AdaME: Adaptive learning of multisource adaptation ensembles

Scott Yak
Google Research
New York, NY 10011
scottyak@google.com

Xavi Gonzalvo
Google Research
New York, NY 10011
xavigonzalvo@google.com

Corinna Cortes
Google Research
New York, NY 10011
corinna@google.com

Mehryar Mohri
Google & Courant Institute
New York, NY 10012
mohri@google.com

Abstract

We present a new adaptive algorithm to build multisource domain adaptation neural networks ensembles. Since the standard convex combination ensembles cannot succeed in this scenario, we present a learnable domain-weighted combination and new learning guarantees based on the deep boosting algorithm. We introduce and analyze a new algorithm, ADAME, for this scenario and show that it benefits from favorable theoretical guarantees, is risk-averse and reduces the worst-case mismatch between the inference and training distributions. We also report the results of several experiments demonstrating its performance in the FMOW-WILDS dataset.

1 Introduction

Generalization in supervised machine learning dramatically deteriorates as a function of the divergence between the training and test distributions. Transferring knowledge from multiple source domains to a target domain is appealing but performance decays due to domain shift (Torralba and Efros, 2011). In this paper, we are focusing on minimizing the impact of domain shift between the source and target domains via multi-source domain adaptation (MSA) where the labeled data may be collected from multiple sources with different distributions. The goal is to do well even on the worst-case subpopulation (e.g., a standard model performing poorly on under-represented demographics). We assume that the target domain can typically be viewed as a combination of the source domains, that is a mixture of their joint distributions, or it may be close to such mixtures. We focus on how the learner can adaptively combine relatively accurate predictors for each source domain to derive an accurate ensemble predictor for any new mixture target domain.

MSA was first theoretically studied by Mansour et al. (2008) and subsequently by Hoffman et al. (2018). Many methods have been proposed to address different challenges: distribution-weighted combination via conditional probabilities (Cortes et al., 2020a), semi-supervised domain adaptation (Saito et al., 2019) or multisource boosting (Cortes et al., 2021a). Recently, a common approach to transfer the task knowledge to the unlabeled target domain is an auxiliary feature alignment loss (Guo et al., 2018; Zhao et al., 2020).

Existing ensembling solutions have focused on combining domain-specific models via a distribution-weighted combining rule (Hoffman et al., 2012; Cortes et al., 2020a) which benefited from favorable theoretical guarantees (Hoffman et al., 2018). A successful approach to the combination of source predictors is the so-called Q-ensembles (Cortes et al., 2021a). These are convex combinations

Workshop on Distribution Shifts, 36th Conference on Neural Information Processing Systems (NeurIPS 2022).

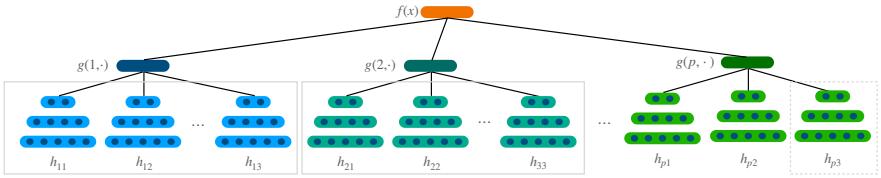

Figure 1: Illustration of the algorithm's incremental construction of a neural network.

weighted by a domain classifier Q, that is, $\mathsf{Q}(k|x)$ is the conditional probability of domain k given input point x.

Our contributions are as follows. (1) We extend the idea of example-dependent distribution-weighted Q-ensembles to a learnable gating mechanism. While a distribution-weighted combination with a domain classifier is reported to perform well (Cortes et al., 2020b, 2021a), low accuracy due to domain overlap can affect the performance of the domain-specific predictors; (2) We extend the work in (Cortes et al., 2021a) and present tighter theoretical guarantees based on deep boosting (Kuznetsov et al., 2014); (3) We generalize the idea of adaptively learning ensembles (i.e., AdaNet by Cortes et al. (2017b)) to the MSA case while maintaining the agnostic loss principle from (Cortes et al., 2021a). This reduces the worst-case mismatch between the test and training distributions;

2 Our approach

2.1 Learning scenario

Let $\mathcal{X}$ denote the input space and $\mathcal{Y} = \{-1, +1\}$ the output space associated to binary classification. We consider a scenario where the learner receives labeled samples from p source domains, each defined by a distribution $\mathcal{D}_k$ over $\mathcal{X} \times \mathcal{Y}$, $k \in [p]$. For any function $f \colon \mathcal{X} \to \mathbb{R}$ and distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, let $\mathcal{L}(\mathcal{D}, f)$ be the expected loss of f, that is $\mathcal{L}(\mathcal{D}, f) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(f(x), y)]$, where ℓ is the binary loss $\ell(f(x), y) = \mathbb{I}(yf(x) \leq 0)$.

For any $k \in [p]$, let $\mathcal{H}_k$ denote a hypothesis set of functions mapping from $\mathcal{X}$ to $[-1, +1]$, $|\mathcal{H}_k| = N_k$. The objective of the learner is to find a predictor f that is accurate for *any* target distribution $\mathcal{D}_\lambda$ that is a mixture of the source distributions, where λ may be in a subset Λ of the simplex. Thus, $\mathcal{D}_\lambda$ can be written as $\mathcal{D}_\lambda = \sum_{k=1}^{p} \lambda_k \mathcal{D}_k$, with $\lambda = (\lambda_1, \ldots, \lambda_p) \geq 0$ and $\sum_{k=1}^{p} \lambda_k = 1$.

To come up with a predictor f, the learner seeks an ensemble of functions from the base classes $\mathcal{H}_k$. The natural solution of a convex combinations (i.e., $\sum_t \alpha_t h_t$) would lead to a poor solution for some distributions (Proposition 1 in Cortes et al. (2021b)). Instead, it is possible to consider a combination of base predictors that takes into consideration the conditional probability of domain k given x, or $\mathsf{Q}(k|x)$. In practice, $\mathsf{Q}(k|x)$ can be approximated by training a dedicated classifier that maps the inputs $\mathcal{X}$ to the respective domain $k \in [p]$. In this case, the model becomes a domain-weighted ensemble where the learning process makes each base predictor domain-independent (Cortes et al., 2020b, 2021b).

In this paper, we propose to extend the Q-ensemble mechanism with a function g that combines the discriminative effect of $\mathsf{Q}(k|\cdot)$ and the performance of h_k across domains, $\forall k' \in [p]$, such that $k \neq k'$. Function g aims at mitigating two issues: (1) facilitate the combination of multiple sources in the base predictors and use transfer learning; (2) alleviate the detrimental effect of $\mathsf{Q}(k|\cdot)$ to the stability of the α's when the loss of the discriminators for a domain k is zero, and the α for cross-domain influence would be forced to be really large.

In the general case, this function g can be seen as a gating mechanism defining the problem as a mixture of experts (MoE) framework where each expert is trained with a base predictor of a domain. This function g is learned during training to serve as a gating function for a multi-source adaptation ensemble model. In this context, we consider the following form for the ensembles of base predictors:

$$\forall x \in \mathcal{X}, \quad f(x) = \sum_{l=1}^{p} g(l, x) \sum_{r=1}^{N_l} \alpha_{l,r} h_{l,r}(x), \tag{1}$$

where $g(l,x)$ refers to the gating function for domain l given x, $h_{l,r} \in \mathcal{H}_r^l$ and $\alpha_{l,r} \geq 0$, $\sum_{l=1}^p \sum_{r=1}^{N_l} \alpha_{l,r} = 1$. For any $k \in [p]$, define the family $\mathcal{J}_k$ as follows: $\mathcal{J}_k = \left\{g(k,\cdot)\, h \colon h \in \mathcal{H}_k, g \in \mathcal{G}\right\}$ and $\mathcal{H}_k = \left\{\bigcup_{r=1}^{N_k} \mathcal{H}_r^k\right\}$. Then, the family of ensemble functions $\mathcal{F}$ that we consider can be defined as $\mathcal{F} = \operatorname{conv}(\bigcup_{k=1}^p \mathcal{J}_k)$.

2.2 Learning guarantees

For any $\lambda \in \Delta$, let $\overline{\mathcal{D}}_\lambda$ be the distribution defined by $\overline{\mathcal{D}}_\lambda = \sum_{k=1}^p \lambda_k \widehat{\mathcal{D}}_k$, where $\widehat{\mathcal{D}}_k$ is the empirical distribution associated to an i.i.d. sample S_k drawn from $\mathcal{D}_k$. $\overline{\mathcal{D}}_\lambda$ is distinct from the distribution associated to $\mathcal{D}_\lambda$. We denote by $S_k = ((x_{k,1}, y_{k,1}), \ldots, (x_{k,m_k}, y_{k,m_k})) \in \left(\mathcal{X} \times \mathcal{Y}\right)^{m_k}$ the labeled sample of size m_k received from source k, which is drawn i.i.d. from $\mathcal{D}_k$.

Theorem 1. *Fix $\rho > 0$. Then, for any $\delta > 0$, with probability at least $1 - \delta$ over the draw of a sample $S = (S_1, \ldots, S_p) \sim \mathcal{D}_1^{m_1} \otimes \cdots \otimes \mathcal{D}_p^{m_p}$, the following inequality holds for all ensemble functions $f = \sum_{t=1}^T \alpha_t g(k_t,\cdot) h_t \in \mathcal{F}$ and all $\lambda \in \Delta$:*

$$\mathcal{L}(\mathcal{D}_\lambda, f) \leq \mathcal{L}_\rho(\overline{\mathcal{D}}_\lambda, f) + \sum_{k=1}^p \lambda_k \frac{8}{\rho}\left(\mathfrak{R}_{m_k}(\mathcal{G}) + \sum_{l=1}^p \sum_{r=1}^{N_l} \alpha_{l,r} \mathfrak{R}_{m_k}(\mathcal{H}_r^l)\right) + \frac{2}{\rho}\sum_{k=1}^p \sqrt{\frac{\lambda_k^2}{m_k}\log N_p} + C(\rho, N_p, \lambda_k, m_k, \delta/p). \tag{2}$$

where $N_p = \sum_{l=1}^p N_l$ and $C(\rho, N_p, \lambda_k, m_k, \delta/p) = \widetilde{O}\left(\frac{1}{\rho}\sqrt{\sum_{k=1}^p \frac{\lambda_k^2}{m_k}\log N_p}\right)$.

The bound of the theorem can be generalized to hold uniformly for all $\rho \in (0, 1]$, at the price of an additional term of the form $\sqrt{\log\log_2(2/\rho)/m_k}$ using standard techniques (Koltchinskii and Panchenko, 2002).

The proof of Theorem 1 can be found in Appendix A. This is the result of applying the deep boosting learning bounds (Kuznetsov et al., 2014) to our framework using the set of p multidomain ensembles in function class $\mathcal{J}_k$, $k \in [p]$, as the sub-families composing the base classifier set. The learning bounds in Eq. 2 represent an improvement to the guarantees presented by Cortes et al. (2021b). Our algorithm benefits from the mixture weight assigned to each sub-family, instead of depending on the worst-case domain Rademacher complexity in Eq. 5 of Theorem 1 in (Cortes et al., 2021b).

2.3 Ensemble learning

ADAME seeks to find a function $f = \sum_{t=1}^T \alpha_t g(k_t,\cdot) h_t \in \mathcal{F}$ that directly minimizes the data-dependent generalization bound of Theorem 1. For the k-th domain, the following objective function for an ensemble f is defined for any $\alpha_{l,j} \geq 0$:

$$F_k(\alpha) = \frac{1}{m_k}\sum_{i=1}^{m_k} \Phi\left(-y_{k,i} \sum_{l=1}^p \sum_{r=1}^{N_l} \alpha_{l,r} g(l, x_{k,i}) h_{r,l}\right) + \sum_{l=1}^p \sum_{r=1}^{N_l} (\beta_1 R_{l,r} + \beta_2) \tag{3}$$

where Φ is a convex, increasing and differentiable function such as the logistic function, and $R_{l,r} = \mathfrak{R}_{m_k}(\mathcal{H}_r^l)$ is the regularization term dependent on the Rademacher complexity of a base predictor r in domain l.

Given the *agnostic loss* (Mohri et al., 2019) of a predictor f leads to the following objective function:

$$F(\alpha) = \max_{\lambda \in \Lambda} \sum_{k=1}^p \lambda_k F_k(\alpha). \tag{4}$$

We will consider the case where the set Λ coincides with the full simplex, that is $\Lambda = \Delta$. F can then be expressed more simply as $F = \max_{k=1}^p F_k$. Since a convex function composed with an affine function is convex and a sum of convex functions is convex, F is convex as the maximum of a set of convex functions.

2.4 Domain-weighted ensemble learning

Model $g(k, \cdot)$ is trained using model distillation from the statically learned domain-weight $\mathsf{Q}(k|\cdot)$. For domain $k \in [p]$ and input $x \in \mathcal{X}$, model g with weights $w \in \mathbb{R}^d$ is defined as:

$$g(k, x) = \frac{\exp g_k(x)}{\sum_{j=0}^{p} \exp g_j(x)},$$

where $g_k(x)$ is the linear output for input x and domain k. The loss function for ensemble output f is defined as $\ell(f(x), y)$:

$$w^* = \operatorname*{argmin}_{w \in \mathbb{R}^d} \sum_{k=1}^{p} \sum_{i=1}^{m_k} \gamma \ell(f(x_{k,i}), y_{k,i}) + (1-\gamma) \mathsf{D}_{KL}(\mathsf{Q}(k|x_{k,i}), g(k, x_{k,i})), \tag{5}$$

where γ is a hyperparameter for the linear combination, D_{KL} is the Kullback-Leibler Divergence loss between the softened probability distributions of $\mathsf{Q}(k|x)$ acting as the teacher model and the $g(k, x)$ functioning as a student model with a temperature scaling hyperparameter τ.

2.5 Algorithm

The algorithm is described in Figure 2. The problem consists in alternating the optimization of the objective function F in Eq. 4 to find the α's (line 4) and Eq. 5 (line 14) to optimize w^*.

Since F is defined over a very large space of base functions, the first part of ADAME consists of applying coordinate descent to Eq. 4. The WEAKLEARNER function generates a number of candidate models from $\mathcal{H}_k$. For simplicity, the algorithm in Figure 2 generates 2 candidates (h and h', line 5) for which we are going to select the most optimal given Eq. 4 and the complexity from the theoretical guarantees in Eq. 2 (see line 11). The algorithm continues adding candidates until the complexity of a new candidate is larger that its improvement on the overall model. This adaptive process guarantees that the model is only going to be extended with new candidates within the learning bounds of Theorem 1. This is equivalent to minimizing the following objective function over α and $\{h, h'\}$:

ADAME($S = S_1, \ldots, S_p$)
1 $\alpha_0 \leftarrow 0$
2 **for** $t \leftarrow 1$ **to** T **do**
3 $f_{t-1} \leftarrow \sum_{l=1}^{p} g(l, \cdot) \sum_{j=1}^{N_l} \alpha_{t-1,l,j} h_{l,j}$
4 $k \leftarrow \operatorname{argmax}_{k \in [p]} F_k(\alpha_{t-1})$
5 $h, h' \leftarrow$ WEAKLEARNER(S, f_{t-1})
6 **if** $\min_\alpha F(\alpha, h) \leq \min_\alpha F(\alpha, h')$ **then**
7 $\alpha^* \leftarrow \operatorname{argmin}_\alpha F(\alpha, h)$
8 $h^* \leftarrow h$
9 **else** $\alpha^* \leftarrow \operatorname{argmin}_\alpha F(\alpha, h')$
10 $h^* \leftarrow h'$
11 **if** $F(\alpha_{t-1} + \alpha^*) < F(\alpha_{t-1})$ **then**
12 $f_{t-1} \leftarrow f_t + \alpha_k^* h_k^*$
13 **else return** f_{t-1}
14 $w^* \leftarrow Eq.\ 5$
15 **return** f_T

Figure 2: Pseudocode of the ADAME algorithm simplified for two base predictors h and h'.

$$F(\alpha, h) = \frac{1}{m_k} \sum_{i=1}^{m_k} \Phi\left(-y_{k,i} f_{t-1}(x_{k,i}) - y_{k,i} \alpha h(x_{k,i})\right) + \sum_{l=1}^{p} \sum_{r=1}^{N_l} (\beta_1 R_{l,r} + \beta_2)$$

The candidate generator in WEAKLEARNER produces base predictors trained with data from all domains that are adapted to a particular domain via standard transfer learning (Zhuang et al., 2019).

3 Experiments

In this Section, we present results for the FMOW-WILDS dataset (Christie et al., 2017; Koh et al., 2020). The "Baseline" figures are taken from the WILDS paper. For the ADANET and ADAME(G+Q) figures, we used ensembles of NFNets-F0 (Brock et al., 2021) that were pretrained on ImageNet, then finetuned on the FMoW training set, then further finetuned to adapt to each geographic region to produce 5 per-region ensembles with 5 NFNet-F0s each. The g model used for ADAME(G+Q) is also an NFNet-F0 (details in Appendix B). Table 1 reports average and worst-region accuracies (%) under time shifts in FMOW-WILDS. Models are trained on data before 2013 and tested on held-out location coordinates from in-distribution (ID) or out-of-distribution (OOD) test

sets. ID results correspond to the train-to-train setting. Parentheses show standard deviation across 3 replicates. In Appendix C we present an extensive set of results for different datasets (e.g., MNIST and sentiment analysis).

Table 1: Average and worst-region accuracies (%) for FMoW-WILDS.

	Algorithm	Validation (ID)	Validation (OOD)	Test (ID)	Test (OOD)
Average	Baseline	61.2 (0.52)	59.5 (0.37)	59.7 (0.65)	53.0 (0.55)
	ADANET	72.67 (0.43)	64.47 (0.35)	72.84 (0.61)	57.64 (0.46)
	ADAME(G+Q)	**73.04** (0.46)	**65.02** (0.51)	**72.97** (0.21)	**58.19** (0.47)
Worst	Baseline	59.2 (0.69)	48.9 (0.62)	58.3 (0.92)	32.3 (1.25)
	ADANET	67.71 (0.38)	52.80 (0.29)	65.68 (0.65)	39.79 (0.80)
	ADAME(G+Q)	**68.39** (0.52)	**53.18** (0.39)	**66.41** (0.51)	**40.46** (0.76)

4 Conclusions

We presented ADAME, an algorithm that provides a principled solution for adaptive structural learning of neural network ensembles for multiple source adaptation (MSA). This is an increasingly common and important learning problem due to the possibilities to improve the performance of the worst-case subdomains. Our solution extends the idea of adaptive ensembles to MSA by introducing a learnable distribution-weighted ensemble function (g). We presented tighter theoretical guarantees based on deep boosting so that the complexity of the ensemble doesn't depend on the complexity of the worst cross domain model. Our experimental results on the FMoW-WILDS dataset further demonstrate the effectiveness of our solution.

References

A. Brock, S. De, S. L. Smith, and K. Simonyan. High-performance large-scale image recognition without normalization. *CoRR*, abs/2102.06171, 2021. URL https://arxiv.org/abs/2102.06171.

G. A. Christie, N. Fendley, J. Wilson, and R. Mukherjee. Functional map of the world. *CoRR*, abs/1711.07846, 2017. URL http://arxiv.org/abs/1711.07846.

C. Cortes, M. Mohri, and U. Syed. Deep boosting. In *ICML*, pages 1179 – 1187, 2014.

C. Cortes, G. DeSalvo, and M. Mohri. Learning with rejection. In *ALT 2016*, 2016. URL https://cs.nyu.edu/~mohri/pub/rej.pdf.

C. Cortes, X. Gonzalvo, V. Kuznetsov, M. Mohri, and S. Yang. Adanet: Adaptive structural learning of artificial neural networks. In *Proceedings of ICML*, pages 874–883, 2017a.

C. Cortes, X. Gonzalvo, V. Kuznetsov, M. Mohri, and S. Yang. Adanet: Adaptive structural learning of artificial neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 874–883. JMLR. org, 2017b.

C. Cortes, M. Mohri, A. T. Suresh, and N. Zhang. Multiple-source adaptation with domain classifiers. *CoRR*, abs/2008.11036, 2020a. URL https://arxiv.org/abs/2008.11036.

C. Cortes, M. Mohri, A. T. Suresh, and N. Zhang. Multiple-source adaptation with domain classifiers. *CoRR*, abs/2008.11036, 2020b. URL https://arxiv.org/abs/2008.11036.

C. Cortes, M. Mohri, D. Storcheus, and A. T. Suresh. Boosting with multiple sources. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 17373–17387. Curran Associates, Inc., 2021a. URL https://proceedings.neurips.cc/paper/2021/file/9103820024efb30b451d006dc4ab3370-Paper.pdf.

C. Cortes, M. Mohri, D. Storcheus, and A. T. Suresh. Boosting with multiple sources. In *Proceedings of NeurIPS*, 2021b.

J. Guo, D. J. Shah, and R. Barzilay. Multi-source domain adaptation with mixture of experts. *CoRR*, abs/1809.02256, 2018. URL http://arxiv.org/abs/1809.02256.

J. Hoffman, B. Kulis, T. Darrell, and K. Saenko. Discovering latent domains for multisource domain adaptation. In *Proceedings, Part II, of the 12th European Conference on Computer Vision — ECCV 2012 - Volume 7573*, page 702–715, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 9783642337086.

J. Hoffman, M. Mohri, and N. Zhang. Algorithms and theory for multiple-source adaptation. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/2e2079d63348233d91cad1fa9b1361e9-Paper.pdf.

P. W. Koh, S. Sagawa, H. Marklund, S. M. Xie, M. Zhang, A. Balsubramani, W. Hu, M. Yasunaga, R. L. Phillips, S. Beery, J. Leskovec, A. Kundaje, E. Pierson, S. Levine, C. Finn, and P. Liang. WILDS: A benchmark of in-the-wild distribution shifts. *CoRR*, abs/2012.07421, 2020. URL https://arxiv.org/abs/2012.07421.

V. Koltchinskii and D. Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*, 30, 2002.

V. Kuznetsov, M. Mohri, and U. Syed. Multi-class deep boosting. In *NIPS*, 2014.

Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation with multiple sources. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2008. URL https://proceedings.neurips.cc/paper/2008/file/0e65972dce68dad4d52d063967f0a705-Paper.pdf.

M. Mohri, G. Sivek, and A. T. Suresh. Agnostic federated learning. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 4615–4625. PMLR, 2019.

K. Saito, D. Kim, S. Sclaroff, T. Darrell, and K. Saenko. Semi-supervised domain adaptation via minimax entropy. *CoRR*, abs/1904.06487, 2019. URL `http://arxiv.org/abs/1904.06487`.

A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528, 2011. doi: 10.1109/CVPR.2011.5995347.

S. Zhao, B. Li, C. Reed, P. Xu, and K. Keutzer. Multi-source domain adaptation in the deep learning era: A systematic survey. *CoRR*, abs/2002.12169, 2020. URL `https://arxiv.org/abs/2002.12169`.

F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He. A comprehensive survey on transfer learning. *CoRR*, abs/1911.02685, 2019. URL `http://arxiv.org/abs/1911.02685`.

A Proof of Theorem 1

Theorem 1. *Fix $\rho > 0$. Then, for any $\delta > 0$, with probability at least $1 - \delta$ over the draw of a sample $S = (S_1, \ldots, S_p) \sim \mathcal{D}_1^{m_1} \otimes \cdots \otimes \mathcal{D}_p^{m_p}$, the following inequality holds for all ensemble functions $f = \sum_{t=1}^{T} \alpha_t g(k_t, \cdot) h_t \in \mathcal{F}$ and all $\lambda \in \Delta$:*

$$\mathcal{L}(\mathcal{D}_\lambda, f) \leq \mathcal{L}_\rho(\overline{\mathcal{D}}_\lambda, f) + \sum_{k=1}^{p} \lambda_k \frac{8}{\rho} \left(\mathfrak{R}_{m_k}(\mathcal{G}) + \sum_{l=1}^{p} \sum_{r=1}^{N_l} \alpha_{l,r} \mathfrak{R}_{m_k}(\mathcal{H}_r^l) \right) + \frac{2}{\rho} \sum_{k=1}^{p} \sqrt{\frac{\lambda_k^2}{m_k} \log N_p} + C(\rho, N_p, \lambda_k, m_k, \delta/p). \tag{2}$$

where $N_p = \sum_{l=1}^{p} N_l$ and $C(\rho, N_p, \lambda_k, m_k, \delta/p) = \widetilde{O}\left(\frac{1}{\rho} \sqrt{\sum_{k=1}^{p} \frac{\lambda_k^2}{m_k} \log N_p} \right)$.

Proof. Assuming a single domain denoted by k and l, being $k, l \in [p]$, let $\mathcal{F}_l$ be a hypothesis set admitting a decomposition $\mathcal{F}_l = \bigcup_{r=1}^{N_l} \mathcal{G}_r^l$ for some $N_l > 1$. Fix $\rho > 0$, then for any $\delta > 0$, with probability at least $1 - \delta$ over the draw of a sample S_k of size m_k from $\mathcal{D}^{m_k}$, the following inequality holds for all $f_l = \sum_{r=1}^{N_l} \alpha_r \cdot g_r \in \mathcal{F}_l$ (Cortes et al., 2014, 2017a):

$$\mathcal{L}(\mathcal{D}_k, f) \leq \mathcal{L}_\rho(\widehat{\mathcal{D}}_k, f) + \frac{4}{\rho} \sum_{r=1}^{N_l} \alpha_{l,r} \mathfrak{R}_{m_k}(\mathcal{J}_r^l) + \frac{2}{\rho} \sqrt{\frac{\log N_l}{m_k}} + C(\rho, N_l, m_k, \delta).$$

Fix $\lambda \in \Delta$, for any $k \in [p]$, all ensembles $f = \sum_{t=1}^{T} \alpha_t g(k_t, \cdot) h_t \in \mathcal{F}$:

$$\mathcal{L}(\mathcal{D}_k, f) \leq \mathcal{L}_\rho(\widehat{\mathcal{D}}_k, f) + \frac{4}{\rho} \sum_{l=1}^{p} \sum_{r=1}^{N_l} \alpha_{l,r} \mathfrak{R}_{m_k}(\mathcal{J}_r^l) + \frac{2}{\rho} \sqrt{\frac{\log N_p}{m_k}} + C(\rho, N_p, m_k, \delta),$$

where $N_p = \sum_{l=1}^{p} N_l$.

By the union bound, the following inequalities hold simultaneously for all $k \in [p]$:

$$\mathcal{L}(\mathcal{D}_k, f) \leq \mathcal{L}_\rho(\widehat{\mathcal{D}}_k, f) + \frac{4}{\rho} \sum_{l=1}^{p} \sum_{r=1}^{N_l} \alpha_{l,r} \mathfrak{R}_{m_k}(\mathcal{J}_r^l) + \frac{2}{\rho} \sqrt{\frac{\log N_p}{m_k}} + C(\rho, N_p, m_k, \delta/p),$$

Multiplying each by λ_k and summing them up yields:

$$\mathcal{L}(\mathcal{D}_\lambda, f) \leq \mathcal{L}_\rho(\overline{\mathcal{D}}_\lambda, f) + \sum_{k=1}^{p} \lambda_k \frac{4}{\rho} \sum_{l=1}^{p} \sum_{r=1}^{N_l} \alpha_{l,r} \mathfrak{R}_{m_k}(\mathcal{J}_r^l) + \frac{2}{\rho} \sum_{k=1}^{p} \sqrt{\frac{\lambda_k^2}{m_k} \log N_p} + C(\rho, N_p, \lambda_k, m_k, \delta/p).$$

The Rademacher complexity of a product of two (or more) families of functions relates to the sum of the Rademacher complexity of each family (see Lemma 1 and (Cortes et al., 2016)). The Rademacher complexity of the family of functions $\mathcal{J}_{\mathcal{H},\mathcal{G}}$ is $\mathfrak{R}_{m_k}(\mathcal{J}_r^l) = 2(\mathfrak{R}_{m_k}(\mathcal{H}_r^l) + \mathfrak{R}_{m_k}(\mathcal{G}))$ which finalizes the proof.

Lemma 1. *Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be two families of functions mapping $\mathcal{X}$ to $[-1, +1]$. Let $\mathcal{F} = \{f_1 f_2 \colon f_1 \in \mathcal{F}_1, f_2 \in \mathcal{F}_2\}$. Then the empirical Rademacher complexities of $\mathcal{F}$ for any sample S of size m are bounded:*

$$\widehat{\mathfrak{R}}_S(\mathcal{F}) \leq 2(\widehat{\mathfrak{R}}_S(\mathcal{F}_1) + \widehat{\mathfrak{R}}_S(\mathcal{F}_2))$$

□

B FMoW-WILDS Experiments

Here we describe how the FMoW-WILDS models were trained. First, we loaded an NFNets-F0 (Brock et al., 2021) pretrained from ImageNet from Github[1], and zero-initialized the head weights. Next, we trained the head weights, then fine-tuned all weights on the training set of all domains using 5 random seeds to produce 5 different checkpoints. During this training process, we used CutMix, MixUp, random brightness, random contrast, random flip, and 90-degree rotations for image augmentation. For each checkpoint, we then further fine-tuned on each domain using the same image augmentation but with stochastic-depth dropout rate of 0.5. The Q-model is also an NFNet, but trained from random initialization to predict the domain id, using the same image augmentation and stochastic-depth dropout using 5 random seeds. Since we have five checkpoints, this gives us 5 domain-specific models *per domain*, which we ensemble together, giving us 5 domain-specific *ensembles*. Finally, we use the Q-model to ensemble the 5 domain-specific ensembles to produce the final ADAME(G+Q) ensemble.

B.1 Hyperparameters for finetuning FMoW(all)

- Model architecture: NFNets-F0 (pretrained from ImageNet)
- Base learning rate: 1e-3
- Batch size: 128
- Use batch norm: True
- Learning rate schedule: Linear warmup for 600 steps, then decay by 0.995 every 600 steps.
- Stochastic depth dropout rate: 0.1
- Num steps: 120, 000

B.2 Hyperparameters for finetuning FMoW(per-region)

- Model architecture: NFNets-F0 (pretrained from FMoW (all))
- Base learning rate: 1e-5
- Batch size: 64
- Use batch norm: True
- Learning rate schedule: Linear warmup for 400 steps, then decay by 0.96 every 300 steps.
- Stochastic depth dropout rate: 0.5
- Num steps: 60, 000

B.3 Hyperparameters for finetuning FMoW(Q)

- Model architecture: NFNets-F0 (random init)
- Base learning rate: 2e-4
- Batch size: 128
- Use batch norm: False
- Learning rate schedule: Linear warmup for 600 steps, then decay by 0.96 every 600 steps.
- Stochastic depth dropout rate: 0.1
- Num steps: 60, 000

[1] `https://github.com/deepmind/deepmind-research/tree/master/nfnets#pre-trained-weights`

Table 2: Test errors for multiple benchmarks.

Algorithm	Error-1	Error-2	Error-3	Error-Uniform	Error-Agnostic
			Sentiment Analysis		
AdaBoost-1	0.326 ± 0.019	0.300 ± 0.008	0.360 ± 0.017	0.329 ± 0.017	0.360 ± 0.019
AdaBoost-2	0.354 ± 0.020	$\mathbf{0.266 \pm 0.009}$	0.336 ± 0.023	0.318 ± 0.019	0.357 ± 0.019
AdaBoost-3	0.402 ± 0.015	0.334 ± 0.008	0.258 ± 0.015	0.331 ± 0.016	0.402 ± 0.018
AdaBoost-all	0.354 ± 0.020	0.325 ± 0.011	0.313 ± 0.022	0.324 ± 0.021	0.354 ± 0.016
DMSA	0.332 ± 0.021	0.308 ± 0.017	0.314 ± 0.015	0.318 ± 0.019	0.332 ± 0.021
MultiBoost	0.332 ± 0.027	0.288 ± 0.018	0.284 ± 0.027	0.301 ± 0.027	0.332 ± 0.024
AdaME(G+Q)	$\mathbf{0.291 \pm 0.014}$	0.278 ± 0.020	$\mathbf{0.246 \pm 0.017}$	$\mathbf{0.2662 \pm 0.012}$	$\mathbf{0.296 \pm 0.012}$
			Digits Recognition (4 vs. 9)		
AdaBoost-1	0.044 ± 0.007	0.615 ± 0.012	0.476 ± 0.022	0.379 ± 0.008	0.615 ± 0.012
AdaBoost-2	0.455 ± 0.014	0.299 ± 0.011	0.504 ± 0.015	0.420 ± 0.011	0.504 ± 0.015
AdaBoost-3	0.549 ± 0.034	0.488 ± 0.015	0.300 ± 0.013	0.446 ± 0.013	0.549 ± 0.034
AdaBoost-all	0.060 ± 0.009	0.374 ± 0.015	0.353 ± 0.012	0.262 ± 0.009	0.374 ± 0.015
DMSA	0.069 ± 0.005	0.351 ± 0.012	0.310 ± 0.011	0.243 ± 0.009	0.351 ± 0.015
MultiBoost	0.096 ± 0.008	0.283 ± 0.028	0.246 ± 0.014	0.209 ± 0.013	0.284 ± 0.027
AdaME(G+Q)	$\mathbf{0.008 \pm 0.003}$	$\mathbf{0.064 \pm 0.007}$	$\mathbf{0.036 \pm 0.005}$	$\mathbf{0.036 \pm 0.002}$	$\mathbf{0.064 \pm 0.007}$
			Digits Recognition (1 vs. 7)		
AdaBoost-1	0.005 ± 0.002	0.613 ± 0.007	0.519 ± 0.012	0.379 ± 0.004	0.613 ± 0.007
AdaBoost-2	0.431 ± 0.022	0.252 ± 0.009	0.479 ± 0.012	0.387 ± 0.010	0.479 ± 0.012
AdaBoost-3	0.680 ± 0.031	0.490 ± 0.014	0.244 ± 0.012	0.474 ± 0.013	0.680 ± 0.031
AdaBoost-all	0.014 ± 0.003	0.286 ± 0.010	0.306 ± 0.012	0.202 ± 0.005	0.306 ± 0.011
DMSA	0.012 ± 0.003	0.288 ± 0.017	0.286 ± 0.015	0.195 ± 0.013	0.288 ± 0.017
MultiBoost	0.026 ± 0.004	0.261 ± 0.013	0.257 ± 0.015	0.181 ± 0.005	0.261 ± 0.011
AdaME(G+Q)	$\mathbf{0.003 \pm 0.002}$	$\mathbf{0.082 \pm 0.005}$	$\mathbf{0.015 \pm 0.004}$	$\mathbf{0.033 \pm 0.002}$	$\mathbf{0.083 \pm 0.005}$
			Objects Recognition (Fashion-MNIST)		
AdaBoost-1	0.015 ± 0.003	0.251 ± 0.026	0.602 ± 0.028	0.288 ± 0.017	0.602 ± 0.028
AdaBoost-2	0.435 ± 0.007	0.015 ± 0.002	0.169 ± 0.012	0.173 ± 0.003	0.435 ± 0.007
AdaBoost-3	0.311 ± 0.018	0.097 ± 0.005	0.014 ± 0.002	0.140 ± 0.006	0.311 ± 0.018
AdaBoost-all	0.036 ± 0.004	0.020 ± 0.002	0.025 ± 0.003	0.027 ± 0.002	0.036 ± 0.004
DMSA	0.033 ± 0.008	0.015 ± 0.002	0.022 ± 0.003	0.023 ± 0.007	0.033 ± 0.009
MultiBoost	0.028 ± 0.003	0.015 ± 0.003	0.022 ± 0.002	0.021 ± 0.001	0.028 ± 0.003
AdaME(G+Q)	$\mathbf{0.006 \pm 0.002}$	$\mathbf{0.001 \pm 0.001}$	$\mathbf{0.001 \pm 0.001}$	$\mathbf{0.003 \pm 0.001}$	$\mathbf{0.006 \pm 0.002}$

C Additional Benchmarks

In this section, we present experimental results for the AdaME(G+Q) algorithm on several other multiple-source datasets (see Table 2). Note that the previous study (including AdaBoost, DMSA, and MultiBoost) is restricted to learning an ensemble of decision stump, which AdaME learns an ensemble of neural networks. For Sentiment Analysis, AdaME uses multi-head self-attention for the base models, the Q-model, and the G-model. For Digits Recognition and Objects Recognition, AdaME uses a feed-forward CNN for the base models, the Q-model, and the G-model.