Uncertainty-Guided Online Test-Time Adaptation via Meta-Learning

Kyubyung Chae [1] **Taesup Kim** [1]

Abstract

In real-world scenarios, machine learning systems often encounter continual distributional shifts due to various factors in the test environment, resulting in unreliable predictions. Consequently, it is crucial to develop a model that can robustly adapt to the environment in an online manner. In this work, we propose a method to meta-learn how to guide unsupervised online adaptation, taking into account the uncertainty of predictions. Typically, all unlabeled test samples are equally integrated into online test-time adaptation. However, samples with high uncertainty can negatively impact the adaptation performance. Thus, we enable the model to adaptively learn from test samples by quantifying uncertainty during test-time online adaptation. Our experimental results demonstrate that our uncertainty-guided online adaptation enhances both robustness and adaptation performance during test-time on image classification tasks experiencing distributional shifts.

1. Introduction

In real-world applications, distributional shifts during test-time challenge the reliability of predictions in machine learning systems. Previous methods (Sun & Saenko, 2016; Li et al., 2018; Tzeng et al., 2017; Sun et al., 2016) have concentrated on aligning feature distributions between labeled source domains and unlabeled target domains, otherwise known as unsupervised domain adaptation. However, these methods require prior knowledge of the target distribution, which limits their practicality.

In contrast, test-time adaptation methods hold the advantage of not requiring target data during the training phase. Existing techniques (Sun et al., 2020; Zhang et al., 2022) generally depend on data augmentation methods. In addition, batch normalization (Li et al., 2016; Kaku et al., 2020; Nado et al., 2020; Schneider et al., 2020) and fine-tuning strategies (Wang et al., 2020; Liang et al., 2020) are primarily tailored for offline scenarios, and they lack effectiveness in online applications where test data is accrued incrementally (additional details can be found in the Appendix A.3). Moreover, the practice of treating all unlabeled test samples equally during test-time adaptation can have adverse effects, especially when considering the challenges posed by samples with uncertain predictions.

We propose an innovative approach to unsupervised online test-time adaptation that leverages both meta-learning and uncertainty estimation. Our goal is to enable the model to robustly adapt to distributional shifts in an online manner during test time by taking prediction uncertainty into account. We introduce a mechanism that selectively incorporates test samples during online adaptation by quantifying and utilizing their prediction uncertainty.

In summary, our work tackles the problem of distributional shifts by employing a meta-learning framework that facilitates uncertainty-guided online test-time adaptation. Through the integration of uncertainty estimation, we empower the model to adaptively use test samples during online adaptation, thereby enhancing its robustness and performance in real-world scenarios.

2. Preliminaries

In this section, we define our problem setting and describe an adaptive model for domain generalization by providing an explanation of the objective of *Adaptive Risk Minimization*. In this section, we establish our problem setting and describe an adaptive model for domain generalization, providing a comprehensive explanation of the objective of *Adaptive Risk Minimization*. In Section 3, we delve into uncertainty measurement and present our strategy for understanding the context of inputs. Ultimately, in Section 4, we empirically validate our methodology.

Problem setting We consider a classification problem where $x \in \mathcal{X}$ represents the input data and $y \in \mathcal{Y}$ stands for the output (target or label) data. The set of distributions on $\mathcal{X}$ X is denoted as $\mathcal{P}_x$, and the set of joint distributions on $\mathcal{X} \times \mathcal{Y}$ is denoted as $\mathcal{P}_{xy}$. We sample a joint distribution $p_{xy} \sim \mathcal{P}_{xy}$ and regard it as a *domain* defined by a specific

[1]Graduate School of Data Science, Seoul National University. Correspondence to: Taesup Kim <taesup.kim@snu.ac.kr>.

Published at the ICML 2023 Workshop on Spurious Correlations, Invariance, and Stability. Honolulu, Hawaii, USA.

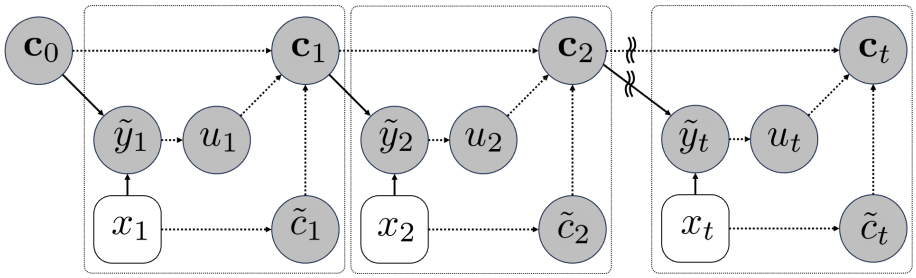


Figure 1. Overall procedure: The context vector $\tilde{c}_t$ extracted from the input image x_t is updated based on the uncertainty weight u_t (i.e., context update c_t). To measure the uncertainty, we employ predictive entropy, approximating the posterior predictive distribution by averaging the softmax output, denoted as $\tilde{y}_k$.

dataset or a user. Consequently, the distribution $\mathcal{P}_{xy}$ encompasses all potential domains. Under suitable regularity conditions (Kallenberg & Kallenberg, 1997), p_{xy} can be decomposed into $p_{xy} = p_x \cdot p_{y|x}$, where $p_{y|x}$ represents the conditional probability distributions of y given x. In our problem setting, we initially assume that a single underlying distribution p_{xy} exists during test-time. We then define our objective as enabling our model to adapt to the distribution p_{xy} by solely experiencing samples from p_x without any label information, and to maximize expected performance in an online setting during test-time (*i.e.*,unsupervised online adaptation).

Adaptive Risk Minimization (ARM) To achieve our objective of test-time adaptation, we first employ an adaptive model f that incorporates both the input data x and the marginal input distribution p_x from which x is sampled. This model is designed to adapt its prediction for x by utilizing p_x. In practice, it can be challenging to have direct knowledge of the marginal input distribution p_x and utilize it effectively. To address this challenge, Zhang et al. (2021) propose an approximation method. Rather than having full access to p_x, this approach initializes the model f with a batch of K input samples $x_1, ..., x_K$, which are assumed to be sampled from the same distribution p_x. The model f then makes predictions based on this batch, serving as an empirical approximation (e.g., a histogram) $\hat{p}_x$ of the true marginal distribution p_x. Based on this approach, the Adaptive Risk Minimization (ARM) framework (Zhang et al., 2021) was proposed to train the adaptive model f, which can effectively manage domain shifts. It comprises two modules: (1) the context model $h(\cdot;\phi) : \mathcal{X}^K \to \mathbb{R}^d$, which extracts a context feature $\mathbf{c}$, and (2) the prediction model $g(\cdot,\cdot;\theta) : \mathcal{X} \times \mathbb{R}^d \to \mathcal{Y}$, which makes a prediction based on the context feature $\mathbf{c}$. These models are parameterized by θ and ϕ, respectively. The context model $h\left(\{x_k\}_{k=1}^K;\phi\right)$ aims to approximate the marginal input distribution p_x with a batch of samples from it and extract the corresponding context feature $\mathbf{c}$. In turn, the prediction model $g(x', \mathbf{c};\theta)$ generates a prediction based on the extracted context in an adaptive manner. The ARM objective can be defined as:

$$\min_{\theta,\phi} \mathbb{E}_{\mathcal{P}_{xy}} \left[\mathbb{E}_{p_{xy}} \left[\frac{1}{K} \sum_{k=1}^{K} \ell\left(g\left(x_k, \mathbf{c};\theta\right), y_k\right) \right] \right],$$

where the overall risk is computed over all possible domains $p_{xy} \sim \mathcal{P}_{xy}$. Moreover, a batch of K samples is treated as a single episode sampled from a given domain p_{xy}. The objective can then be optimized within the meta-learning framework by utilizing multiple episodes. For a detailed explanation of a meta-learning approach to optimize the ARM objective, please refer to Appendix D.

3. Method

When a batch of K samples is given, the context model h in ARM approximates the marginal input distribution p_x by extracting context features from them. To accomplish this, it first separately processes each sample x_k in the batch to extract a sample-level context feature vector $c_k = \tilde{h}(x_k;\phi)$. A set of context feature vectors $c_k k = 1^K$ is then aggregated into a single context feature $\mathbf{c}$ using an average (pooling) operation: $\mathbf{c} = \frac{1}{K}\sum k = 1^K c_k$. This is often referred to as *context aggregation*, the goal of which is to provide an adapted context to the prediction model g. However, this approach is somewhat impractical in real-world scenarios, as it assumes that in each episode a batch of multiple samples is simultaneously and identically provided at test-time. To address this limitation, we modify this approach by leveraging a *cumulative sum* to effectively handle different batch sizes and improve robustness in online scenarios, which are more realistic than batch settings at test-time.

With a cumulative sum, we continually accumulate sample-level context feature vectors in an online manner, denoted as:

$$\mathbf{c}_k = \mathbf{c}_{k-1} + \tilde{h}(x_k;\phi),$$

where the initial context $\mathbf{c}_0 = \mathbf{0} \in \mathbb{R}^d$ is zero-initialized or can be learned as additional trainable parameters. This enables the ARM framework to extend the adaptation problem to more realistic settings, where samples are provided in an

online manner during test-time, such as in streaming or online settings. Consequently, each prediction $g(x_k, \mathbf{c}k, \theta)$ is conditioned on the context feature $\mathbf{c}k$, which is solely based on the experienced samples $xk'_{k' \leq k}$. Since the scale of the context feature $\mathbf{c}_k$ can proportionally increase with the number of experienced samples k, we apply Layer Normalization (Ba et al., 2016) to normalize its scale and stabilize the learning procedure. In summary, this modified approach enables the adaptive model to handle a wider range of online and few-shot scenarios, more akin to real-world situations, during meta-training in the ARM framework, thus leading to more robust learning.

3.1. Uncertainty-Guided Context Aggregation

We need a model that can adapt to a target (test) domain while only using unlabeled samples in an online setting. To achieve this, it's crucial to minimize the influence of unknown samples that may negatively impact overall performance. For instance, if a given sample, denoted as x_k, is uncertain due to its noisiness or if it comes from a different source distribution (*i.e.*,domain), it can adversely affect the overall performance by erroneously accumulating in the context feature.

To alleviate this issue, we utilize predictive entropy as a measure of uncertainty (Gal et al., 2017; Nair et al., 2020) to adaptively guide the procedure of accumulating context from the experienced samples. For each sample, predictive entropy measures the information in the model's categorical predictive distribution. It is defined as the negative sum of probabilities weighted by their logarithms, *i.e.*,$H(p) = -\sum_j p_j \log p_j$.

To correctly estimate the entropy and utilize it as a measure of uncertainty, we adopt a two-step process inspired by Gal & Ghahramani (2016) using Dropout layers. First, we approximate the posterior predictive distribution $\tilde{p}(y|x_k, \mathbf{c}_{k-1})$, where the context feature $\mathbf{c}_{k-1}$ includes context from samples $x_{k'k' < k}$. This is done by averaging the predictive distribution computed by softmax outputs $p(y|x_k, \mathbf{c}_{k-1}, \theta) = \mathrm{softmax}\left(f(x_k, \mathrm{c}_{k-1}; \theta)\right)$ obtained from T Monte Carlo samples with different Dropout masks ${\theta_t}_{t=1}^{T}$ sampled from the Dropout distribution $q(\theta)$.

More precisely, our approximate posterior predictive distribution is given by:

$$\begin{aligned}\tilde{p}(y|x_k, \mathbf{c}_{k-1}) &= \int p(y|x_k, \mathbf{c}_{k-1}, \theta) q(\theta) d\theta \\ &\approx \frac{1}{T} \sum_{t=1}^{T} \mathrm{softmax}\left(f(x_k, \mathrm{c}_{k-1}; \theta_t)\right).\end{aligned}$$

Based on this approximated distribution, we estimate its entropy and convert it into a measure of uncertainty for a given sample x_k as:

$$u_k = e^{-\beta \cdot H(\tilde{p}(y|x_k, \mathbf{c}_{k-1}))}$$

where β is a trainable positive value. Subsequently, we use this measure to adaptively guide the context aggregation as:

$$\mathbf{c}_k = \mathbf{c}_{k-1} + u_k \cdot \tilde{h}(x_k; \phi),$$

where the context aggregation assigns less weight when the prediction of a given sample is uncertain and vice versa. This aggregation method allows the adaptive model to handle uncertain samples effectively, minimizing their impact on its overall performance.

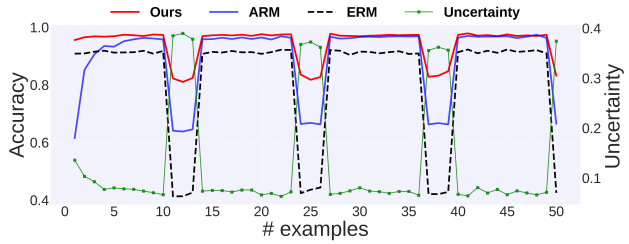
4. Experiments

4.1. Datasets and Evaluation protocol

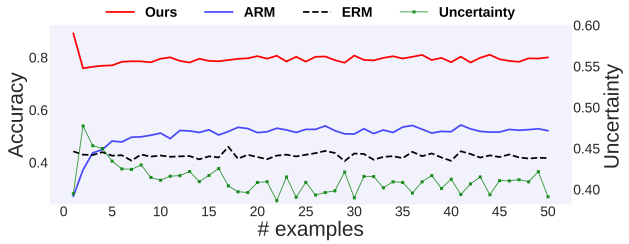
Rotated MNIST. In our study, we examine a variant of the MNIST dataset known as Rotated MNIST. The images in this dataset are rotated in 10-degree increments, spanning from 0 to 130 degrees. Each rotation is considered a distinct domain. We have limited training data for the smaller domains, with only 108 data points for rotations of 120 and 130 degrees, and 324 points for rotations between 90 and 110 degrees. The overall training set comprises 32,292 points. During testing, we generate images from the MNIST test set using various rotations and measure both the worst-case and average accuracy across domains.

Federated Extended MNIST (FEMNIST). We analyze the FEMNIST dataset, an extension of the EMNIST dataset that includes handwritten uppercase and lowercase letters, as well as digits (Cohen et al., 2017). FEMNIST further provides metadata indicating the user who generated each data point (Caldas et al., 2018). In our approach, we treat each user as a separate domain. To evaluate the performance of various methods, we measure both the worst-case and average accuracy across 35 test users. These users are distinct from the training users, which ensures a fair evaluation.

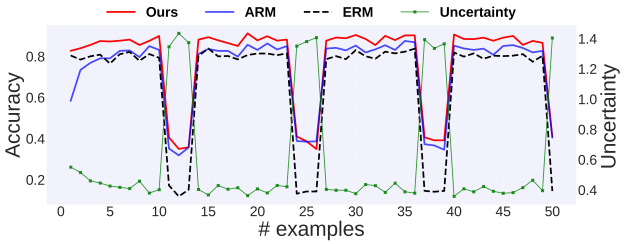
Online Scenarios In Figure 2 and Table 2, each *episode* refers to a scenario in which one sample arrives. The length of an episode is 50, which corresponds to the batch size used during training. The context feature vector accumulates within the episode. These episodes occur within the same domain, and the presented plots represent averaged results across different domains and three distinct seeds. In *Clean* episodes, no noise is present in any of the samples. Conversely, *Always* episodes contain salt-and-pepper noise in all samples. In *Cycle* episodes, there is a sequence of ten normal samples followed by three noisy instances, which then repeats.



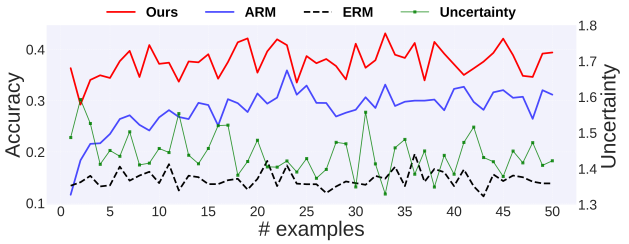
(a) Average Accuracy (MNIST, Cycle)



(b) Average Accuracy (MNIST, Always)



(c) Average Accuracy (FEMNIST, Cycle)



(d) Average Accuracy (FEMNIST, Always)

Figure 2. Comparison of Average Accuracy in Online scenarios

	MNIST		FEMNIST	
Method	**WC**	**Avg**	**WC**	**Avg**
ERM	74.5 ± 1.4	93.6 ± 0.4	62.4 ± 0.4	79.1 ± 0.3
UW	80.3 ± 1.2	95.1 ± 0.1	65.7 ± 0.7	80.3 ± 0.6
DRNN	79.9 ± 0.7	94.9 ± 0.1	57.5 ± 1.7	76.5 ± 1.2
DANN	78.8 ± 0.3	94.9 ± 0.1	65.4 ± 1.0	81.7 ± 0.3
MMD	82.4 ± 0.9	95.3 ± 0.3	62.4 ± 0.7	79.8 ± 0.4
BN adaptation	78.0 ± 0.3	94.4 ± 0.1	65.7 ± 1.5	80.0 ± 0.5
TTT	81.1 ± 0.3	95.4 ± 0.1	68.6 ± 0.4	84.2 ± 0.1
ARM	88.0 ± 0.8	96.3 ± 0.4	70.9 ± 1.4	86.4 ± 0.3
ARM (cusum)	88.1 ± 0.3	96.7 ± 0.1	71.8 ± 0.2	86.2 ± 0.0
Ours	$\mathbf{91.0 \pm 0.3}$	$\mathbf{97.2 \pm 0.1}$	$\mathbf{73.5 \pm 1.5}$	$\mathbf{87.8 \pm 0.2}$

Table 1. There are four categories of methods based on their use of components: none, training domains only, test batches only, or both training domains and test batches. These categories are separated by horizontal lines. The worst-case (WC) and average (Avg) top-1 accuracy across all datasets for three separate runs of each method are reported in terms of means and standard errors.

4.2. Comparisons and ablations

We compare our methods against robustness, invariance, and adaptation baselines including ERM, UW, DRNN (Sagawa et al., 2019), DANN (Ganin & Lempitsky, 2015), MMD (Li et al., 2018), BN adaptation (Ioffe & Szegedy, 2015; Schneider et al., 2020; Bronskill et al., 2020; Nado et al., 2020), TTT (Sun et al., 2020), and ARM (Zhang et al., 2021). ARM (cumsum) represents the result of using only the cumulative sum without measuring uncertainty. In other words, the uncertainty for all samples is set to zero. In the context aggregation, all samples carry the same weight of 1. More details can be found in Appendix C.

Method	**Type**	**MNIST**	**FEMNIST**
ARM (Offline)	Clean	96.3 ± 0.4	86.4 ± 0.3
	Cycle	90.5 ± 0.2	74.9 ± 0.1
	Always	52.5 ± 1.9	32.7 ± 4.5
ARM (Online)	Clean	95.3 ± 0.1	83.7 ± 0.1
	Cycle	89.1 ± 0.4	73.6 ± 0.4
	Always	50.7 ± 1.6	30.5 ± 4.3
Ours (Online)	Clean	$\mathbf{97.2 \pm 0.1}$	$\mathbf{87.8 \pm 0.2}$
	Cycle	$\mathbf{94.3 \pm 0.5}$	$\mathbf{78.3 \pm 0.4}$
	Always	$\mathbf{79.3 \pm 3.8}$	$\mathbf{39.0 \pm 2.0}$

Table 2. Comparison with ARM according to the scenario and noise episode.

4.3. Quantitative evaluation and comparisons

Our method in Table 1 excels in *offline* scenarios, surpassing other worst-case and average accuracy methods, where batch samples can be accessed simultaneously. It's significant to note that our approach was mainly intended for demanding online situations.

In general, methods that prioritize robustness and invariance may lead to unsatisfactory performance. While techniques like BN adaptation and TTT are widely recognized for their effectiveness in addressing image corruption, adaptations involving angles or users as domains do not yield favorable results. Our method and ARM rely less on favorable inductive biases yet consistently deliver superior results.

As shown on the left side of Figure 2, the ARM method displays notably weaker performance during the initial stages of an episode when there are limited samples available. This decline is further accentuated by its significant underperformance compared to ERM. In contrast to the ARM method,

our model demonstrates robust performance. Remarkably, our proposed model exhibits the ability to identify the occurrence of periodic incoming noise.

When looking at Figure 2, it is clear that the model's uncertainty level increases when there is noise, as shown by the green line. In the *Always* episode, as the episodes became consistently noisy and more noise samples accumulate, the uncertainty decreased. These results demonstrate that our model effectively handles challenging scenarios.

5. Conclusion

Our research focuses on addressing distributional shifts and enhancing model reliability for uncertainty measurement in testing scenarios. The empirical results validate the efficacy of our uncertainty-guided online adaptation approach in handling distributional shifts. Future work should prioritize overcoming architectural limitations in prediction models and exploring alternative approaches to diverse datasets. Additionally, improving uncertainty measurement with out-of-distribution samples is crucial. By addressing these areas, we can advance the reliability and performance of machine learning systems in real-world scenarios.

Acknowledgements

This work was supported by the New Faculty Startup Fund from Seoul National University and IITP (RS-2023-00232046).

References

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization, 2016.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural networks, 2015.

Bronskill, J., Gordon, J., Requeima, J., Nowozin, S., and Turner, R. E. Tasknorm: Rethinking batch normalization for meta-learning, 2020.

Caldas, S., Duddu, S. M. K., Wu, P., Li, T., Konečnỳ, J., McMahan, H. B., Smith, V., and Talwalkar, A. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

Cohen, G., Afshar, S., Tapson, J., and Van Schaik, A. Emnist: Extending mnist to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pp. 2921–2926. IEEE, 2017.

Dou, Q., Castro, D. C., Kamnitsas, K., and Glocker, B. Domain generalization via model-agnostic learning of semantic features, 2019.

Fernando, B., Habrard, A., Sebban, M., and Tuytelaars, T. Unsupervised visual domain adaptation using subspace alignment. In *Proceedings of the IEEE international conference on computer vision*, pp. 2960–2967, 2013.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

Fleuret, F. et al. Test time adaptation through perturbation robustness. In *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning, 2016.

Gal, Y., Islam, R., and Ghahramani, Z. Deep bayesian active learning with image data, 2017.

Ganin, Y. and Lempitsky, V. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pp. 1180–1189. PMLR, 2015.

Gopalan, R., Li, R., and Chellappa, R. Unsupervised adaptation across domain shifts by generating intermediate data representations. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2288–2302, 2013.

Hazan, E. et al. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

Hoffman, J., Kulis, B., Darrell, T., and Saenko, K. Discovering latent domains for multisource domain adaptation. In *ECCV (2)*, pp. 702–715, 2012.

Hoffman, J., Darrell, T., and Saenko, K. Continuous manifold based adaptation for evolving visual domains. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 867–874, 2014.

Hoffman, J., Mohri, M., and Zhang, N. Algorithms and theory for multiple-source adaptation. *Advances in Neural Information Processing Systems*, 31, 2018.

Hu, X., Uzunbas, G., Chen, S., Wang, R., Shah, A., Nevatia, R., and Lim, S.-N. Mixnorm: Test-time adaptation through online normalization estimation. *arXiv preprint arXiv:2110.11478*, 2021.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

Iwasawa, Y. and Matsuo, Y. Test-time classifier adjustment module for model-agnostic domain generalization. *Advances in Neural Information Processing Systems*, 34: 2427–2440, 2021.

Jiang, L. and Lin, T. Test-time robust personalization for federated learning, 2023.

Kaku, A., Mohan, S., Parnandi, A., Schambra, H., and Fernandez-Granda, C. Be like water: Robustness to extraneous variables via adaptive feature normalization. *arXiv preprint arXiv:2002.04019*, 2020.

Kallenberg, O. and Kallenberg, O. *Foundations of modern probability*, volume 2. Springer, 1997.

Kendall, A. and Gal, Y. What uncertainties do we need in bayesian deep learning for computer vision? In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Kurmi, V. K., Subramanian, V. K., and Namboodiri, V. P. Domain impression: A source data free domain adaptation method. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 615–625, 2021.

Lee, D.-H. et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, pp. 896, 2013.

Li, D., Yang, Y., Song, Y.-Z., and Hospedales, T. M. Learning to generalize: Meta-learning for domain generalization, 2017.

Li, H., Pan, S. J., Wang, S., and Kot, A. C. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5400–5409, 2018.

Li, P., Li, D., Li, W., Gong, S., Fu, Y., and Hospedales, T. M. A simple feature augmentation for domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8886–8895, 2021.

Li, R., Jiao, Q., Cao, W., Wong, H.-S., and Wu, S. Model adaptation: Unsupervised domain adaptation without source data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9641–9650, 2020.

Li, Y., Wang, N., Shi, J., Liu, J., and Hou, X. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016.

Liang, J., Hu, D., and Feng, J. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning*, pp. 6028–6039. PMLR, 2020.

Long, M., Cao, Y., Wang, J., and Jordan, M. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pp. 97–105. PMLR, 2015.

Nado, Z., Padhy, S., Sculley, D., D'Amour, A., Lakshminarayanan, B., and Snoek, J. Evaluating prediction-time batch normalization for robustness under covariate shift. *arXiv preprint arXiv:2006.10963*, 2020.

Nair, T., Precup, D., Arnold, D. L., and Arbel, T. Exploring uncertainty measures in deep networks for multiple sclerosis lesion detection and segmentation. *Medical Image Analysis*, 59:101557, 2020. ISSN 1361-8415. doi: https://doi.org/10.1016/j.media.2019.101557.

Neal, R. M. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Nichol, A., Achiam, J., and Schulman, J. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Patel, V. M., Gopalan, R., Li, R., and Chellappa, R. Visual domain adaptation: A survey of recent advances. *IEEE Signal Processing Magazine*, 32(3):53–69, 2015. doi: 10.1109/MSP.2014.2347059.

Qiao, F. and Peng, X. Uncertainty-guided model generalization to unseen domains. *CoRR*, abs/2103.07531, 2021.

Qiao, F., Zhao, L., and Peng, X. Learning to learn single domain generalization, 2020.

Roy, S., Trapp, M., Pilzer, A., Kannala, J., Sebe, N., Ricci, E., and Solin, A. Uncertainty-guided source-free domain adaptation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXV*, pp. 537–555. Springer, 2022.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

Schneider, S., Rusak, E., Eck, L., Bringmann, O., Brendel, W., and Bethge, M. Improving robustness against common corruptions by covariate shift adaptation, 2020.

Shalev-Shwartz, S. et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

Silver, D. L. and Mercer, R. E. The task rehearsal method of life-long learning: Overcoming impoverished data. In *Advances in Artificial Intelligence: 15th Conference of the Canadian Society for Computational Studies of Intelligence, AI 2002 Calgary, Canada, May 27–29, 2002 Proceedings 15*, pp. 90–101. Springer, 2002.

Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

Sun, B. and Saenko, K. Deep coral: Correlation alignment for deep domain adaptation. In *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14*, pp. 443–450. Springer, 2016.

Sun, B., Feng, J., and Saenko, K. Return of frustratingly easy domain adaptation. In *Proceedings of the AAAI conference on artificial intelligence*, 2016.

Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A., and Hardt, M. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pp. 9229–9248. PMLR, 2020.

Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., and Darrell, T. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

Tzeng, E., Hoffman, J., Saenko, K., and Darrell, T. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., and Darrell, T. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020.

Yeh, H.-W., Yang, B., Yuen, P. C., and Harada, T. Sofa: Source-data-free feature alignment for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 474–483, 2021.

Yin, D., Gontijo Lopes, R., Shlens, J., Cubuk, E. D., and Gilmer, J. A fourier perspective on model robustness in computer vision. *Advances in Neural Information Processing Systems*, 32, 2019.

Zhang, M., Marklund, H., Dhawan, N., Gupta, A., Levine, S., and Finn, C. Adaptive risk minimization: Learning to adapt to domain shift. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 23664–23678. Curran Associates, Inc., 2021.

Zhang, M., Levine, S., and Finn, C. Memo: Test time robustness via adaptation and augmentation, 2022.

Zhou, A. and Levine, S. Training on test data with bayesian adaptation for covariate shift. *arXiv preprint arXiv:2109.12746*, 2021.

A. Related Work

A.1. Domain Adaptation

Unsupervised domain adaptation (UDA) (Patel et al., 2015) is a technique that helps improve the performance of a target model when there is a discrepancy between the labeled source domain and the unlabeled target domain. During adaptation, UDA methods try to align the feature distributions between the two domains using discrepancy losses (Tzeng et al., 2014; Long et al., 2015; Sun & Saenko, 2016; Li et al., 2018) or adversarial training (Ganin & Lempitsky, 2015; Tzeng et al., 2017), or by aligning the input space (Fernando et al., 2013; Gopalan et al., 2013; Sun et al., 2016). This requires the assessment of the target distribution. In other words, DA assumes that the density function $p(\mathbf{x})$ is already known during training, which is often impractical for real-world applications.

A.2. Test-time Adaptation

Test-time adaptation methods are different from UDA methods as they do not require source data for adaptation. Some previous studies have used generative models to align features when source data is unavailable (Li et al., 2020; Kurmi et al., 2021; Yeh et al., 2021). However, anticipating diverse and challenging test distributions in advance can be difficult, limiting the generalization performance (Hoffman et al., 2012; 2014; 2018). Other methods, such as TTT (Sun et al., 2020), MEMO (Zhang et al., 2022; Jiang & Lin, 2023), and Fleuret et al. (2021), rely on image augmentation techniques, but these may not be applicable or effective for modalities like text and audio (Li et al., 2021).

Instead of seeking consistent features across domains, some methods perform fine-tuning using batches or entire datasets of test inputs. Batch normalization (BN) adaptations (Li et al., 2016; Kaku et al., 2020; Nado et al., 2020; Schneider et al., 2020) compute BN statistics on the test set, while others, like Iwasawa & Matsuo (2021), update only the final classification layer using class prototypes during test-time. Techniques like TENT (Wang et al., 2020) and SHOT (Liang et al., 2020) minimize entropy and employ diversity regularization to improve adaptation performance. In a Bayesian perspective, Zhou & Levine (2021) propose a regularized entropy minimization procedure that requires approximating density during training.

However, these methods are primarily designed for *offline* scenarios where all test data is available for training, which is not practical for *online* applications where test data is acquired incrementally over time.

A.3. Online Test-time Adaptation (OTA)

Online test-time adaptation refers to the scenario where test samples are received incrementally, either one by one or in small batches, rather than having access to all test samples simultaneously. In traditional *online learning* (Shalev-Shwartz et al., 2012; Hazan et al., 2016), the learning algorithm receives a test sample, predicts its label, and then receives the true label from an oracle to minimize regret. However, in OTA, true labels are not available during testing, rendering the concept of regret irrelevant. Consequently, in OTA, the learning algorithm must account for uncertainty since there is no direct feedback from the environment. Considering uncertainty enables the model to adapt and make informed decisions without relying on true labels. It is worth noting that the term *online learning* can have different interpretations outside the context of learning theory, leading to potential confusion or ambiguity (Sun et al., 2020).

A.4. Meta learning

Meta-learning is a powerful approach within machine learning that enables models to learn new tasks effectively by learning algorithms that can adapt and generalize to different domains. While most research in meta-learning has focused on few-shot learning (Finn et al., 2017; Nichol et al., 2018; Snell et al., 2017), a growing interest lies in extending meta-learning to tackle domain shift challenges. The Adaptive Risk Minimization (ARM) framework is a notable methodology that tackles domain shift (Zhang et al., 2021). By employing meta-learning paradigms, ARM focuses on unlabeled adaptation tasks and specifically addresses distribution shifts. However, ARM approaches are not designed to effectively learn from the source data when provided with limited target examples.

A.5. Uncertainty Quantification

In contrast with traditional deep learning models that produce deterministic predictions, Bayesian deep learning (Neal, 2012; Blundell et al., 2015; Kendall & Gal, 2017) is a popular approach that incorporates prior knowledge and expresses uncertainty in model predictions, especially in cases involving domain-shift or out-of-distribution samples. Although

mathematically rigorous, conventional Bayesian methods for deep learning have yet to be widely utilized in applications due to implementation challenges and lengthy training times.

Prior works in UDA (Silver & Mercer, 2002; Lee et al., 2013; Yin et al., 2019; Hu et al., 2021) have used deep ensembles for uncertainty quantification, while source-free DA scenarios have utilized feature augmentation and label mixup probabilistically (Qiao & Peng, 2021). Laplace approximation was applied in USDA (Roy et al., 2022), but Kronecker-factorization was assumed for large numbers of classes.

Gal & Ghahramani (2016) introduced a simpler approach to uncertainty estimation in deep learning, by training a dropout network and taking Monte Carlo (MC) samples of the prediction using dropout at test time. This approach produces an approximation of the posterior of the network's weights. In computer vision, this manner of modeling uncertainty has demonstrated improved performance in standard scene understanding networks without additional parameterization (Kendall & Gal, 2017). Therefore, while our framework can accommodate various uncertainty measurement methods, we adopt MC dropout due to its ease of implementation and minimal assumptions that could compromise domain adaptation.

B. Additional Experimental Details

We conducted our tests using various computing clusters equipped with NVIDIA RTX 3090 GPUs, allocating one GPU for each experiment. To ensure reliable results, we performed multiple runs for each method using three different seeds and reported the mean and standard error across these runs. The training hyperparameters were meticulously chosen based on preliminary experiments, with a focus on optimizing validation accuracy.

In our experiments, we followed a similar setup as described in the work by Zhang et al. (2021). We provide detailed information on how we constructed the dataset splits. These splits were designed independently of the training, validation, and test accuracies of any method. The design decisions were made intuitively, such as preserving the original data splits for MNIST or randomly, such as selecting users for different splits in FEMNIST. In some cases, we chose disjoint sets of corruptions with mostly different types for specific purposes.

Table 2 and Figure 2 display the results for the *Cycle* and *Always* noise episode, where the noise level is 10% salt-and-pepper noise. However, for FEMNIST, the noise level is set to 5%.

B.1. Rotated MNIST details

We create a training set comprising 32,292 data points by using 90% of the original training set. To construct this set, we sample and apply random rotations to each image. The rotations are independent of the image or label, but certain rotations are less frequently sampled. Specifically, rotations from 0 to 20 degrees (inclusive) have 7,560 data points each, rotations from 30 to 50 degrees have 2,160 points each, rotations from 60 to 80 degrees have 648 points each, rotations from 90 to 110 degrees have 324 points each, and rotations from 120 to 130 degrees have 108 points each.

We conducted training for all models over 200 epochs using mini-batches of size 300. Our optimization approach involved Adam updates with a learning rate of 0.0001, momentum of 0.9, and a cosine scheduler. To introduce an additional level of mini-batching similar to the ARM method, we adjusted the batch dimensions to 6×50, where each inner mini-batch contained examples from the same rotation. The image dimensionality remained at $28 \times 28 \times 1$, and we normalized the inputs by dividing them by 256. Convolutional neural networks were employed across all methods, with varying depths, to ensure fairness in terms of parameters.

In our evaluation process, we monitored validation accuracy every ten epochs. For each rotation, we estimated validation accuracy by randomly sampling 300 points from the held-out original training set and applying the specific rotation. This estimation procedure mirrored the test evaluation, which randomly sampled 3,000 test points and applied a specific rotation.

The network architecture consisted of two convolution layers with 128 filters of size 5×5, followed by 4×4 average pooling. Additionally, there was a fully connected layer of size 200 and a linear output layer. ReLU nonlinearities were used throughout the network, and batch normalization (BN) was applied to the convolution layers. Dropout layers were added after the ReLU activation in all layers, while the two convolution layers utilized 2×2 max pooling. Regarding the context network, it comprised two convolution layers with 64 filters of size 5×5, incorporating ReLU nonlinearities, BN, and padding. It was then followed by a final convolution layer with 12 filters of size 5×5 and padding.

B.2. FEMNIST details

Compared to MNIST, FEMNIST (as well as EMNIST) poses greater challenges due to its larger label space (62 classes compared to 10), label imbalance (with digits accounting for nearly half of the data), and inherent ambiguities (such as distinguishing lowercase and uppercase "o"). In our dataset processing, we filtered out users with fewer than 100 examples, resulting in 262, 50, and 35 unique users and a total of 62,732, 8,484, and 8,439 data points in the training, validation, and test splits, respectively. The most minor users contain 104, 119, and 140 data points, respectively.

With regard to the hyperparameters and evaluation process, we followed the same settings as for rotated MNIST, except for one modification: we set the meta batch size to 2. As for the network architectures, they remained unchanged, except for instances where the context network's last layer consisted of only one filter with a size of 5×5.

C. Comparisons Details

Empirical risk minimization (ERM) is a standard deep network training approach that aims to minimize the average loss on the training data. *Up-weighting* (UW) approach is an extension of ERM where groups are up-weighted to follow a uniform distribution. *Distributionally robust neural networks (DRNN)* (Sagawa et al., 2019) is a state-of-the-art method, which uses distributional regularization to build robust neural network models for group shifts. *Domain-Adversarial Neural Networks (DANN)* (Ganin & Lempitsky, 2015) is a method that utilizes a domain discriminator to learn domain-invariant representations. *Maximum Mean Discrepancy (MMD)* (Li et al., 2018) is a statistical measure used to quantify the difference between two distributions, often utilized in domain adaptation and domain generalization to minimize the distribution discrepancy across different domains.

BN adaptation refers to the process of updating the batch normalization statistics on the target data during testing, based on the approach introduced by Ioffe et al. (Ioffe & Szegedy, 2015), as further developed and discussed in the works (Schneider et al., 2020; Bronskill et al., 2020; Nado et al., 2020). *Test-Time Training (TTT)* is a method that involves the joint training of supervised and self-supervised tasks on the source domain, and during testing, it continues training only the self-supervised task on the target domain, as proposed by Sun et al. (Sun et al., 2020).

Adaptive Risk Minimization (ARM) (Zhang et al., 2021) is a domain generalization method that adapts to unseen domains by using context variables to capture domain-specific information and improve adaptation performance.

D. Optimizing the ARM objective via Meta-Learning

Unlabeled adaptation and meta-learning have traditionally been studied separately. However, the ARM framework (Zhang et al., 2021) integrates meta-learning tools to address domain shift problems, distinguishing it from previous meta-learning approaches focused on non-adaptive generalization performance (Li et al., 2017; Dou et al., 2019; Qiao et al., 2020). These earlier methods aimed to meta-train models for generalization across given training domains, while the ARM framework extends existing meta-learning techniques to effectively handle domain shift.

In our study, we adopt this framework. Algorithm 1 presents a comprehensive meta-learning approach for optimizing the ARM objective. As described earlier, the function h generates context variables c using an unlabeled batch of data (line 5). This mimics the adaptation process during testing, where label availability is not assumed (lines 7-8). However, the training update itself relies on labeled data (line 6). We assume that h is differentiable with respect to its input parameters θ and ϕ, allowing us to utilize gradient updates on both θ and ϕ to optimize the model's post-adaptation performance on a mini-batch of data sampled from a specific domain ρ.

Algorithm 1 Meta-Learning for ARM

`// Training Phase`

Require: # training steps T, batch size K, learning rate η

Initialize: θ, ϕ
for $t = 1$ to T **do**
  Sample ρ uniformly from training domains
  Sample $(x_k, y_k) \sim p(\cdot, \cdot|\rho)$ for $k = 1$ to K
  $c \leftarrow h(\theta, x_1, \ldots, x_K; \phi)$
  $(\theta, \phi) \leftarrow (\theta, \phi) - \eta \nabla_{(\theta,\phi)} \sum_{k=1}^{K} \ell(g(c, x_k; \theta), y_k)$
end for

`// Test time adaptation Phase`

Require: θ, ϕ, test batch $x_1, \ldots, x_K$

$c' \leftarrow h(\theta, x_1, \ldots, x_K; \phi)$
$\hat{y}_k \leftarrow g(c', x_k; \phi)$ for $k = 1, \ldots, K$