Multiple-source adaptation with the Hedge algorithm

Anonymous Authors

Abstract

We consider the problem of unsupervised domain adaptation when multiple labeled source domains are available, but the target domain is an unknown mixture of these sources. A seminal theoretical framework by [25] shows the existence of a universal distribution-weighted ensemble that competes with the best source-specific model on any such target mixture. Subsequent work [37, 5] proposed practical algorithms based on difference-of-convex (DC) programming, which, however, does not guarantee global optimality. We propose an algorithmic solution to the same multiple-source domain adaptation problem, which is based on Hedge [1] and has global convergence guarantee. We also conducted experiments on standard image and text benchmarks to evaluate our method against DC programming and other baselines.

1 Introduction

Domain adaptation (often discussed alongside transfer learning) [28] addresses the challenge of learning under a distribution shift between training and testing data. In the unsupervised domain adaptation setting, a learner is provided with a labeled source dataset and an unlabeled target dataset, with the objective of achieving strong predictive performance on the target domain.

When multiple source domains are available alongside access to the target distribution (or a sample thereof), the multiple-source domain adaptation problem can often be reduced to a single-source setting by simply pooling the sources. However, this reduction may overlook structural relationships between domains that could be leveraged through careful causal or distributional analysis [36]. In contrast, the multiple-source domain adaptation framework introduced by [25] offers a distinct advantage: it enables test-time adaptation [32, 33, 10] with theoretical guarantees, without requiring model re-training. This paradigm shares strong conceptual ties with domain generalization [34, 41, 3, 27], which similarly aims to achieve robust performance on unseen target distributions.

Under the framework of [25], the learner is given access to k pre-trained models, each optimized on a distinct source domain. The objective is to combine these models into a single ensemble that generalizes well to any target distribution formed as an unknown mixture of the sources. The authors theoretically

established the existence of a universal, distribution-weighted ensemble whose error on any target mixture is bounded by the error of the best source-specific model evaluated on its respective domain.

The first algorithmic realization of this universal classifier was proposed by [37], who formulated the ensemble weighting problem as a difference-of-convex (DC) program. Although effective, DC optimization generally does not guarantee convergence to a global optimum. In this work, we propose an alternative approach: we frame the problem as an online learning game and solve it using the Hedge algorithm [22, 8, 1]. Our method is conceptually straightforward, easy to implement, and comes with global convergence guarantees. We provide a rigorous theoretical analysis demonstrating that the Hedge-based ensemble converges to the optimal solution, and we empirically validate its effectiveness on standard digit and text classification benchmarks.

2 Preliminaries

Problem setup We assume there are k labeled source domains $D_i, i \in [k]$. The target distribution is defined as $D_\lambda(x) = \sum_{i=1}^{k} \lambda_i D_i(x)$ for any λ in the probability simplex Δ_k. We have access to models $h_i, i \in [k]$, each of which is already well-trained on its respective source domain. Concretely, for each i, the expected loss of h_i on D_i is bounded by ϵ:

$$L_{D_i}(h_i) := \mathbb{E}_{x \sim D_i}\, \ell(h_i(x), f(x)) \leq \epsilon.$$

We assume the labeling function f is the same across domains, so the only distribution shift is in the marginal over x (covariate shift). The goal is to predict well on samples drawn from an unknown mixture D_λ for any $\lambda \in \Delta_k$.

Similar assumptions also emerged in relevant fields; in distributional robust optimization, mixture-domain assumption was utilized to develop an optimization procedure that can handle latent distribution variation in robust learning [7]. In federated learning, client-side local data can have different distribution than the global data distribution, which can be effectively viewed as a mixture of client data [35, 21, 18, 26].

Distribution-weighted adaptation is a simple principle to achieve domain adaptation. It was introduced to single-source domain adaptation problems by [30, 29, 16]. The idea is to shift the source distribution D_s towards the target distribution D_t in the training objective [31]:

$$\mathbb{E}_{x \sim D_t}\, \ell(h(x), y) = \mathbb{E}_{x \sim D_s}\, \frac{D_t(x)}{D_s(x)} \ell(h(x), y) \tag{1}$$

where $\mathbb{E}_D\, \ell(h(x), y)$ is the expected loss of predictor h on samples (x, y) from a domain D. When there is no labeling-distribution shift conditional on x, the unlabeled data in both the source and target domains can be used to estimate the density ratio, $\frac{D_t(x)}{D_s(x)}$, which re-weighs loss point-wise.

In [25], the authors extended the idea of distribution-weighted adaptation to leverage multi-source dataset. A key result of [25] is the proof of existence of a distribution-weighted ensemble predictor of the form

$$h_z(x) = \sum_{i=1}^{k} \frac{z_i D_i(x)}{\sum_{j=1}^{k} z_j D_j(x)} h_i(x), \tag{2}$$

Given that each h_i is a model trained from source i, the authors proved the existence of a parameter z so that h_z has error competitive to the worst in-domain model in the ensemble on any target mixture.

3 Related works

Transfer learning. Principled approaches include density-ratio reweighting [29, 30, 31] and feature alignment based on kernel mean matching [11, 16]. The latter is grounded in the theory of [2] and has been realized via domain-adversarial training [9, 15], deep adaptation networks [24], and generative models [42].

Multiple-source adaptation. Most practical methods assumed the availability of unlabeled target domain data and extended feature alignment methods to multiple sources, treating them as additional distribution-shifted training data (see the survey [39]). Most of them are specifically tailored to image data, whose data generating mechanism can be similarly modeled. Theoretically grounded works include [25, 38, 37, 5]. Our work is most closely related to the latter two, as we also focus on the distribution-weighted ensemble but propose a globally convergent alternative to DC programming.

Learning by playing games. Online learning algorithms are known to be connected to minimax optimization and games [4]. To find z in Eq. (2) we use Hedge, a multiplicative update algorithm, which has been applied to solve problems in theoretical computer science [14, 1]. Recently, similar ideas have been applied to learning in games [13], but our algorithm was developed independently.

3.1 Technical priors and our contributions

The authors of [25] showed the existence of z that guarantees

$$\min_{z} \max_{\lambda} \mathbb{E}_{x \sim D_\lambda} \, \ell(h_z(x), y) \leq \epsilon$$

provided each base learner in the ensemble has at most ϵ-error. By reformulating the minimax optimization problem as a DC (Difference of Convex) objective, the first work that algorithmically realized h_z was [37]. However, DC programming does not guaranty to find h_z that satisfies the above error bound. Several practical variants of domain-weighted ensembling methods were proposed for

multiple-source adaptation, which also took advantage of the available unlabeled target data [40, 12, 19]. Our main contribution is theoretical: by applying Hedge algorithm, we find h_z that achieves the minimax error guarantee.

Domain modeling As part of the construction of h_z, we need to estimate the domain densities $D_i(x)$. A particularly effective approach to model domain densities is by learning a domain classifier [5, 38, 9, 2]. For adversarial multiple-source adaptation, [38] adopted k binary classifiers to model the probability of data generated by k source domains. In [5], it was shown that the modeling of $D_i(x)$ can be replaced by modeling of $D(i|x)$, using a multi-class classifier (MaxEnt classifier), without affecting the performance of h_z. In this work, we adopted the approach of [5] but modeled $D(i|x)$ using a softmax classifier for $i \in [k]$, whose backbone has the same architecture as that of the base model h_i's.

4 Hedge for Domain-Adversarial Boosting

We now show how to apply the Hedge algorithm [1] to the multiple-source adaptation problem. Our algorithm, which we call DAB-HEDGE (Domain-Adversarial Boosting with Hedge), is presented as Algorithm 1.

Algorithm 1: DAB-HEDGE

Input: Trained classifiers $h_1, \ldots, h_k$; learning rate β; estimated loss upper bound ρ; total rounds T

Output: Ensemble $\frac{1}{T}\sum_{t=1}^{T} h_{\lambda_t}$

Initialize weight vector $\mathbf{w}_1 \in \mathbb{R}^k$ with $w_1(i) = 1$ for all i
for $t \leftarrow 1$ **to** T **do**
  Compute normalization $Z_t \leftarrow \sum_{j=1}^{k} w_t(j)$
  for $i \leftarrow 1$ **to** k **do**
    $\lambda_t(i) \leftarrow w_t(i)/Z_t$
  end
  Form ensemble classifier $h_{\lambda_t}(x) = \sum_{i=1}^{k} \frac{\lambda_t(i) D_i(x)}{\sum_j \lambda_t(j) D_j(x)} h_i(x)$
  for $i \leftarrow 1$ **to** k **do**
    $\ell_t(i) \leftarrow \mathbb{E}_{x \sim D_i}\left[\ell(h_{\lambda_t}(x), f(x))\right]$
    $w_{t+1}(i) \leftarrow w_t(i) \cdot \exp\left(\beta \frac{\ell_t(i)}{\rho}\right)$
  end
end

We make two standard assumptions.

Assumption 1 (Bounded loss): The per-domain expected losses are bounded in $[0, \rho]$, i.e., $\max_{i,z} \mathbb{E}_{x \sim D_i} \ell(h_z(x), f(x)) \leq \rho$.

Assumption 2 (Convex loss): The loss function ℓ is convex in its first argument.

These assumptions are mild. Assumption 1 can be satisfied by taking $\rho = k \max_{i,j} \mathbb{E}_{D_i} \ell(h_j(x), f(x))$. Moreover, in practice it can be estimated from observed loss sequence $\max_{i,t} \ell_t(i)$. Assumption 2 holds for many common losses (cross-entropy, squared error, hinge loss) but excludes the 0-1 loss.

Theorem 4.1. *Suppose Assumptions 1 and 2 hold. Running Algorithm 1 with* $T \geq \frac{4\rho^2 \ln k}{\delta^2}$ *and* $\beta = \sqrt{\frac{\log k}{T}} \leq \frac{1}{2}$ *yields the following guarantee: for any* λ *and* D_λ*,*

$$\mathbb{E}_{x \sim D_\lambda} \left[\ell \left(\frac{1}{T} \sum_{t=1}^{T} h_{\lambda_t}(x), f(x) \right) \right] \leq \epsilon + \delta.$$

Proof. In [25], the goal of finding a good ensemble predictor is formulated as a game between *Nature* and *Learner*. Since the hypothesis space of all possible ensemble classifiers is infinite, to apply Hedge, we switch the role of learner and nature. The learner wants to find a distribution λ over domains that maximizes the loss of an ensemble produced by nature. The nature instead wants to minimize the loss by forming a good ensemble classifier upon observing λ. Unlike the worst-case scenario, here nature has a fixed strategy: upon observing learner's distribution weighting parameter λ, it always produces the λ-parametrized ensemble h_λ (see Eq (2)), guaranteed to have small loss by previous analysis (Theorem 2 of [25]).

We now set up the domain adversarial boosting problem under the online learning framework. For any h, define a k-dimensional loss vector ℓ_h with each entry representing loss of h on domain D_i:

$$\ell_h(i) := \mathbb{E}_{D_i} \ell(h(x), f(x)) = L_{D_i}(h), \quad \forall i \in [k]$$

Then we can rewrite $L_{D_\lambda}(h)$, loss of h over a λ-mixture of sources, as below:

$$L_{D_\lambda}(h) = \sum_i \lambda_i \mathbb{E}_{D_i} \ell(h(x), f(x)) = \langle \ell_h, \lambda \rangle$$

In a multiple-round game between nature and learner, the loss of nature at the t-th round of the game is always bounded, since

$$\langle \ell_{h_t}, \lambda_t \rangle = \sum_i \lambda_t(i) \ell_{h_t}(i) = \sum_i \lambda_t(i) L_{D_i}(h_t) = L_{D_{\lambda_t}}(h_t)$$

When we set

$$h_t := h_{\lambda_t}, \quad \text{i.e., the } \lambda_t\text{-ensemble defined according to Eq (2)}$$

By Theorem 2 of [25], we get

$$\langle \ell_{h_t}, \lambda_t \rangle = L_{D_{\lambda_t}}(h_{\lambda_t}) \leq \epsilon$$

Next, we define $R(T)$ as the difference below:

$$R(T) := \max_{\lambda} \frac{1}{T} \sum_{t=1}^{T} \langle \ell_{h_t}, \lambda \rangle - \frac{1}{T} \sum_{t=1}^{T} \langle \ell_{h_t}, \lambda_t \rangle$$

If we can prove $R(T) \to 0$ as $T \to \infty$, then the ensemble $\bar{h} := \frac{1}{T} \sum_{t=1}^{T} h_{\lambda_t}$ has the following guarantee: For any λ,

$$\begin{aligned} \langle \ell_{\bar{h}}, \lambda \rangle &\leq \frac{1}{T} \sum_{t=1}^{T} \langle \ell_{h_t}, \lambda \rangle \quad \text{(convexity of loss)} \\ \leq \max_{\lambda} \frac{1}{T} \sum_{t=1}^{T} \langle \ell_{h_t}, \lambda \rangle &\leq \frac{1}{T} \sum_{t=1}^{T} \langle \ell_{h_t}, \lambda_t \rangle + R(T) \\ &\leq \epsilon + R(T) \to \epsilon \quad \text{as } T \to \infty \end{aligned}$$

Treating ℓ_{h_t} as gains, we may adapt Hedge to losses $\frac{-\ell_{h_t}}{\rho}$ as in [20] and prove convergence of $R(T)$. The Hedge algorithm, its original and extended analysis, and a complementary proof (Proposition B.2) can be found in the Appendix. □

4.1 Fixed-Point Iteration versus Hedge

We note that the Hedge update resembles the fixed-point mapping proposed for theoretical analysis in [25]. That mapping is $\phi : \Delta_k \to \Delta_k$ with

$$\phi(z)_i = \frac{z_i L_{D_i}(h_z) + \eta/k}{\sum_j z_j L_{D_j}(h_z) + \eta/k}.$$

While this mapping could be used as an iterative update, its convergence properties have not been explored. In contrast, the Hedge update in Algorithm 1 is a softmax variant that enjoys provable global convergence.

4.2 When Is Hedge Unnecessary?

In some practical cases, the uniform weighting $z = (\frac{1}{k}, \ldots, \frac{1}{k})$ already performs well. We provide a theoretical justification: when the source domains have disjoint supports, the uniform ensemble achieves the same ϵ error bound as the optimal mixture.

Proposition 4.2 (ϵ upper bound for h_u on disjoint sources)**.** *Suppose $D_i, i \in [k]$, satisfy $\mathrm{Supp}(D_i) \cap \mathrm{Supp}(D_j) = \emptyset$ for all $i \neq j$. Then for any $\lambda \in \Delta_k$,*

$$L_{D_\lambda}(h_u) \leq \epsilon.$$

Proof. For any x, there is a unique domain i such that $D_i(x) > 0$ and $D_j(x) = 0$ for $j \neq i$. Hence $h_u(x) = h_i(x)$. Consequently, $L_{D_j}(h_u) = \mathbb{E}_{D_j}\, \ell(h_j(x), f(x)) \leq \epsilon$ for every j, and the same holds for any mixture. □

5 Experiments

We evaluate our Hedge-based ensemble on two standard benchmarks: a digit dataset (four domains: SVHN, MNIST, USPS, MNIST-M) and the Amazon review dataset (four domains: Books, DVD, Electronics, Kitchen). For digits, we use a CNN architecture similar to [38]. For Amazon reviews, we use BERT [6] as the base model. We compare against simple averaging of base model outputs, h_z with a random z, h_z with z being equally weighted (uniform z-ens), h_z found by DC programming, and h_z found by Hedge (hedge z-ens using the last iterate, and hedge agg-z-ens using the average of iterates).

5.1 Digits Dataset

We first train a source-specific model on each domain. Table 1 reports the accuracy and cross-entropy loss of these models on each test domain. Then we evaluate the ensemble methods on four random mixtures and four adversarial mixtures (obtained from the Hedge iterates). Table 2 summarizes the results. Our Hedge-based ensembles consistently achieve the highest accuracy and lowest loss, outperforming DC programming and all baselines.

Table 1: Accuracy and cross-entropy loss of domain-specific models on each test domain (digits). Each cell shows `accuracy (loss)`.

Model \ Dataset	svhn	mnist	usps	mnistm
svhn	0.9435 (0.286)	0.7363 (1.287)	0.7464 (1.449)	0.5083 (2.694)
mnist	0.4285 (6.810)	0.9947 (0.022)	0.8685 (1.118)	0.6049 (4.307)
usps	0.3498 (7.760)	0.6185 (2.684)	0.9671 (0.275)	0.2653 (8.429)
mnistm	0.5700 (3.180)	0.9817 (0.081)	0.6562 (2.460)	0.9330 (0.297)

5.2 Amazon Review Dataset

We use the same four domains as [25]. Each domain has 2000 labeled samples; we randomly select 20% for testing and use the rest for training and validation. BERT is fine-tuned on each source domain. Table 3 shows the performance of the source-specific BERT models. Table 4 compares our Hedge ensembles with the UDALM model [17], which is a strong single-source adaptation baseline. Our method achieves competitive or superior results, even without using unlabeled target data during meta-training.

Table 2: Test error (accuracy and cross-entropy loss) for different methods on digits mixtures.

Model \	Type	Mixture over sources			
		Mixture 1	Mixture 2	Mixture 3	Mixture 4
svhn	random	0.7446 (0.0899)	0.7474 (0.0828)	0.7544 (0.0838)	0.7279 (0.0884)
	adversarial	0.7554 (0.0834)	0.7801 (0.0751)	0.7953 (0.0697)	0.8197 (0.0609)
mnist	random	0.8361 (0.0969)	0.6964 (0.2174)	0.8277 (0.1084)	0.8203 (0.1242)
	adversarial	0.5762 (0.3031)	0.5077 (0.3591)	0.4888 (0.3708)	0.4748 (0.3843)
usps	random	0.8683 (0.0828)	0.4768 (0.3473)	0.8176 (0.1127)	0.5712 (0.2566)
	adversarial	0.3959 (0.4324)	0.3368 (0.4867)	0.3286 (0.4936)	0.3250 (0.4962)
mnistm	random	0.6866 (0.1392)	0.7989 (0.0885)	0.7018 (0.1310)	0.8606 (0.0594)
	adversarial	0.7402 (0.1134)	0.7087 (0.1292)	0.6942 (0.1375)	0.6746 (0.1477)
simple ens	random	0.9002 (0.3636)	0.8551 (0.5490)	0.9001 (0.3725)	0.9088 (0.4151)
	adversarial	0.8064 (0.6835)	0.7770 (0.7604)	0.7700 (0.7674)	0.7639 (0.7794)
random z-ens	random	0.9631 (0.2541)	0.9437 (0.2679)	0.9640 (0.2414)	0.9656 (0.1736)
	adversarial	0.9307 (0.3333)	0.9167 (0.3928)	0.9178 (0.3848)	0.9217 (0.3714)
uniform z-ens	random	0.9632 (0.2552)	0.9457 (0.2570)	0.9642 (0.2408)	0.9648 (0.1768)
	adversarial	0.9298 (0.3347)	0.9196 (0.3760)	0.9186 (0.3806)	0.9183 (0.3885)
dc z-ens	random	0.9643 (0.2470)	0.9513 (0.2348)	0.9659 (0.2314)	0.9672 (0.1650)
	adversarial	0.9372 (0.3022)	0.9288 (0.3385)	0.9278 (0.3412)	0.9287 (0.3471)
hedge z-ens	random	0.9651 (0.2435)	0.9553 (0.2182)	0.9669 (0.2279)	0.9699 (0.1534)
	adversarial	0.9430 (0.2780)	0.9352 (0.3131)	0.9343 (0.3168)	0.9340 (0.3241)
hedge agg-z-ens	random	0.9645 (0.2468)	0.9525 (0.2267)	0.9661 (0.2313)	0.9679 (0.1588)
	adversarial	0.9389 (0.2906)	0.9303 (0.3264)	0.9293 (0.3301)	0.9297 (0.3376)

Table 3: Accuracy and cross-entropy loss of BERT models on Amazon test domains. Each cell shows `accuracy (loss)`.

Model \ Domain	books	dvd	electronics	kitchen
books	0.9525 (0.1178)	0.9025 (0.2749)	0.8875 (0.3231)	0.8750 (0.3196)
dvd	0.9275 (0.2026)	0.9525 (0.1617)	0.9275 (0.2306)	0.8900 (0.2944)
electronics	0.8975 (0.3289)	0.8700 (0.4428)	0.9650 (0.0802)	0.9050 (0.2664)
kitchen	0.9150 (0.2286)	0.8750 (0.3097)	0.9350 (0.1812)	0.9525 (0.1438)

Table 4: Test accuracy (and cross-entropy loss) on Amazon OOD evaluation.

Method	→ books	→ dvd	→ electronics	→ kitchen
UDALM-worst [17]	90.29	89.54	91.69	93.21
UDALM-best [17]	91.00	90.97	94.34	94.43
simple ens	0.9175 (0.2027)	0.8950 (0.2801)	0.9150 (0.1940)	0.9100 (0.2238)
dc z-ens	0.9300 (0.1917)	0.9000 (0.2736)	0.9375 (0.1785)	0.9050 (0.2415)
hedge z-ens	0.9275 (0.1977)	0.9000 (0.2755)	0.9375 (0.1729)	0.9075 (0.2415)
hedge agg-z-ens	0.9275 (0.1971)	0.9000 (0.2746)	0.9375 (0.1729)	0.9075 (0.2448)

References

[1] Arora, S., Hazan, E., and Kale, S. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing 8*, 6 (2012), 121–164.

[2] Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., and Vaughan, J. W. A theory of learning from different domains. *Mach. Learn. 79*, 1–2 (May 2010), 151–175.

[3] Blanchard, G., Lee, G., and Scott, C. Generalizing from several related classification tasks to a new unlabeled sample. In *Advances in Neural Information Processing Systems* (2011), J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., vol. 24, Curran Associates, Inc.

[4] Cesa-Bianchi, N., and Lugosi, G. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[5] Cortes, C., Mohri, M., Suresh, A. T., and Zhang, N. A discriminative technique for multiple-source adaptation. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event* (2021), M. Meila and T. Zhang, Eds., vol. 139 of *Proceedings of Machine Learning Research*, PMLR, pp. 2132–2143.

[6] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (Minneapolis, Minnesota, June 2019), J. Burstein, C. Doran, and T. Solorio, Eds., Association for Computational Linguistics, pp. 4171–4186.

[7] Duchi, J. C., Hashimoto, T. B., and Namkoong, H. Distributionally robust losses for latent covariate mixtures. *Oper. Res. 71* (2020), 649–664.

[8] Freund, Y., and Schapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences 55*, 1 (1997), 119–139.

[9] Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. Domain-adversarial training of neural networks. *J. Mach. Learn. Res. 17*, 1 (Jan. 2016), 2096–2030.

[10] Goyal, S., Sun, M., Raghunathan, A., and Kolter, J. Z. Test time adaptation via conjugate pseudo-labels. In *Advances in Neural Information Processing Systems* (2022), S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35, Curran Associates, Inc., pp. 6204–6218.

[11] Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *Journal of Machine Learning Research 13*, 25 (2012), 723–773.

[12] Guo, J., Shah, D., and Barzilay, R. Multi-source domain adaptation with mixture of experts. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* (Brussels, Belgium, Oct.-Nov. 2018), E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, Eds., Association for Computational Linguistics, pp. 4694–4703.

[13] Gupta, Y., Zhai, R., Suggala, A., and Ravikumar, P. Responsible ai (rai) games and ensembles. In *Advances in Neural Information Processing Systems* (2023), A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., vol. 36, Curran Associates, Inc., pp. 72717–72749.

[14] Hardt, M., and Rothblum, G. N. A multiplicative weights mechanism for privacy-preserving data analysis. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science* (2010), pp. 61–70.

[15] Hoffman, J., Tzeng, E., Park, T., Zhu, J.-Y., Isola, P., Saenko, K., Efros, A., and Darrell, T. CyCADA: Cycle-consistent adversarial domain adaptation. In *Proceedings of the 35th International Conference on Machine Learning* (10–15 Jul 2018), J. Dy and A. Krause, Eds., vol. 80 of *Proceedings of Machine Learning Research*, PMLR, pp. 1989–1998.

[16] Huang, J., Smola, A. J., Gretton, A., Borgwardt, K. M., and Scholkopf, B. Correcting sample selection bias by unlabeled data. In *Proceedings of the 20th International Conference on Neural Information Processing Systems* (Cambridge, MA, USA, 2006), NIPS'06, MIT Press, p. 601–608.

[17] Karouzos, C., Paraskevopoulos, G., and Potamianos, A. UDALM: Unsupervised domain adaptation through language modeling. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Online, June 2021), K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tur, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, and Y. Zhou, Eds., Association for Computational Linguistics, pp. 2579–2590.

[18] Khaled, A., Mishchenko, K., and Richtarik, P. Tighter theory for local sgd on identical and heterogeneous data. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics* (26–28 Aug 2020), S. Chiappa and R. Calandra, Eds., vol. 108 of *Proceedings of Machine Learning Research*, PMLR, pp. 4519–4529.

[19] Kim, Y.-B., Stratos, K., and Kim, D. Domain attention with an ensemble of experts. In *Proceedings of the 55th Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers) (Vancouver, Canada, July 2017), R. Barzilay and M.-Y. Kan, Eds., Association for Computational Linguistics, pp. 643–653.

[20] Li, J. Online learning: Experts and bandits. Lecture notes, Course 15-850, Carnegie Mellon University.

[21] Lin, T., Kong, L., Stich, S. U., and Jaggi, M. Ensemble distillation for robust model fusion in federated learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Red Hook, NY, USA, 2020), NIPS '20, Curran Associates Inc.

[22] Littlestone, N., and Warmuth, M. The weighted majority algorithm. *Information and Computation 108*, 2 (1994), 212–261.

[23] Livni, R. Lecture 18: Expert advice (hedge algorithm) & online to batch. Lecture notes for COS-511: Learning Theory, Spring 2017.

[24] Long, M., Cao, Y., Wang, J., and Jordan, M. Learning transferable features with deep adaptation networks. In *Proceedings of the 32nd International Conference on Machine Learning* (Lille, France, 07–09 Jul 2015), F. Bach and D. Blei, Eds., vol. 37 of *Proceedings of Machine Learning Research*, PMLR, pp. 97–105.

[25] Mansour, Y., Mohri, M., and Rostamizadeh, A. Domain adaptation with multiple sources. In *Advances in Neural Information Processing Systems* (2008), D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds., vol. 21, Curran Associates, Inc.

[26] Mohri, M., Sivek, G., and Suresh, A. T. Agnostic federated learning. In *Proceedings of the 36th International Conference on Machine Learning* (09–15 Jun 2019), K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97 of *Proceedings of Machine Learning Research*, PMLR, pp. 4615–4625.

[27] Muandet, K., Balduzzi, D., and Schölkopf, B. Domain generalization via invariant feature representation. In *Proceedings of the 30th International Conference on Machine Learning* (Atlanta, Georgia, USA, 17–19 Jun 2013), S. Dasgupta and D. McAllester, Eds., vol. 28 of *Proceedings of Machine Learning Research*, PMLR, pp. 10–18.

[28] Pan, S. J., and Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering 22*, 10 (2010), 1345–1359.

[29] Shimodaira, H. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference 90*, 2 (2000), 227–244.

[30] Sugiyama, M., Krauledat, M., and Müller, K.-R. Covariate shift adaptation by importance weighted cross validation. *J. Mach. Learn. Res. 8* (Dec. 2007), 985–1005.

[31] Sugiyama, M., Suzuki, T., and Kanamori, T. Density ratio estimation : A comprehensive review (statistical experiment and its related topics).

[32] 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 *Proceedings of the 37th International Conference on Machine Learning* (13–18 Jul 2020), H. D. III and A. Singh, Eds., vol. 119 of *Proceedings of Machine Learning Research*, PMLR, pp. 9229–9248.

[33] Wang, D., Shelhamer, E., Liu, S., Olshausen, B., and Darrell, T. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations* (2021).

[34] Wang, J., Lan, C., Liu, C., Ouyang, Y., Qin, T., Lu, W., Chen, Y., Zeng, W., and Yu, P. S. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering 35*, 8 (2023), 8052–8072.

[35] Woodworth, B., Patel, K. K., and Srebro, N. Minibatch vs local sgd for heterogeneous distributed learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Red Hook, NY, USA, 2020), NIPS '20, Curran Associates Inc.

[36] Zhang, K., Gong, M., and Scholkopf, B. Multi-source domain adaptation: a causal view. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence* (2015), AAAI'15, AAAI Press, p. 3150–3157.

[37] Zhang, N., Mohri, M., and Hoffman, J. Multiple-source adaptation theory and algorithms. *Ann. Math. Artif. Intell. 89*, 3-4 (2021), 237–270.

[38] Zhao, H., Zhang, S., Wu, G., Moura, J. M. F., Costeira, J. P., and Gordon, G. J. Adversarial multiple source domain adaptation. In *Advances in Neural Information Processing Systems* (2018), S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31, Curran Associates, Inc.

[39] Zhao, S., Chen, H., Huang, H., Xu, P., and Ding, G. More is better: Deep domain adaptation with multiple sources. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24* (8 2024), K. Larson, Ed., International Joint Conferences on Artificial Intelligence Organization, pp. 8354–8362. Survey Track.

[40] Zhao, S., Wang, G., Zhang, S., Gu, Y., Li, Y., Song, Z., Xu, P., Hu, R., Chai, H., and Keutzer, K. Multi-source distilling domain adaptation, 2020.

[41] Zhou, K., Liu, Z., Qiao, Y., Xiang, T., and Loy, C. C. Domain generalization: A survey. *IEEE Trans. Pattern Anal. Mach. Intell. 45*, 4 (Apr. 2023), 4396–4415.

[42] Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), pp. 2242–2251.

A Hedge regret bound for nonnegative loss sequences

In this section, we provide background of the Hedge algorithm [1], as described in Algorithm 2.

Algorithm 2: Hedge Algorithm (Multiplicative Weights)

Input: Learning rate $\epsilon > 0$
Number of experts n
Time horizon T
Output: Sequence of actions $\{i_t\}_{t=1}^{T}$

Initialize weight vector $\mathbf{W}_1 \leftarrow (1, 1, \ldots, 1) \in \mathbb{R}^n$
Set probability distribution $\mathbf{x}_1 \leftarrow \frac{1}{n}\mathbf{W}_1$ `// Uniform initialization`
for $t \leftarrow 1$ **to** T **do**
  Choose action i_t with probability $\mathbb{P}(i_t = i) = x_t(i)$
  Observe loss vector $\mathbf{g}_t = (g_t(1), \ldots, g_t(n))$
  Update weights: $\mathbf{W}_{t+1} \leftarrow \mathbf{W}_t \odot \exp(-\epsilon \mathbf{g}_t)$ `// Element-wise multiplication`
  Normalize distribution: $\mathbf{x}_{t+1} \leftarrow \frac{\mathbf{W}_{t+1}}{\|\mathbf{W}_{t+1}\|_1}$ `//` $\|\cdot\|_1$ `is L1-norm`
end

We first review the analysis of regret bounds for the Hedge algorithm when the losses are nonnegative, based on lecture note [23].

Lemma A.1 (18.1)**.** *Let $\mathbf{g}_t^2$ denote the n-dimensional vector of pointwise square losses (i.e., $\mathbf{g}_t^2(i) = (g_t(i))^2$), let $\epsilon > 0$, and assume all losses $g_t(i)$ are nonnegative. The Hedge Algorithm satisfies for every expert i^*:*

$$\sum_{t=1}^{T}(\mathbf{x}_t \cdot \mathbf{g}_t) \leq \sum_{t=1}^{T} g_t(i^*) + \epsilon \sum_{t=1}^{T}(\mathbf{x}_t \cdot \mathbf{g}_t^2) + \frac{\log n}{\epsilon} \tag{18.1}$$

Proof. Define $\Phi_t = \sum_{i=1}^{n} W_t(i)$ as the total weight at the **beginning** of round t, with $\Phi_1 = n$ (uniform initialization). The weight update rule is:

$$W_{t+1}(i) = W_t(i)\exp(-\epsilon g_t(i))$$

The total weight at the beginning of round $t+1$ is:

$$\begin{aligned}
\Phi_{t+1} &= \sum_{i=1}^{n} W_{t+1}(i) \\
&= \sum_{i=1}^{n} W_t(i)\exp(-\epsilon g_t(i)) \\
&= \Phi_t \sum_{i=1}^{n} \underbrace{\frac{W_t(i)}{\Phi_t}}_{\mathbf{x}_t(i)} \exp(-\epsilon g_t(i)) \\
&= \Phi_t(\mathbf{x}_t \cdot \exp(-\epsilon \mathbf{g}_t))
\end{aligned}$$

Using $e^{-x} \leq 1 - x + x^2$ for $x \geq 0$:

$$\begin{aligned}\Phi_{t+1} &\leq \Phi_t \sum_{i=1}^{n} \mathbf{x}_t(i)(1 - \epsilon g_t(i) + \epsilon^2 g_t(i)^2) \\ &= \Phi_t \left(1 - \epsilon(\mathbf{x}_t \cdot \mathbf{g}_t) + \epsilon^2(\mathbf{x}_t \cdot \mathbf{g}_t^2)\right) \\ &\leq \Phi_t \exp\left(-\epsilon(\mathbf{x}_t \cdot \mathbf{g}_t) + \epsilon^2(\mathbf{x}_t \cdot \mathbf{g}_t^2)\right)\end{aligned}$$

Unfolding this recursion from $t = 1$ to $t = T$:

$$\Phi_{T+1} \leq n \exp\left(-\epsilon \sum_{t=1}^{T}(\mathbf{x}_t \cdot \mathbf{g}_t) + \epsilon^2 \sum_{t=1}^{T}(\mathbf{x}_t \cdot \mathbf{g}_t^2)\right)$$

For expert i^*:

$$W_{T+1}(i^*) = W_1(i^*) \exp\left(-\epsilon \sum_{t=1}^{T} g_t(i^*)\right) = \exp\left(-\epsilon \sum_{t=1}^{T} g_t(i^*)\right)$$

since $W_1(i^*) = 1$. The key inequality holds because:

$$W_{T+1}(i^*) \leq \sum_{i=1}^{n} W_{T+1}(i) = \Phi_{T+1}$$

as the weight of a single expert cannot exceed the total weight. Thus:

$$\exp\left(-\epsilon \sum_{t=1}^{T} g_t(i^*)\right) \leq n \exp\left(-\epsilon \sum_{t=1}^{T}(\mathbf{x}_t \cdot \mathbf{g}_t) + \epsilon^2 \sum_{t=1}^{T}(\mathbf{x}_t \cdot \mathbf{g}_t^2)\right)$$

Taking logs and rearranging proves the lemma. □

Theorem A.2 (18.2)**.** *Apply the Hedge algorithm to the Online Expert problem with* $\epsilon = \sqrt{\frac{\log n}{T}}$*. Then the regret satisfies:*

$$\text{Regret}_T = O\left(\sqrt{T \log n}\right)$$

Proof. Assuming losses $g_t(i) \in [0, 1]$, we have $g_t(i)^2 \leq g_t(i) \leq 1$, so:

$$\mathbf{x}_t \cdot \mathbf{g}_t^2 \leq \mathbf{x}_t \cdot \mathbf{1} = 1$$

Applying Lemma 18.1:

$$\begin{aligned}\sum_{t=1}^{T}(\mathbf{x}_t \cdot \mathbf{g}_t) &\leq \sum_{t=1}^{T} g_t(i^*) + \epsilon T + \frac{\log n}{\epsilon} \\ \text{Regret}_T(i^*) &\leq \epsilon T + \frac{\log n}{\epsilon}\end{aligned}$$

Substituting $\epsilon = \sqrt{\frac{\log n}{T}}$:

$$\begin{aligned}\text{Regret}_T(i^*) &\le \sqrt{\frac{\log n}{T}} \cdot T + \frac{\log n}{\sqrt{\frac{\log n}{T}}} \\ &= \sqrt{T \log n} + \sqrt{T \log n} \\ &= 2\sqrt{T \log n}\end{aligned}$$

Since this holds for all i^*, we get $\text{Regret}_T = O(\sqrt{T \log n})$. □

B Hedge regret bounds for symmetric loss sequences [20]

The lecture note of [20] extended the analysis of Hedge to the cases when the losses are symmetric around zero and bounded between $[-1, 1]$.

Theorem B.1 (see Theorem 14.7, [20])**.** *Consider any fixed $\epsilon \in (0, \frac{1}{2}]$. For any sequences of loss vectors $[-1, 1]^n$ and all indices $i \in n$, the Hedge algorithm guarantees:*

$$\sum_{t=1}^{T} \langle p^t, \ell^t \rangle \le \sum_{t=1}^{T} \ell_i^t + \epsilon T + \frac{\ln n}{\epsilon}$$

Proof. As in previous proofs, let $\Phi^t = \sum_j w_j^t$ and so $\Phi^1 = n$, and

$$\begin{aligned}\Phi^{t+1} &= \sum_i w_i^{t+1} = \sum_i w_i^t e^{-\epsilon \ell_i^t} \\ &\le \sum_i w_i^t (1 - \epsilon \ell_i^t + \epsilon^2 (\ell_i^t)^2) \quad (e^x \le 1 + x + x^2 \ \forall x \in [-1, 1]) \\ &\le \sum_i w_i^t (1 + \epsilon^2) - \epsilon \sum_i w_i^t \ell_i^t \quad (\text{since } |\ell_i^t| \le 1) \\ &= (1 + \epsilon^2)\Phi^t - \epsilon \Phi^t \langle p^t, \ell^t \rangle \quad (\text{since } w_i^t = p_i^t \Phi^t) \\ &= \Phi^t (1 + \epsilon^2 - \epsilon \langle p^t, \ell^t \rangle) \\ &\le \Phi^t e^{\epsilon^2 - \epsilon \langle p^t, \ell^t \rangle} \quad (\text{using } 1 + x \le e^x)\end{aligned}$$

Comparing to the final weight of the i-th coordinate,

$$w_i^{T+1} = e^{-\epsilon \sum_t \ell_i^t} \le \Phi^{T+1} = \Phi^1 e^{\epsilon^2 T - \epsilon \sum_t \langle p^t, \ell_i^t \rangle}$$

Taking logs and divide by ϵ proves the theorem. □

This implies again when choosing learning rate ϵ in Algorithm 2 to be $\epsilon = \sqrt{\frac{\ln n}{T}}$, the average regret is upper bounded by $2\sqrt{\frac{\ln n}{T}}$. The lecture note in [20] also introduces some useful corollaries.

Corollary B.1.1 (Corolloary 14.8 [20]). *For $T \geq \frac{4 \log n}{\epsilon^2}$, the average loss of the Hedge algorithm is*

$$\frac{1}{T}\sum_t \langle p^t, \ell^t \rangle \leq \min_i \frac{1}{T}\sum_t \ell_i^t + \epsilon = \min_{p \in \Delta_n} \frac{1}{T}\sum_t \langle p, \ell^t \rangle + \epsilon$$

Below is a corollary for the application of symmetric Hedge to maximize gains $g^t \in [-\rho, \rho]$. Applying the corollary above with losses $\ell^t = \frac{-g^t}{\rho}$ yields a corollary below.

Corollary B.1.2 (Corollary 14.9 [20], Average Gain). *Let $\rho \geq 1$ and $\varepsilon \in (0, 1/2)$. For any sequence of gain vectors $g_1, \ldots, g_T \in [-\rho, \rho]^n$ with*

$$T \;\geq\; \frac{4\rho^2 \ln n}{\varepsilon^2},$$

the gains version of the Hedge algorithm produces probability vectors $p_t \in \Delta_n$ such that

$$\frac{1}{T}\sum_{t=1}^{T} \langle g_t, p_t \rangle \;\geq\; \max_{i \in [n]} \frac{1}{T}\sum_{t=1}^{T} \langle g_t, e_i \rangle \;-\; \varepsilon.$$

To complement the proof of Theorem 4.1, we state and prove the following result.

Proposition B.2. *Let $\rho \geq 1$. For any sequence of gain vectors $g_1, \ldots, g_T \in [-\rho, \rho]^k$ with*

$$T \geq \frac{4\rho^2 \ln k}{\delta^2}$$

Choose $\beta = \sqrt{\frac{\ln k}{T}}$. The gains version of the Hedge algorithm produces probability vectors $p_t \in \Delta_k$ such that

$$\frac{1}{T}\sum_{t=1}^{T} \langle g_t, p_t \rangle \geq \max_{p \in \Delta_k} \frac{1}{T}\sum_{t=1}^{T} \langle g_t, p \rangle - \delta$$

Proof. By Theorem B.1, for $\ell^t \in [-1, 1]^k$ and for all $i \in [k]$,

$$\sum_{t=1}^{T} -\ell_i^t - \sum_{t=1}^{T} \langle p^t, -\ell^t \rangle \leq \beta T + \frac{\ln k}{\beta}$$

Substitute ℓ^t with $-\frac{g_t}{\rho}$, we get

$$\sum_t \frac{g_i^t}{\rho} - \sum_{t=1}^{T} \langle p^t, \frac{g^t}{\rho} \rangle \leq \beta T + \frac{\ln k}{\beta} = 2\sqrt{T \ln k}$$

And $\forall i \in [k]$

$$\frac{1}{T}\sum_{t=1}^{T} g_i^t - \frac{1}{T}\sum_{t=1}^{T} \langle p^t, g^t \rangle \leq 2\rho\sqrt{\frac{\ln k}{T}} \leq 2\rho\frac{\delta}{2\rho} = \delta$$

Since $\max_{p \in \Delta_k} \frac{1}{T}\sum_{t=1}^{T} \langle g^t, p \rangle = \max_i \frac{1}{T}\sum_{t=1}^{T} g_i^t$, the proof is finished. □