Unsupervised Learning under Latent Label Shift

Anonymous Authors[1]

Abstract

What sorts of structure might enable a learner to discover classes from unlabeled data? Traditional unsupervised learning approaches risk recovering incorrect classes based on spurious data-space similarity. In this paper, we introduce unsupervised learning under *Latent Label Shift* (LLS), where the label marginals $p_d(y)$ shift but the class conditionals $p(\mathbf{x}|y)$ do not. This setting suggests a new principle for identifying classes: elements that shift together across domains belong to the same true class. For finite input spaces, we establish an isomorphism between LLS and topic modeling; for continuous data, we show that if each label's support contains a separable region, analogous to an anchor word, oracle access to $p(d|\mathbf{x})$ suffices to identify $p_d(y)$ and $p_d(y|\mathbf{x})$ up to permutation. Thus motivated, we introduce a practical algorithm that leverages domain-discriminative models as follows: (i) push examples through domain discriminator $p(d|\mathbf{x})$; (ii) discretize the data by clustering examples in $p(d|\mathbf{x})$ space; (iii) perform non-negative matrix factorization on the discrete data; (iv) combine recovered $p(y|d)$ with discriminator outputs $p(d|\mathbf{x})$ to compute $p_d(y|\mathbf{x})$ $\forall d$. In semi-synthetic experiments, we show that our algorithm can use domain information to overcome a failure mode of standard unsupervised classification in which data-space similarity does not indicate true groupings.

1. Introduction

Discovering systems of categories from unlabeled data is a fundamental but ill-posed challenge in machine learning. Typical unsupervised learning methods group instances together based on feature-space similarity. Accordingly, given a collection of photographs of animals, a practitioner might hope that, in some feature space, images of animals of the same species should be somehow similar to each other. But why should we expect a clustering algorithm to recognize that dogs viewed in sunlight and dogs viewed at night belong to the same category? Why should we expect that butterflies and caterpillars lie close together in feature space?

In this paper, we offer an alternative principle according to which we might identify a set of classes: we exploit distribution shift across times and locations to reveal otherwise unrecognizable groupings among examples. For example, if we noticed that whenever we found ourselves in a location where butterflies are abundant, caterpillars were similarly abundant, and that whenever butterflies were scarce, caterpillars had a similar drop in prevalence, we might conclude that the two were tied to the same underlying concept, no matter how different they appear in feature space. In short, our principle suggests that latent classes might be uncovered whenever *instances that shift together group together*.

Formalizing this intuition, we introduce the problem of unsupervised learning under *Latent Label Shift* (LLS). Here, we assume access to a collection of domains $d \in \{1, \ldots, r\}$, where the mixture proportions $p_d(y)$ vary across domains but the class conditional distribution $p(x|y)$ is domain-invariant. Our goals are to recover the underlying classes up to permutation, and thus to identify both the per-domain mixture proportions $p_d(y)$ and optimally adapted per-domain classifiers $p_d(y|x)$. We prove that under mild assumptions, knowledge of this underlying structure is sufficient for inducing the full set of categories.

First, we focus on the *tabular setting*, demonstrating that when the input space is discrete and finite, LLS is isomorphic to topic modeling (Blei et al., 2003). In this case, we can apply standard identification results for topic modeling (Donoho & Stodden, 2003; Arora et al., 2012b; Gillis & Vavasis, 2014; Huang et al., 2016; Chen et al., 2021) that rely only on the existence of anchor words within each topic (i.e., for each label y_i there is at least one x in the support of y_i, that is not in the support of any $y_j \neq y_i$). Here, standard methods based on Non-negative Matrix Factorization (NMF) can recover each domain's underlying mixture proportion $p_d(y)$ and optimal predictor $p_d(y|x)$. However, the restriction to discrete inputs, while appropriate for topic modeling, proves restrictive when our interests extend to high-dimensional continuous input spaces.

Then, to handle high-dimensional inputs, we propose

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the SCIS workshop.

Discriminate-Discretize-Factorize-Adjust (DDFA), a general framework that proceeds in the following steps: (i) pool data from all domains to produce a mixture distribution $q(x, d)$; (ii) train a domain discriminative model f to predict $q(d|x)$; (iii) push all data through f, cluster examples in the pushforward distribution, and tabularize the data based on cluster membership; (iv) solve the resulting discrete topic modeling problem (e.g., via NMF), estimating $q(y, d)$ up to permutation of the latent labels; (v) combine the predicted $q(d|x)$ and $q(y, d)$ to estimate $p_d(y)$ and $p_d(y|x)$. In developing this approach, we draw inspiration from recent works on distribution shift and learning from positive and unlabeled data that (i) leverage black box predictors to perform dimensionality reduction (Lipton et al., 2018; Garg et al., 2020; 2021); and (ii) work with *anchor sets*, separable subsets of continuous input spaces that belong to only one class's support (Scott, 2015; Liu & Tao, 2016; du Plessis et al., 2016)

Our key theoretical result shows that domain discrimination ($q(d|x)$) provides a sufficient representation for identifying parameters of interest. Given oracle access to $q(d|x)$ (which is identified without labels), our procedure is consistent. Our analysis reveals that the true $q(d|x)$ maps all points in the same anchor set to a single point mass in the push-forward distribution. This motivates our approach of discretizing data by hunting for tight clusters in $\widehat{q}(d|x)$ space.

In semi-synthetic experiments, we adapt existing image classification benchmarks to the LLS setting. We note that training a domain discriminative classifier is a difficult task, and find that warm starting the initial layers of our model with pretrained weights from unsupervised approaches can significantly boost performance. We show that warm-started DDFA outperforms state-of-the-art (SOTA) unsupervised approaches when domain marginals $p_d(y)$ are sufficiently sparse. In particular, we observe improvements of as much as 30% accuracy over unsupervised SOTA on CIFAR-20. Further, on subsets of FieldGuide dataset, where similarity between species and diversity within a species leads to failure of unsupervised learning, we show that DDFA recovers the true distinctions. To be clear, these are not apples-to-apples comparisons: our methods are specifically tailored to the LLS setting. The takeaway is that the structure of the LLS setting can be exploited to outperform the best unsupervised learning heuristics, in particular when these heuristics detect classes by exploiting spurious similarities.

2. Problem Formulation

For a vector $v \in \mathbb{R}^p$, we use v_j to denote its jth entry. For an event E, we let $\mathbb{I}[E]$ denote the binary indicator of the event. We use $[A]_{i,j}$ to access the element at (i, j) in A. Let $\mathcal{X}$ be the input space and $\mathcal{Y} = \{1, 2, \ldots, k\}$ the output space for classification. We use capital letters (e.g., X) to denote random variables and small case letters (e.g., x) to denote the values they take. We now introduce the problem of unsupervised learning under LLS. We assume that we observe unlabeled data from $\mathcal{R} = \{1, 2, \ldots, r\}$ domains. By p_d, we denote the probability density (or mass) function for each domain $d \in \mathcal{R}$.

Definition 2.1 (Latent label shift)**.** We observe data from r domains. While the label distribution among these domains can change, for all $d, d' \in \mathcal{R}$ and for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$, we have $p_d(x|y) = p_{d'}(x|y) = p(x|y)$.

Simply put, Definition 2.1 states that the conditional distribution $p_d(x|y)$ remains invariant across domains, i.e., they satisfy the label shift assumption. Crucially, under LLS, $p_d(y)$ can vary across domains, and we observe unlabeled data with domain label $\{(x_1, d_1), (x_2, d_2), \ldots, (x_n, d_n)\}$. We now aim to (i) estimate the label marginal in each domain $p_d(y)$; and (ii) estimate the optimal per-domain predictor $p_d(y|x)$, upto some permutation of labels.

Mixing distribution Q A key step in our algorithm will be to train a domain discriminative model. Towards this end we define Q, a distribution over $\mathcal{X} \times \mathcal{Y} \times \mathcal{R}$, constructed by taking a uniform mixture over all domains. By q, we denote the probability density (or mass) function of Q. Define Q such that $q(x, y|D = d) = p_d(x, y)$. For all $d \in \mathcal{R}$, we define $\gamma_d = q(d)$, i.e., the prevalence of each domain in our distribution Q. Notice that $q(x, y)$ is a mixture over the distributions $\{p_d(x, y)\}_{d \in \mathcal{R}}$, with mixture weights $\{\gamma_d\}_{d \in \mathcal{R}}$.

Notation for the discrete case To begin, we setup notation for discrete input spaces. Without loss of generality, we assume that $\mathcal{X} = \{1, 2, \ldots, m\}$. The label shift assumption allows us to formulate the label marginal estimation problem in matrix form. Let $\mathbf{Q}_{X|D}$ be an $m \times r$ matrix such that $[\mathbf{Q}_{X|D}]_{i,d} = p_d(X = i)$, i.e., the d-th column of $\mathbf{Q}_{X|D}$ is $p_d(x)$. Let $\mathbf{Q}_{X|Y}$ be an $m \times k$ matrix such that $[\mathbf{Q}_{X|Y}]_{i,j} = p(X = i|Y = j)$, the j-th column is a distribution over X given $Y = j$. Similarly, define $\mathbf{Q}_{Y|D}$ as a $k \times r$ matrix whose d-th column is the marginal $p_d(y)$. With Definition 2.1, we have $p_d(x) = \sum_y p_d(x, y) = \sum_y p_d(x|y)p_d(y) = \sum_y p(x|y)p_d(y)$. Since this is true $\forall d \in \mathcal{R}$, we have the matrix form as $\mathbf{Q}_{X|D} = \mathbf{Q}_{X|Y}\mathbf{Q}_{Y|D}$.

Assumptions We introduce four additional assumptions:

A.1 As many domains as classes, i.e., $|\mathcal{R}| \geqslant |\mathcal{Y}|$.

A.2 Matrix formed by label marginals (as columns) across domains is full-rank, i.e., $\text{rank}(\mathbf{Q}_{Y|D}) = k$.

A.3 Equal representation of domains, i.e., $\gamma_d = 1/r$.

A.4 Fix $\epsilon > 0$. For all $y \in \mathcal{Y}$, there exists a subdomain $A_y \subseteq \mathcal{X}$, such that $q(A_y) \geqslant \epsilon$ with $q(A_y|y) > 0$ and for all $y' \in \mathcal{Y} \backslash \{y\}$, $q(A_y|y') = 0$. For any $x \in \mathcal{X}$, if $q(x|y) > 0$ and $q(x|y') = 0$ for all $y' \in \mathcal{Y} \backslash \{y\}$, then $x \in A_y$. This is the ϵ-anchor sub-domain condition.

A.1–A.2 are benign, these assumptions just imply that the

matrix $\mathbf{Q}_{Y|D}$ is full row rank. Without loss of generality, A.3 can be assumed when dealing with data from a collection of domains. Intuitively, A.4 states that for each label $y \in \mathcal{Y}$, we have some subset of inputs that only belong to that class y. The anchor word condition is related to the positive sub-domain in PU learning, which requires that there exists a subset of $\mathcal{X}$ in which all examples only belong to the positive class (Scott, 2015; Liu & Tao, 2016; du Plessis et al., 2016).

3. Theoretical Analysis

In this section, we establish identifiability of LLS problem. We begin by considering the case where the input space is discrete and formalize the isomorphism to topic modeling. Then we establish the identifiability of the system in this discrete setting by appealing to existing results in topic modeling (Huang et al., 2016). Finally, extending results from discrete case, we provide novel analysis to establish our identifiability result for the continuous setting.

Isomorphism to topic modeling Recall that for the discrete input setting, we have the matrix formulation: $\mathbf{Q}_{X|D} = \mathbf{Q}_{X|Y}\mathbf{Q}_{Y|D}$. Consider a corpus of r documents, consisting of terms from a vocabulary of size m. Let $\mathbf{D}$ be an $\mathbb{R}^{m\times r}$ matrix such that $[\mathbf{D}]_{i,j}$ represents the frequency of term i in document j. Topic modeling (Hofmann, 1999; Blei et al., 2003) considers each document to be composed as a mixture of k topics. Given a topic, each term has a probability of occurring that is document-invariant, but the proportion of topics themselves vary across documents. This can be expressed as: $\mathbf{D} = \mathbf{CW}$, where $\mathbf{C}$ is an $\mathbb{R}^{m\times k}$ matrix, $[\mathbf{C}]_{i,j}$ is the probability of term i given topic j, and $\mathbf{W}$ is an $\mathbb{R}^{k\times r}$ matrix, where $[\mathbf{W}]_{i,j}$ is the proportion of topic i in document j. The isomorphism is then: document $\equiv$ domain, topic $\equiv$ label, term $\equiv$ input-sample, and $\mathbf{D} = \mathbf{CW} \equiv \mathbf{Q}_{X|D} = \mathbf{Q}_{X|Y}\mathbf{Q}_{Y|D}$. We leverage this isomorphism to extend identifiability conditions of the topic modeling problem (Donoho & Stodden, 2003; Chen et al., 2021) to our LLS-setting. We formalize the adaption here:

Theorem 3.1. *(adapted from Proposition 1 in Huang et al. (2016)) Assume A.1, A.2 and A.4 hold (A.4 in the discrete setting is referred to as the anchor word condition). Then the solution to $\mathbf{Q}_{X|D} = \mathbf{Q}_{X|Y}\mathbf{Q}_{Y|D}$ is uniquely identified.*

We refer readers to Huang et al. (2016) for a proof of this theorem. Theorem 3.1 states that if each label y has at least one token in the input space that has support only in y, and A.1, A.2 hold, then the solution to $\mathbf{Q}_{X|Y}$, $\mathbf{Q}_{Y|D}$ is unique. Furthermore, under this condition, there exist algorithms that can recover $\mathbf{Q}_{X|Y}$, $\mathbf{Q}_{Y|D}$ within some permutation (Arora et al., 2012a;b; Gillis & Vavasis, 2014).

Extensions to the continuous case We will prove identifiability in the continuous setting, when $\mathcal{X} = \mathbb{R}^p$ for some $p \geqslant 1$. In addition to A.1–A.4, we make the assumption that we have oracle access to $q(d|x)$, i.e., the true domain discriminator for mixture distribution Q. This is implied by assuming access to the marginal $q(x, d)$ from which we observe samples. We formalize this extension in the following theorem:

Theorem 3.2. *Let the distribution Q over X, Y, D satisfy Assumptions A.1–A.4. Assuming access to the joint distribution $q(x, d)$, we show that the following quantities are identifiable: (i) $\mathbf{Q}_{Y|D}$, (ii) $q(y|X = x)$, for all $x \in \mathcal{X}$ that lies in the support (i.e. $q(x) > 0$); and (iii) $q(y|X = x, D = d)$, for all $x \in \mathcal{X}$ and $d \in \mathcal{R}$ such that $q(x, d) > 0$.*

We present a proof sketch for Theorem 3.2 in App. B, and a full proof in App. D. The core idea is to show that with access to an oracle domain discriminator $f(x) = q(d|x)$, we can construct a map from a continuous space which satisfies assumption A.4, A.1 to a discrete space which satisfies anchor word condition. We also provide a geometric perspective on identifiability in the continuous setting in App. H.

4. DDFA Framework

Our algorithm estimates label marginal matrix $\mathbf{Q}_{Y|D}$ and domain-specific classifier $q(y|x, d)$, for continuous X.

Discriminate We begin Algorithm 1 by using pairs of (sample, domain from which sample came) to train a domain discriminative classifier $\widehat{f}$ on cross-entropy loss to predict the domain of each sample. This classifier outputs a distribution over domains for a given input. With unlimited data, the minimizer of this loss is the true f (shown in App. E). To avoid overfitting, we stop training $\widehat{f}$ when the cross-entropy domain loss on unlabeled validation data stops decreasing.

Discretize We push all train and validation data through the domain discriminator to get vector $\widehat{f}(x_i)$ for each sample x_i. In the proof of Theorem 3.2, we show that when working with true f, and the entire marginal $q(x)$, we can find a discretization satisfying the anchor word condition by selecting point masses of at least ϵ size in the distribution of $f(x)$. In the practical setting, with a finite set of data and a noisy $\widehat{f}$, we use clustering to approximately find point masses. We find $m \geqslant k$ clusters with a standard clustering procedure (e.g. K-means). If the clustering is sufficiently granular, we intuit that these m clusters include k clusters corresponding to the anchor subdomains $f(A_y)$. We use the cluster mapping (denoted c) to discretize samples into finite space $[m]$. Combining cluster id $c(x_i)$ with domain source d_i for each sample, we estimate $\widehat{\mathbf{Q}}_{c(X)|D}$ by computing, for each domain, the fraction of its samples assigned to each cluster.

Factorize We apply an NMF algorithm to $\widehat{\mathbf{Q}}_{c(X)|D}$ to obtain our estimates of $\widehat{\mathbf{Q}}_{c(X)|Y}$ and $\widehat{\mathbf{Q}}_{Y|D}$.

Adjust We invoke Algorithm 2 on each test point (x', d'). We obtain a plug-in estimate $\widehat{q}(y|x')$ by left-multiplying $\widehat{f}(x')$ with the pseudo-inverse of $\widehat{\mathbf{Q}}_{D|Y}$ (if our estimates are perfect, this recovers the true solution, according to Lemma B.1). Using the steps from the proof of Lemma B.2,

Algorithm 1 DDFA Training

input $r \geqslant k \geqslant 1, \{(x_i, d_i)\}_{i\in[n]} \sim q(x, d)$,
A class of functions $\mathcal{F}$ from $\mathbb{R}^p \to \mathbb{R}^r$
Split into train set T and validation set V
Train $\widehat{f} \in \mathcal{F}$ to minimize cross entropy loss to predict $d|x$ on T with early stopping on V
Push all $\{x_i\}_{i\in[n]}$ through $\widehat{f}$
Cluster the n points $\{\widehat{f}(x_i)\}_{i\in[n]}$, into m clusters.
$c(x_i) \leftarrow$ Cluster id of $\widehat{f}(x_i)$
$\widehat{q}(c(X) = a|D = b) \leftarrow \frac{\sum_{i\in[n]} \mathbb{I}[c(x_i)=a,\ d_i=b]}{\sum_{j\in[n]} \mathbb{I}[d_j=b]}$
For all a, b, $[\widehat{\mathbf{Q}}_{c(X)|D}]_{a,b} \leftarrow \widehat{q}(c(X) = a|D = b)$
$\widehat{\mathbf{Q}}_{c(X)|Y}, \widehat{\mathbf{Q}}_{Y|D} \leftarrow$ NMF $(\widehat{\mathbf{Q}}_{c(X)|D})$
output $\widehat{\mathbf{Q}}_{Y|D}, \widehat{f}$

Algorithm 2 DDFA Prediction

input $\widehat{\mathbf{Q}}_{Y|D}, \widehat{f}, (x', d') \sim q(x, d)$
Populate $\widehat{\mathbf{Q}}_{D|Y}$ as $[\widehat{\mathbf{Q}}_{D|Y}]_{d,y} \leftarrow \frac{[\widehat{\mathbf{Q}}_{Y|D}]_{y,d}}{\sum_{d''=1}^{d''=r}[\widehat{\mathbf{Q}}_{Y|D}]_{y,d''}}$
Assign $\widehat{q}(y|x') \leftarrow \left[(\widehat{\mathbf{Q}}_{D|Y})^{\dagger}\, \widehat{f}(x')\right]_y$
Assign $\widehat{q}(y|x', d') \leftarrow \dfrac{[\widehat{\mathbf{Q}}_{D|Y}]_{d',y}\widehat{q}(y|x')}{\sum_{y''\in[k]}[\widehat{\mathbf{Q}}_{D|Y}]_{d',y''}\widehat{q}(y''|x')}$
$y_{\text{pred}} \leftarrow \arg\max_{y\in[k]} \widehat{q}(y|x', d')$
output : $\widehat{q}(y|x', d') = \widehat{p}_{d'}(y|x')$, $\widehat{q}(y|x')$, y_{pred}

we get a plug-in estimate $\widehat{q}(y|x', d') = \widehat{p}_{d'}(y|x')$.

5. Experiments

Datasets To explore a failure mode of unsupervised classification, we use FieldGuide (https://tinyurl.com/4muwt7k9), which contains images of moths and butterflies. Each species is a class, and each class contains images from youth (caterpillar) and adult stages of life. We intuit that butterflies from a given species look more like butterflies from other species than caterpillars from their own species, and expect that unsupervised methods will learn incorrect class boundaries which distinguish caterpillars from butterflies, instead of true species boundaries. We assume each class has an anchor sub-domain, i.e., some images can only belong to that species. We build FieldGuide-2 and FieldGuide-28 subsets, with two and 28 species, respectively. Results from FieldGuide-2 and benchmarks CIFAR-10, CIFAR-20 (Krizhevsky & Hinton, 2009), and ImageNet-50 (Deng et al., 2009) are available in App. G.

LLS Setup Full semi-synthetic experiment setup is found in App. F. We sample $p_d(y)$ from a symmetric Dirichlet distribution with concentration α/k and enforce max condition number κ on $\mathbf{Q}_{Y|D}$. Small α and small κ encourage sparsity in $\mathbf{Q}_{Y|D}$, so each label tends to only appear in a few domains. Larger α encourages each $p_d(y)$ to tend toward uniform. We draw from test, train, and valid datasets without replacement to match these distributions.

Table 1. Results on FieldGuide-28. By DDFA we refer to DDFA initialized with pretext training adopted by SCAN. In DDFA, we do not use SCAN to cluster. α is Dirichlet parameter used for generating label marginals in each domain, κ is maximum allowed condition number of generated $\mathbf{Q}_{Y|D}$ matrix, r is number of domains.

		$\alpha : 0.5,\ \kappa : 12$		$\alpha : 3,\ \kappa : 20$		$\alpha : 10,\ \kappa : 28$	
r	Approach	Test acc	$\mathbf{Q}_{Y\|D}$ err	Test acc	$\mathbf{Q}_{Y\|D}$ err	Test acc	$\mathbf{Q}_{Y\|D}$ err
28	SCAN	0.288	-	0.292	-	**0.314**	-
	DDFA	**0.547**	0.036	**0.310**	0.034	0.314	0.036
37	SCAN	0.295	-	0.313	-	0.315	-
	DDFA	**0.760**	0.028	**0.521**	0.032	**0.326**	0.041
47	SCAN	0.281	-	0.318	-	**0.304**	-
	DDFA	**0.709**	0.035	**0.473**	0.035	0.299	0.039

Baseline Unsupervised classification method SCAN (Van Gansbeke et al., 2020) is trained on train and valid splits of each dataset, then evaluated on sampled test subsets.

Training and Evaluation The domain discriminator is ResNet-18, pretrained with SCAN contrastive pre-text weights. We train this network on images x_i to predict domain indices d_i, choose best iteration by valid loss, pass train and valid data through $\widehat{f}$, and cluster pushforward predictions $\widehat{f}(x_i)$. We compute the $\widehat{\mathbf{Q}}_{c(X)|D}$ matrix and run NMF to obtain $\widehat{\mathbf{Q}}_{c(X)|Y}$, $\widehat{\mathbf{Q}}_{Y|D}$. We predict class labels with Algorithm 2, then compute the highest true accuracy among any permutation of these labels (denoted "Test acc"). Permuting rows of $\widehat{\mathbf{Q}}_{Y|D}$ to match this label permutation, we compute the average absolute difference between corresponding entries of $\widehat{\mathbf{Q}}_{Y|D}$ and $\mathbf{Q}_{Y|D}$ (denoted "$\mathbf{Q}_{Y|D}$ err"). See App. F for full details, as well as setup for other base domain discriminator models and initialization strategies.

Results On FieldGuide-28 (Table 1), DDFA outperforms SCAN when $\mathbf{Q}_{Y|D}$ is sufficiently sparse (sampled with $\alpha : 0.5$ or $\alpha : 3$), with the highest observed accuracy difference ranging above 30-40%. We do not claim that SCAN is too weak to find image groupings on this data; instead we acknowledge that the domain information available to DDFA (and not to SCAN) is helpful for finding the true class distinctions between species as opposed to spurious distinctions between adult and immature life stages. Results from all experiments are available in App. G.

Conclusion Our theoretical results demonstrate that under LLS, we can leverage shifts among previously seen domains to recover correct class distinctions in a purely unsupervised manner. We believe that this work is just the first step in a new direction for leveraging structural assumptions together with distribution shift to perform unsupervised learning.

References

Alexandari, A., Kundaje, A., and Shrikumar, A. Adapting to label shift with bias-corrected calibration. In *International Conference on Machine Learning (ICML)*, 2019.

Arjovsky, M., Bottou, L., Gulrajani, I., and Lopez-Paz, D. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Arora, S., Ge, R., Kannan, R., and Moitra, A. Computing a nonnegative matrix factorization – provably. In *Symposium on Theory of Computing (STOC)*, 2012a. doi: 10.1145/2213977.2213994. URL https://doi.org/10.1145/2213977.2213994.

Arora, S., Ge, R., and Moitra, A. Learning topic models–going beyond svd. In *Foundations of Computer Science (FOCS)*. IEEE, 2012b.

Azizzadenesheli, K., Liu, A., Yang, F., and Anandkumar, A. Regularized learning for domain adaptation under label shifts. In *International Conference on Learning Representations (ICLR)*, 2019.

Bekker, J. and Davis, J. Learning from positive and unlabeled data: a survey. *Machine Learning*, 109(4):719–760, apr 2020. doi: 10.1007/s10994-020-05877-5. URL https://doi.org/10.1007%2Fs10994-020-05877-5.

Blei, D. M., Ng, A. Y., and Jordan, M. I. Latent dirichlet allocation. *Journal of Machine Learning research*, 3(Jan): 993–1022, 2003.

Caron, M., Bojanowski, P., Joulin, A., and Douze, M. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 132–149, 2018.

Caron, M., Bojanowski, P., Mairal, J., and Joulin, A. Unsupervised pre-training of image features on non-curated data, 2019. URL https://www.cv-foundation.org/openaccess/content_iccv_2015/papers/Doersch_Unsupervised_Visual_Representation_ICCV_2015_paper.pdf.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning (ICML)*, pp. 1597–1607. PMLR, 2020.

Chen, Y., He, S., Yang, Y., and Liang, F. Learning topic models: Identifiability and finite-sample analysis. *arXiv preprint arXiv:2110.04232*, 2021.

Cichocki, A. and Phan, A.-H. Fast local algorithms for large scale nonnegative matrix and tensor factorizations. *IEICE Transactions*, 92-A:708–721, 03 2009. doi: 10.1587/transfun.E92.A.708.

Crouse, D. F. On implementing 2d rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4):1679–1696, 2016.

Darlow, L. N., Crowley, E. J., Antoniou, A., and Storkey, A. J. Cinic-10 is not imagenet or cifar-10, 2018. URL https://arxiv.org/abs/1810.03505.

de Paulo Faleiros, T. and de Andrade Lopes, A. On the equivalence between algorithms for non-negative matrix factorization and latent dirichlet allocation. In *European Symposium on Artificial Neural Networks (ESANN)*, 2016.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Ieee, 2009.

Ding, C., Li, T., and Peng, W. On the equivalence between non-negative matrix factorization and probabilistic latent semantic indexing. *Computational Statistics & Data Analysis*, 52(8):3913–3927, 2008.

Doersch, C., Gupta, A., and Efros, A. A. Unsupervised visual representation learning by context prediction. In *International Conference on Computer Vision (ICCV)*, 2015. URL https://www.cv-foundation.org/openaccess/content_iccv_2015/papers/Doersch_Unsupervised_Visual_Representation_ICCV_2015_paper.pdf.

Donoho, D. and Stodden, V. When does non-negative matrix factorization give a correct decomposition into parts? *Advances in Neural Information Processing Systems (NeurIPS)*, 16, 2003.

du Plessis, M. C., Niu, G., and Sugiyama, M. Class-prior estimation for learning from positive and unlabeled data. *Machine Learning*, 106(4):463–492, nov 2016. doi: 10.1007/s10994-016-5604-6. URL https://doi.org/10.1007%2Fs10994-016-5604-6.

Elkan, C. and Noto, K. Learning classifiers from only positive and unlabeled data. In *SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 213–220, 2008.

Garg, S., Wu, Y., Balakrishnan, S., and Lipton, Z. A unified view of label shift estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. URL https://arxiv.org/abs/2003.07554.

Garg, S., Wu, Y., Smola, A., Balakrishnan, S., and Lipton, Z. Mixture proportion estimation and PU learning: A modern approach. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. URL https://arxiv.org/abs/2111.00980.

Gaussier, E. and Goutte, C. Relation between plsa and nmf and implications. In *Conference on Research and Development in Information Retrieval (SIGIR)*, 2005.

Gidaris, S., Singh, P., and Komodakis, N. Unsupervised representation learning by predicting image rotations, 2018. URL https://arxiv.org/abs/1803.07728.

Gillis, N. and Vavasis, S. A. Fast and robust recursive algorithms for separable nonnegative matrix factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):698–714, apr 2014. doi: 10.1109/tpami.2013.226. URL https://doi.org/10.1137%2F130946782.

Girolami, M. and Kabán, A. On an equivalence between plsi and lda. In *Conference on Research and Development in Information Retrieval (SIGIR)*, 2003.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, 2020.

Hofmann, T. Probabilistic latent semantic indexing. In *Conference on Research and Development in Information Retrieval (SIGIR)*, 1999.

Huang, K., Fu, X., and Sidiropoulos, N. D. Anchor-free correlated topic modeling: Identifiability and algorithm. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Hyvärinen, A. and Oja, E. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5): 411–430, 2000.

Ivanov, D. DEDPUL: Difference-of-estimated-densities-based positive-unlabeled learning. *arXiv preprint arXiv:1902.06965*, 2019.

Johnson, J., Douze, M., and Jégou, H. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7 (3):535–547, 2019.

Krizhevsky, A. and Hinton, G. Learning Multiple Layers of Features from Tiny Images. Technical report, Citeseer, 2009.

Kumar, A., Sindhwani, V., and Kambadur, P. Fast conical hull algorithms for near-separable non-negative matrix factorization. In *International Conference on Machine Learning*, pp. 231–239. PMLR, 2013.

Lipton, Z., Wang, Y.-X., and Smola, A. Detecting and correcting for label shift with black box predictors. In *International Conference on Machine Learning (ICML)*. PMLR, 2018.

Liu, T. and Tao, D. Classification with noisy labels by importance reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 38 (3):447–461, mar 2016. doi: 10.1109/tpami.2015.2456899. URL https://doi.org/10.1109%2Ftpami.2015.2456899.

MacQueen, J. et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pp. 281–297. Oakland, CA, USA, 1967.

Muandet, K., Balduzzi, D., and Schölkopf, B. Domain generalization via invariant feature representation. In *International Conference on Machine Learning (ICML)*. PMLR, 2013.

Noroozi, M. and Favaro, P. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*, pp. 69–84. Springer, 2016.

Paatero, P. and Tapper, U. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

Papadimitriou, C. H., Raghavan, P., Tamaki, H., and Vempala, S. Latent semantic indexing: A probabilistic analysis. *Journal of Computer and System Sciences*, 61(2): 217–235, 2000.

Park, S., Han, S., Kim, S., Kim, D., Park, S., Hong, S., and Cha, M. Improving unsupervised image clustering with robust learning. *CoRR*, abs/2012.11150, 2020. URL https://arxiv.org/abs/2012.11150.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research (JMLR)*, 12, 2011.

Prabhudesai, K. S., Mainsah, B. O., Collins, L. M., and Throckmorton, C. S. Augmented latent dirichlet allocation (lda) topic model with gaussian mixture topics. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2451–2455. IEEE, 2018.

Reynolds, D. A. Gaussian mixture models. *Encyclopedia of biometrics*, 741(659-663), 2009.

Saerens, M., Latinne, P., and Decaestecker, C. Adjusting the Outputs of a Classifier to New a Priori Probabilities: A Simple Procedure. *Neural Computation*, 2002.

Scott, C. A Rate of Convergence for Mixture Proportion Estimation, with Application to Learning from Noisy Labels. In *Artificial Intelligence and Statistics (AISTATS)*, 2015. URL https://proceedings.mlr.press/v38/scott15.html.

Seung, D. and Lee, L. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems (NeurIPS)*, 2001.

Storkey, A. When training and test sets are different: characterizing learning transfer. *Dataset shift in machine learning*, 30:3–28, 2009.

Tan, V. Y. and Févotte, C. Automatic relevance determination in nonnegative matrix factorization with the/spl beta/-divergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 35(7), 2012.

Tian, D. Research on plsa model based semantic image analysis: A systematic review. *J. Inf. Hiding Multim. Signal Process.*, 9(5):1099–1113, 2018.

Van Gansbeke, W., Vandenhende, S., Georgoulis, S., Proesmans, M., and Van Gool, L. Scan: Learning to classify images without labels, 2020. URL https://arxiv.org/abs/2005.12320.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Zhang, K., Schölkopf, B., Muandet, K., and Wang, Z. Domain adaptation under target and conditional shift. In *International Conference on Machine Learning (ICML)*. PMLR, 2013.

A. Related Work

Unsupervised Learning Standard unsupervised learning approaches for discovering labels often rely on similarity in the original data space (MacQueen et al., 1967; Reynolds, 2009). While distances in feature space become meaningless for high-dimensional data, deep learning researchers have turned to similarity in a representations space learned via self-supervised contrastive tasks (Doersch et al., 2015; Noroozi & Favaro, 2016; Gidaris et al., 2018; Chen et al., 2020), or similarity in a feature space learned end-to-end for a clustering task (Caron et al., 2018; 2019; Park et al., 2020; Van Gansbeke et al., 2020). Our problem setup closely resembles independent component analysis (ICA), where one seeks to identify statistically independent signal components from mixtures (Hyvärinen & Oja, 2000). However, ICA's assumption of statistical independence among the components does not obtain in our setup. In topic modeling (Papadimitriou et al., 2000; Blei et al., 2003; Arora et al., 2012b; Huang et al., 2016; Chen et al., 2021), documents are modeled as mixtures of topics, and topics as categorical distributions over a finite vocabulary. Early topic modeling approaches papers with relaxed assumptions on the distribution of topic mixing coefficients (pLSI) (Hofmann, 1999; Papadimitriou et al., 2000) as well as the well-known Latent Dirichlet Allocation (LDA) which assumes that topic mixing coefficients are drawn from a Dirichlet distribution (Blei et al., 2003). The topic modeling literature often draws on non-negative Matrix Factorization (NMF) methods (Paatero & Tapper, 1994; Seung & Lee, 2001), which decompose a given matrix into a product of two matrices with non-negative elements (Girolami & Kabán, 2003; Gaussier & Goutte, 2005; Ding et al., 2008; de Paulo Faleiros & de Andrade Lopes, 2016). In both Topic Modeling and NMF, a fundamental problem has been to characterize the precise conditions under which the system is uniquely identifiable (Donoho & Stodden, 2003; Arora et al., 2012b; Huang et al., 2016; Chen et al., 2021). The anchor condition (also referred to as separability) is known to be instrumental for identifying topic models (Donoho & Stodden, 2003; Arora et al., 2012b; Huang et al., 2016; Chen et al., 2021). In this work, we extend these ideas, leveraging separable subsets of each label's support (the anchor sets) to produce anchor words in the discretized problem. Existing methods have attempted to extend latent variable modeling to continuous input domains by making assumptions about the functional forms of the class-conditional densities, e.g., restricting to Gaussian mixtures (Reynolds, 2009; Prabhudesai et al., 2018). A second line of approach involves finding an appropriate discretization of the continuous space (Tian, 2018).

Distribution Shift under the Label Shift Assumption The label shift assumption, where $p_d(y)$ can vary but $p(x|y)$ cannot, has been extensively studied in the domain adaptation literature (Saerens et al., 2002; Storkey, 2009; Zhang et al., 2013; Lipton et al., 2018; Garg et al., 2020) and also obtains in the problem of learning from positive and unlabeled data (Elkan & Noto, 2008; Bekker & Davis, 2020; Garg et al., 2021). For both problems, many classical approaches suffer from the curse of dimensionality, failing in the settings where deep learning prevails. Our solution strategy draws inspiration from recent work on label shift (Lipton et al., 2018; Alexandari et al., 2019; Azizzadenesheli et al., 2019; Garg et al., 2020) and PU learning (Scott, 2015; Liu & Tao, 2016; Bekker & Davis, 2020; Garg et al., 2021) that leverage black-box predictors to produce sufficient low-dimensional representations for identifying target distributions of interest (other works leverage black box predictors heuristically (Ivanov, 2019)). **Key differences:** While PU learning requires identifying *one* new class for which we lack labeled examples provided that the positive class contains an anchor set (Garg et al., 2021), LLS can identify an arbitrary number of classes (up to permutation) from completely unlabeled data, provided a sufficient number of domains.

Domain Generalization The related problem of Domain Generalization (DG) also addresses learning with data drawn from multiple distributions and where the domain identifiers play a key role (Muandet et al., 2013; Arjovsky et al., 2019). However in DG, we are given *labeled* data from multiple domains, and our goal is to learn a classifier that can generalize to new domains. By contrast, in LLS, we work with unlabeled data only, leveraging the problem structure to identify the underlying labels.

B. Proof Sketch

In this section we present a proof sketch for Theorem 3.2. We begin by first presenting key lemmas (we include their proofs in App. C). The full proof of Theorem 3.2 is in App. D.

Lemma B.1. *Under the same assumptions as Theorem 3.2, the matrix* $\mathbf{Q}_{Y|D}$ *and* $f(x) = q(d|x)$ *uniquely determine* $q(y|x)$ *for all* $y \in \mathcal{Y}$ *and* $x \in \mathcal{X}$ *such that* $q(x) > 0$.

Lemma B.1 states that given matrix $\mathbf{Q}_{Y|D}$ and oracle domain discriminator, we can uniquely identify $q(y|x)$. In particular, we show that for any $x \in \mathcal{X}$, $q(d|x)$ can be expressed as a convex combination of the k columns of $\mathbf{Q}_{D|Y}$ (which is computed from $\mathbf{Q}_{Y|D}$ and is column rank k) and the coefficients of the combination are $q(y|x)$. Combining this with the linear independence of the columns of $\mathbf{Q}_{D|Y}$, we show that these coefficients are unique. In the following lemma, we show

how the identified $q(y|x)$ can then be used to identify $q(y|x, d)$:

Lemma B.2. *Under the same assumptions as Theorem 3.2, for all $y \in \mathcal{Y}$ and $x \in \mathcal{X}$ such that $q(x, d) > 0$, the matrix $\mathbf{Q}_{Y|D}$ and $q(y|x)$ uniquely determine $q(y|x, d)$.*

To prove Lemma B.2, we show that we can combine the conditional distribution over the labels given a sample $x \in \mathcal{X}$ with the prior distribution of the labels in each domain to determine the posterior distribution over labels given the sample x and the domain of interest. Next, we introduce a key property of the domain discriminator classifier f:

Lemma B.3. *Under the same assumptions as Theorem 3.2, for all x, x' in anchor sub-domain, i.e., $x, x' \in A_y$ for a given label $y \in \mathcal{Y}$, we have $f(x) = f(x')$. Further, for any $y \in \mathcal{Y}$, if $x \in A_y, x' \notin A_y$, then $f(x) \neq f(x')$.*

Lemma B.3 implies that the oracle domain discriminator f maps all points in an anchor subdomain, and only those points in that anchor subdomain to the same point in $f(x) = q(d|x)$ space. We can now present a proof sketch for Theorem 3.2 (full proof in App. D):

Proof sketch of Theorem 3.2. The key idea of the proof lies in proposing a discretization such that some subset of anchor subdomains for each label y in the continuous space map to distinct anchor words in discrete space. In particular, if there exists a discretization of the continuous space $\mathcal{X}$ that for any $y \in \mathcal{Y}$, maps all $x \in A_y$ to the same point in the discrete space, but no $x \notin A_y$ maps to this point, then this point serves as an anchor word. From Lemma B.3, we know that for each $y \in \mathcal{Y}$, there is a unique point in $f(x)$ space to which all $x \in A_y$ are mapped and to which no $x' \notin A_y$ is mapped. Pushing all the $x \in \mathcal{X}$ through f, we know from A.4 that there exists k point masses of size ϵ, one for each $f(A_y)$ in the $f(x) = q(d|x)$ space. We can now inspect this space for point masses of size at least ϵ to find at most $\mathcal{O}(1/\epsilon)$ such point masses among which are contained the k point masses corresponding to the anchor subdomains. Discretizing this space by assigning each point mass to a group (and non-point masses to a single additional group), we have k groups that have support only in one y each. Thus, we have achieved a discretization with anchor words. Further, since the discrete space arises from a pushforward of the continuous space through f, the discrete space also satisfies the latent label shift assumption A.1. We now use Theorem 3.1 to claim identifiability of $\mathbf{Q}_{Y|D}$. We then use Lemmas B.1 and B.2 to prove parts (ii) and (iii).

C. Proofs of Lemmas

In this section, we present several new lemmas which are required to prove Theorem 3.2, and provide proofs. We also provide proofs for Lemmas B.1, B.2, and B.3.

Lemma C.1. *Let distribution Q over random variables X, Y, D satisfy A.1–A.4. Then for all $y \in \mathcal{Y}$, $q(y) > 0$. That is, all labels have nonzero probability under Q.*

Proof of Lemma C.1. Proof by contradiction. Let $y \in \mathcal{Y}$. Assume $q(y) = 0$. We can write $q(y) = \sum_{d \in \mathcal{R}} q(d)q(y|D = d)$. Then $q(y) = \sum_{d \in \mathcal{R}} \gamma_y q(y|D = d) = \sum_{d \in \mathcal{R}} \frac{1}{r} q(y|D = d)$. Then $q(y) = \frac{1}{r} \sum_{d \in \mathcal{R}} q(y|D = d)$. Since $q(y|D = d) \geqslant 0$ for all $d \in \mathcal{R}$, we see that $q(y) = 0 \implies q(y|D = d) = 0$ for all $d \in \mathcal{R}$.

Then $[\mathbf{Q}_{Y|D}]_{y,d} = 0$ for all $d \in \mathcal{R}$. Then there is a row (row d) in the matrix $\mathbf{Q}_{Y|D}$ in which every entry is 0, so $\mathbf{Q}_{Y|D}$ cannot be rank k. This violates assumption A.2. Then by contradiction we have shown $q(y) > 0$. □

Lemma C.2. *Let distribution Q over random variables X, Y, D satisfy Assumptions A.1–A.4. Let $x \in \mathcal{X}$ such that $q(x) > 0$. Then if $x \in A_y$ for some $y \in \mathcal{Y}$, we have that $q(y|X = x) = 1$, and for all $y' \in \mathcal{Y} \backslash \{y\}$, $q(y'|X = x) = 0$. The converse is also true: if $q(y|X = x) = 1$ for some $y \in \mathcal{Y}$ and $q(y'|X = x) = 0$ $\forall y' \in \mathcal{Y} \backslash \{y\}$, then we know that $x \in A_y$.*

Proof of Lemma C.2. We prove directions one at a time.

- **Forward direction.** We seek to show that:

$$x \in A_y \implies \left[q(y|X = x) = 1,\ \forall y' \in \mathcal{Y} \backslash \{y\}, q(y'|X = x) = 0\right].$$

Assume $x \in A_y$. We recall that we earlier assumed $q(x) > 0$.

$$q(x) = \sum_{y'' \in \mathcal{Y}} q(y'')q(x|Y = y'') = q(y)q(x|Y = y) + \sum_{y' \in \mathcal{Y}\setminus\{y\}} q(y')q(x|Y = y')\,.$$

Now $q(x) = q(y)q(x|Y = y) + \sum_{y' \in \mathcal{Y}\setminus\{y\}} q(y')\,(0) = q(y)q(x|Y = y)$. Because $q(x|y) > 0$ (by A.4) and $q(y) > 0$ (by Lemma C.1), we know that $q(x) = q(y)q(x|Y = y) > 0$. Then $q(y|X = x) = \dfrac{q(y)q(x|Y = y)}{q(x)} = \dfrac{q(x)}{q(x)} = 1$ (Bayes' rule). By the properties of a probability distribution, $q(y|X = x) + \sum_{y' \in \mathcal{Y}\setminus\{y\}} q(y'|X = x) = 1$. Then because $q(y|X = x) = 1$, we have : $\sum_{y' \in \mathcal{Y}\setminus\{y\}} q(y'|X = x) = 0$. Then for all $y' \in \mathcal{Y}\setminus\{y\}$, it must be that $q(y'|X = x) = 0$ Then we have shown $q(y|X = x) = 1$, and for all $y' \in \mathcal{Y}\setminus\{y\},\ q(y'|X = x) = 0$.

- **Converse.**

  We seek to show that:

$$\big[q(y|X = x) = 1,\ \forall y' \in \mathcal{Y}\setminus\{y\}, q(y'|X = x) = 0\big] \implies x \in A_y.$$

  Assume $q(y|X = x) = 1$ and for all $y' \in \mathcal{Y}\setminus\{y\}, q(y'|X = x) = 0$. We recall that we earlier assumed $q(x) > 0$. Also, $q(y) > 0$ by Lemma C.1. Then $q(x|Y = y) = \dfrac{q(y|X = x)q(x)}{q(y)} = \dfrac{(1)q(x)}{q(y)} > 0$. Let $y' \in \mathcal{Y}\setminus\{y\}$. Then $q(x|Y = y') = \dfrac{q(y'|X = x)q(x)}{q(y')} = \dfrac{(0)q(x)}{q(y')} = 0$. Then because $q(x|Y = y) > 0$ and $\forall y' \in \mathcal{Y}\setminus\{y\},\ q(x|Y = y) = 0$, we see that $x \in A_y$.

□

Lemma C.3. *Let random variables X, Y, D and distribution Q satisfy Assumptions A.1–A.4. Then, the matrix $\mathbf{Q}_{D|Y}$, defined as an $r \times k$ matrix whose elements are $[\mathbf{Q}_{D|Y}]_{i,j} = Q(D = i|Y = j)$, and in which each column is a conditional distribution over the domains given a label, has linearly independent columns. Furthemore, $\mathbf{Q}_{D|Y}$ can be computed directly from only $\mathbf{Q}_{Y|D}$.*

Proof of Lemma C.3. Let random variables X, Y, D and distribution Q satisfy Assumptions A.1–A.4.

Each $[\mathbf{Q}_{D|Y}]_{d,y} = q(d|Y = y) = \dfrac{q(y|D = d)q(d)}{q(y)} = \dfrac{q(y|D = d)\gamma_d}{q(y)} = \dfrac{q(y|D = d)}{rq(y)}$.

Since each column of $\mathbf{Q}_{D|Y}$ is a probability distribution that sums to 1, and $rq(y)$ is constant down each column y, we can obtain $\mathbf{Q}_{D|Y}$ by simply taking $\mathbf{Q}_{Y|D}^\top$, in which each $[\mathbf{Q}_{Y|D}^\top]_{d,y} = [\mathbf{Q}_{Y|D}]_{y,d} = q(y|D = d)$, and normalizing the columns so they sum to 1.

The matrix $\mathbf{Q}_{Y|D}$ has linearly independent rows by Assumption A.2. Then $\mathbf{Q}_{Y|D}^\top$ has linearly independent columns. Scaling these columns by a nonzero value does not change their linear independence, so the columns of $\mathbf{Q}_{D|Y}$ are also linearly independent.

Then matrix $\mathbf{Q}_{D|Y}$ has linearly independent columns, and can be computed by taking $\mathbf{Q}_{Y|D}^\top$ and normalizing its columns.

□

Lemma C.4. *Let random variables X, Y, D and distribution Q satisfy Assumptions A.1–A.4. Let $d \in \mathcal{R}, x \in \mathcal{X}, y \in \mathcal{Y}$. Then $q(d|X = x, Y = y) = q(d|Y = y)$.*

Proof of Lemma C.4.

$$\begin{aligned} q(d|X = x, Y = y) &= \frac{q(x|D = d, Y = y)q(D = d|Y = y)}{q(x|Y = y)} \\ &= \frac{p_d(x|Y = y)q(d|Y = y)}{q(x|Y = y)} \\ &= \frac{p(x|Y = y)q(d|Y = y)}{q(x|Y = y)} \\ &= \frac{q(x|Y = y)q(d|Y = y)}{q(x|Y = y)} \\ &= q(d|Y = y)\,. \end{aligned}$$

□

Proof of Lemma B.1. Let distribution Q over random variables X, Y, D satisfy Assumptions A.1-A.4. Let $x \in \mathcal{X}$ with $q(x) > 0$, and $y \in \mathcal{Y}$.

Assume we know $\mathbf{Q}_{Y|D}$ and $[f(x)]_d = q(d|X = x)$. With $\mathbf{Q}_{Y|D}$, we know $q(y|D = d)\forall y, d$. Also, with the oracle f, we are able to obtain $q(d|X = x)\forall x, d$. For all $d \in \mathcal{R}$, we can write that $q(d|X = x) = \sum_{y' \in \mathcal{Y}} q(d|X = x, Y = y')q(y'|X = x) = \sum_{y' \in \mathcal{Y}} q(d|Y = y')q(y'|X = x)$, using Lemma C.4.

Define the vector-valued function $g : \mathcal{X} \to \mathbb{R}^k$ such that $[g(x)]_y = q(y|X = x)$ for all $x \in \text{supp}(X)$. $\mathbf{Q}_{D|Y}$ is a matrix of shape $r \times k$, with $[\mathbf{Q}_{D|Y}]_{i,j} = Q(D = i|Y = j)$. It can be computed from $\mathbf{Q}_{Y|D}$ and has linearly independent columns—both facts shown in Lemma C.3.

Then $[f(x)]_d = q(d|X = x) = \mathbf{Q}_{D|Y}[d, :]g(x)$, a product between the dth row vector of $\mathbf{Q}_{D|Y}$ and the column vector $g(x)$. Then $f(x) = \mathbf{Q}_{D|Y}g(x)$.

This system is a linear system with $r \geqslant k$ equations. Recalling that $\mathbf{Q}_{D|Y}$ has k linearly independent columns, we can select any k linearly independent rows of $\mathbf{Q}_{D|Y}$ to solve the equation for the true, unique solution for the unknown vector $g(x)$. Another way to describe this is with the pseudo-inverse: $g(x) = (\mathbf{Q}_{D|Y})^{\dagger} f(x)$. Then we have $[g(x)]_y = q(y|X = x)$ for all $y \in \mathcal{Y}$.

□

Proof of Lemma B.2. Let distribution Q over random variables X, Y, D satisfy Assumptions A.1-A.4. Let $x \in \mathcal{X}, d \in \mathcal{R}$ with $q(x, d) > 0$, and $y \in \mathcal{Y}$.

Assume we know matrix $\mathbf{Q}_{Y|D}$ and $q(y'|X = x),\ \forall y' \in \mathcal{Y}$. We can compute $\mathbf{Q}_{D|Y}$ from $\mathbf{Q}_{Y|D}$ via Lemma C.3.

$$q(y|X = x, D = d) = \frac{q(y, x, d)}{q(x, d)} = \frac{q(d|X = x, Y = y)q(y|X = x)q(x)}{q(d|X = x)q(x)}.$$

Using Lemma C.4, $q(d|X = x, Y = y) = q(d|Y = y)$.

$$\begin{aligned} q(y|X = x, D = d) &= \frac{q(d|Y = y)q(y|X = x)q(x)}{q(d|X = x)q(x)} \\ &= \frac{q(d|Y = y)q(y|X = x)}{q(d|X = x)}. \end{aligned}$$

The denominator $q(d|X = x)$ is constant across all values of y, so we can write that $q(y|X = x, D = d) \propto q(d|Y = y)q(y|X = x)$ and normalize to find the probability:

$$q(y|X = x, D = d) = \frac{q(d|Y = y)q(y|X = x)}{\sum_{y' \in \mathcal{Y}} q(d|Y = y')q(y'|X = x)}.$$

We know $q(d|Y = y)$ as $[\mathbf{Q}_{D|Y}]_{d,y}$, and every $q(d|Y = y')$, where $y' \in \mathcal{Y}$, as $[\mathbf{Q}_{D|Y}]_{d,y'}$. We also know $q(y|X = x)$ and every $q(y'|X = x)$ where $y' \in \mathcal{Y}$, by the lemma assumptions. Then we can compute $q(y|X = x, D = d)$. □

Proof of Lemma B.3. Let distribution Q over random variables X, Y, D satisfy Assumptions A.1-A.4. Recall $f : \mathbb{R}^p \to \mathbb{R}^r$ is a vector-valued oracle function such that $[f(x)]_d = q(d|X = x)$ for all $x \in \text{supp}_Q(X)$. Also let us recall that $\mathbf{Q}_{D|Y}$ is defined as an $r \times k$ matrix whose elements $[\mathbf{Q}_{D|Y}]_{i,j} = Q(D = i|Y = j)$, and each column is a conditional distribution over the domains given a label . It has linearly independent columns by Lemma C.3.

First recognize that $\forall d \in \mathcal{R}, x \in \mathcal{X} : q(x) > 0$,

$$\begin{aligned}
[f(x)]_d = q(d|X = x) &= \sum_{y'' \in \mathcal{Y}} q(d, y''|X = x). \\
&= \sum_{y'' \in \mathcal{Y}} q(d|Y = y'', X = x)q(y''|X = x) \\
&= \sum_{y'' \in \mathcal{Y}} q(d|Y = y'')q(y''|X = x), \text{ by Lemma C.4.}
\end{aligned}$$

Then recognize that we can write $f(x) = \sum_{y'' \in \mathcal{Y}} q(y''|X = x)\mathbf{Q}_{D|Y}[:, y'']$, where $\mathbf{Q}_{D|Y}[:, y'']$ is the y''th column of $\mathbf{Q}_{D|Y}$. Now we could also rewrite $f(x) = \mathbf{Q}_{D|Y} \left[Q(Y = 1|X = x) \ldots Q(Y = k|X = x)\right]^\top$.

We now prove two key components of the statement. Let $y \in \mathcal{Y}$. Let $x \in A_y : q(x) > 0$.

- **Points in same anchor sub-domain map together.**

  Let $x' \in A_y$ such that $q(x') > 0$. We now seek to show that $f(x) = f(x')$. Recall that $x, x' \in A_y$. By Lemma C.2, $q(y|X = x) = q(y|X = x') = 1$. Also, $\forall y'' \in \mathcal{Y}\backslash\{y\}, q(y''|X = x) = q(y''|X = x') = 0$. Then for all $y'' \in \mathcal{Y}$, $q(y''|X = x) = q(y''|X = x')$.

  Therefore, $\forall d \in \mathcal{R}$,

$$\begin{aligned}
[f(x)]_d = q(d|X = x) &= \sum_{y'' \in \mathcal{Y}} q(d|Y = y'')q(y''|X = x) \\
&= \sum_{y'' \in \mathcal{Y}} q(d|Y = y'')q(y''|X = x') \\
&= q(d|X = x') = [f(x')]_d.
\end{aligned}$$

  Then $f(x) = f(x')$.

- **Point outside of the anchor sub-domain does not map with points in the anchor sub-domain**. Let $x_0 \notin A_y$ such that $q(x_0) > 0$. We now seek to show that $f(x) \neq f(x_0)$. Because $x_0 \notin A_y$ with $q(x_0) > 0$, and because A_y is maximal, then by Lemma C.2, it must be that one of the following cases is true:

  - **Case 1:** $q(y|X = x_0) \neq 1$
  - **Case 2:** $q(y'|X = x_0) > 0$ for some $y' \in \mathcal{Y}\backslash\{y\}$.

In all circumstances, $\exists y'' \in \mathcal{Y} : q(y''|x_0) \neq Q(Y = y''|x)$.

$$[Q(Y = 1|X = x)...Q(Y = k|X = x)]^\top \neq [Q(Y = 1|X = x_0)...Q(Y = k|X = x_0)]^\top .$$

Because $\mathbf{Q}_{D|Y}$ has linearly independent columns (shown in Lemma C.3), we now know that

$$\begin{aligned} f(x) &= \mathbf{Q}_{D|Y} \left[Q(Y = 1|X = x) \ ... \ Q(Y = k|X = x)\right]^\top \\ &\neq \mathbf{Q}_{D|Y} \left[Q(Y = 1|X = x_0) \ ... \ Q(Y = k|X = x_0)\right]^\top = f(x_0) \,. \end{aligned}$$

So $f(x) \neq f(x_0)$.

□

D. Proof of Theorem 3.2

Proof of Theorem 3.2. Let distribution Q over random variables X, Y, D satisfy Assumptions A.1-A.4.

Recall $f : \mathcal{X} \to \mathbb{R}^r$ is a vector-valued oracle function such that $[f(x)]_d = q(d|X = x) \forall x \in \text{supp}_Q(X)$. It is known because we know the marginal $q(x, d)$. Let $y \in \mathcal{Y}$. Then by Lemma B.3, f sends every $x \in A_y$ (and no other $x \notin A_y$) to the same value. We overload notation to denote this as $f(A_y)$. Then $Q(f(X) = f(A_y)) = Q(X \in A_y) \geqslant \epsilon$. Then in the marginal distribution of $f(X)$ with respect to distribution Q, there is a distinct point mass on each $f(A_y)$, with mass at least ϵ.

Because we know the marginal $q(x, d)$, we know the marginal $q(x)$, so we can obtain the distribution of $f(X)$ with respect to distribution Q. If we analyze the marginal distribution of $f(X)$ with respect to distribution Q, and recover all point masses with mass at least ϵ, we can recover no more than $\mathcal{O}\left(1/\epsilon\right)$ such points. We set $m \in \mathbb{Z}^+$ so that the number of points we recovered is $m - 1$.

We denote a mapping $\psi : \mathbb{R}^r \to [m]$. This mapping sends each value of $f(x)$ corresponding to a point mass with mass at least ϵ to a unique index in $\{1, ..., m - 1\}$. It sends any other value in $\mathbb{R}^p$ to m. We note that the ordering of the point masses might have $(m - 1)!$ permutations.

Notice that the point mass on each $f(A_y)$ must be recovered among these $m - 1$ masses. Recall that $\forall y \in \mathcal{Y}$, $[f(x) = f(A_y) \iff x \in A_y]$. Then for each $y \in \mathcal{Y}$, $[\psi(f(X)) = \psi(f(A_y)) \iff X \in A_y]$, because ψ does not send any other value in $\mathbb{R}^r$ besides $f(A_y)$ to $\psi(f(A_y))$.

For convenience, we now define a mapping $c : \mathcal{X} \to [m]$ such that $c = \psi \circ f$. We will also abuse notation here to denote $c(A_y) = \psi(f(A_y)) = \psi(f(x)), \ \forall x \in A_y$ Then $c(X)$ is a discrete, finite random variable that takes values in $[m]$. As c is a pushforward function on X, $c(X)$ satisfies the label shift assumption because X does.

We might now define a matrix $\mathbf{Q}_{c(X)|D}$ in which each entry $[\mathbf{Q}_{c(X)|D}]_{i,d} = Q(c(X) = i|D = d)$. Because $c(X)$ satisfies label shift, we know that we can decompose $\mathbf{Q}_{c(X)|D} = \mathbf{Q}_{c(X)|Y}\mathbf{Q}_{Y|D}$.

$$\begin{aligned} Q(c(x) = c(A_y)|Y = y) &= Q(X \in A_y|Y = y) > 0 \,. \\ Q(c(x) = c(A_y)|Y \neq y) &= Q(X \in A_y|Y \neq y) = 0 \,. \end{aligned}$$

Then for each $y \in \mathcal{Y}$, the row with row index $c(A_y)$ is positive in the yth column, and zero everywhere else. Restated, for each $y \in \mathcal{Y}$, there is some row with positive entry exactly in yth column. This is precisely the anchor word assumption for a discrete, finite random variable. We already know that $\mathbf{Q}_{Y|D}$ is full row-rank, so because $\mathbf{Q}_{c(X)|Y}$ satisfies the anchor word assumption, we can identify $\mathbf{Q}_{Y|D}$ up to permutation of rows by Theorem 3.1.

□

E. Minimizing Cross-Entropy Loss yields Domain Discriminator

Let distribution Q over random variables X, Y, D satisfy Assumptions A.1-A.4. We here examine the behavior of the cross-entropy loss, in the infinite data case (when we can work with expectations over the entire distribution instead of empirical expectations over a finite set of datapoints). Define the vector-valued function $z : \mathcal{R} \to \mathbb{R}^r$ such that $z(d)$ is a one-hot vector of length r, such that $[z(d)]_i = 1,$ iff $d = i$. Then we write the cross-entropy loss with targets as true domains as

$$\mathcal{L}_{CE} = \mathbb{E}_{(X,D)\sim Q}[-\sum_{i=1}^{i=r}[z(D)]_i \log([f(X)]_i)]$$

$$\mathcal{L}_{CE} = \mathbb{E}_X \mathbb{E}_{D|X}[-\sum_{i=1}^{i=r}[z(D)]_i \log([f(X)]_i)]$$

$$\mathcal{L}_{CE} = \mathbb{E}_X[-\sum_{i=1}^{i=r}\mathbb{E}_{D|X}[[z(D)]_i \log([f(X)]_i)]]$$

$$\mathcal{L}_{CE} = \mathbb{E}_X[-\sum_{i=1}^{i=r}\log([f(X)]_i)\mathbb{E}_{D|X}[[z(D)]_i]]$$

$$\mathcal{L}_{CE} = \mathbb{E}_X[-\sum_{i=1}^{i=r}\log([f(X)]_i)(1 \times Q(D=i|X) + 0 \times (1 - Q(D=i|X)))]$$

$$\mathcal{L}_{CE} = \mathbb{E}_X[-\sum_{i=1}^{i=r}\log([f(X)]_i)Q(D=i|X)]$$

In order to find the minimizer of the cross entropy loss over the class of all functions from $R^p \to [0,1]^r$, we formulate the following objective with the Lagrange constraint:

$$J = \min_{[f(X)]_1 \ldots [f(X)]_r} \mathbb{E}_X[-\sum_{i=1}^{i=r}\log([f(X)]_i)Q(D=i|X)] + \lambda(\sum_{i=1}^{i=r}[f(X)]_i - 1)$$

Setting partial derivative with respect to $[f(X)]_r$ to 0, we get $-\frac{Q(D=i|X)}{[f^\star(X)]_i} + \lambda = 0$ and $[f^\star(X)]_i = \frac{1}{\lambda}Q(D=i|X)$.

From KKT condition, the optimal solution lies on constraint surface, giving:

$$\sum_{i=1}^{i=r}[f^\star(X)]_i = 1$$

$$\sum_{i=1}^{i=r}\frac{1}{\lambda}Q(D=i|X) = 1$$

$$\frac{1}{\lambda}\sum_{i=1}^{i=r}Q(D=i|X) = 1$$

$$\frac{1}{\lambda} = 1$$

$$\lambda = 1$$

Finally, we get $[f^\star(X)]_i = Q(D=i|X)$, so the optimal f^* by the cross entropy loss as defined will in fact recover the oracle domain discriminator.

$$\begin{bmatrix} 0.17 & 0.65 \\ 0.83 & 0.35 \end{bmatrix}$$

(a) $\alpha : 0.5, \kappa : 3$

$$\begin{bmatrix} 0.37 & 0.06 \\ 0.63 & 0.94 \end{bmatrix}$$

(b) $\alpha : 3, \kappa : 5$

$$\begin{bmatrix} 0.42 & 0.25 \\ 0.58 & 0.75 \end{bmatrix}$$

(c) $\alpha : 10, \kappa : 7$

Figure 1. Example $\mathbf{Q}_{Y|D}$ matrices sampled for FieldGuide-2 with 2 classes and 2 domains. Each column represents the distribution across classes $p_y(d)$ for a given domain. At small α, each $p_y(d)$ is likelier to be "sparse", meaning that a few classes are heavily represented, and at large α, $p_y(d)$ tends toward a uniform distribution in which classes are represented evenly.

F. Additional Experimental Details

Our code is available at https://github.com/latentlabelshift-anonymous/latentlabelshift. Here we present the full generation procedure for semisynthetic example problems, and discuss the parameters.

1. Choose a Dirichlet concentration parameter $\alpha > 0$, maximum condition number $\kappa \geqslant 1$ (with respect to 2-norm), and domain count $r \geqslant k$.
2. For each $y \in [k]$, sample $p_d(y) \sim \text{Dir}(\frac{\alpha}{k}\mathbf{1}_k)$.
3. Populate the matrix $\mathbf{Q}_{Y|D}$ with the computed $p_d(y)$s. If $\text{cond}(\mathbf{Q}_{Y|D}) \geqslant k$, return to step 2 and re-sample.
4. Distribute examples across domains according to $\mathbf{Q}_{Y|D}$, for each of train, test, and valid sets. This procedure entails creating a quota number of examples for each (class, domain) pair, and drawing datapoints without replacement to fill each quota. We must discard excess examples from some classes in the dataset due to class imbalance in the $\mathbf{Q}_{Y|D}$ matrix. Due to integral rounding, domains may be *slightly* imbalanced.
5. Conceal true class information and return (x_i, d_i) pairs.

It is important to note the role of κ and α in the above formulation. Although they are unknown parameters to the classification algorithm, they affect the sparsity of the $\mathbf{Q}_{Y|D}$ and difficulty of the problem. Small α encourages high sparsity in $p_d(y)$, and large α causes $p_d(y)$ to tend towards a uniform distribution. We observe an example of the effects of α in Fig. 1. κ has a strong effect on the difficulty of the problem. Consider the case when $k = 2$. When $\kappa = 1$, the only potential $\mathbf{Q}_{Y|D}$ matrices are $\mathbf{I}_2$ up to row permutation (which means that domains and classes are exactly correlated, so the domain indicates the class and the problem is supervised). In the other limit, if $\kappa \to +\infty$, we may generate $\mathbf{Q}_{Y|D}$ matrices that are singular, breaking needed assumptions for domain discriminator output to uniquely identify true class of anchor subdomains. κ also helps control the class imbalance (if a row of $\mathbf{Q}_{Y|D}$ is small, indicating that the class is heavily under-represented across all domains, the condition number will increase).

F.1. FieldGuide-2 and FieldGuide-28 Datasets

The dataset and description is available at https://sites.google.com/view/fgvc6/competitions/butterflies-moths-2019. From this data we create two datasets FieldGuide-2 and FieldGuide-28. For FieldGuide-28 we select the 28 classes which have 1000 datapoints in the training file. Since the test set provided in the website does not have annotations, we manually create a test set by sampling 200 datapoints from training file of each of the 28 classes. Therefore, we finally have 22400 trainging points and 5600 testing points. The FieldGuide-2 dataset is created by considering two classes from the created FieldGuide-28 dataset.

F.2. Hyperparameters and Implementation Details: SCAN baseline

We select the unsupervised classification method SCAN as a state-of-the-art baseline (Van Gansbeke et al., 2020). SCAN pretrains a ResNet (He et al., 2016) backbone using SimCLR (Chen et al., 2020) and MoCo (He et al., 2020) setups (pretext tasks). SCAN then trains a clustering head to minimize the SCAN loss. SCAN code is available at https://github.com/wvangansbeke/Unsupervised-Classification. To evaluate SCAN, we use the public pretrained weights available for CIFAR-10, CIFAR-20, and ImageNet-50. We also train SCAN ourselves on the train and

validation portions of the FieldGuide2 and FieldGuide28 datasets with a ResNet18 backbone and SimCLR pretext task. We replicate the hyperparameters used for CIFAR training.

We make sure to evaluate SCAN classification on the same potentially class-imbalanced test subset we create for each experiment. Since SCAN is fit on a superset of the data DDFA sees, we believe this gives a slight data advantage to the SCAN baseline (although we acknowledge that the class balance for SCAN training is also potentially different from its evaluation class balance).

With the Hungarian algorithm, implemented in (Crouse, 2016; Virtanen et al., 2020), we compute the highest true accuracy among any permutation of predicted class labels (denoted "Test acc").

- **CIFAR-10 and CIFAR-20 Datasets (Krizhevsky & Hinton, 2009)**

  We use ResNet-18 (He et al., 2016) backbone with weights trained by SCAN-loss and obtained from the SCAN repo `https://github.com/wvangansbeke/Unsupervised-Classification`.

  We use the same transforms present in the repo for test data.

- **ImageNet-50 Dataset (Deng et al., 2009)**

  We use ResNet-50 backbone with weights trained by SCAN-loss and obtained from the SCAN repo.

  We use the same transforms present in the repo for test data.

- **FieldGuide-2 and FieldGuide-28 Datasets**

  For each of the two datasets, we pretrain a different SCAN baseline network (including pretext and SCAN-loss steps) on all available data from the dataset. The backbone for each is ResNet-18.

  For training the pretext task, we use the same transform strategy used in the repo for CIFAR-10 data (with mean and std values as computed on the Fieldguide-28 dataset, and crop size 224). For training SCAN, we resize the smallest image dimension to 256, perform a random horizontal flip and random crop to size 224. We also normalize. For validation we resize smallest image dimension to 256, center crop to 224, and normalize.

F.3. Hyperparameters and Implementation Details: DDFA (RI)

This is the DDFA procedure with random initialization.

We train ResNet-50 (He et al., 2016) (with random initialization and added dropout) based on the implementation from `https://github.com/kuangliu/pytorch-cifar` on images x_i with domain indices d_i as the label, choose best iteration by valid loss, pass all training and validation data through $\widehat{f}$, and cluster pushforward predictions $\widehat{f}(x_i)$ into $m \geqslant k$ clusters with Faiss K-Means (Johnson et al., 2019). We compute the $\widehat{\mathbf{Q}}_{c(X)|D}$ matrix and run NMF to obtain $\widehat{\mathbf{Q}}_{c(X)|Y}$, $\widehat{\mathbf{Q}}_{Y|D}$. To make columns sum to 1, we normalize columns of $\widehat{\mathbf{Q}}_{c(X)|Y}$, multiply each column's normalization coefficient over the corresponding row of $\widehat{\mathbf{Q}}_{Y|D}$ (to preserve correctness of the decomposition), and then normalize columns of $\widehat{\mathbf{Q}}_{Y|D}$.

Some NMF algorithms only output solutions satisfying the anchor word property (Arora et al., 2012a; Kumar et al., 2013; Gillis & Vavasis, 2014). We found the strict requirement of an exact anchor word solution to lead to low noise tolerance. We therefore use the Sklearn implementation of standard NMF (Cichocki & Phan, 2009; Pedregosa et al., 2011; Tan & Févotte, 2012).

We predict class labels with Algorithm 2. With the Hungarian algorithm, implemented in (Crouse, 2016; Virtanen et al., 2020), we compute the highest true accuracy among any permutation of these labels (denoted "Test acc"). With the same permutation, we reorder rows of $\widehat{P}_{Y|D}$, then compute the average absolute difference between corresponding entries of $\widehat{\mathbf{Q}}_{Y|D}$ and $\mathbf{Q}_{Y|D}$ (denoted "$\mathbf{Q}_{Y|D}$ err").

In order to make hyperparameter choices for final experiments, such as the choice of the NMF solver, clustering algorithm, and learning rate, we consulted CIFAR-10 and CINIC-10 (similar to an extension of CIFAR-10) (Darlow et al., 2018) final test task accuracy. We were also able to confirm our intuition that minimum validation loss on domain discriminator training corresponds to good final task performance. Final evaluation runs on CIFAR-10 were made with the best hyperparameters found here. We acknowledge that this may lead to test-set overfitting on CIFAR-10, but point out that on all other datasets, we consulted only validation loss on domain discriminator training when adjusting the hyperparameters. Final evaluation runs used the following fixed hyperparameters:

Common Hyperparameters

Architecture: ResNet-50 with added dropout

Faiss KMeans number of iterations (niter): 100

Faiss Kmeans number of clustering redos (nredo): 5

Learning Rate: 0.001

Learning Rate Decay: Exponential, parameter 0.97

SKlearn NMF initialization: random

Dataset-Specific Hyperparameters

- **CIFAR-10 Dataset**
  Training Epochs: 100
  Number of Clusters (m): 30
- **CIFAR-20 Dataset**
  Training Epochs: 100
  Number of Clusters (m): 60
- **ImageNet-50 Dataset**
  Not evaluated.
- **FieldGuide-2 and FieldGuide-28 Datasets**
  Not evaluated.

F.4. Hyperparameters and Implementation Details: DDFA (SI) and DDFA (SPI)

In Sec. 5, we used DDFA in the results section as a shorthand for DDFA (SPI). This is the DDFA procedure with SCAN initialization.

The procedure is identical to the standard DDFA procedure, except that SCAN (Van Gansbeke et al., 2020) pre-trained weights or SCAN (Van Gansbeke et al., 2020) contrastive pre-text weights are used to initialize the domain discriminator before it is fine-tuned on the domain discrimination task. Hyperparameters used also differ.

When SCAN pretrained weights are available, we use those. When they are not, we train SCAN ourselves.

Much like SCAN (RI), we used CIFAR-10 final test acuracy to choose hyperparameters and make algorithm decisions. For all other datasets, we consulted only validation domain discrimination loss. Final evaluation runs used the following fixed hyperparameters:

Common Hyperparameters

Faiss KMeans number of iterations (niter): 100

Faiss Kmeans number of clustering redos (nredo): 5

Learning Rate: 0.00001

Learning Rate Decay: Exponential, parameter 0.97

SKlearn NMF initialization: random

Dataset-Specific Hyperparameters

- **CIFAR-10 Dataset**
  Architecture: ResNet-18

Pre-seed: Weights trained with SCAN pretext and SCAN-loss on entirety of CIFAR-10 (from SCAN repo).
Training Epochs: 25
Number of Clusters (m): 10
Transforms used: Same as SCAN repo.

- **CIFAR-20 Dataset**
  Architecture: ResNet-18
  Pre-seed: Weights trained with SCAN pretext and SCAN-loss on entirety of CIFAR-20 (from SCAN repo).
  Training Epochs: 25
  Number of Clusters (m): 20
  Transforms used: Same as SCAN repo.
- **ImageNet-50 Dataset**
  Architecture: ResNet-50
  Pre-seed: Weights trained with SCAN pretext and SCAN-loss on entirety of ImageNet-50 (from SCAN repo).
  Training Epochs: 25
  Number of Clusters (m): 50
  Transforms used: Same as SCAN repo.
- **FieldGuide-2 Dataset**
  Architecture: ResNet-18
  Pre-seed: Weights trained with SCAN pretext on entirety of FieldGuide-2 (trained by us).
  Training Epochs: 30
  Number of Clusters (m): 2
  Transforms used for pretext: Same strategy as CIFAR-10 in SCAN repo with appropriate mean, std, and crop size 224.
  Transform used for SCAN: Resize to 256, Random horizontal flip, Random crop to 224, normalize
  Learning rate used for SCAN: 0.001 (other hyperparameters were same as in SCAN repo for CIFAR-10)
- **FieldGuide-28 Dataset**
  Architecture: ResNet-18
  Pre-seed: Weights trained with SCAN pretext on entirety of FieldGuide-28 (trained by us).
  Training Epochs: 60
  Number of Clusters (m): 28
  Transforms used for pretext: Same strategy as CIFAR-10 in SCAN repo with appropriate mean, std, and crop size 224.
  Transform used for SCAN: Resize to 256, Random horizontal flip, Random crop to 224, normalize
  Learning rate used for SCAN: 0.01 (other hyperparameters were same as in SCAN repo for CIFAR-10)

G. Additional Experimental Results

Table 2. Results on CIFAR-10. Each entry is produced with the averaged result of 3 different random seeds. With DDFA (RI) we refer to DDFA with randomly initialized backbone. With DDFA (SI) we refer to DDFA's backbone initialized with SCAN. Note that in DDFA (SI), we do not leverage SCAN for clustering. α is the Dirichlet parameter used for generating label marginals in each domain, κ is the maximum allowed condition number of the generated $\mathbf{Q}_{Y|D}$ matrix, r is number of domains.

r	Approaches	$\alpha : 0.5,\ \kappa : 4$		$\alpha : 3,\ \kappa : 4$		$\alpha : 10,\ \kappa : 8$	
		Test acc	$\mathbf{Q}_{Y\|D}$ err	Test acc	$\mathbf{Q}_{Y\|D}$ err	Test acc	$\mathbf{Q}_{Y\|D}$ err
10	SCAN	0.8205	-	**0.8222**	-	**0.8025**	-
	DDFA (RI)	0.7361	0.0354	0.5393	0.0481	0.3135	0.0736
	DDFA (SI)	**0.8987**	**0.0233**	0.7566	**0.0401**	0.5359	**0.0542**
15	SCAN	0.8245	-	0.8172	-	**0.8107**	-
	DDFA (RI)	0.7732	0.0326	0.5319	0.0464	0.275	0.0739
	DDFA (SI)	**0.9605**	**0.0163**	**0.8443**	**0.0256**	0.7327	**0.0376**
20	SCAN	0.8048	-	0.8109	-	**0.8164**	-
	DDFA (RI)	0.6883	0.0471	0.5651	0.0461	0.2704	0.0705
	DDFA (SI)	**0.9656**	**0.0158**	**0.904**	**0.0193**	0.7979	**0.0298**
25	SCAN	0.7974	-	0.8131	-	0.8094	-
	DDFA (RI)	0.7235	0.0389	0.5616	0.0444	0.2802	0.0863
	DDFA (SI)	**0.9701**	**0.0132**	**0.9173**	**0.0166**	**0.8204**	**0.027**

Table 3. Results on CIFAR-20. Each entry is produced with the averaged result of 3 different random seeds. With DDFA (RI) we refer to DDFA with randomly initialized backbone. With DDFA (SI) we refer to DDFA's backbone initialized with SCAN. Note that in DDFA (SI), we do not leverage SCAN for clustering. α is the Dirichlet parameter used for generating label marginals in each domain, κ is the maximum allowed condition number of the generated $\mathbf{Q}_{Y|D}$ matrix, r is number of domains.

r	Approaches	$\alpha : 0.5,\ \kappa : 8$		$\alpha : 3,\ \kappa : 12$		$\alpha : 10,\ \kappa : 20$	
		Test acc	$\mathbf{Q}_{Y\|D}$ err	Test acc	$\mathbf{Q}_{Y\|D}$ err	Test acc	$\mathbf{Q}_{Y\|D}$ err
20	SCAN	0.4416	-	0.4447	-	**0.433**	-
	DDFA (RI)	0.517	0.0423	0.3355	0.0451	0.1632	0.0568
	DDFA (SI)	**0.7838**	**0.0225**	**0.5926**	**0.0271**	0.3904	**0.0344**
25	SCAN	0.4403	-	0.4439	-	0.4398	-
	DDFA (RI)	0.4886	0.0487	0.292	0.0485	0.0753	0.0806
	DDFA (SI)	**0.8372**	**0.0204**	**0.6685**	**0.0253**	**0.4869**	**0.0299**
30	SCAN	0.4317	-	0.4611	-	0.4327	-
	DDFA (RI)	0.5121	0.0462	0.2992	0.0477	0.0869	0.0765
	DDFA (SI)	**0.8197**	**0.0219**	**0.7425**	**0.0207**	**0.5434**	**0.0283**

Table 4. Results on ImageNet-50. Each entry is produced with the averaged result of 3 different random seeds. With DDFA (SI) we refer to DDFA's backbone initialized with SCAN. Note that in DDFA (SI), we do not leverage SCAN for clustering. α is the Dirichlet parameter used for generating label marginals in each domain, κ is the maximum allowed condition number of the generated $\mathbf{Q}_{Y|D}$ matrix, r is number of domains.

r	Approaches	$\alpha : 0.5,\ \kappa : 200$		$\alpha : 3,\ \kappa : 205$		$\alpha : 10,\ \kappa : 210$	
		Test acc	$\mathbf{Q}_{Y\|D}$ err	Test acc	$\mathbf{Q}_{Y\|D}$ err	Test acc	$\mathbf{Q}_{Y\|D}$ err
50	SCAN	**0.7366**	-	**0.754**	-	**0.7372**	-
	DDFA (SI)	0.7204	0.0132	0.6322	0.0149	0.3431	0.0217
60	SCAN	0.757	-	0.7344	-	**0.7315**	-
	DDFA (SI)	**0.8179**	0.0101	**0.7434**	0.0124	0.5784	0.0175

Table 5. Results on FieldGuide-2. Each entry is produced with the averaged result of 3 different random seeds. With DDFA (SPI) we refer to DDFA initialized with pretext training adopted by SCAN. Note that in DDFA (SPI), we do not leverage SCAN for clustering. α is the Dirichlet parameter used for generating label marginals in each domain, κ is the maximum allowed condition number of the generated $\mathbf{Q}_{Y|D}$ matrix, r is number of domains.

r	Approaches	$\alpha : 0.5,\ \kappa : 3$		$\alpha : 3,\ \kappa : 5$		$\alpha : 10,\ \kappa : 7$	
		Test acc	$\mathbf{Q}_{Y\|D}$ err	Test acc	$\mathbf{Q}_{Y\|D}$ err	Test acc	$\mathbf{Q}_{Y\|D}$ err
2	SCAN	0.5784	-	0.5635	-	0.5731	-
	DDFA (SPI)	**0.7757**	0.2411	**0.7729**	0.1497	**0.6578**	0.264
3	SCAN	0.598	-	0.5889	-	0.5935	-
	DDFA (SPI)	**0.9599**	0.0545	**0.8304**	0.1478	**0.6933**	0.2238
5	SCAN	0.6004	-	0.5767	-	0.5792	-
	DDFA (SPI)	**0.9534**	0.0934	**0.7835**	0.1406	**0.6166**	0.2575
7	SCAN	0.5932	-	0.5923	-	0.5861	-
	DDFA (SPI)	**0.9037**	0.1151	**0.8157**	0.1448	**0.6608**	0.1975
10	SCAN	0.5831	-	0.5848	-	**0.582**	-
	DDFA (SPI)	**0.9071**	0.1547	**0.7139**	0.1698	**0.582**	0.164

Table 6. Extended Results on FieldGuide-28. Each entry is produced with the result of a single trial. With DDFA (SPI) we refer to DDFA initialized with pretext training adopted by SCAN. Note that in DDFA (SPI), we do not leverage SCAN for clustering. α is the Dirichlet parameter used for generating label marginals in each domain, κ is the maximum allowed condition number of the generated $\mathbf{Q}_{Y|D}$ matrix, r is number of domains.

r	Approaches	$\alpha : 0.5,\ \kappa : 12$		$\alpha : 3,\ \kappa : 20$		$\alpha : 10,\ \kappa : 28$	
		Test acc	$\mathbf{Q}_{Y\|D}$ err	Test acc	$\mathbf{Q}_{Y\|D}$ err	Test acc	$\mathbf{Q}_{Y\|D}$ err
28	SCAN	0.2882	-	0.2915	-	**0.314**	-
	DDFA (SPI)	**0.5472**	0.0359	**0.3102**	0.0338	0.3136	0.0355
37	SCAN	0.295	-	0.3126	-	0.3145	-
	DDFA (SPI)	**0.7595**	0.0276	**0.5214**	0.0318	**0.3257**	0.0409
42	SCAN	0.2856	-	0.3177	-	0.2883	-
	DDFA (SPI)	**0.6699**	0.0317	**0.4714**	0.0373	**0.4082**	0.0311
47	SCAN	0.2807	-	0.3184	-	**0.3044**	-
	DDFA (SPI)	**0.7087**	0.0352	**0.4728**	0.0353	0.2992	0.0394

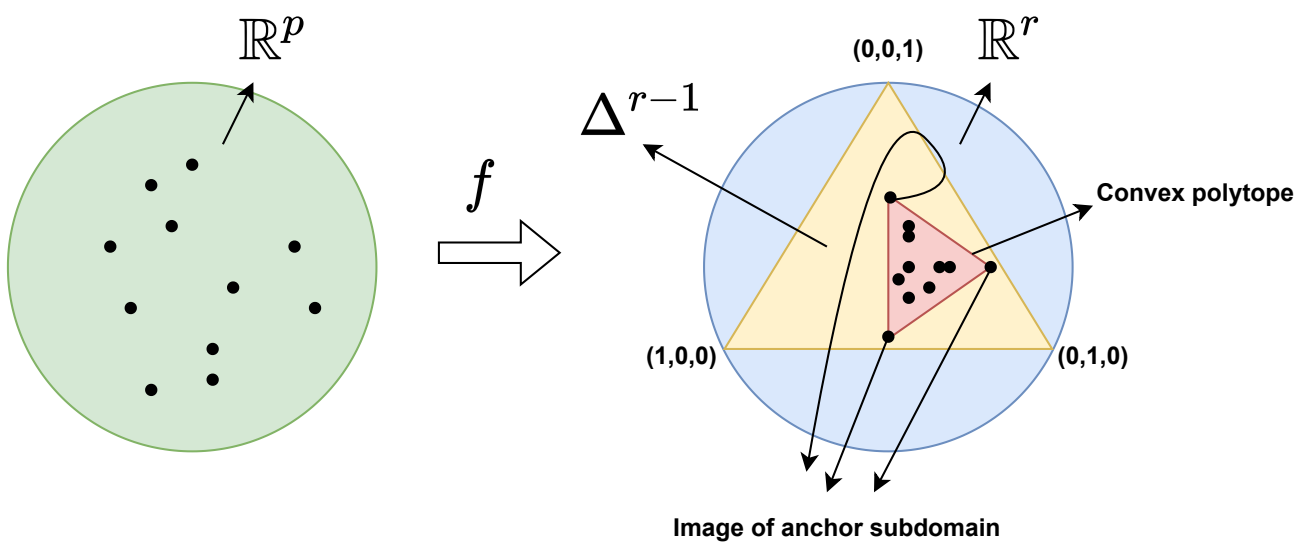

Figure 2. This figure illustrates the case with 3 domains and 3 classes. The oracle domain discriminator maps points from a high-dimensional input space to a $k = 3$ vertex convex polytope (shaded red) embedded in $\Delta^{r-1}, r = 3$ (shaded yellow). The anchor subdomains map to the vertices of this polytope.

H. Discussion of Convex Polytope Geometry

The geometric properties of topic modeling for finite, discrete random variables has been explored in depth in related works ((Donoho & Stodden, 2003; Huang et al., 2016; Chen et al., 2021)). The observation that columns in $\mathbf{Q}_{X|D}$ are convex combinations of columns in $\mathbf{Q}_{X|Y}$ leads to a perspective on identification of the matrix decomposition as identification of the convex polytope in $\mathbb{R}^m$ which encloses all of the columns of $\mathbf{Q}_{X|D}$ (the corners of which correspond to columns of $\mathbf{Q}_{X|Y}$ under certain identifiability conditions).

Here, we briefly discuss an interesting but somewhat different application of convex polytope geometry. Instead of a convex polytope in $\mathbb{R}^m$ with corners as columns of $\mathbf{Q}_{X|Y}$, we concern ourselves with the convex polytope in $\mathbb{R}^r$ with corners as columns in $\mathbf{Q}_{D|Y}$, which must enclose all values taken by the oracle domain discriminator $f(x)$ for $x \in \mathcal{X}, q(x) > 0$.

Let us assume that Assumptions A.1–A.4 are satisfied. We recall the oracle domain discriminator f defined such that $[f(x)]_d = q(d|X = x)$. Let $x \in \mathcal{X} = \mathbb{R}^p$. Now, since the r values $q(d|X = x)$ for $d \in \{1, 2, ..., r\}$ together constitute a categorical distribution, each of these r values lie between 0 and 1, and also their sum adds to 1. Therefore the vector $f(x)$ lies on the simplex Δ^{r-1}. We now express $f(x)$ as a convex combination of the k columns of $\mathbf{Q}_{D|Y}$. We denote these column vectors $\mathbf{Q}_{D|Y}[:, y]$ for each $y \in \mathcal{Y} = [k]$. Note that each such vector also lies in the Δ^{r-1} simplex.

As an intermediate step in the proof of Lemma B.3 given in App. C, we showed that each $f(x)$ is a linear combination of these columns of $\mathbf{Q}_{D|Y}$ with coefficients $q(y|X = x)$ for all $y \in \mathcal{Y}$. That is, we can rewrite $f(x) = \mathbf{Q}_{D|Y} \left[Q(Y = 1|X = x) \ldots Q(Y = k|X = x)\right]^\top$

Since the coefficients in the linear combination are probabilities which, taken together, form a categorical distribution, they lie between 0 and 1 and sum to 1. Thus, for all $x \in \mathcal{X}$ with $q(x) > 0$, $f(x)$ can be expressed as a *convex* combination of the columns of $\mathbf{Q}_{D|Y}$. Therefore, for any x with $q(x) > 0$, $f(x)$ lies inside the $k-$vertex convex polytope with corners as the columns of $\mathbf{Q}_{D|Y}$ (which are linearly independent by Lemma C.3). This polytope is embedded in Δ^{r-1}.

Now consider x in an anchor sub-domain, that is $x \in A_y$ for some $y \in \mathcal{Y}$. We know that if $q(x) > 0$, $q(y|X = x) = 1$, $q(y'|X = x) = 0$ for all $y' \neq y$ (Lemma C.2). Since the $q(y|X = x)$ are now one-hot, we have that $f(x) = \mathbf{Q}_{D|Y}[:, y]$ for $x \in A_y$. In words, this means that $f(A_y)$ is precisely the yth column of $\mathbf{Q}_{D|Y}$. It follows that the domain discriminator maps each of the k anchor sub-domains exactly to a unique vertex of the polytope. The situation is described in Fig. 2.

We could now recover the columns of $\mathbf{Q}_{D|Y}$, up to permutation, with the following procedure:

1. Push all $x \in \mathcal{X}$ through f.
2. Find the minimum volume convex polytope that contains the resulting density of points on the simplex. The vectors that compose the vertices of this polytope are the columns of $\mathbf{Q}_{D|Y}$, up to permutation.

Note that from Assumption A.4, we are guaranteed to have a region of the input space with at least $\epsilon > 0$ mass that gets mapped to each of the vertices when carrying out step (i). Therefore, our discovered minimum volume polytope must enclose all of these vertices. Since no mass will exist outside of the true polytope, requiring a minimum volume polytope will ensure that the recovered polytope fits the true polytope's vertices precisely (as any extraneous volume outside of the true polytope must be eliminated). Then step (ii) recovers $\mathbf{Q}_{D|Y}$, up to permutation of columns. Having recovered $\mathbf{Q}_{D|Y}$, we can use Lemmas B.1 and B.2 to recover $q(y|x, d)$.

This procedure is a geometric alternative to the clustering approach outlined in Algorithm 1. In practice, fitting a convex hull around the outputs of a noisy, non-oracle estimated domain discriminator may be computationally expensive, and noise may lead this sensitive procedure fail to recover the true vertices.