42].

6. Conclusion

We present a meta-learning algorithm to regress model parameters of a novel task for which no ground truth is available (*zero-shot task*). We evaluated our learned model on the Taskonomy [42] dataset, with four zero-shot tasks: surface normal estimation, room layout estimation, depth estimation and camera pose estimation. Our future work will involve closer analysis of the implications of obtaining task correlation from various sources, and the corresponding results for zero-shot task transfer. In particular, negative transfer in task space is a particularly interesting direction of future work.

Acknowledgement: We gratefully acknowledge the support of the Google Travel Grant for presenting this work, and Intel India for funding this project.

References

[1] E. Alfonseca, K. Filippova, J.-Y. Delort, and G. Garrido. Pattern learning for relation extraction with a hierarchical topic model. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 54–59. Association for Computational Linguistics, 2012. 3

[2] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008. 2

[3] Y. Aytar and A. Zisserman. Tabula rasa: Model transfer for object category detection. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2252–2259. IEEE, 2011. 2

[4] S. H. Bach, B. He, A. Ratner, and C. Ré. Learning the structure of generative models without labeled data. *arXiv preprint arXiv:1703.00854*, 2017. 3

[5] A. Bansal, B. Russell, and A. Gupta. Marr revisited: 2d-3d alignment via surface normal prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5965–5974, 2016. 6, 7

[6] Q. Chen, Y. Liu, Z. Wang, I. Wassell, and K. Chetty. Re-weighted adversarial adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7976–7985, 2018. 2

[7] H. Cohen and K. Crammer. Learning multiple tasks in parallel with a shared annotator. In *Advances in Neural Information Processing Systems*, pages 1170–1178, 2014. 2

[8] 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. 8

[9] G. Csurka. Domain adaptation for visual applications: A comprehensive survey. *arXiv preprint arXiv:1702.05374*, 2017. 2

[10] K. G. Derpanis. Overview of the ransac algorithm. *Image Rochester NY*, 4(1):2–3, 2010. 6

[11] D. Eigen and R. 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. 6

[12] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017. 3, 6

[13] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *European Conference on Computer Vision*, pages 597–613. Springer, 2016. 2, 3

[14] A. Gupta, M. Hebert, T. Kanade, and D. M. Blei. Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. In *Advances in neural information processing systems*, pages 1288–1296, 2010. 6

[15] R. Held and A. Hein. Movement-produced stimulation in the development of visually guided behavior. *Journal of comparative and physiological psychology*, 56(5):872, 1963. 1

[16] D. Isele, M. Rostami, and E. Eaton. Using task features for zero-shot knowledge transfer in lifelong learning. In *IJCAI*, pages 1620–1626, 2016. 3

[17] T. Kim, J. Yoon, O. Dia, S. Kim, Y. Bengio, and S. Ahn. Bayesian model-agnostic meta-learning. *arXiv preprint arXiv:1806.03836*, 2018. 3

[18] S. Korman and R. Litman. Latent ransac. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6693–6702, 2018. 6, 7

[19] A. Kumar and H. Daume III. Learning task grouping and overlap in multi-task learning. *arXiv preprint arXiv:1206.6417*, 2012. 2, 8

[20] C.-Y. Lee, V. Badrinarayanan, T. Malisiewicz, and A. Rabinovich. Roomnet: End-to-end room layout estimation. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 4875–4884. IEEE, 2017. 6

[21] J.-H. Lee, M. Heo, K.-R. Kim, and C.-S. Kim. Single-image depth estimation based on fourier domain analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 330–339, 2018. 6, 7

[22] S.-I. Lee, V. Chatalbashev, D. Vickrey, and D. Koller. Learning a meta-level prior for feature relevance from multiple related tasks. In *Proceedings of the 24th international conference on Machine learning*, pages 489–496. ACM, 2007. 2

[23] J. J. Lim, R. R. Salakhutdinov, and A. Torralba. Transfer learning by borrowing examples for multiclass object detection. In *Advances in neural information processing systems*, pages 118–126, 2011. 2

[24] M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In *Advances in neural information processing systems*, pages 469–477, 2016. 2

[25] S. Liu, S. J. Pan, and Q. Ho. Distributed multi-task relationship learning. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 937–946. ACM, 2017. 2

[26] M. Long, Z. Cao, J. Wang, and S. Y. Philip. Learning multiple tasks with multilinear relationship networks. In *Advances in Neural Information Processing Systems*, pages 1594–1603, 2017. 2

[27] Z. Luo, Y. Zou, J. Hoffman, and L. F. Fei-Fei. Label efficient learning of transferable representations acrosss domains and tasks. In *Advances in Neural Information Processing Systems*, pages 165–177, 2017. 2

[28] D. K. Naik and R. Mammone. Meta-neural networks that learn by learning. In *Neural Networks, 1992. IJCNN., International Joint Conference on*, volume 1, pages 437–442. IEEE, 1992. 3

[29] J. Oh, S. Singh, H. Lee, and P. Kohli. Zero-shot task generalization with multi-task deep reinforcement learning. *arXiv preprint arXiv:1706.05064*, 2017. 3

[30] A. Pal and V. N. Balasubramanian. Adversarial data programming: Using gans to relax the bottleneck of curated labeled data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1556–1565, 2018. 2, 3

[31] X. Qi, R. Liao, Z. Liu, R. Urtasun, and J. Jia. Geonet: Geometric neural network for joint depth and surface normal estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 283–291, 2018. 6, 7

[32] A. J. Ratner, C. M. De Sa, S. Wu, D. Selsam, and C. Ré. Data programming: Creating large training sets, quickly. In *Advances in neural information processing systems*, pages 3567–3575, 2016. 2, 3

[33] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 2, 3

[34] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 2

[35] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada. Maximum classifier discrepancy for unsupervised domain adaptation. *arXiv preprint arXiv:1712.02560*, 3, 2017. 2

[36] O. Sener, H. O. Song, A. Saxena, and S. Savarese. Learning transferrable representations for unsupervised domain adaptation. In *Advances in Neural Information Processing Systems*, pages 2110–2118, 2016. 2

[37] S. Thrun and L. Pratt. *Learning to learn*. Springer Science & Business Media, 2012. 3

[38] P. Varma, B. He, D. Iter, P. Xu, R. Yu, C. De Sa, and C. Ré. Socratic learning: Augmenting generative models to incorporate latent subsets in training data. *arXiv preprint arXiv:1610.08123*, 2016. 2, 3

[39] 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

[40] Y.-X. Wang, D. Ramanan, and M. Hebert. Learning to model the tail. In *NIPS*, 2017. 4, 8

[41] 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, 4, 6, 7

[42] 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, 4, 5, 6, 7, 8

[43] 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. 1, 6, 8

[44] W. Zhang, W. Zhang, K. Liu, and J. Gu. Learning to predict high-quality edge maps for room layout estimation. *IEEE Transactions on Multimedia*, 19(5):935–943, 2017. 6

[45] C. Zou, A. Colburn, Q. Shan, and D. Hoiem. Layoutnet: Reconstructing the 3d room layout from a single rgb image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2051–2059, 2018. 6