Multi-Task Generalization and Adaptation between Noisy Digit Datasets: An Empirical Study

Steffen Schneider[*1,2], **Alexander S. Ecker**[1], **Jakob H. Macke**[2], **Matthias Bethge**[1]
[1] University of Tübingen
[2] Technical University of Munich
[*] steffen.schneider@tum.de

Abstract

Transfer learning for adaptation to new tasks is usually performed by either fine-tuning all model parameters or parameters in the final layers. We show that good target performance can also be achieved on typical domain adaptation tasks by adapting only the normalization statistics and affine transformations of layers throughout the network. We apply this adaptation scheme to supervised domain adaptation on common digit datasets and study robustness properties under perturbation by noise. Our results indicate that (1) adaptation to noise exceeds the difficulty of widely used digit benchmarks in domain adaptation, (2) the similarity of the optimal adaptation parameters for different domains is strongly predictive of generalization performance, and (3) generalization performance is highest with training on a rich environment or high noise levels.

1 Introduction

After the initial success of deep learning algorithms in computer vision (Krizhevsky et al., 2012), transfer by fine-tuning either the final layers or all network parameters on a new visual domains became an important scheme for many application settings (Donahue et al., 2013).

While adaptation of earlier network layers is crucial for good performance on a target task that differs from the source domain (Kornblith et al., 2018), the question of *how* to perform this adaptation is less well investigated. In the context of multi-task, sequential, or meta-learning settings where the objective is to transfer and keep knowledge between different tasks, this question is of importance. Motivated by Geirhos et al. (2018) and previous results by Rebuffi et al. (2017), Cariucci et al. (2017) and Li et al. (2016), which indicate that adaptation of normalization statistics and a feature-map specific offset and scaling throughout a convolutional network is sufficient for adaptation with reasonable performance, we study this kind of multi-task generalization in typical domain adaptation scenarios.

We make the following contributions: First, we corroborate previous results that conditioning a model on a new task by only adapting a small set of normalization and affine parameters is sufficient for multi-task learning on reasonable performance levels. Second, we investigate the interplay between the "richness" of a task and the effects on generalization. Third, in controlled training settings with two different noise models, we find that (1) applying parameters obtained in a low-noise setting to the high-noise setting leads to degrading performance levels strongly correlated with the cosine similarity between the parameter vector and the optimally adapted parameter vector, and (2) parameters obtained in a high-noise setting generalize well to low-noise settings, even in our minimal fine-tuning scheme.

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.
Continual Learning Workshop.

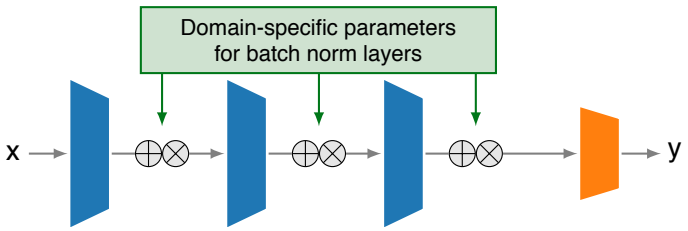

Figure 1: Model architecture using conditional batch normalization: By only adapting normalization statistics and the following affine transformation to the target domain, the task-specific behaviour is distilled into a few thousand parameters.

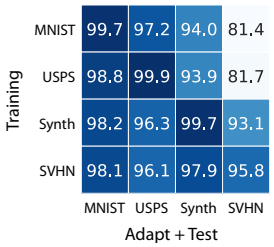

Figure 2: Domain adaptation matrix between four digit datasets.

Our experiments differ from previous work: In contrast to Rebuffi et al. (2017), we consider the classical domain adaptation setting, in which the label space is shared between tasks and the classifier does not have to be adapted.

2 Methods

We consider multi-task learning among a source task $\mathcal{S}$ and a set of target tasks $\{\mathcal{T}_n\}_{n=1}^N$, where by *task*, we consider an underlying data distribution of causes Y (here: labels), effects X (here: images) and conditions T, given by the task-specific probability density function (pdf) $\mathcal{T}_i(x, y) := p(x|y, t = i)p(y|t = i)$. As in many machine learning problems, we aim at inferring the conditional distribution of $Y|X$ with pdf $p(y|x)$ for anti-causal inference from a set of samples. In this work, we consider a "best-case" setting with access to samples from all tasks.

Structural Learning in Deep Networks. Structural learning is a term originally derived from sensorimotor research (Braun et al., 2009) and denotes a training setting in which by learning from many different environments, adaptation to a novel setting no longer requires an adaptation of all model parameters. Instead, adaptation within a lower-dimensional subspace of parameters is sufficient for fast and sufficiently accurate performance.

This motivates us to approach the process of fine-tuning deep neural networks not by adapting all parameters, but only a constrained (small) subset. We first train on a large labeled training set and then fine-tune on the labeled target data set.

Distilling multi-task knowledge into 0.1% of parameters. For simplification, we use a model $h_{\theta,\phi}$ with two sets of parameters. We optimize θ along with $\phi_{\mathcal{S}}$ on the source task $\mathcal{S}$. While usual fine-tuning would adapt both sets of parameters on a target task $\mathcal{T}$, we refrain from doing so and only fine-tune the selected subset ϕ for each target domain, solving the (independent) ERMs

$$\min_{\theta,\phi_{\mathcal{S}}} \mathbb{E}_{x^s,y^s\sim\mathcal{S}}[\ell(h_{\theta,\phi_{\mathcal{S}}}(x^s); y^s)] \quad \text{and} \quad \min_{\phi_1,\ldots,\phi_n} \sum_{n=1}^{N} \mathbb{E}_{x^t,y^t\sim\mathcal{T}_n}[\ell(h_{\theta,\phi_n}(x^t); y^t)]], \tag{1}$$

which can be either trained in parallel or sequentially as data becomes available, depending on the exact learning setup. In our experiments, we solve both ERMs jointly, but stress that no information about the target domains $\mathcal{T}_i$ is used to derive either θ or $\phi_{\mathcal{S}}$, as evident from Eq. 1. While different choices for this conditioning are possible, we choose to update the statistics for batch normalization, along with fine-tuning the following affine transformation, similar to previous work by Li et al. (2016) and Rebuffi et al. (2017), and as depicted in Fig. 2. Note that in this case, the dimensionality of ϕ is considerably smaller than θ, and fine-tuning to a particular task *is exclusively reflected by these parameters*, since θ was never trained on the target tasks. We used $\dim \phi_i \approx 8000$ and $\dim \theta \approx 10^6$ in the experiments presented below.

Tasks. We perform two different experiments with different underlying motivations. For the first set of experiments, we consider four popular digits datasets: MNIST (LeCun et al., 1998), Street View House Numbers (SVHN) (Netzer et al., 2011), USPS (Hull, 1994) and Synthetic Images (SYNTH)

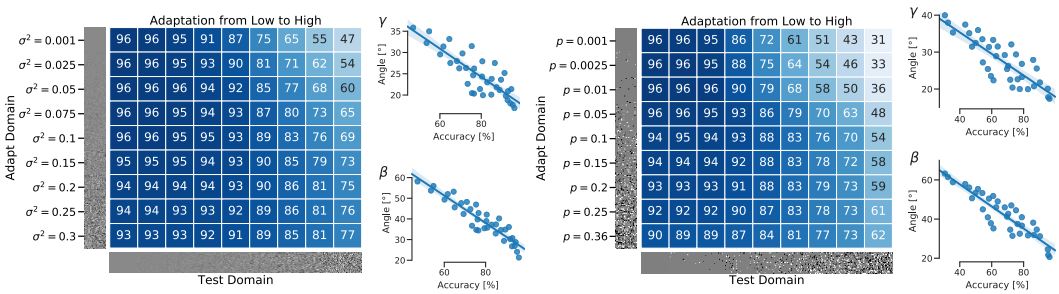

Figure 3: Adaptation from low noise to varying (higher) levels of Gaussian noise (left) and salt & pepper noise (right). Matrices show domain adapted on in rows and domain tested on in columns. Scatter plots show that performance correlates with the angles between the optimal adaptation parameters for source and target domain (scale: γ and offset: β).

(Ganin et al., 2016), yielding four distinct tasks. For the second set, we augment the SVHN dataset with either additive point-wise Gaussian noise (with varying variance $\sigma \in [0.001, 0.3]$) or point-wise Salt-and-Pepper (S&P) noise (with varying $p_{\text{S}} = p_{\text{P}} =: p/2 \in [0, 0.18]$). These perturbations yield nine tasks for each noise model. We select either the lowest or the highest noise variance as the source task $\mathcal{S}$.

Quantitative Comparison of Task-Specific Parameters. Due to the previously described conditioning of the model on a task, it is possible to compare models adapted to a particular task in a much smaller parameter space compared to a full neural network. For the problems and network model considered here, this amounts to around 8,000 parameters evenly split between bias, scale, mean and variance parameters. As a metric to compare tasks $\mathcal{T}_i$ and $\mathcal{T}_j$, we use the cosine similarity, i. e., the angle between the fine-tuned parameter vectors denoted as

$$\alpha_{ij} = \arccos \frac{\phi_i^T \phi_j}{\|\phi_i'\| \|\phi_j'\|}. \tag{2}$$

Domain Adaptation The fully supervised scenario provides an *upper-bound* for the performance of a multi-task learner under the chosen conditioning setting. Beyond this supervised scenario, we are interested in the reverse question of how much information can actually be leveraged without any target labels. This is the setting considered in unsupervised domain adaptation (cf. Ben-David et al. (2010) for a review). In addition to our main study, we apply several recent state-of-the-art domain adaptation algorithms on the considered task and conclude that surprisingly, the noise adaptation setting seems to be a harder challenge than adaptation between digit domains on SVHN. We evaluate the approaches of Sun et al. (2017), French et al. (2017), Haeusser et al. (2017) and Shu et al. (2018).

Implementation and model availability. We implemented all experiments using the `salad` toolbox (Schneider et al., 2018) for PyTorch (Paszke et al., 2017) and release the code for reproducing our results as well as trained models at `https://github.com/stes/nips2018-continual`.

3 Experiments and Results

Internal adaptation (almost) matches fine-tuning performance on digit benchmarks. In this first part, we demonstrate that by simply adapting the offset and scale of each feature map (Fig. 1), we achieve a performance comparable to that of recent domain adaptation algorithms on several digit adaptation tasks. We train a model on different image domains and fine-tune it to various target tasks with the proposed adaptation scheme. Our results show that training on "visually rich" domains is crucial for obtaining good transfer performance (Fig. 2), presumably because it enables the model to learn more diverse feature representations. This is a crucial insight for the development of domain adaptation algorithms and motivates the use of generative models and translation-based approaches when synthetic data is used during training time (cf. Shrivastava et al. (2016)).

Cosine Similarity is predictive of model performance. In our second experiment, we compare network properties when training on images perturbed by different types of noise (Gaussian and salt

& pepper) at different noise levels. We train networks in two different configurations on each of the two noise models. In the first part, we train a model in the low Gaussian or low S&P noise setting $\mathcal{S} := \mathcal{T}_1$, and fine-tune the network jointly on the remaining tasks $\mathcal{T}_2, \ldots, \mathcal{T}_9$. After convergence, the parameter configuration is given by the large set of parameters θ derived from the source task (training configuration), along with the small subset of parameters ϕ_i (scales and offsets) adapted to a particular target task $\mathcal{T}_i$ (adaptation). For each test task $\mathcal{T}_j$, we evaluate the performance of all adaptation configurations (θ, ϕ_i), yielding a total of 81 comparisons. As depicted in Fig. 3 and further in §C we find that, for both noise models, the parameter angle α_{ij} is predictive of the (degradation of) model performance, in cases where the noise during test time has greater variance than during training ($\sigma_j > \sigma_i$ and $p_j > p_i$).

Noise Robustness of Different Parameter Settings. This correlation is only observed in cases where the adaptation noise variance σ_i is lower than the testing noise variance σ_j, i.e., for all combinations of tasks in $\{\mathcal{T}_i, \mathcal{T}_j : \sigma_j > \sigma_i\}$, or the upper triangular part in Fig. 3. The observation does not hold for the lower triangular part, i.e., for cases where the adaptation noise was higher than the testing noise. In general, this result corroborates the general intuition that training with additionally "injected" noise variance, e.g. by means of data augmentation (Krizhevsky et al., 2012), improves model robustness under this particular noise model.

Noise Adaptation as Unsupervised Domain Adaptation. So far we considered the case where we have full label information on the target tasks to obtain upper bounds on the target performance. We now evaluate the noisy digit benchmark in an unsupervised setting. We test recent algorithms on the perturbed versions of SVHN and SYNTH with fixed S&P noise ($p_S = p_P = 0.15$) and find that especially the SVHN benchmark is surprisingly hard (Fig. 4). While transfer from SVHN to SYNTH is easily solved by most algorithms ($>95\%$ performance with most algorithms considered by Schneider et al. (2018)), transfer from SYNTH to a noisy version of SYNTH is already challenging for some algorithms. Similarly, transfer from SVHN to noisy SVHN seems to be a harder task than transfer from SYNTH to SVHN (88.1%/78.8%/89.7%/85.3% for the different solvers, cf. Schneider et al. (2018)). We conclude that unsupervised adaptation to noise is an interesting and challenging problem that should be considered in future work on domain adaptation.

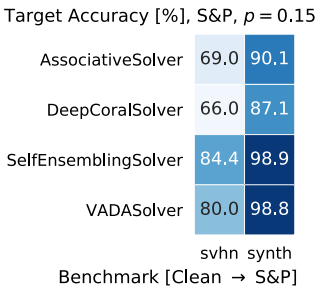

Figure 4: Performance of various domain adaptation algorithms adapting from clean to S&P-perturbed images.

Control Experiments, obtaining θ in the high noise case. In the second part, serving as a control experiment, we also train models in the high Gaussian and high S&P noise setting, $\mathcal{S} := \mathcal{T}_9$, and fine-tune jointly on $\mathcal{T}_1, \ldots, \mathcal{T}_8$. For the Gaussian noise model, performance degrades with the introduction of noise (from 96% to 85%). However, the adaptation scheme considered here is very effective for Gaussian noise, as indicated by the main diagonals in Fig. S1, with a performance gap of 77% vs. 85% for θ obtained on low and high noise, respectively. For the S&P model, this is not true, and full performance of 96% accuracy can be obtained in both settings. However, adaptation with our proposed scheme is not as effective as before, and the performance gap for evaluation in the high noise setting is much bigger (62% vs. 96%, cf. Fig. S2). Full results can be found in §C.2 and §C.4.

4 Conclusion

We studied adaptation and generalization properties of convolutional neural networks in different benchmark settings using digit datasets. We conclude that adaptation of a very limited subset of parameters (as little as 0.1% of network parameters) can be sufficient to provide competitive performance, *but only given* that the source task to train on is sufficiently "rich", thereby extending the previous results by Rebuffi et al. (2017). These findings motivate (1) the use of real world datasets over synthetic ones as well as (2) rethinking the use of translation-based domain adaptation algorithms as a method to "enrich" the distribution of a synthetic dataset.

To further investigate the latter result, we considered this behaviour more closely in the context of a synthetic, point-wise noise model in which the "richness" of a task is a controllable parameter. Our results in this controlled setting suggest that within the 8000-dimensional space of adaptation

parameters, configurations derived on high-noise domains attain a much lower empirical risk on any the target tasks than configurations obtained on low-noise domains. Given the limited dimensionality of the adaptation parameters, this motivates follow-up work in meta-learning, few shot adaptation and multi-task learning investigating whether it is possible to directly infer these configurations from properties of the data without relying on any label information.

Acknowledgements

We thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting St.S.

References

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175, 2010. ISSN 15730565. doi: 10.1007/s10994-009-5152-4. URL https://link.springer.com/content/pdf/10.1007/s10994-009-5152-4.pdf.

Daniel A. Braun, Ad Aertsen, Daniel M. Wolpert, and Carsten Mehring. Motor Task Variation Induces Structural Learning. *Current Biology*, 19(4):352–357, 2009. ISSN 09609822. doi: 10.1016/j.cub.2009.01.036. URL https://www.cell.com/current-biology/pdf/S0960-9822(09)00608-3.pdf.

Fabio Maria Cariucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulo. AutoDIAL: Automatic Domain Alignment Layers. *Proceedings of the IEEE International Conference on Computer Vision*, 2017-Octob(4):5077–5085, 2017. ISSN 15505499. doi: 10.1109/ICCV.2017.542.

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. 2013. ISBN 9781634393973. doi: 10.1007/978-3-319-51844-2{_}3. URL https://arxiv.org/pdf/1310.1531.pdfhttp://arxiv.org/abs/1310.1531.

Geoffrey French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for visual domain adaptation. pages 1–15, 2017. URL http://arxiv.org/abs/1706.05208.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, Victor Lempitsky, Urun Dogan, Marius Kloft, Francesco Orabona, and Tatiana Tommasi. Domain-Adversarial Training of Neural Networks. *Journal of Machine Learning Research*, 17:1–35, 2016. ISSN 1475-7516. doi: 10.1088/1475-7516/2015/08/013. URL https://arxiv.org/pdf/1505.07818.pdf.

Robert Geirhos, Carlos R. Medina Temme, Jonas Rauber, Heiko H. Schuett, Matthias Bethge, and Felix A. Wichmann. Generalisation in humans and deep neural networks. *Neural Information Processing Systems (NIPS) 2018*, 8 2018. URL http://arxiv.org/abs/1808.08750.

Philip Haeusser, Thomas Frerix, Alexander Mordvintsev, and Daniel Cremers. Associative Domain Adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2017-Octob, pages 2784–2792, 2017. ISBN 9781538610329. doi: 10.1109/ICCV.2017.301. URL http://arxiv.org/abs/1708.00938.

Jonathan J. Hull. A Database for Handwritten Text Recognition Research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 5 1994. ISSN 01628828. doi: 10.1109/34.291440. URL http://ieeexplore.ieee.org/document/291440/.

Simon Kornblith, Jonathon Shlens, Quoc V Le, and Google Brain. Do Better ImageNet Models Transfer Better? *CoRR*, 2018. URL https://arxiv.org/pdf/1805.08974.pdf.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks, 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 1998.

Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting Batch Normalization for Practical Domain Adaptation. Technical report, 2016. URL https://arxiv.org/pdf/1603.04779.pdf.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading Digits in Natural Images with Unsupervised Feature Learning. Technical report, 2011. URL http://ufldl.stanford.edu/housenumbers/.

Adam Paszke, Gregory Chanan, Zeming Lin, Sam Gross, Edward Yang, Luca Antiga, and Zachary Devito. Automatic differentiation in PyTorch. In *Advances in Neural Information Processing Systems 30*, number Nips, pages 1–4, 2017.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In I Guyon, U V Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 506–516. Curran Associates, Inc., 2017. URL http://arxiv.org/abs/1705.08045.

Steffen Schneider, Alexander S Ecker, Jakob H Macke, and Matthias Bethge. Salad: A Toolbox for Semi-supervised Adaptive Learning Across Domains, 2018. URL https://openreview.net/forum?id=S1lTifykqm.

Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Josh Susskind, Wenda Wang, and Russ Webb. Learning from Simulated and Unsupervised Images through Adversarial Training. Technical report, 2016. URL https://arxiv.org/pdf/1612.07828.pdf.

Rui Shu, Hung H Bui, Hirokazu Narui, and Stefano Ermon. A DIRT-T Approach to Unsupervised Domain Adaptation. *International Conference on Learning Representations*, 2018. URL https://arxiv.org/pdf/1802.08735.pdfhttp://arxiv.org/abs/1802.08735.

Baochen Sun, Jiashi Feng, and Kate Saenko. Correlation alignment for unsupervised domain adaptation. In *Advances in Computer Vision and Pattern Recognition*, number 9783319583464, pages 153–171. 2017. ISBN 2191-6586. doi: 10.1007/978-3-319-58347-1{_}8. URL https://arxiv.org/pdf/1612.01939.pdf.

Supplementary Material

A Network Modules

A.1 Conditional Layers

The adaptable part of our network architecture outlined below is a `ConditionalBatchNorm` layer, performing the operation:

$$f(x; c) = \frac{x - \mu(c)}{\sqrt{\sigma(c)^2 + \epsilon}} \odot \gamma(c) + \beta(c) \tag{3}$$

We note that other choices for f are certainly possible, i.e., depth-separated convolutions, residual connections as used by Rebuffi et al. (2017), and other adaptation modules. Extending our experimental setup is easily possible for these kind of experiments.

A.2 Model Architecture

For specific questions regarding model parameters, the exact training setups and hyperparameters, please refer to the software repository we are releasing along with this paper. As a first insight, we share the model architecture here:

```
DigitModel(
  (features): DigitFeatures(
    (norm):       InstanceNorm2d(3, eps=1e-05, momentum=0, affine=False,
                          track_running_stats=False)

    (conv1_1):    Conv2d(3, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (conv1_1_bn): ConditionalBatchNorm()
    (conv1_2):    Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (conv1_2_bn): ConditionalBatchNorm()
    (conv1_3):    Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (conv1_3_bn): ConditionalBatchNorm()
    (pool1):      MaxPool2d(kernel_size=(2, 2), stride=(2, 2), padding=0, dilation=1,
                       ceil_mode=False)

    (drop1):      Dropout(p=0.5)
    (conv2_1):    Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (conv2_1_bn): ConditionalBatchNorm()
    (conv2_2):    Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (conv2_2_bn): ConditionalBatchNorm()
    (conv2_3):    Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (conv2_3_bn): ConditionalBatchNorm()
    (pool2):      MaxPool2d(kernel_size=(2, 2), stride=(2, 2), padding=0, dilation=1,
                       ceil_mode=False)

    (drop2):      Dropout(p=0.5)
    (conv3_1):    Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1))
    (conv3_1_bn): ConditionalBatchNorm()
    (nin3_2):     Conv2d(512, 256, kernel_size=(1, 1), stride=(1, 1), padding=(1, 1))
    (nin3_2_bn):  ConditionalBatchNorm()
    (nin3_3):     Conv2d(256, 128, kernel_size=(1, 1), stride=(1, 1), padding=(1, 1))
    (nin3_3_bn):  ConditionalBatchNorm()
  )
  (classifier):   Linear(in_features=128, out_features=10, bias=True)
)
```

B Control Experiments: High Noise Training Settings

As a control experiments, we train networks in the high noise setting to get an understanding of upper-bound performance.

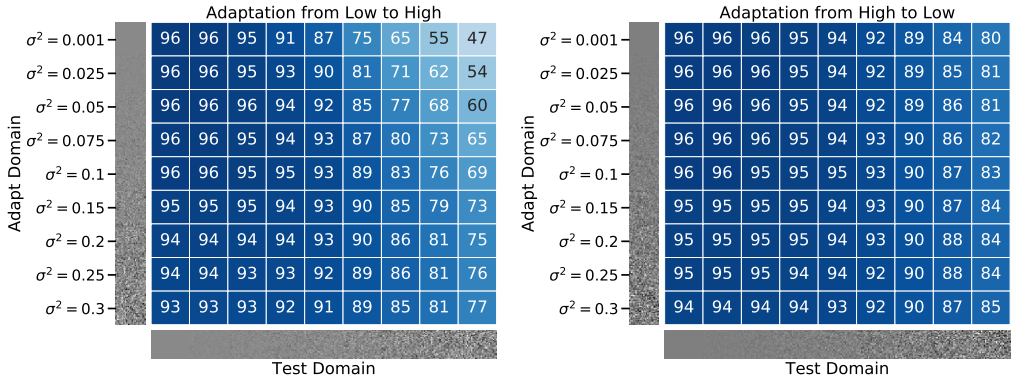

Figure S1: Comparision between adaptation results based on the training domain for θ: The left figure shows adaptation results when θ is derived on the low noise domain $\sigma_1^2 = 0.001$, and ϕ_i is adapted to the noise setting σ_i^2. The right figure shows adaptation results when θ is derived on the high noise domain $\sigma_9^2 = 0.3$, and ϕ_i is adapted to the noise setting σ_i^2. Notably, in the Gaussian noise setting, overall training performance degrades even in the case when all parameters θ, ϕ_9 are adapted to match the high noise setting, yielding a maximum achievable performance of 85% with the chosen model.

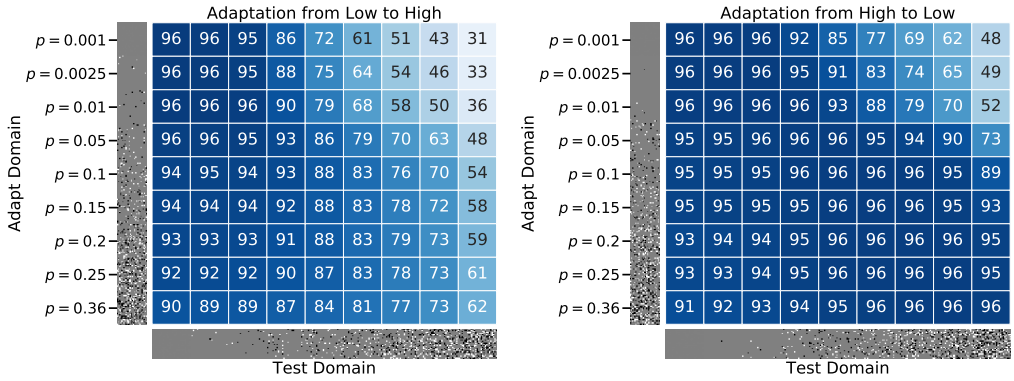

Figure S2: Similar organization as in Fig. S1, but for the Salt&Pepper case, where the main parameters are trained on low noise in the left, and in the high noise setting in the right plot. Crucially, in contrast to Gaussian noise, the adaptation scheme works less well when transferring from very severe noise to low noise, indicating that this perturbation scheme more closely resembles distinct tasks that require a larger amount of adaptation, at least in the extreme cases. Up to $p = 0.15$, the results more closely match the Gaussian case. Very interestingly, the model is able to fully adapt to the high S&P noise model, in contrast to the high variance Gaussian model.

C Parameter Angle is Predictive of Performance

We provide additional material for extending Fig. 3. The left part of Fig. 3 is a part of the full results depicted in §C.1, the right part of Fig. 3 is part of §C.3. For the control experiments, we show the same plots in §C.2 and §C.4 for Gaussian and Salt and Pepper noise, respectively. We note that in these settings, the correlation is less clear, especially for the Gaussian noise setting.

C.1 Gaussian Noise, Training on low noise domain

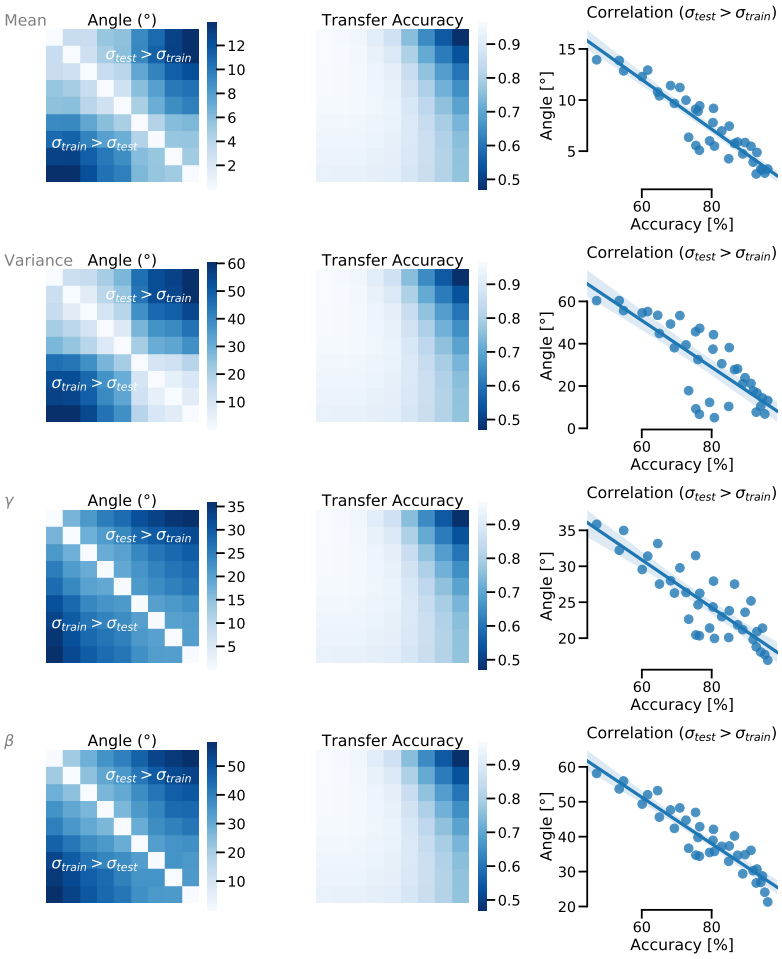

Figure S3: Relationship between angles of different adaptation parameters (top to bottom: $\mu(c)$, $\sigma^2(c)$, $\gamma(c)$, $\beta(c)$). Regarding transfer accuracy (middle), parameter angle is predictive of accuracy when adaptation is performed on a domain richer (with higher noise variance) than the test domain.

C.2 Gaussian, Training on high noise domain

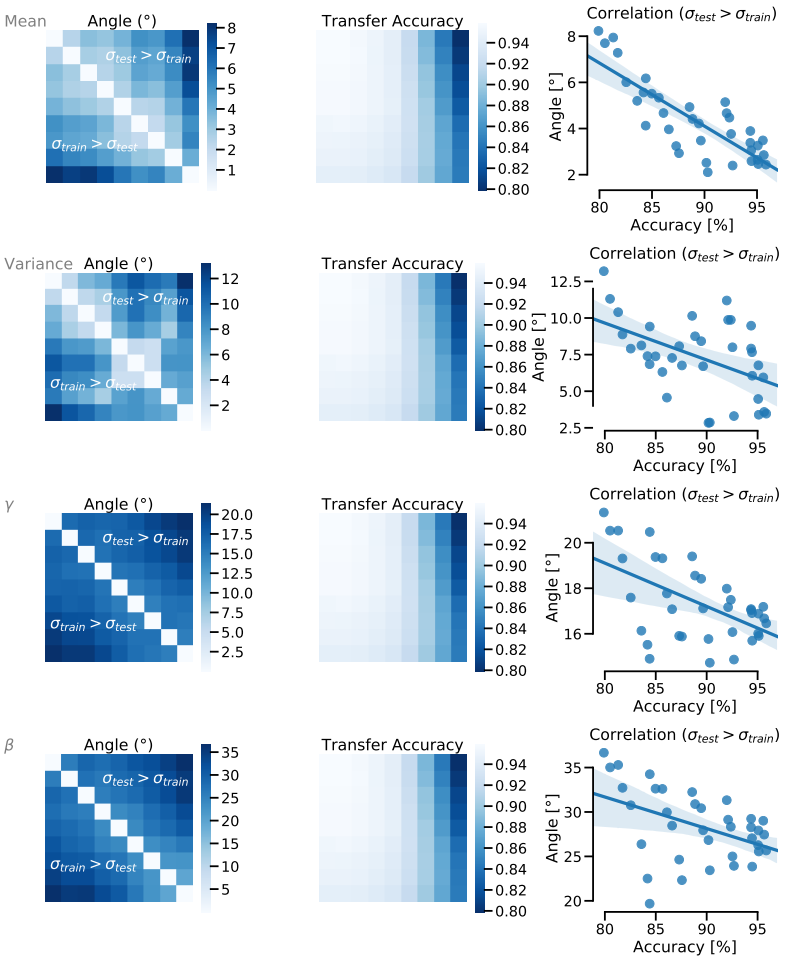

Figure S4: Control experiments with most parameters tuned on the high noise case: Relationship between angles of different adaptation parameters (top to bottom: $\mu(c)$, $\sigma^2(c)$, $\gamma(c)$, $\beta(c)$). Note that the resulting parameter angles are much more similar compared to the low-noise case, indicating the model weights transfer better.

C.3 Salt and Pepper Noise, Training on low noise domain

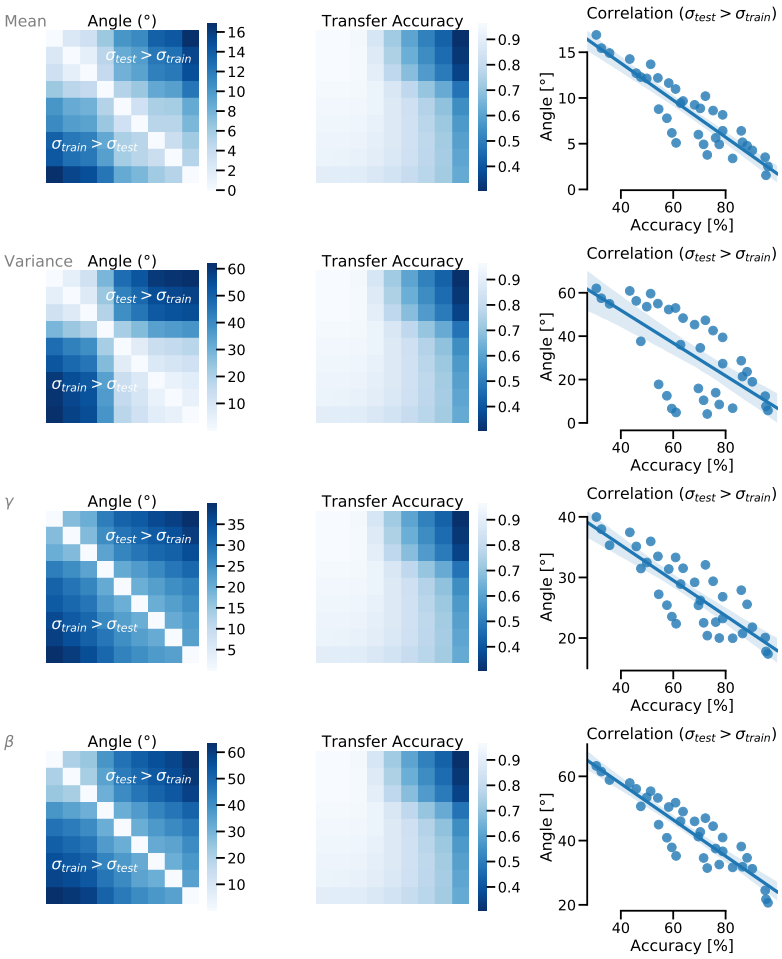

Figure S5: Relationship between angles of different adaptation parameters (top to bottom: $\mu(c)$, $\sigma^2(c)$, $\gamma(c)$, $\beta(c)$). Results are consistent with the Gaussian noise case, albeit the values are more scattered for the variance parameter.

C.4 Salt and Pepper Noise, Training on high noise domain

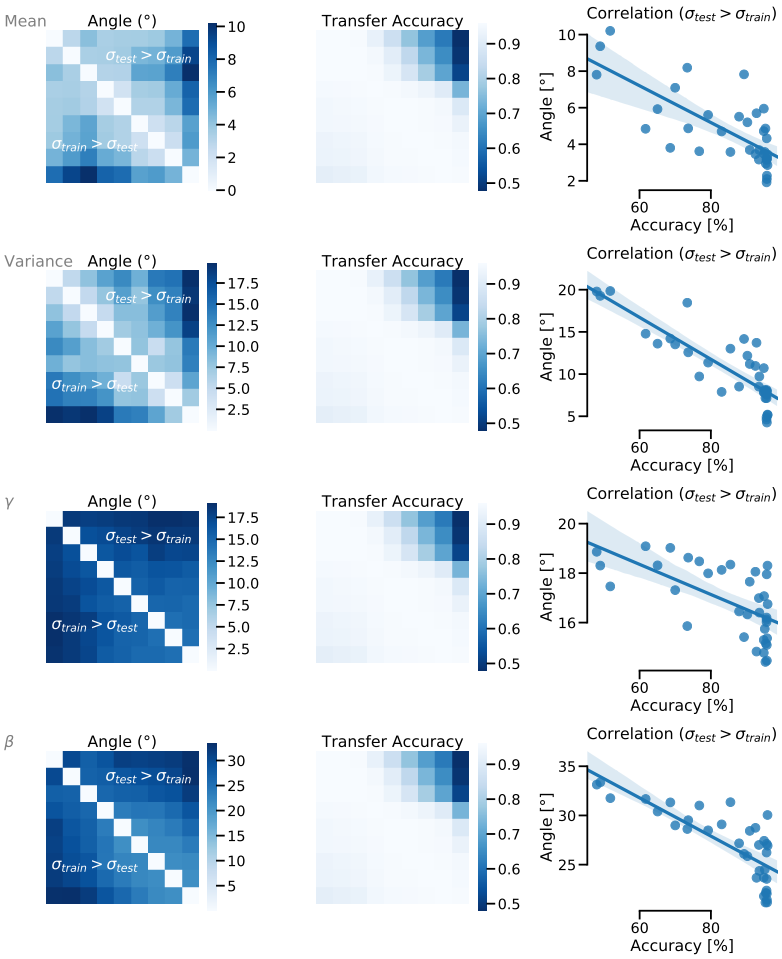

Figure S6: Relationship between angles of different adaptation parameters (top to bottom: $\mu(c)$, $\sigma^2(c)$, $\gamma(c)$, $\beta(c)$). As in the Gaussian noise case, less adaptation in the parameters is required. However, difference in parameter angles is bigger compared to the high Gaussian noise setting, which aligns with the result that transfer is more challenging in the S&P setting.

D Digit Adaptation

We show the full data for the supervised digit adaptation experiments in Fig. 2. In particular, for all tuples (i, j, k) denoting indices of the task for training, adaptation and testing, we derive parameters θ_i on the training task $\mathcal{T}_i$, adapt ϕ_j on the adaptation task $\mathcal{T}_j$, following the objective

$$\theta_i, \phi_i = \arg\min_{\theta_i, \phi_i} \mathbb{E}_{x,y\sim\mathcal{T}_i}[\ell(h_{\theta_i,\phi_i}(x); y)], \quad \phi_j = \arg\min_{\phi_j} \mathbb{E}_{x,y\sim\mathcal{T}_j}[\ell(h_{\theta_i,\phi_j}(x); y)], \tag{4}$$

and evaluate the risk on the target or test task $\mathcal{T}_k$ as

$$\mathbb{E}_{x,y\in\mathcal{T}_k}\ell(h_{\theta_i,\phi_i}(x); y). \tag{5}$$

For 4 domains, this yields 64 experiments in total.

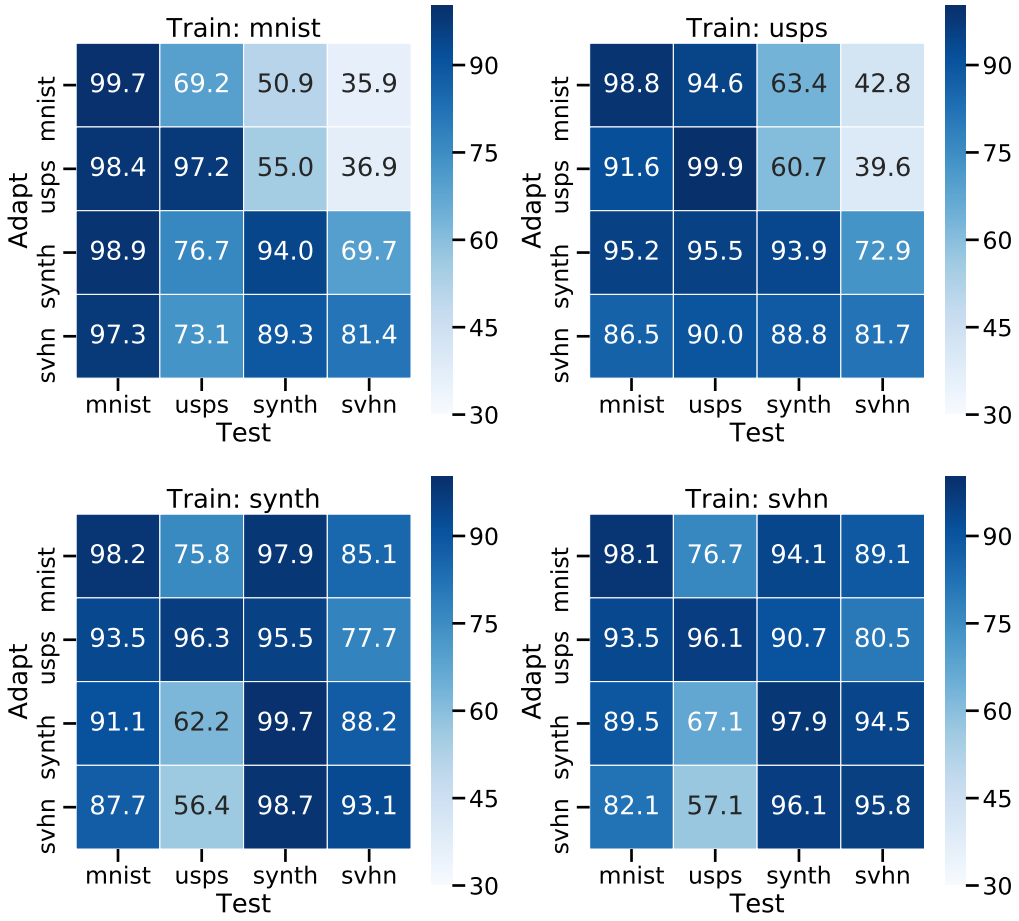

Figure S7: From left to right, we show the full results for training θ on MNIST, USPS, SYNTH or SVHN, respectively, along with all combinations for adaptation and testing. Note that in the main text, we only discuss the cases where adaptation and test domains are equivalent.