2016 23rd International Conference on Pattern Recognition (ICPR)
Cancún Center, Cancún, México, December 4-8, 2016

Leveraging Multiple Tasks to Regularize Fine-Grained Classification

Riddhiman Dasgupta Anoop M. Namboodiri
CVIT, International Institute of Information Technology, Hyderabad, India.
riddhiman.dasgupta@research.iiit.ac.in, anoop@iiit.ac.in

***Abstract*—Fine-grained classification is an extremely challenging problem in computer vision, compounded by subtle differences in shape, pose, illumination and appearance. While convolutional neural networks have become the versatile jack-of-all-trades tool in modern computer vision, approaches for fine-grained recognition still rely on localization of keypoints and parts to learn discriminative features for recognition. In order to achieve this, most approaches use a localization module and subsequently learn classifiers for the inferred locations, thus necessitating large amounts of manual annotations for bounding boxes and keypoints. In order to tackle this problem, we aim to leverage the (taxonomic and/or semantic) relationships present among fine-grained classes. The ontology tree is a free source of labels that can be used as auxiliary tasks to train a multi-task loss. Additional tasks can act as regularizers, and increase the generalization capabilities of the network. Multiple tasks try to take the network in diverging directions, and the network has to reach a common minimum by adapting and learning features common to all tasks in its shared layers. We train a multi-task network using auxiliary tasks extracted from taxonomical or semantic hierarchies, using a novel method to update task-wise learning rates to ensure that the related tasks aid and unrelated tasks does not hamper performance on the primary task. Experiments on the popular CUB-200-2011 dataset show that employing super-classes in an end-to-end model improves performance, compared to methods employing additional expensive annotations such as keypoints and bounding boxes and/or using multi-stage pipelines.** [1]

I. Introduction

Convolutional neural networks(CNNs) first tasted mainstream success with their impressive performance on large scale image recognition challenges, starting with Krizhevsky *et al*. [1], which brought them into the limelight. Training a convnet from scratch is usually too expensive and will not result in the same discriminative power of one that is trained on a large dataset like Imagenet. A far more effective strategy is to fine-tune a convnet pre-trained on Imagenet to new datasets and/or tasks. Consequently, researchers have adapted convnets that were pre-trained on Imagenet for a vast plethora of tasks, ranging from object detection and semantic segmentation to pose estimation, depth estimation, attribute prediction, part localization, and many more. The works by Donahue *et al*. [2], Ravazian *et al*. [3], Chatfield *et al*. [4], and Oquab *et al*. [5] have shown beyond any reasonable doubt that convnets are ripe for transfer learning via fine-tuning.

The primary challenges of fine grained recognition are large variations in pose and illumination, subtle intra-class differences and striking inter-class similarities. Most modern methods for fine grained recognition rely on a combination of localizing discriminative regions and learning corresponding discriminative features. This in turn requires strong supervision such as keypoint or attribute annotations, which are expensive and difficult to obtain at scale. On the other hand, since fine grained recognition deals with subordinate-level classification, there exists an implied relationships among labels. These relationships may be taxonomical (such as super classes) or semantic (such as attributes) in nature. The ontology obtained in this manner contains rich latent knowledge about finer differences between classes that can be exploited for visual classification. The model we propose consists of a single deep convolutional neural network, with each level of the ontology giving rise to an additional set of labels for the input images. These additional labels are used as auxiliary tasks for a multi-task network, which can be trained end-to-end using a simple weighted objective function. We also propose a novel method to dynamically update the learning rates (hereforth referred to as the task coefficients) for each task in the multi-task network, based on its relatedness to the primary task.

In this work, we analyze the utility of jointly learning multiple related/auxiliary tasks that could regularize each other to prevent over-fitting, while ensuring that the network retains its discriminative capability. Much like dropout is bagging taken to the extreme, multi-task learning is analogous to boosting, if each task is considered a weak learner. We note that our model can be plugged into or used in conjunction with more complex multi-stage pipeline methods such as [7]–[10]

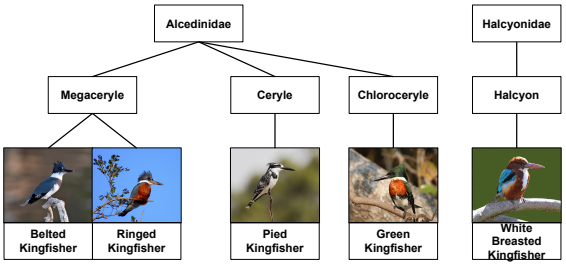


Fig. 1. Leveraging the taxonomic ontology of birds for fine grained recognition. From top to bottom, we have family, order and species for five classes of kingfishers in the CUB-200-2011 dataset [6]. Observe how identifying the family or order can help identifying the class, e.g. in case of ringed kingfisher and green kingfisher. Best viewed enlarged, in color.

[1] Additional details can be found at cvit.iiit.ac.in/multitaskhierarchy.

to further improve performance for fine grained recognition. Furthermore, this enables us to learn a single model that can be used for multiple tasks, effectively reducing the training time by a factor of T for T tasks. Our experimental results show that adding additional tasks are effective as regularizers, especially for convnets where there is not enough training data available. This is often the case in fine grained datasets, where labelled data is scarce and expensive to obtain.

II. Related Work

A. Deep Multi-task Learning

Collobert *et al*. [11] trained a deep neural network on three related tasks in the domain of natural language processing.. This seminal work paved the way for deep multi-task networks, ranging from joint prediction of depth, surface normals and semantic labels [12] and joint learning of facial landmark localization, pose estimation and gender recognition [13], to instance segmentation [14] and immediacy prediction [15]. More recently, Misra *et al*. [16] presented a detailed study of how and where to share tasks across layers in convnets.
Li *et al*. [17] exploit localization as an additional task to find human pose keypoints. Zhang *et al*. [18] use pose and keypoints as additional tasks for multiview face detection. However, both methods consider all tasks to have fixed learning rates. Zhang *et al*. [19] use auxiliary tasks such as attribute prediction to make facial keypoint detection more robust. Tian *et al*. [20] use a task-assistant CNN to jointly learn attributes to detect pedestrians. Both methods resort to multi-step alternating gradient descent methods to tweak task weights, resulting in increased training time and complexity.
Gkioxari *et al*. [21] study the effect of multi-task learning using R-CNNs fine-tuned for pose estimation and action classification. Teterwak [22] provides a succinct attempt at regularizing deep neural networks using multi-task learning.

B. Fine Grained Recognition

There exists a plethora of literature tackling fine grained recognition with the help of convnets, with most approaches relying on alignment of keypoints [23], [24], [25], localization of keypoints [10], [9], [7] or part based models [26], [27]. Branson *et al*. [23] use keypoint based templates to align bird parts and learn separate part based networks, which are then combined. Zhang *et al*. [27] extend R-CNNs [28] by combining region proposals with geometric constraints to train part based networks. Both of these works rely on keypoints and bounding boxes requiring expensive labor-intensive annotations, which is not a strict requirement for our proposed method. Krause *et al*. [25] use cosegmentation followed by alignment, while Xiao *et al*. [26] combine bottom-up part extraction with top-down part-based attention. While both methods forego the requirement of annotated regions, they end up being complex multi-stage pipelines in contrast to our simple end-to-end trainable models.
Lin *et al*. [8] on the other hand use two separate convnets to extract deep features and subsequently combine them using a bilinear layer. Jaderberg *et al*. [10] employ spatial transformer networks for the task. While these models are end-to-end trainable, they are large, slow and difficult to interpret. Our model is at a clear advantage here because it can be plugged into any of these end-to-end methods to potentially achieve even higher efficacy.
Lin *et al*. [24] formulate a complex non-parametric valve linkage function to connect localization and classification networks by aligning predicted parts and keypoints. More recent methods such as Huang *et al*. [9] and Zhang *et al*. [7] aim to combine attention based models and part based models by cropping parts corresponding to predicted keypoints and feeding them to discriminative feature extractors for fine grained recognition. Yet again, our proposed approach does not require such additional annotations, but can be merged with these end-to-end approaches combining localisation, attention and classification.

C. Taxonomy Based Classification

Perhaps closest to our approach lies the work done by Wang *et al*. [29], Deng *et al*. [30] and Srivastava *et al*. [31]. Wang *et al*. [29] claim that a set of classification labels at the subordinate level implies a hierarchy of labels. Their work involves separate models being learnt for each level of the hierarchy and fused for global recognition. We instead aim to employ multi-task learning to regularize the subordinate level classification using the other levels of the ontology tree. This makes our proposed method work with a single model, which can be much smaller as well as end-to-end trainable.

D. Dynamic Multi Task Coefficients

Zhang *et al*. [19] apply task-wise early stopping, but do not tune the task-wise rates throughout training. Zhang *et al*. [34] employ both dynamic task coefficients and task correlations, but end up requiring multiple alternating gradient updates for each mini-batch. Abdulnabi *et al*. [35] propose a latent task matrix to capture the relationship among tasks, which can only be trained via a combination of multiple separate gradient descent steps for each mini-batch, rendering training highly impractical.
We overcome many of the aforementioned restrictions in our model. We consider no constraints such as binary tasks or equal task coefficients. Our work aims to present multi-task learning as not just an easy way of training a single network for multiple tasks, but as an effective regularizer.

III. Proposed Method

A. Multiple Related Tasks

Learning multiple tasks jointly is a natural way of regularization [32] for deep neural networks that typically have a large number of parameters or weights. The crux of the idea stems from the notion that if tasks are related, then features representing the task should be related as well. Multi-task learning requires a way to share features across tasks. In neural networks, including deep convolutional ones, this can be accomplished by branching or bifurcation. All layers before the branching or bifurcation are shared, while the subsequent

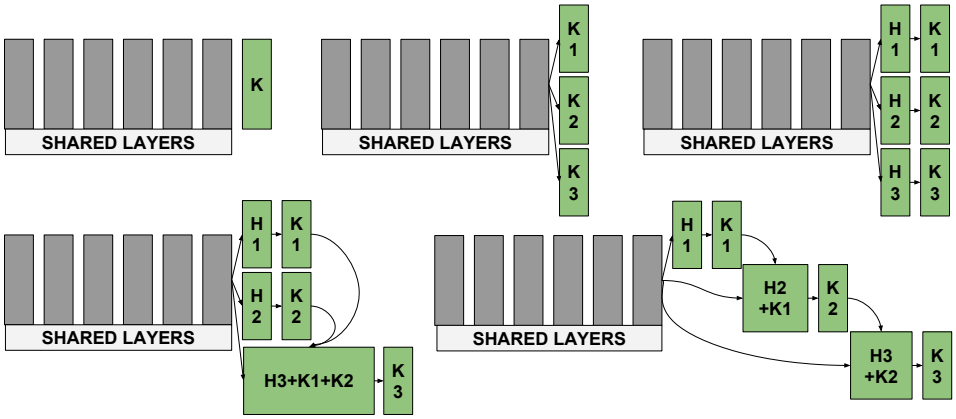


Fig. 2. A few examples of typical multi-task neural network architectures. Note that the first few layers are shared by all the tasks, and there is no constraint on what type of task can be added. Shared layers are represented in grey, while layers in task-specific branches are shown in green. Hi means the hidden layer and Ki denotes the final loss layer for task i. In the top row, from left to right, we have (a) a single task network, followed by (b) a plain multi-task network, folowed by (c) a multi-task network with task-specific hidden layers. In the bottom row, from left to right, we have (d) a network where features from some tasks are concatenated before being passed to another task, and (e) a cascading multi-task network where each task feeds into the next one.

layers are task dependent. Figure 2 shows a few examples of typical convolutional neural networks designed for learning multiple tasks. Here, we aim to optimize the performance of a main or primary task T_0 with the aid of additional related/auxiliary tasks $\tau = \{1, ..., T\}$. The general form of the objective function that we aim to minimize here is:

$$\underset{W_0, W_i \forall t \in \tau}{argmin} \sum_{i=1}^{N} \left[\alpha_0 l_0(y_i^0, f_0(x_i, W_0)) + \sum_{t=1}^{T} \alpha_t l_t(y_i^t, f_t(x_i, W_t)) \right] \tag{1}$$

The 0^{th} index refers to the primary task. For N input samples, W_t denotes the weights of the network with respect to task t, while y_i^t denotes the ground truth for the input representation x_i. f_t represents the feature transformation of the input x_i with respect to the task t and the corresponding weights W_t, and l_t is the corresponding loss function for the task. It is to be noted that $W_t = (W_s, W_{tt})$, where W_s is the shared representation, *i.e.*, the weights of the shared layers, while W_{tt} is the set of weights from the task specific layers.

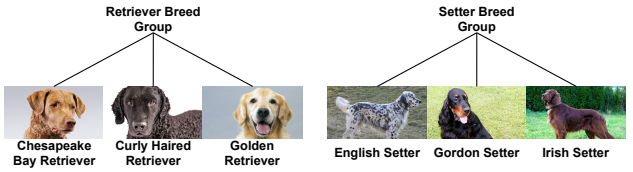
Compared to traditional multi-task learning, in this formulation we can employ different loss functions for each task as appropriate. This is slightly different from traditional multi-task learning where all the tasks might be considered to be equally important. For this purpose, we associate each task t with its loss function l_t and a coefficient α_t, which acts as a coefficient determining the relative importance of the corresponding task.

Even with disparate loss functions, the entire convnet can still be trained in end-to-end fashion using vanilla backpropagation. Each loss function l_t will compute the error E_t^i with respect to an input x_i and a target y_t^i, along with a set of error gradients ∇_t^i. These gradients for each task t are then back-propagated all the way to the point of branching and bifurcation, where they are combined and propagated backwards through the shared layers. The task-specific coefficients α_t are applied to the errors E_t^i and gradients ∇_t^i to ensure that each task contributes accordingly to the global loss or objective function. This formulation of multi-task learning faces two challenges, namely that of finding related tasks, and that of setting the proper task-specific coefficients. We now discuss our proposed way of dealing with these two hurdles.

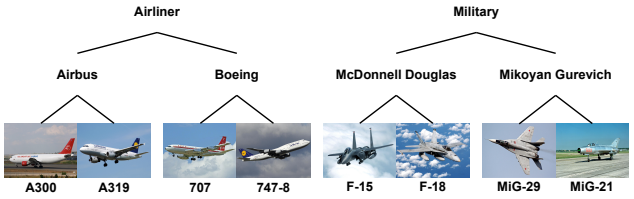
B. Hierarchy as a Related Task

Multi-task learning works only with the correct set of tasks to learn jointly. Trying to learn unrelated tasks leads to negative transfer, resulting in poor generalization. We opine that inherent relationships present among classes can be effectively used as related tasks to regularize the primary classification task. For example, the semantic relationships in case of automobiles, *i.e.* the type of car (sedan *vs.* hatchback), manufacturer (Ford *vs.* BMW), model (Tesla Model S *vs.* Tesla Roadster), form a three level hierarchy that can be exploited for fine grained recognition. From fig. 3, we can observe that distinguishing commercial airliners from military fighters is easier than distinguishing a Boeing airliner from an Airbus one. One can also obtain a multiple tasks from attributes, such as ingredients of food items, or from superclasses, as shown in fig. 3 for breeds of dogs. Even when relationships cannot be obtained automatically, and require domain knowledge, it is far cheaper to have experts extract a ontology among classes, than to have them manually annotate each image in a dataset for keypoints, attributes, etc. We use this to our advantage, and use a hierarchy based on the scientific names according to the Linnaean taxonomy [33] which is in effect a taxonomical hierarchy. Thus, for any organism, traversing the hierarchy tree results in multiple labels depending on the level in the tree, where each label is a classification task that needs to be learnt by our model. As an example, human beings belong to Homo sapiens at the species level, Homo at the genus level, Primates at the order level, Mammals at the class level, and so on. Naturally, classifiction can benefit from such a hierarchy of classes, since going to a higher level enables one to leverage

inter-class differences to distinguish classes, while intra-class variations can be tackled by traversing to a lower level in the hierarchy.



(a) Hierarchy among dogs



(b) Hierarchy among aircraft

Fig. 3. Examples of relationships among classes that can be exploited for multi-task learning. *a* and *b* show relationships inherent among breeds of dogs, and types of aircrafts respectively. Best viewed digitally, in color.

C. Task-specific Coefficients

We are now left with the daunting problem of figuring out how to specify task importance. Tasks need to be initialised with a proper task-specific coefficient, which effectively weighs its contribution to the total loss and gradients. Furthermore, these coefficients need to be monitored and tuned during training based on whether a task is helping or hurting the performance of the primary task. We adapt the work by Silver *et al*. [36] where a separate dynamic task coefficient is introduced for each task based on a measure of relatedness of each auxiliary task with the primary task.

We extend Silver *et al*.'s measure of relatedness to work for any task with any loss function and with any number of hidden layers in its own task-specific branch, as long as each task specific branch has a linear layer with the same number of units. Like Silver *et al*., we consider that the primary task has a coefficient of α_0, and each task has a coefficient of α_t for $t = \{1, ..., T\}$, where we now consider α_t to be a measure of relatedness between the t^{th} task and the primary task, *i.e*.,

$$\alpha_t = tanh\Big(\frac{accuracy_t}{distance_t + \epsilon} \times \frac{1}{RELMIN}\Big) \qquad (2)$$

where $accuracy_t$ is the performance measure of the t^{th} task and $distance_t$ is the Euclidean distance between the weights of the t^{th} task and the primary task. $RELMIN$ is a hyperparameter, and ϵ is a small constant $(1e-6)$ to prevent division by zero. The hyperbolic tangent clamps the task specific coefficient between 0 and 1, while the primary task has a coefficient of $\alpha_0 = 1$ in our experiments. This ensures that the auxiliary tasks always contribute less than the primary task to the weighted loss.

In our model, each task specific branch has a final hidden layer with the same number of units. Thus, mathematically, $distance_t$ is computed as the distance between the weights of the final hidden layers in task t's branch and primary task's branch. We further introduce task competition by applying a softmax on the task coefficients of the auxiliary tasks. As a result, the primary task has $\alpha_0 = 1$ and for the auxiliary tasks, $\sum_{t=1}^{T} \alpha_t = 1$. Since each auxiliary task now has its contribution clamped further, this inter-task competition acts as an additional regularizer. Even though our model has task specific branches and multi-output tasks unlike [19], [20], [34], [35], simply by using making the hidden layer of each task have the same size we can dynamically update task specific coefficients, and by smoothening the aforementioned coefficients using a softmax function, we obtain task wise feature competition.

IV. Experiments

We show quantitative results on the Caltech-UCSD Birds-200-2011 dataset. We use the Torch [37] machine learning library, on an NVIDIA Titan X GPU.

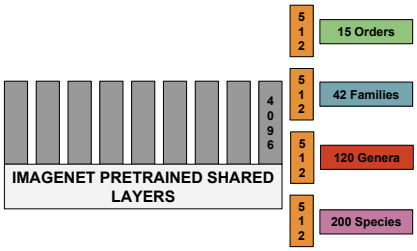


Fig. 4. The task specific layers of the model for the CUB dataset. Shared layers are shown in grey, and are usually taken from a model pre-trained on Imagenet. Each task specific branch is color-coded, viz. orders is in green, families is in blue, genera is in red, and species is in purple.

A. Caltech-UCSD Birds-200-2011

The Caltech-UCSD Birds-200-2011 dataset contains $11,788$ images of birds belonging to 200 species, where each image contains additional annotations consisting of 15 pairs of keypoint coordinates, 312 binary attributes and one bounding box for localization per image. Unlike most existing methods, we do not use these labels at all. The training and test splits are roughly equal in size (5994 vs. 5794), and so this dataset has limited training data. Regularization via multi-task learning should therefore boost the performance of the primary task. We refer to the American Ornithologists' Union Checklist of North American Birds to obtain the taxonomic hierarchy for avian species. For each of the 200 species present in the dataset, we extract the order, family and genus, for a total of 15 orders, 42 families, 120 genera and 200 species in the dataset. Classifying each bird image into the corresponding genus, family, order and species make up the four tasks in our multi-task model, with species level classification being the primary task. The remaining three tasks contribute to the main goal of identifying the species of the bird present in the image.

We use an Imagenet pre-trained model to initialise our network. We take all the layers till the last 4096-dimensional fully connected layer. Right after this layer, we create four

branches, whose details are provided in figure 4 below. Each branch has a linear layer mapping the 4096 dimensional feature of the shared layer to 512 dimensions, followed by another linear layer mapping the 512 dimensional feature to the corresponding number of classes. All task specific layers except the final ones have ReLU non-linearities and Dropout with a probability of 0.5. The 512 dimensional linear layer in each branch is shown in orange because it is used to compute the relatedness of the auxiliary tasks with the primary task. figure 4 shows the base model, which we also adapt to form concatenated and cascaded models, similar to the ones shown in figure 2.

B. Training Details

The joint objective function is thus a weighted sum of four terms, with the weights for species, genus, family and order classification being denoted by $\alpha_{class}, \alpha_{genus}, \alpha_{family}$ & α_{order} respectively. α_{class} is set to 1.0, while all other task coefficients are set to 0.1 initially, and updated at the end of every epoch. We also employ task wise early stopping, where we set the coefficient for a task to 0 if its performance saturates in a fixed number of epochs. The learning rate for the branches is 10 times the learning rate for the pre-trained layers. Mini-batch gradient descent is used for training, with an initial learning rate of 0.001 and a batch size of 32. We set the value of $RELMIN$ to 0.05 following Silver *et al.* [36], who show that their method is robust to changes in the value of $RELMIN$. Accuracies for the 200 class fine-grained classification task are shown in table I. We use the VGG-16 model of [38] for fine-tuning our models, and compare our results with the current state-of-the-art for simple fine-tuning of pre-trained model on the CUB-200-2011 dataset for a fair comparison. It should be noted that both [8] and [10] use pretrained models as a building block in their models, and our model can easily be incorporated into their systems for even more improved fine-grained classification.

C. Analysis

From table I, it is evident that forcing a deep network to learn multiple tasks simultaneously results in increased testing accuracy on the primary task(s). The added regularization increases the generalization of the model. A closer look at the learning procedure reveals some more benefits of having multiple tasks. While performance from our multi-task models does not reach state of the art, it surpasses or provides close competition to [7], [23], [27], [29] which rely on expensive additional annotations of keypoints and bounding boxes, and/or are multi-stage methods. In contrast, our model is an order of magnitude simpler, and can easily be utilised in other end-to-end fine grained recognition pipelines, such as [8]–[10]. Additionally, our results are purely in the weakest regime of evaluation, without any bounding boxes provided during either training or testing, unlike methods such as [25]. Supervision in the form of bounding boxes is bound to increase accuracy, and remains in the scope of future work.

TABLE I

ACCURACY FOR FINE-GRAINED RECOGNITION WITH THE PROPOSED APPROACH. PLEASE NOTE THAT CLASSIFICATION HAS A CONSTANT WEIGHT $\alpha_{class} = 1$. MTL REFERS TO SIMPLE MULTI-TASK LEARNING, WITH EACH TASK SPECIFIC BRANCH HAVING JUST A SINGLE CLASSIFIER LAYER. $MTL - H$ REFERS TO THE ARCHITECTURE SHOWN IN FIGURE 4 WHERE EACH TASK SPECIFIC BRANCH HAVING ITS OWN HIDDEN LAYERS. CONCATENATING AND CASCADING ARE APPLIED AS SHOWN IN FIGURE 2.

Methods	Additional Details	Accuracy
Fine-tuning VGG-16	-	73.52
MTL	-	74.85
MTL-H	No re-weighting	74.75
MTL-H	Re-weighting	75.09
MTL-H	Re-weighting, smoothing	75.28
MTL-H	Re-weighting, smoothing, stopping	**75.66**
MTL-H	Re-weighting, smoothing, stopping, concatenating tasks	**75.76**
MTL-H	Re-weighting, smoothing, stopping, cascading tasks	**76.66**
BCNN	[M,M] model from [8]	78.10
BCNN-MTL	MTL on top of BCNN [8]	**79.04**
[27]	Oracle unknown scheme	73.89
[7]	Without bilinear features	75.04
[23]	Head+Body+Whole image model	75.73
[9]	Keypoints+BBox cropping	**76.27**

From our results, it can be seen that recognition is aided by not only adding multiple related tasks, but by task coefficient re-weighting and smoothing as well. Further, it seems that concatenating and cascading tasks improves performance even further, courtesy the knowledge embedded in the manifolds of the auxiliary task branches.

We further implement the $BCNN[M, M]$ method of [8], and replace the single classifier with our baseline MTL classifier, *i.e.* with 4 branches corresponding to the 4 tasks we intend to learn jointly. We show that combining a hierarchy based MTL classifier with the $BCNN$ gives a slight boost in accuracy. Compared to [8] that runs at 8 frames/sec, our model runs at 32 frames/sec while training.

Fig. 5. Rate of convergence plotted as a function of accuracy versus epochs, for the baseline fine-tuned model and out $MTL - H$ model.

D. Regularization of Primary Task

We also address the claim of multi-task learning providing better regularization by analyzing the gap between trainng and testing accuracies, as shown in figure 5. Here, we plot training and testing accuracies for the baseline single task fine-tuned model (STL in the figure) and our $MTL - H$ model for the first 20 epochs of training, when both models have

saturated in terms of performance. One can clearly observe from figure 5 that not only does the testing accuracy improve in the $MTL-H$ model, but the difference between training and testing accuracies is also lower for the $MTL-H$ model. This proves that multi-task learning has indeed regularized the model, leading to increased generalization. Indeed, in table II we show that auxiliary tasks in the form of super-classes actually help resolve confusing samples, and end up achieving better performance. Our experiments show that while training accuracy reaches 99.41% in the fine-tuned model, it only reaches 96.36% in the final $MTL-H$ model.

TABLE II
ACCURACIES FOR THE ADDITIONAL TASKS OF ORDER, FAMILY AND GENUS CLASSIFICATION USING A SINGLE TASK MODEL FINETUNED FROM VGG-16, AND A MULTI TASK MODEL WITH TASK SPECIFIC HIDDEN LAYERS. IMPROVED ACCURACIES REPRESENT BETTER GENERALIZATION.

Methods	Order	Family	Genus
Single Task Model	94.80	86.58	79.39
MTL-H	96.78	90.49	83.43

V. CONCLUSION

Use of a label hierarchy to generate auxiliary tasks is an effective strategy for regularization of a learning algorithm. We exploit the inter-dependency and relatedness among tasks to train deep convolutional networks that are more accurate and robust. Our experiments show that even with limited training data, the presence of hints in the form of additional tasks and a joint objective function causes the convnet to learn meaningful features that generalize well on testing data. Additionally, multiple tasks can be learnt faster than it takes to learn a single task. Possible extensions to this would involve attempting to analyse the effect of relatedness of tasks has on generalization. A deterministic method to assign appropriate importance to each auxiliary task is much needed as well.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[2] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," *arXiv preprint arXiv:1310.1531*, 2013.

[3] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *CVPRW*, 2014.

[4] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *BMVC*, 2014.

[5] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *CVPR*, 2014.

[6] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 Dataset," California Institute of Technology, Tech. Rep. CNS-TR-2011-001, 2011.

[7] N. Zhang, E. Shelhamer, Y. Gao, and T. Darrell, "Fine-grained pose prediction, normalization, and recognition," *arXiv preprint arXiv:1511.07063*, 2015.

[8] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear cnn models for fine-grained visual recognition," *arXiv preprint arXiv:1504.07889*, 2015.

[9] S. Huang, Z. Xu, D. Tao, and Y. Zhang, "Part-stacked cnn for fine-grained visual categorization," *arXiv preprint arXiv:1512.08086*, 2015.

[10] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *NIPS*, 2015.

[11] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *ICML*, 2008.

[12] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *ICCV*, 2015.

[13] R. Ranjan, V. M. Patel, and R. Chellappa, "Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition," *arXiv preprint arXiv:1603.01249*, 2016.

[14] J. Dai, K. He, and J. Sun, "Instance-aware semantic segmentation via multi-task network cascades," *arXiv preprint arXiv:1512.04412*, 2015.

[15] X. Chu, W. Ouyang, W. Yang, and X. Wang, "Multi-task recurrent neural network for immediacy prediction," in *ICCV*, 2015.

[16] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *CVPR*, 2016.

[17] S. Li, Z.-Q. Liu, and A. B. Chan, "Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network," in *CVPR*, 2014.

[18] C. Zhang and Z. Zhang, "Improving multiview face detection with multi-task deep convolutional neural networks," in *WACV*, 2014.

[19] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Facial landmark detection by deep multi-task learning," in *ECCV*, 2014.

[20] Y. Tian, P. Luo, X. Wang, and X. Tang, "Pedestrian detection aided by deep learning semantic tasks," *arXiv preprint arXiv:1412.0069*, 2014.

[21] G. Gkioxari, B. Hariharan, R. Girshick, and J. Malik, "R-cnns for pose estimation and action detection," *arXiv preprint arXiv:1406.5212*, 2014.

[22] P. Teterwak and L. Torresani, "Shared Roots: Regularizing Deep Neural Networks through Multitask Learning," Master's thesis, Dartmouth College, 2014.

[23] S. Branson, G. Van Horn, P. Perona, and S. Belongie, "Improved bird species recognition using pose normalized deep convolutional nets," in *BMVC*, 2014.

[24] D. Lin, X. Shen, C. Lu, and J. Jia, "Deep lac: Deep localization, alignment and classification for fine-grained recognition," in *CVPR*, 2015.

[25] J. Krause, H. Jin, J. Yang, and L. Fei-Fei, "Fine-grained recognition without part annotations," in *CVPR*, 2015.

[26] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang, "The application of two-level attention models in deep convolutional neural network for fine-grained image classification," *arXiv preprint arXiv:1411.6447*, 2014.

[27] N. Zhang, J. Donahue, R. Girshick, and T. Darrell, "Part-based r-cnns for fine-grained category detection," in *ECCV*, 2014.

[28] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014.

[29] D. Wang, Z. Shen, J. Shao, W. Zhang, X. Xue, and Z. Zhang, "Multiple granularity descriptors for fine-grained categorization," in *ICCV*, 2015.

[30] J. Deng, N. Ding, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, and H. Adam, "Large-scale object classification using label relation graphs," in *ECCV*, 2014.

[31] N. Srivastava and R. R. Salakhutdinov, "Discriminative transfer learning with tree-based priors," in *NIPS*, 2013.

[32] I. Goodfellow, Y. Bengio, and A. Courville, "Deep learning," 2016, book in preparation for MIT Press. [Online]. Available: http://www.deeplearningbook.org

[33] C. Linnaeus *et al.*, "Systema naturae, vol. 1," *Systema naturae, Vol. 1*, 1758.

[34] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Learning deep representation for face alignment with auxiliary attributes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 5, pp. 918–930, May 2016.

[35] A. H. Abdulnabi, G. Wang, J. Lu, and K. Jia, "Multi-task cnn model for attribute prediction," *Multimedia, IEEE Transactions on*, 2015.

[36] D. L. Silver and R. E. Mercer, "The parallel transfer of task knowledge using dynamic learning rates based on a measure of relatedness," *Connection Science*, 1996.

[37] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS Workshop*, 2011.

[38] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.