
Dataset Dynamics via Gradient Flows in Probability Space

David Alvarez-Melis¹ Nicolò Fusi¹

Abstract

Various machine learning tasks, from generative modeling to domain adaptation, revolve around the concept of dataset transformation and manipulation. While various methods exist for transforming unlabeled datasets, principled methods to do so for *labeled* (e.g., classification) datasets are missing. In this work, we propose a novel framework for dataset transformation, which we cast as optimization over data-generating joint probability distributions. We approach this class of problems through Wasserstein gradient flows in probability space, and derive practical and efficient particle-based methods for a flexible but well-behaved class of objective functions. Through various experiments, we show that this framework can be used to impose constraints on classification datasets, adapt them for transfer learning, or to re-purpose fixed or black-box models to classify—with high accuracy—previously unseen datasets.

1. Introduction

One of the hallmarks of machine learning practice is the relative scarcity of domain- and task-specific data, compared to the abundance of general-purpose data. Thus, many problems in machine learning involve dataset manipulation or transformation to re-weight, sub-sample, augment, compress, or build generative models of data.

Recent work on these problems has focused on unlabeled datasets, either by considering only unsupervised learning settings or ignoring labels in supervised ones. Manipulation of labeled datasets, while less explored, is appealing for its various potential applications. On the one hand, it can unify, formalize, and cast under a new light various data manipulation heuristics core to state-of-the-art supervised learning pipelines, such as data augmentation (Simard et al., 2003; Krizhevsky et al., 2012; Cubuk et al.,

¹Microsoft Research. Correspondence to: David Alvarez-Melis <alvarez.melis@microsoft.com>.

2019), dataset pooling, or instance mixing (Zhang et al., 2018). On the other hand—and our principal motivation in this work—is the potential of such a framework to tackle less explored or entirely novel problems, such as ‘shaping’ datasets by imposing certain (e.g., geometric or privacy) constraints, or transforming them from one domain to the other to recast transfer learning as transfer of data to the domain of expertise of a model, rather than the other way around. Current data transformation methods are ill suited for these tasks, since they either operate exclusively on features or use labels sparingly and heuristically. In addition, most of them are tailored to specific problems and cannot be easily generalized. Thus, a unified framework for transforming and manipulating datasets is missing.

In response to this, in this work we propose a principled, flexible, and computationally feasible approach to *labeled* dataset transformation. Based on the motivating applications described above, we seek a framework that is model-independent, depending only on intrinsic geometric properties of the data, and applicable to any classification dataset, regardless of size, dimensionality, or number of classes. Our first step towards achieving this is to view datasets as samples from a joint distribution, and to focus on manipulating these distributions instead. Indeed, while a dataset might consist of finitely many samples, the precise number is seldom relevant and often unspecified (e.g., in streaming settings). Instead, the true object of interest is the generative process that gives rise to the dataset. Thus, we characterize a dataset as a collection of samples from an (unknown) joint probability distribution $\rho(x, y)$, and cast its transformation as an optimization problem in probability space $\mathcal{P}(\mathcal{X} \times \mathcal{Y})$. Formally, we seek to solve problems of the form $\min_{\rho \in \mathcal{P}(\mathcal{X} \times \mathcal{Y})} F(\rho)$, for some functional F over distributions that encodes the transformation of interest.

Formalizing and solving optimization problems over such—infinite dimensional, non-euclidean—probability spaces is conceptually challenging. In this work, we do so by means of *gradient flows*, a lynchpin of applied mathematics for modeling dynamics in very general infinite-dimensional spaces (Ambrosio et al., 2005), which have in fact been extensively studied in the context of metric measure spaces (Santambrogio, 2017), and have deep connections to partial differential equations (PDE) (Jordan et al., 1998). Gradient flows come with various appealing proper-

ties: they are inherently flexible in terms of the underlying space and types of dynamics they can model, they admit rigorous convergence analysis, and they produce—in addition to a final minimizing solution—a full trajectory of iterates, which are often useful too. It is no surprise then that gradient flows have recently become a popular tool to analyze and derive (parameter) optimization methods in machine learning (Javanmard et al., 2020; Chizat, 2019).

But harnessing gradient flows for labeled dataset optimization poses various challenges. First, a suitable representation of feature-label pairs and a meaningful metric between datasets must be defined. Then, one must find a class of functionals that are expressive enough to model interesting objective functions on datasets yet sufficiently well-behaved to enjoy theoretical guarantees and allow for tractable optimization. In response to the first of these challenges, we leverage a recently proposed notion of distance between datasets (Alvarez-Melis & Fusi, 2020) based on optimal transport (OT). Conceptually, this notion endows the space of joint (feature-label) distributions with a metric, allowing us to cast flows in this space in the more general setting of metric measure spaces described above. On a practical level, we can use this distance as a functional to define gradient flows that minimize it, i.e., to encode similarity to a given dataset as an optimization objective. Doing so requires generalizing this distance, which has previously only been used for *static* dataset comparison, and making it differentiable, yielding a loss function that we can efficiently optimize using automatic differentiation.

To address the second challenge—defining suitable functional objectives—we borrow a family of flexible functionals over measures that are well-studied in the gradient flow literature, show how they can be re-purposed to encode various handy dataset-related objectives, and discuss in detail how to make use of them within a particle-based approximation of the gradient flow that relies on automatic differentiation. As a result, we end up with a framework for labeled dataset optimization that is flexible, efficient, and has a solid theoretical foundation.

In our experiments, we first show how this framework for dataset transformation yields novel ways to approach transfer learning problems. For example, it can be used in combination with traditional model adaptation, as a means of data-preprocessing before fine-tuning (§7.2), but also *instead of it*, for settings where model adaptation is infeasible (e.g., frozen, black-box, or extremely large models). For such challenging settings, our flow-based methods allow us to ‘re-purpose’ already-trained models to classify previously unseen datasets with high accuracy (§7.3). We also show this framework can be used to generate datasets with various geometric constraints (§7.1), showing its promise as a principled tool for dataset synthesis.

2. Related Work

Task dynamics. Achille et al. (2018) propose a dynamic notion of distance between tasks based on Kolmogorov complexity, and interpret it as a stochastic differential equation. Their notion is defined on parameter space, and is driven by a model-dependent empirical risk minimization loss. Similarly, Wu & Fischer (2020) study phase transitions of the information bottleneck, but do so on a learnt representation space, and do not consider gradient flows.

Dataset adaptation with optimal transport. Most prior work that uses OT distances for comparing datasets operates exclusively on features (not labels) (Seguy et al., 2018), or relies on a classification loss for the latter (Courty et al., 2017; Damodaran et al., 2018), which assumes the two label sets are identical. In contrast, the notion of distance between datasets we use in this work, and therefore the resulting framework, does not. Furthermore, these works rely on the static formulation of OT, on a barycentric mapping to transform datasets, and consider only an OT distance as objective. Here instead we rely on more general functional objectives, which might contain an OT distance as one of their components, and use an explicit dynamic formulation of OT to transform the datasets.

Computational Wasserstein gradient flows. Recent work leverages gradient flows in Wasserstein space for machine learning (Liutkus et al., 2019; Liu et al., 2019; Arbel et al., 2019; Kolouri et al., 2019; Chizat & Bach, 2018; Chizat, 2019). Most of these consider flows on model (i.e., parameter) space or feature space, so they are not suitable for handling labeled dataset transformation.

Generative modeling. Generative modeling can be understood as optimization in probability space too: its goal is to find a parametrized distribution ρ_θ that minimizes some notion of dissimilarity to a true data distribution ρ_{data} , i.e., to minimize $F(\rho_\theta) = D(\rho_\theta, \rho_{\text{data}})$ for some discrepancy D . The typical setting assumes that samples from $\rho_{\text{data}} = \text{argmin } F(\rho)$ are available, and that the ultimate goal is to generate this distribution *exactly*. Here we consider more general settings, including those for which the optimal distribution ρ^* is unknown, defined only implicitly as the minimizer of some functional objective F . Within generative modeling, there is a flourishing line of work that learns the target distribution by means of *normalizing flows* (Dinh et al., 2014; 2017; Kingma & Dhariwal, 2018; Germain et al., 2015; Papamakarios et al., 2017), which are sequences of invertible transformations typically parametrized through neural networks. These are different from the *gradient flows* considered here, which are non-parametric, defined implicitly through a functional, and whose infinitesimal dynamics are well understood.

3. Technical Background

We begin by introducing the two key notions behind our framework: optimal transport and gradient flows.

3.1. Setting and Notation

Let \mathcal{X} be a Polish space equipped with metric d , and $\mathcal{P}(\mathcal{X})$ the set of non-negative Borel measures with finite second-order moment on that space. We consider both continuous and discrete measures, the latter represented as an empirical distribution: $\sum_{i=1}^N p_i \delta_{x_i}$, where δ_x is a Dirac at position $x \in \mathcal{X}$. When clear from the context, we use α to denote both the measure and its density ρ_α . For a measure α and measurable map $T : \mathcal{X} \rightarrow \mathcal{X}$, $T_\# \alpha$ denotes the push-forward measure of α under T . For a joint measure $\pi \in \mathcal{P}(\mathcal{X} \times \mathcal{X})$, we can express its marginals as $P_1 \# \pi$ and $P_2 \# \pi$ using the maps $P_1(x, x') = x$ and $P_2(x, x') = x'$. We focus on supervised learning, so we define a dataset as a collection of feature-label pairs $\{(x^{(i)}, y^{(i)})\}_i$, which we assume are sampled from some $\alpha \in \mathcal{P}(\mathcal{X} \times \mathcal{Y})$. We denote the dataset as D_α to make this dependence explicit. We will use the shorthand notations: $\mathcal{Z} \triangleq \mathcal{X} \times \mathcal{Y}$ and $z \triangleq (x, y)$. Finally, $\nabla \cdot$ denotes the divergence operator.

3.2. Optimal Transport

For measures $\alpha, \beta \in \mathcal{P}(\mathcal{X})$ and cost function $c : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}_+$, the optimal transport problem is

$$\text{OT}_c(\alpha, \beta) \triangleq \min_{\pi \in \Pi(\alpha, \beta)} \int c(x_1, x_2) d\pi(x_1, x_2), \quad (1)$$

where $\Pi(\alpha, \beta)$ is the set of transport plans between α and β , i.e., couplings with these two measures as marginals:

$$\Pi(\alpha, \beta) \triangleq \{\pi \in \mathcal{P}(\mathcal{X} \times \mathcal{X}) \mid P_1 \# \pi = \alpha, P_2 \# \pi = \beta\}. \quad (2)$$

When $c(x, y) = d(x, y)^p$ for $p \geq 1$, $\mathbf{W}_p \triangleq \text{OT}_c(\alpha, \beta)^{1/p}$ is called the p -Wasserstein distance. As its name implies, \mathbf{W}_p defines a *true* distance on $\mathcal{P}(\mathcal{X})$ (Villani, 2008). Thus, the latter equipped with the former is a metric space $\mathbb{W}_p(\mathcal{X}) = (\mathcal{P}(\mathcal{X}), \mathbf{W}_p)$, called the (p -)Wasserstein space. In practice, a regularized version of Problem (1), with an added entropy term $\lambda H(\pi)$, is often solved instead (Cuturi, 2013).

The dual formulation of problem (1) is:

$$\text{OT}_c(\alpha, \beta) = \sup_{\varphi \in \mathcal{C}(\mathcal{X})} \int_{\mathcal{X}} \varphi d\alpha + \int_{\mathcal{X}} \varphi^c d\beta, \quad (3)$$

where $\varphi : \mathcal{X} \rightarrow \mathbb{R}$ is known as the Kantorovich potential, and φ^c is its c -conjugate: $\varphi^c(x) = \inf_{x' \in \mathcal{X}} c(x, x') - \varphi(x)$. For $c(x, x') = \|x - x'\|^2$, φ^c is the Fenchel conjugate.

In seminal work, Benamou & Brenier (2000) showed that OT has a *dynamic* formulation too:

$$\mathbf{W}_p^p(\alpha, \beta) = \min_{\mu_t, V_t} \int_0^1 \int_{\mathcal{X}} \|V_t(x)\|^p d\mu_t(x) dt, \quad (4)$$

where the minimum is taken over measure-field pairs satisfying $\mu_0 = \alpha, \mu_1 = \beta$, and the continuity equation:

$$\partial_t \mu_t = -\nabla \cdot (\mu_t V_t). \quad (5)$$

This formulation amounts to finding, among *paths* of measures μ_t advecting from α to β and velocity fields V_t satisfying a conservation of mass constraint, the ‘shortest’ one, i.e., that which minimizes the path length (formally, the integral of the metric derivative). Thus, the dynamic formulation focuses on *local transfer* (via μ_t), compared to *global correspondence* (via π) in the static one (Equation (1)).

3.3. The Optimal Transport Dataset Distance

It is appealing to use OT to define a distance between datasets, but this is non-trivial for labeled datasets. The main issue is that problem (1) would require an element-wise metric d , which for labeled datasets means defining a distance between pairs of feature-label pairs. For the general case where \mathcal{Y} might be a discrete set (i.e., classification), this seems daunting. In recent work, Alvarez-Melis & Fusi (2020) propose a hybrid metric on this joint space that relies on representing the labels y as distributions over features α_y . E.g., for a digit classification dataset, α_1 would be a distribution over images with label $y = 1$. With this, they define a metric on \mathcal{Z} as $d_{\mathcal{Z}}(z, z')^p \triangleq d_{\mathcal{X}}(x, x')^p + W_p^p(\alpha_y, \alpha_{y'})$. Using $d_{\mathcal{Z}}$ as the ground cost in eq. (1) yields a distance between measures on $\mathcal{P}(\mathcal{Z})$, and therefore between datasets, which they refer to as the Optimal Transport Dataset Distance (OTDD):

$$\text{OTDD}(D_\alpha, D_\beta) \triangleq \left(\min_{\pi \in \Pi(\alpha, \beta)} \int_{\mathcal{Z} \times \mathcal{Z}} d_{\mathcal{Z}}(z, z') d\pi(z, z') \right)^{\frac{1}{p}}. \quad (6)$$

The main appeal of this distance is that it is defined even if the label sets of the two data sets are non-overlapping, or if there is no explicit known correspondence between them (e.g., digits to letters). It achieves this through a purely geometric treatment of features and labels. Another advantage is its computational scalability, which relies on using a Gaussian approximation on the per-label distributions, i.e., modeling each α_y as $\mathcal{N}(\mu_y, \Sigma_y)$, whose mean and covariance are estimated from samples. In that case, the distances $W_2^2(\alpha_y, \alpha_{y'})$ can be computed in closed form, so no optimization is needed to evaluate $d_{\mathcal{Z}}(z, z')$ inside problem (6).

3.4. Gradient Flows

Consider a functional $F : \mathcal{X} \rightarrow \mathbb{R}$ and a point $x_0 \in \mathcal{X}$. A *gradient flow* is an absolutely continuous curve $x(t)$ that evolves from x_0 in the direction of steepest descent of F . When \mathcal{X} is Hilbertian and F is sufficiently smooth, its gradient flow can be succinctly expressed as the solution of a differential equation $x'(t) = -\nabla F(x(t))$ with initial condition $x(0) = x_0$. Different discretizations of this equation yield popular gradient descent schemes, such as momentum and acceleration (Su et al., 2016; Wilson et al., 2016).

4. Dataset Dynamics through Optimization

Using the notation and concepts introduced in Section 3, we now formalize the motivating problem of dataset transformation. We do so first in general terms here, and discuss specific objective functions in the next section.

4.1. Functional Minimization via Gradient Flows

Given a dataset objective expressed as a functional $F : \mathcal{P}(\mathcal{Z}) \rightarrow \mathbb{R}$, we seek a joint measure $\rho \in \mathcal{P}(\mathcal{Z})$ realizing:

$$\min_{\rho \in \mathcal{P}(\mathcal{Z})} F(\rho) \quad (7)$$

We propose to approach this problem via gradient flows, i.e., by moving along a curve of steepest descent starting at ρ_0 until reaching a solution ρ^* . Unlike Euclidean settings, here the underlying space $\mathbb{W}_p(\mathcal{Z})$ is infinite-dimensional and non-Hilbertian, thus requiring stronger tools.

First, the notion of derivative can be extended to functionals on measures through the first variation, denoted by $\frac{\delta F}{\delta \rho}$ (and defined in detail in Appendix B). With this, we characterize the gradient flow $(\rho_t)_{t \geq 0}$ of F as the solution of:

$$\partial_t \rho_t = \nabla_W F(\rho_t) \triangleq \nabla \cdot \left(\rho_t \nabla \frac{\delta F}{\delta \rho}(\rho_t) \right), \quad (8)$$

which can also be seen as a continuity equation (4) for the measure ρ_t and the velocity field $-\nabla \frac{\delta F}{\delta \rho}(\rho_t)$.

Our main functional of interest will be the Wasserstein distance to a target distribution: $\mathcal{T}_\beta(\rho) \triangleq W_2(\rho, \beta)$, which we realize using the OTDD (Section 3.3). In addition, following the literature on gradient flows (Santambrogio, 2015; 2017), we consider three other functional families:

$$\mathcal{F}(\rho) = \int f(\rho(z)) dz \quad (9)$$

$$\mathcal{V}(\rho) = \int V(z) d\rho \quad (10)$$

$$\mathcal{W}(\rho) = \frac{1}{2} \iint W(z - z') d\rho(z) d\rho(z') \quad (11)$$

where $f : \mathbb{R} \rightarrow \mathbb{R}$ is convex and superlinear, $V, W : \mathcal{X} \rightarrow \mathbb{R}$ are convex and sufficiently smooth. These terms have a physical interpretation as internal, potential and interaction energies, respectively.

This choice of functionals obeys both theoretical and computational motivations. First, gradient flows on these functionals have provable convergence (Appendix A), which makes them appealing as optimization objectives. Second, their first variation is simple, making them amenable to gradient-based approaches and automatic differentiation, as discussed in Section 6.2. Yet, despite being tractable and well-behaved, these functionals are still sufficiently general to encode various useful objectives and constraints on datasets, as we show in Section 5.

Hence, for the remainder of this work we assume the objective of interest can be cast as:

$$F(\rho) = \mathcal{T}_\beta(\rho) + \mathcal{F}(\rho) + \mathcal{V}(\rho) + \mathcal{W}(\rho). \quad (12)$$

As mentioned before, the first variations of these four functionals are simple (Ambrosio et al., 2005):

$$\frac{\delta \mathcal{F}}{\delta \rho} = f'(\rho), \quad \frac{\delta \mathcal{V}}{\delta \rho} = V, \quad \frac{\delta \mathcal{W}}{\delta \rho} = W * \rho, \quad \frac{\delta \mathcal{T}_\beta}{\delta \rho} = \varphi_\rho, \quad (13)$$

where $*$ denotes the usual convolution operator between a measurable function and a measure, and φ_ρ is the optimal Kantorovich potential in the dual OT formulation (3). In light of this, the gradient flow (8) for functionals of the form (12) corresponds to the solution of:

$$\partial_t \rho = \nabla \cdot (\rho \nabla (f'(\rho) + V + W * \rho + \varphi_\rho)). \quad (14)$$

Appendix B provides a PDE interpretation of this flow.

4.2. Numerical Solution of Gradient Flows

The gradient flows defined above have a probability theory counterpart in terms of random variables. Consider a stochastic process $(Z_t)_t$, where each $Z_t \triangleq (X_t, Y_t)$ is a random variable with law $\rho_t \in \mathcal{P}(\mathcal{Z})$. Equation (8) is thus associated with a stochastic differential equation (SDE) on this random variable:

$$dZ_t = \phi(Z_t, \rho_t) dt, \quad Z_0 \sim \rho_0, \quad (15)$$

for $\phi(Z_t, \rho_t) = -\nabla \frac{\delta F}{\delta \rho}(\rho_t)(Z_t)$. This is a particular (diffusion-less) case of a McKean-Vlasov process (Kac, 1956; McKean, 1966). Eq. (15) can be interpreted as the trajectory of a single *particle*, with initial position drawn from ρ_0 , moving according to a potential function that captures its intrinsic dynamics and interactions with other particles, all driven by F . Two key observations are in order: (i) this *particle view* of the gradient flow lends itself to computational schemes that operate on finitely many samples, and (ii) the process in eq. (15) is defined on \mathcal{Z} (a finite-dimensional space here) rather than on the infinite-dimensional $\mathcal{P}(\mathcal{Z})$, making it amenable to computation.

Numerical approaches to solve SDEs like (15) require time discretization and finite-sample approximation. A simple way to achieve the former is with a forward Euler scheme:

$$Z_{t+1} = Z_t - \gamma \nabla F_{\alpha_t}(Z_t), \quad Z_0 \sim \alpha_0. \quad (16)$$

Computationally, this scheme is approximated by a system of particles that evolve simultaneously. Starting from $\rho_0 \approx \sum_{i=1}^N p_i \delta_{z_0^{(i)}}$, each particle $z^{(i)}$ is evolved according to the Euler scheme above, resulting in a system of n updates:

$$z_{t+1}^{(i)} = z_t^{(i)} - \gamma \nabla_{z_t^{(i)}} F(z_t^{(i)}) \quad \forall i = 1, \dots, n. \quad (17)$$

Then, ρ_t is approximated as $\rho_{N,t} = \sum_{i=1}^N p_i \delta_{z_t^{(i)}}$. For well-behaved functionals, this flow inherits all convexity and stability properties of the exact one, and $\rho_{N,t}(x)$ converges to $\rho_t(x)$ (Carrillo et al., 2019b).

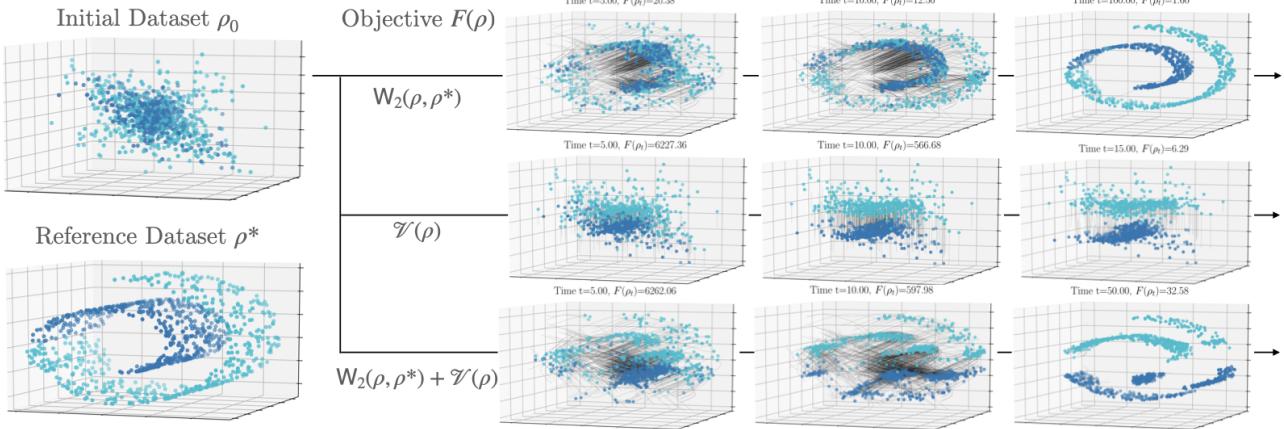


Figure 1. Shaping datasets via flows: our framework allows for simple and principled transformation of classification datasets by following the gradient flow of a functional objective, such as: similarity to a reference dataset (shown here as $W(\cdot, \rho^*)$, via OTDD), a function enforcing linear separability (shown here as $V(\rho)$), or a combination thereof.

5. Encoding Dataset Objectives as Functionals on Measures

Most of the applications motivating this work involve objectives that refer to one or more ‘reference’ datasets. In particular, we are interested in the functional $\mathcal{T}_\beta(\rho) \triangleq \text{OTDD}(\rho, \beta)$, where β is a (fixed) *target* dataset distribution. Various applications can be modeled with this functional, such as dataset interpolation or sample generation for transfer learning (§7.2). This objective can be further combined with some of the other functionals introduced in Section 3 to ‘shape’ the optimal distribution, balancing similarity to β with some other (e.g., size, regularity, or style) constraints imposed on it. This is often called *controlled synthesis* in generative modeling (Wang et al., 2018). Next, we discuss specific examples of dataset objectives encoded using each of the three canonical functionals.

Potential energy. Functionals of the form $V(\rho) = \int V(z) d\rho(z)$ can be used to enforce local (per-instance) constraints on the minimizer ρ^* . For example, one might be interested in constraining the norm of the features of this dataset by taking $V(z) = V(x, y) = \|x\|$, or more generally $V(z) = \|\mathbf{A}x - b\|$ for some $\mathbf{A} \in \mathbb{R}^{d \times d}$, $b \in \mathbb{R}^d$. Alternatively, one might be interested in enforcing such constraints only on certain classes, or have different parameters for every class, e.g., $V(z) = \|\mathbf{A}_y x - b_y\|$. Another possibility is to enforce linear separability in binary classification datasets through a margin-inducing potential such as $V(z) = \max\{0, y(x^\top w - b)\}$ (used in Fig. 1). Note that in all of these cases V is convex, as required for convergence.

Interaction energy. The functional $W(\rho) = \iint W(z - z') d\rho(z) d\rho(z')$ can encode objectives that model interaction or aggregation between samples in the dataset. An illustrative example of this is the class-repulsion functional

obtained by taking:

$$W(z - z') = \begin{cases} \exp\{-\|z - z'\|\} & \text{if } y \neq y' \\ 0 & \text{otherwise} \end{cases},$$

which encourages class separation by penalizing pairs (z, z') that have different labels but similar features. We show an example of a flow using this functional in a simple 2D dataset in Appendix D.

Internal energy. Compared to the previous three, the functionals $\mathcal{F}(\rho) = \int f(\rho(z)) dz$ are less relevant for our purposes, since an explicit density $\rho(z)$ will rarely be available in closed form. Furthermore, care must be taken with these functionals as they yield diffusion terms (e.g., $\Delta \rho$ in the associated PDE (14)) that, despite leading to theoretically well-behaved gradient flows, are challenging to solve numerically via particle methods (Carrillo et al., 2019b). A notable exception is the functional obtained by taking $f(t) = t \log t$, which corresponds to adding an entropy term to the objective (equivalently, a Brownian motion term in Equation (15)), and can be solved via particle methods, e.g., by using an Euler-Maruyama scheme. For more general functionals of this form, recent work circumvents this problem through stochastic (Huang & Liu, 2017; Liu & Yang, 2017), deterministic (Carrillo et al., 2017), and regularization-based schemes (Carrillo et al., 2019a). We leave their exploration for future work.

6. Practical Implementation

The particle scheme (17) provides a template for a gradient-based approach to solve the dataset optimization problem. In this section we discuss how the per-particle gradients are to be computed efficiently, provide practical implementation considerations for the functionals considered in this work, and discuss general computational aspects.

6.1. Dataset Distance

The main challenge in computing $\nabla_{z^{(i)}} \text{OTDD}(\mathbf{D}_{\rho_t}, \mathbf{D}_\beta)$ arises—unsurprisingly—from the discrete nature of the labels y . Recall that each point (particle) is a pair (x, y) , where $x \in \mathbb{R}^d$ and $y \in \{c_1, \dots, c_K\}$. The OTDD framework described in Section 3.3 provides an alternative representation of this point as (x, v_y) where $v_y \in \mathcal{P}(\mathcal{X})$ is a distribution over \mathcal{X} associated with the label y .

For small datasets the Gaussian approximation for OTDD (§3.3) is not needed, so the label-to-label distances $\mathbf{W}(v_y, v_{y'})$ are computed non-parametrically on-the-fly. In this case, only the features need to be updated, so it suffices to compute $\nabla_{x^{(i)}} \text{OTDD}(\mathbf{D}_{\rho_t}, \mathbf{D}_\beta)$. By using a differentiable solver for the inner and outer OT problems, we compute these gradients through automatic differentiation.

If, instead, we use the Gaussian approximation in the OTDD, there is now a parametric representation of the distributions $v_y = \mathcal{N}(\mu_y, \Sigma_y)$ that must be explicitly updated. A simple way to implement this is to backpropagate gradients into x^i ¹, take a gradient step on those, and then recompute means and covariances (and, as a result, the distributional representation of labels v_y). Formally,

$$\begin{aligned} x_{t+1}^i &= x_t^i - \tau \nabla_{x^i} F(\rho) & i \in \{1, \dots, n\} \\ \mu_{t+1}^j &= \text{mean}(\{x_{t+1}^i\}_{i: y^i=j}) & j \in \{1, \dots, k\} \\ \Sigma_{t+1}^j &= \text{cov}(\{x_{t+1}^i\}_{i: y^i=j}) & j \in \{1, \dots, k\} \\ v_{t+1}^j &= \mathcal{N}(\mu_{t+1}^j, \Sigma_{t+1}^j) & j \in \{1, \dots, k\} \end{aligned}$$

Note that here μ^j, Σ^j evolve in response to the dynamics of their associated empirical sample $\{x^i\}_i$, *but not directly because of a gradient step*. Thus, we refer to this type of flow as **feature-driven** (fd) dynamics. The main drawback of this approach is that the label assignments are fixed, i.e., if ρ_0 has k classes so will every ρ_t , and every x_t^i maintains its label throughout. This might be acceptable if \mathbf{D}_{ρ_0} and \mathbf{D}_β have the same number of classes.

In order to relax the approach above to allow direct updates on the label distributions v_y , one can instead apply independent updates on features and labels, while keeping the label assignments fixed throughout. This **joint-driven fixed-label** (jd-f1) scheme corresponds to:

$$\begin{aligned} x_{t+1}^i &= x_t^i - \tau \nabla_{x^i} F(\rho) & i \in \{1, \dots, n\} \\ \mu_{t+1}^j &= \mu_t^j - \tau \nabla_{\mu^j} F(\rho) & j \in \{1, \dots, k\} \\ \Sigma_{t+1}^j &= \Sigma_t^j - \tau \nabla_{\Sigma^j} F(\rho) & j \in \{1, \dots, k\} \\ v_{t+1}^j &= \mathcal{N}(\mu_{t+1}^j, \Sigma_{t+1}^j) & j \in \{1, \dots, k\} \end{aligned}$$

Relaxing the constraint that label assignments are fixed through the flow requires evolving the distribution associated to each point individually. To do so, at time $t = 0$ we

¹Here we drop parentheses in superscripts for convenience.

decouple the distributions v_j , yielding now one per particle instead of one per class. We then evolve each particle (both its feature and label-distribution) independently:

$$\begin{aligned} x_{t+1}^i &= x_t^i - \tau \nabla_{x^i} F(\rho) & i \in \{1, \dots, n\} \\ \mu_{t+1}^i &= \mu_t^i - \tau \nabla_{\mu^i} F(\rho) & i \in \{1, \dots, n\} \\ \Sigma_{t+1}^i &= \Sigma_t^i - \tau \nabla_{\Sigma^i} F(\rho) & i \in \{1, \dots, n\} \\ v_{t+1}^i &= \mathcal{N}(\mu_{t+1}^i, \Sigma_{t+1}^i) & i \in \{1, \dots, n\} \end{aligned}$$

However, recovering discrete labels from the (now decoupled) v_i^i requires aggregating particles based on these. We do so by clustering the pairs (μ_i, Σ_i) . Crucially, we use non-parametric clustering methods that *do not require* the number of clusters be specified, so that they can freely change throughout the flow. We refer to this third approach as **joint-driven variable-label** (jd-v1) dynamics.

Figure 2 shows a comparison of flows driven by two different dynamics on a simple 2D dataset. Additional qualitative comparison can be found in Appendix D.

6.2. Gradients of Energy Functionals

Recall that the first variation of potential functionals \mathcal{V} is given by $\frac{\delta \mathcal{V}}{\delta \rho} = V : \mathcal{Z} \rightarrow \mathbb{R}$, so it suffices to compute gradients of this scalar-valued function, which can be done using automatic differentiation. For interaction functionals, $\frac{\delta \mathcal{W}}{\delta \rho}(z) = (W * \rho)(z) = \int W(z - z') d\rho(z')$, which albeit scalar-valued, now involves an integral over \mathcal{Z} . We approximate it as an empirical expectation over the particles. Finally, for internal energy functionals $\frac{\delta \mathcal{F}}{\delta \rho} = f'(\rho)$, so if the density $\rho(\cdot)$ is available and can be computed with automatic differentiation, so can the gradient of $f' \circ \rho : \mathcal{Z} \rightarrow \mathbb{R}$.

6.3. Flowing Unlabeled Data

In many applications, such as in semi-supervised or unsupervised transfer learning, the data available to initialize the flow might be partially or completely unlabeled. However, so far we have assumed all particles are labeled, as required by the OTDD. Using label-dependent flows even for semi- or unsupervised settings might be desirable to explicitly model class-conditional geometric structure. Thus, we propose two approaches to adapt our flows to settings with limited or no labeled data.

Parametric Flows for Partially Labeled Data. If some labels are available, say, for the data samples indexed by \mathcal{L} , but not for those indexed by \mathcal{U} , one can first run a flow starting from $\{x^i, y^i\}_{i \in \mathcal{L}}$, to obtain a collection of final particles $\{x_T^i, y_T^i\}_{i \in \mathcal{L}}$. Next, the action of the flow $h_{\text{flow}} : x_0 \mapsto x_T$ can be parametrized (e.g., with a neural network) and learnt by fitting $\{x^i\}_{i \in \mathcal{L}}$ to $\{x_T^i\}_{i \in \mathcal{L}}$. This learnt mapping \hat{h} can then be used to flow the unlabeled samples, i.e., as $\hat{x}_T^i = \hat{h}(x^i), i \in \mathcal{U}$. We evaluate this approach for semi-supervised transfer learning in Section 7.2.

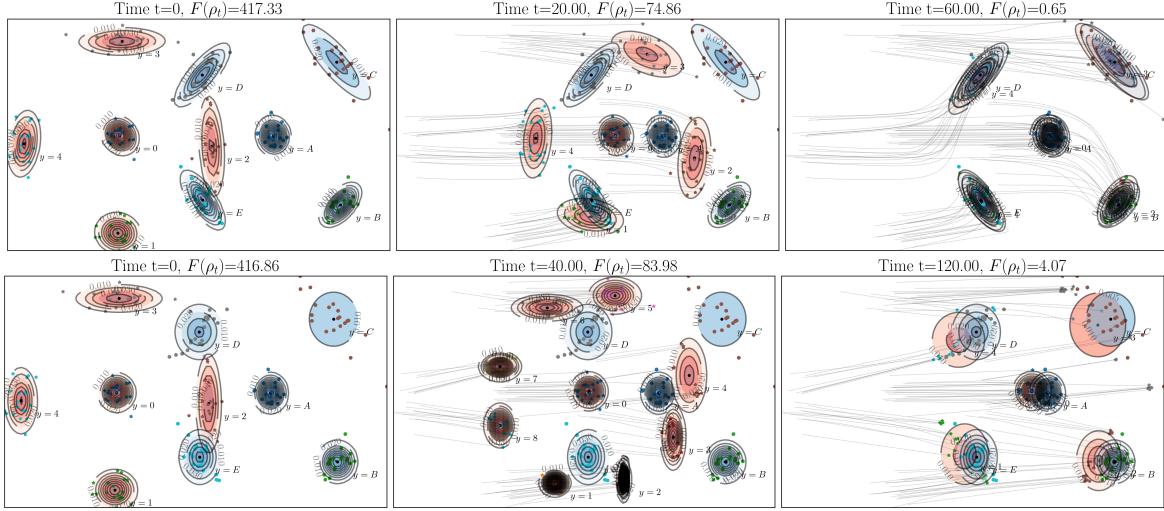


Figure 2. OTDD dynamics comparison: gradient flows driven by $\mathcal{T}_\beta(\rho) = \text{OTDD}(\mathbf{D}_\rho, \mathbf{D}_\beta)$ starting from dataset ρ_0 (red) advecting towards β (blue), using: SGD+jd-f1 (top) and SGD+jd-v1 dynamics (bottom), the latter allowing for variable class assignments.

Pseudo-Labels for Fully Unlabeled Data. If no labels are available at all, we propose to generate *pseudo-labels* for the flow. This can be done, for example, by clustering the initial data and using the identities of the clusters as imputed labels. We evaluate this approach in Section 7.3.

6.4. Miscellaneous Implementation Aspects

Traditional gradient flow implementations rely on constant or scheduled step-size schemes, i.e., vanilla gradient descent. In order to accelerate the flows and account for potential violations in the convexity assumptions of F , here we also consider various adaptive step-size methods like SGD with momentum and ADAM, which have been shown to aid in escaping local minima in deep learning.

In our experiments we will study the behavior of classifiers as we flow data between datasets, often of different dimensionality and with different classes. To deal with the former in image classification applications, we up- or down-sample as necessary to obtain equally-sized images — that is, to put them on the same \mathcal{X} space. A more general approach to deal with mismatch in dimensionality is discussed in (Alvarez-Melis & Fusi, 2020).

We deal with mismatch in classes by means of correspondence. For instance, in order to be able to use a classifier trained on a dataset with k classes on a flow originating in one with k' classes even when using fixed-label dynamics, we propose to take advantage of the label-to-label distances computed by the OTDD under the hood (see Section 3.3) to obtain correspondences between classes. Specifically, we solve an OT problem between the two class probability histograms using these distances, and append the $k' \times k$ optimal coupling π^* as a final linear layer to the classifier. Appendix C provides further computational details.

7. Experiments

We first evaluate our approach for imposing constraints on low-dimensional synthetic datasets (Section 7.1) and then on two settings (Sections 7.2 & 7.3) involving transfer learning with benchmark image classification datasets.

7.1. Imposing Dataset Constraints via Flows

For a synthetic 3D setting, we experiment with flows driven by functionals that combine distance to a reference dataset (here, a swiss roll) and various other functional objectives such as those described in Section 5. The results (Figure 1, bottom row and Figure 12 in the Appendix) show that the flows indeed converge to solutions that are geometrically similar to the reference dataset, but are ‘shaped’ by the additional constraints imposed by the functional objectives.

7.2. Transfer Learning via Flows

Next, we investigate the use of OTDD gradient flows for transfer learning. Specifically, starting from a source domain dataset \mathbf{D}_α , we transform it using a flow driven by the functional $\mathcal{T}_\beta(\rho) = \text{OTDD}(\mathbf{D}_\rho, \mathbf{D}_\beta)$ towards a target dataset of interest \mathbf{D}_β . We consider four classification datasets: MNIST (LeCun et al., 2010), USPS, FASHION-MNIST (Xiao et al., 2017) and KMNIST (Clanuwat et al., 2018), denoted here as M,U,F,K. Figures 3a-d show particle trajectories between these.

Since in most transfer learning settings the number of samples from the target domain is limited, we investigate if the —highly non-linear— action of a flow can be learnt from samples, so as to ‘flow’ additional unlabeled samples, as proposed in Section 6.3. For this purpose, we parametrize $h_{\text{flow}} : x_0 \mapsto x_T$ as a neural net, and fit it

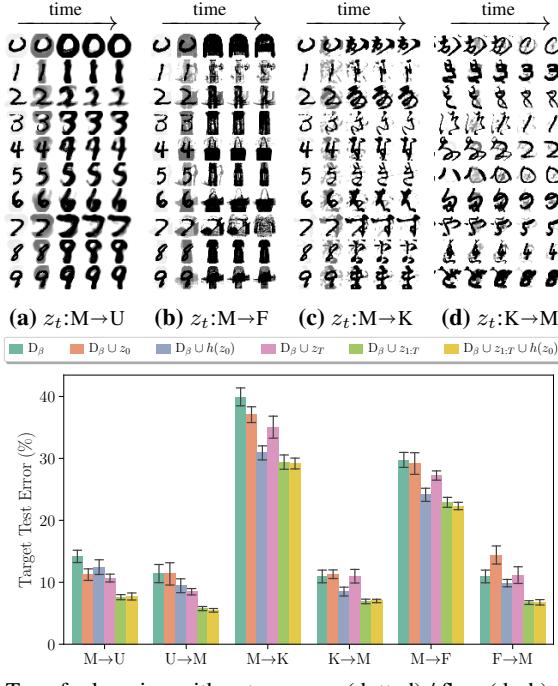


Figure 3. Transfer learning flows: flows between *NIST datasets using 2K particles (see §7.2 for dataset/legend key).

using the initial and final states of the particles ($\{x_0^i\}_{i=1}^n$ and $\{x_T^i\}_{i=1}^n$). Figure 9 (Appendix E) shows that particles mapped with the learnt h reasonably approximate those properly derived from the flow.

For evaluation, we emulate a k-shot learning task, i.e., only k samples per class from the target domain D_β are available, in addition to 2000 from the source domain. With this data, we flow the source samples to the target domain, so that we have at our disposal: the particles in their initial (z_0) and final (z_T) states, the full trajectories $z_{1:T}$, and additional mapped examples $h(z_0)$ using a neural-net push-forward map h as described above.

We compare various adaptation settings: a no-adaptation baseline (i.e., using target data D_β only), and adapting from various combinations of the additional data sources. Details of the classifier architecture and training configuration are provided in Appendix C. We present results for the 100-shot setting here (Figure 3e), and 5- and 10-shot in Appendix E. In most cases all additional data is beneficial, but the full trajectories provide the most gains, indeed improving over using final-state particles only. In addition, using samples mapped via h provides some improvement over the baseline, but less so than using ‘clean’ particles from the flow, as expected. When compared equitably (using the same number of labeled samples) against a usual model-adaptation (fine-tuning) baseline, our data-adaptation approach yields overall better performance gain (Figure 4).

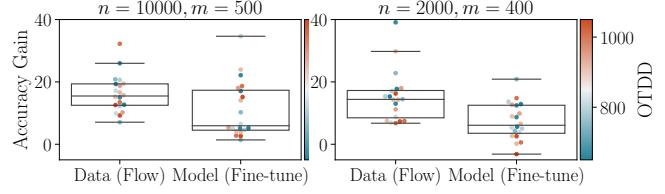


Figure 4. Data vs. model adaptation: both regimes use the same number of source (n) and target (m) labeled samples. Each point is a pair of source/target datasets, color-coded by their distance.

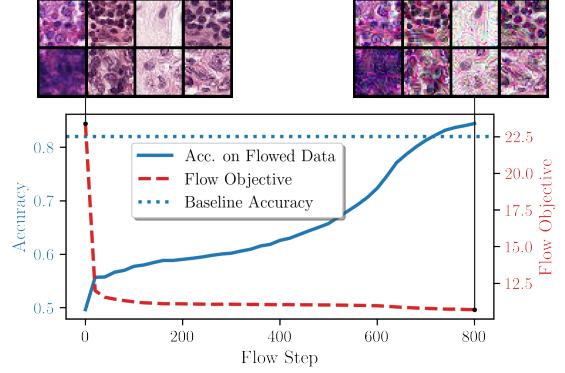


Figure 5. Model re-purposing: we use a model trained on CIFAR10, without any modifications, to classify CAMELYON by flowing the latter to the former. After flowing the data, this model beats a strong baseline model trained only on CAMELYON data.

7.3. Model Re-purposing and Oracle Evaluation

Next, we take the application of flows for transfer learning one step further, foregoing model adaption completely. We now assume the pretrained classifier is frozen and accessible through queries only, and use dataset flows to ‘re-purpose’ it to solve a different task. Concretely, we flow samples from the dataset we seek to classify towards that on which the model was trained, and then we query the model directly on this flowed data without further modifications. Besides providing a litmus test for no-fine-tuning transfer learning, this setting also provides a general strategy to quantitatively evaluate the ‘quality’ of a dataset flow, i.e., by using the model’s accuracy on the flowed data as an oracle measure of evolution of the flow.

In addition to the *NIST datasets, we use CIFAR10, STL10 and the CAMELYON histopathology dataset (Litjens et al., 2018). The results (Figures 6 & 11) show that the flows achieve high classification accuracy, even when the ORACLE is trained on a completely unrelated dataset (e.g., CAMELYON→CIFAR10). Notably, the entropy regularization λ used in OTDD has a much stronger effect on final accuracy than the step size. For the CAMELYON→CIFAR10 setting, Figure 5 shows a sample of flowed particles at different times. Although visible only through subtle artifacts in the images, flowing the data has a clear impact on the pretrained model’s accuracy.

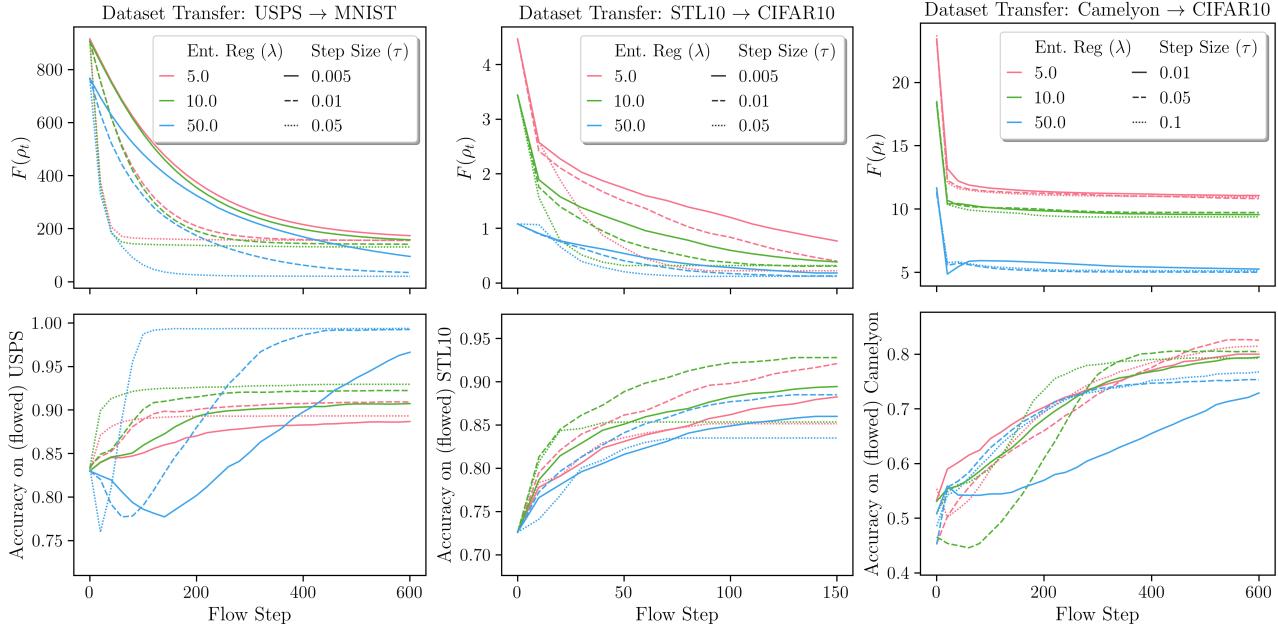


Figure 6. Oracle evaluation of flows: given a large classifier pretrained on a target domain, we use our method to ‘flow’ data from a target domain back to the source domain. The plots show the value of the flow objective and target accuracy throughout the flow.

Table 1. One model to classify them all: we use the exact same model trained on MNIST without modifications to classify samples from various datasets, by flowing all datasets to MNIST, using either our vanilla flow with an OTDD functional objective (supervised) or its unsupervised variant that uses pseudo-labels (§6.3). Shown here is the model’s accuracy (%) on 5K samples.

Dataset	USPS	KMNIST	FMNIST	EMNIST
Original (before flow)	77.1	5.05	8.22	3.62 [†]
Flowed (supervised)	99.1	99.2	99.1	36.7
Flowed (unsupervised)	66.4	59.2	54.9	20.7

[†]stacking a random 26×10 linear layer on top of the classifier.

But adapting datasets rather than models comes with another unexpected bonus: the exact same model can be used to classify *multiple* different datasets *simultaneously*, by flowing all of them to the domain on which the model was trained (stacking, if needed, a class-matching layer §6.4).

Here, we test this ‘model multi-purposing’ setting using a classifier trained on MNIST (the source domain), without modifications, to classify the other *NIST datasets (target domains). For this, we use the functional $\mathcal{T}(\rho) = \text{OTDD}(\rho, \rho_{\text{MNIST}})$ as before in two flow modalities: supervised (using labeled target samples for the flow) and unsupervised (no labels used) with pseudo-labels obtained by k-means clustering as described in Section 6.3. The former is an idealized—and unrealistic—scenario aimed at investigating the intrinsic limits of model-repurposing, and should

be interpreted as an upper bound on transfer accuracy, since labeled data from the target domain is indeed used by the flows, although not by the classifier. The latter, however, is a *bona-fide* method for unsupervised² model-repurposing, applicable in practice.

The results in Table 1 show that the frozen MNIST classifier achieves almost perfect accuracy on most of the other datasets after transforming them with a supervised flow. As expected, when using an unsupervised flow the accuracy is lower, but still significantly higher than if no flow-based data adaptation is performed, except for USPS, which the MNIST model can already classify well without adaptation.

8. Discussion and Future Work

We have shown how gradient flows in probability space can be used to tackle various types of problems in machine learning that involve transformation of labeled datasets. We believe this work lays out a path for a largely uncharted *data-centric* paradigm in machine learning, orthogonal and complementary to the prevailing model-centric one. Exciting directions of research within this paradigm abound. For example, we are interested in understanding meta-learning and related paradigms from a data-centric perspective, and to investigate whether other aspects of the deep learning toolkit (beyond adaptive gradient schemes) might prove useful in the context of dataset optimization.

²However, the identities of the pseudo-labels and true labels must be aligned, manually or automatically (see Appendix C).

References

- Achille, A., Mbeng, G., and Soatto, S. Dynamics and Reachability of Learning Tasks. pp. arXiv:1810.02440, October 2018.
- Alvarez-Melis, D. and Fusi, N. Geometric dataset distances via optimal transport. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 21428–21439. Curran Associates, Inc., 2020.
- Ambrosio, L., Gigli, N., and Savare, G. *Gradient flows in metric spaces and in the Wasserstein space of probability measures*. Lectures in Mathematics. ETH Zürich. Birkhäuser Basel, 2005. ISBN 9783764373092. doi: 10.1007/b137080.
- Arbel, M., Korba, A., Salim, A., and Gretton, A. Maximum mean discrepancy gradient flow. In e (ed.), *Advances in Neural Information Processing Systems 32*, pp. 6484–6494. Curran Associates, Inc., 2019.
- Benamou, J.-D. and Brenier, Y. A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem. *Numer. Math.*, 84(3):375–393, January 2000. ISSN 0029-599X, 0945-3245. doi: 10.1007/s002110050002.
- Carrillo, J. A., Huang, Y., Patacchini, F. S., and Wolansky, G. Numerical study of a particle method for gradient flows. *Kinet. Relat. Models*, 10, 2017. ISSN 1937-5093.
- Carrillo, J. A., Craig, K., and Patacchini, F. S. A blob method for diffusion. *Calc. Var. Partial Differ. Equ.*, 58(2):53, 2019a. ISSN 0944-2669, 1432-0835. doi: 10.1007/s00526-019-1486-3.
- Carrillo, J. A., Craig, K., and Yao, Y. Aggregation-Diffusion equations: Dynamics, asymptotics, and singular limits. In Bellomo, N., Degond, P., and Tadmor, E. (eds.), *Active Particles, Volume 2: Advances in Theory, Models, and Applications*, pp. 65–108. Springer International Publishing, Cham, 2019b. ISBN 9783030202972.
- Chizat, L. Sparse optimization on measures with over-parameterized gradient descent. 2019.
- Chizat, L. and Bach, F. On the global convergence of gradient descent for over-parameterized models using optimal transport. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 3036–3046. Curran Associates, Inc., 2018.
- Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., and Ha, D. Deep learning for classical Japanese literature. December 2018.
- Coates, A., Ng, A., and Lee, H. An analysis of Single-Layer networks in unsupervised feature learning. In Gordon, G., Dunson, D., and Dudík, M. (eds.), *Proceedings of the The 24th International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 215–223, Fort Lauderdale, FL, USA, 2011. JMLR Proceedings.
- Courty, N., Flamary, R., Habrard, A., and Rakotomamonjy, A. Joint distribution optimal transportation for domain adaptation. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 3730–3739. Curran Associates, Inc., 2017.
- Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. AutoAugment: Learning augmentation strategies from data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 113–123, 2019.
- Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 26*, pp. 2292–2300. Curran Associates, Inc., 2013.
- Damodaran, B. B., Kellenberger, B., Flamary, R., Tuia, D., and Courty, N. DeepJDOT: Deep joint distribution optimal transport for unsupervised domain adaptation. In *Computer Vision – ECCV 2018*, pp. 467–483. Springer International Publishing, 2018.
- Dinh, L., Krueger, D., and Bengio, Y. NICE: Non-linear independent components estimation. October 2014.
- Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real NVP. In *International Conference on Learning Representations*, 2017.
- Feydy, J., Séjourné, T., Vialard, F.-X., Amari, S.-I., Trouve, A., and Peyré, G. Interpolating between optimal transport and MMD using sinkhorn divergences. In Chaudhuri, K. and Sugiyama, M. (eds.), *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pp. 2681–2690. PMLR, 2019.
- Flamary, R., Courty, N., Gramfort, A., Alaya, M. Z., Boisbunon, A., Chambon, S., Chapel, L., Corenflos, A., Fazlalizadeh, K., Fournier, N., Gautheron, L., Gayraud, N. T., Janati, H., Rakotomamonjy, A., Redko, I., Rolet, A., Schutz, A., Seguy, V., Sutherland, D. J., Tavenard, R., Tong, A., and Vayer, T. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021.

- Germain, M., Gregor, K., Murray, I., and Larochelle, H. MADE: Masked autoencoder for distribution estimation. In Bach, F. and Blei, D. (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 881–889, Lille, France, 2015. PMLR.
- Huang, H. and Liu, J.-G. Error estimate of a random particle blob method for the Keller-Segel equation. *Math. Comput.*, 86(308):2719–2744, 2017.
- Hull, J. J. A database for handwritten text recognition research. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(5): 550–554, May 1994. ISSN 0162-8828, 1939-3539.
- Javanmard, A., Mondelli, M., and Montanari, A. Analysis of a two-layer neural network via displacement convexity. *Ann. Stat.*, 48(6):3619–3642, 2020. ISSN 0090-5364. doi: 10.1214/20-AOS1945.
- Jordan, R., Kinderlehrer, D., and Otto, F. The variational formulation of the Fokker–Planck equation. *SIAM J. Math. Anal.*, 29(1):1–17, January 1998. ISSN 0036-1410. doi: 10.1137/S0036141096303359.
- Kac, M. Foundations of kinetic theory. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability, Volume 3: Contributions to Astronomy and Physics*. The Regents of the University of California, 1956.
- Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In S. Bengio and H. Wallach and H. Larochelle and K. Grauman and N. Cesa-Bianchi and R. Garnett (ed.), *Advances in Neural Information Processing Systems 31*, pp. 10215–10224. Curran Associates, Inc., 2018.
- Kolouri, S., Nadjahi, K., Simsekli, U., Badeau, R., and Røhde, G. Generalized sliced wasserstein distances. In Wallach, H., Larochelle, H., Beygelzimer, A., d’Alché Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
- Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. 2009.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. Curran Associates, Inc., 2012.
- LeCun, Y., Cortes, C., and Burges, C. J. MNIST handwritten digit database. 2010.
- Litjens, G., Bandi, P., Ehteshami Bejnordi, B., Geessink, O., Balkenhol, M., Bult, P., Halilovic, A., HermSEN, M., van de Loo, R., Vogels, R., Manson, Q. F., Stathonikos, N., Baidoshvili, A., van Diest, P., Wauters, C., van Dijk, M., and van der Laak, J. 1399 H&E-stained sentinel lymph node sections of breast cancer patients: the CAMELYON dataset. *Gigascience*, 7(6), May 2018.
- Liu, C., Zhuo, J., Cheng, P., Zhang, R., and Zhu, J. Understanding and accelerating Particle-Based variational inference. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4082–4092. PMLR, 2019.
- Liu, J.-G. and Yang, R. A random particle blob method for the Keller-Segel equation and convergence analysis. *Math. Comput.*, 86(304):725–745, 2017.
- Liutkus, A., Simsekli, U., Majewski, S., Durmus, A., and Stöter, F.-R. Sliced-Wasserstein flows: Nonparametric generative modeling via optimal transport and diffusions. In Chaudhuri, Kamalika And (ed.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4104–4113, Long Beach, California, USA, 2019. PMLR.
- McKean, H. P. A class of markov processes associated with nonlinear parabolic equations. *Proc. Natl. Acad. Sci. U. S. A.*, 56(6):1907–1911, December 1966. ISSN 0027-8424. doi: 10.1073/pnas.56.6.1907.
- Otto, F. The geometry of dissipative evolution equations: the porous medium equation. *Comm. Partial Differential Equations*, 26(1-2):101–174, January 2001. ISSN 0360-5302. doi: 10.1081/PDE-100002243.
- Papamakarios, G., Pavlakou, T., and Murray, I. Masked autoregressive flow for density estimation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS’17, pp. 2335–2344, USA, December 2017. Curran Associates Inc. ISBN 9781510860964.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d’Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

Santambrogio, F. *Optimal Transport for Applied Mathematicians: Calculus of Variations, PDEs, and Modeling*. Birkhäuser, Cham, 2015. ISBN 9783319208275.

Santambrogio, F. {Euclidean, metric, and Wasserstein} gradient flows: an overview. *Bull. Math. Sci.*, 7(1):87–154, April 2017. ISSN 1664-3607, 1664-3615.

Seguy, V., Damodaran, B. B., Flamary, R., Courty, N., Rolet, A., and Blondel, M. Large-Scale optimal transport and mapping estimation. In *International Conference on Learning Representations*, 2018.

Simard, P. Y., Steinkraus, D., and Platt, J. C. Best practices for convolutional neural networks applied to visual document analysis. In *Seventh International Conference on Document Analysis and Recognition (ICDAR)*, pp. 958–963. IEEE, 2003. doi: 10.1109/ICDAR.2003.1227801.

Su, W., Boyd, S., and Candès, E. J. A differential equation for modeling nesterov’s accelerated gradient method: Theory and insights. *J. Mach. Learn. Res.*, 17(153):1–43, 2016. ISSN 1532-4435, 1533-7928.

Villani, C. *Optimal transport, Old and New*, volume 338. Springer Science & Business Media, 2008. ISBN 9783540710493.

Wang, T.-C., Liu, M.-Y., Zhu, J.-Y., Tao, A., Kautz, J., and Catanzaro, B. High-Resolution image synthesis and semantic manipulation with conditional GANs. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8798–8807, June 2018. doi: 10.1109/CVPR.2018.00917.

Wilson, A. C., Recht, B., and Jordan, M. I. A lyapunov analysis of momentum methods in optimization. November 2016.

Wu, T. and Fischer, I. Phase transitions for the information bottleneck in representation learning. In *International Conference on Learning Representations*, 2020.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. August 2017.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

A. Brief Discussion of Convergence and Guarantees

Proving existence and uniqueness of solutions of gradient flows in Wasserstein space, or convergence of their discretized schemes, is challenging. But it can be achieved through different types of assumptions on the spaces, metrics, and functionals. Here, we will briefly discuss guarantees that depend on one of the simplest such assumptions: geodesic convexity.

Definition A.1. Let \mathcal{X} be a geodesic metric space and $F : \mathcal{X} \rightarrow \mathbb{R} \cup \{+\infty\}$ a functional. We say that F is (λ) -geodesically convex if it is (λ) -convex along geodesics in \mathcal{X} , i.e., for every pair of points $x_0, x_1 \in \mathcal{X}$, there exists a constant-speed geodesic ω connecting $\omega(0) = x_0$ to $\omega(1) = x_1$ such that $t \mapsto F(\omega(t)), t \in [0, 1]$ is (λ) -convex.

Note that if \mathcal{X} is Euclidean, the definition above is simply λ -convexity. On the other hand, this concept is well defined for metric measure spaces like $\mathbb{W}_p(\Omega)$ too. In particular, for $\mathbb{W}_2(\Omega)$, all geodesics are displacement geodesics, so the condition above is also known as *displacement convexity*.

Thus, a functional $F : \mathbb{W}_p(\mathcal{X}) \rightarrow \mathbb{R} \cup \{+\infty\}$ is λ -geodesically convex if and only if for every pair $\mu^1, \mu^2 \in \mathcal{P}(\mathcal{X})$ there exists an optimal transport coupling $\pi \in \Pi(\mu^1, \mu^2)$ such that

$$F(\mu_t^{1 \rightarrow 2}) \leq (1-t)F(\mu^1) + tF(\mu^2) - \frac{\lambda}{2}t(1-t)\mathbb{W}_p^2(\mu^1, \mu^2) \quad \forall t \in [0, 1] \quad (18)$$

where $\mu_t^{1 \rightarrow 2} \triangleq ((1-t)x + ty)_\sharp \pi$ is a geodesic in $\mathbb{W}_p(\mathcal{X})$ interpolating between μ^1 and μ^2 .

It can be shown that the functionals used in this work (i.e., those in Eq. (9)) are displacement convex under suitable conditions (Santambrogio, 2017). Specifically, \mathcal{V} and \mathcal{W} are λ -displacement convex if the underlying potentials V and W are λ -convex. For the internal energy functional \mathcal{F} , some technical assumptions on f are needed, such as requiring that $f(0) = 0$, $s \mapsto s^d f(s^{-d})$ is convex and decreasing, and the underlying space is convex (Santambrogio, 2017, Thm 7.28). It is easy to see that simple functions, such as the entropy term discussed before, or power functions with exponent $q > 1$, all satisfy this condition.

The following result, one of the simplest in such family of guarantees, shows the crucial importance of λ -geodesic convexity for establishing guarantees of gradient flows in Wasserstein space:

Proposition A.2 (Santambrogio 2017, Prop 4.13). Suppose that $F : \mathbb{W}_2(\mathcal{X}) \rightarrow \mathbb{R} \cup \{+\infty\}$ is λ -geodesically convex and that the two curves ρ_t^0 and ρ_t^1 are solutions of (8). Then, setting $\delta(t) = \frac{1}{2}\mathbb{W}_2^2(\rho_t^1, \rho_t^0)$, we have

$$\delta'(t) \leq -2\lambda d(t)$$

This implies uniqueness of the solution of (8) for a fixed initial state, stability and exponential convergence of the flow as $t \rightarrow +\infty$ if $\lambda > 0$.

Unfortunately—and somewhat unexpectedly—the functional $\mathcal{T}_\beta(\rho) = \mathbb{W}_2^2(\rho, \beta)$ turns out to be not displacement convex in general. However, it does satisfy an alternate and more general notion of convexity: along generalized geodesics.

Definition A.3. Let $\rho \in \mathcal{P}(\mathcal{X})$ be fixed. For every pair $\mu^1, \mu^2 \in \mathcal{P}(\mathcal{X})$, a **generalized geodesic** between them with base ρ in $\mathbb{W}_2(\mathcal{X})$ is given by the curve $\mu_t = ((1-t)T_0 + tT_1)_\sharp \rho$ where T_i is the optimal transport map (for the squared cost) from ρ to μ^i .

Thus, a functional $F : \mathbb{W}_p(\mathcal{X}) \rightarrow \mathbb{R} \cup \{+\infty\}$ is λ -geodesically convex along generalized geodesics if it satisfies condition (18) for generalized geodesics. Under the same assumptions as above, the functionals \mathcal{V} , \mathcal{W} , and \mathcal{F} are all convex along generalized geodesics too (Santambrogio, 2017; Ambrosio et al., 2005). But now, as hinted at before, so is $\mathcal{T}_\beta(\rho)$ if we choose β as the base point of the generalized geodesics (Santambrogio, 2015).

The notion of convexity along generalized geodesics can be used to establish results analogous to Proposition A.2 but which apply to more general functionals, including $\mathcal{T}_\beta(\rho)$. Such results usually involve appealing to a characterization of gradient flows known as the evolution variational inequality (EVI):

$$\frac{d}{dt} \frac{1}{2}d(\rho_t, \beta) \leq F(\beta) - F(\rho_t) - \frac{\lambda}{2}d(\mu_t, \beta)^2 \quad \forall \beta \in \mathcal{P}(\mathcal{X}) \quad (19)$$

Convexity along generalized geodesics can be used to prove the EVI conditions holds for a certain functional, which in turn implies uniqueness and stability of the flow. We refer the reader to (Santambrogio, 2017) for further details.

B. First Variations, Gradient Flows, and Connections to PDEs.

B.1. First variation of a functional

As mentioned in Section 4.1, having a notion of derivative of functionals over measures is a crucial step towards defining gradient flows in that space. The notion we rely on here is that of first variation of a functional (Santambrogio, 2017):

Definition B.1. *Given a functional $F : \mathcal{P}(\Omega) \rightarrow \mathbb{R}$, consider perturbations χ such that at least for every $\epsilon \in [0, \epsilon_0]$, $\rho + \epsilon\chi \in \mathcal{P}(\Omega)$. If there exists a function G such that*

$$\frac{d}{d\epsilon} F(\rho + \epsilon\chi) \Big|_{\epsilon=0} = \int G(\rho) d\chi$$

for every such perturbation χ , we call it the **first variation** of F at ρ , and denote it by $\frac{\delta F}{\delta \rho}$.

B.2. Gradient flows and PDEs

The connection between OT and certain diffusive partial differential equations (PDE) has been well studied over the past two decades (Jordan et al., 1998; Otto, 2001). Indeed, equation (8) defines a PDE over densities ρ . As mentioned before, it has a fluid dynamics interpretation as a continuity equation on a density-dependent flow velocity vector field $\mathbf{u} \triangleq -\nabla \frac{\delta F}{\delta \rho}(\rho)$, or a conservation-of-energy PDE for the energy flux $\mathbf{q} \triangleq -\rho \nabla \frac{\delta F}{\delta \rho}(\rho)$. In the context of densities and datasets, this PDE can be roughly understood as a conservation-of-mass principle: no probability mass is created or destroyed in the sequence of densities on $\mathcal{X} \times \mathcal{Y}$ that solve this system.

For a functional of the form (12) with only $\mathcal{F}, \mathcal{V}, \mathcal{W}$ terms, the corresponding PDE (eq. (14)) is known as a diffusion–advection–interaction equation. Certain choices of functionals $\mathcal{F}, \mathcal{V}, \mathcal{W}$ recover familiar PDEs. For example, taking $F(\rho) = \mathcal{F}(\rho) + \mathcal{V}(\rho)$, and $f(t) = t \log t$, the gradient flow of F solves a Fokker-Planck equation (Santambrogio, 2015):

$$\partial_t \rho - \Delta \rho - \nabla \cdot (\rho \nabla V) = 0.$$

In dataset space, this equation can be interpreted as the time evolution of a dataset subject to a drift force imposed by the potential function V and a constant-variance diffusion term ($\Delta \rho$) resulting from the entropy-inducing functional \mathcal{F} . Other choices of functionals allow us to recover the advection equation, porous-media equation, and various other diffusion–advection–interaction PDEs (Santambrogio, 2017). As we did for the Fokker-Planck equation, interpreting these PDEs in our context of dataset dynamics might yield interesting insights for designing objective functions.

C. Implementation and Experimental Details

We implement our method on PyTorch (Paszke et al., 2019), using the `geomloss` (Feydy et al., 2019) and `POT` (Flamary et al., 2021) libraries for OT-related computations, including the OTDD distance needed at every step. The three types of feature-label dynamics described in Section 6 are implemented by detaching parts of the computational graph in order to make gradient updates only in some of them. For the variable label dynamics, there are two options for clustering: fixed-size or nonparametric. We use k-means for the former and density-based spatial clustering of applications with noise (DBSCAN) with parameters $\epsilon = 5$ and minimum points per cluster 4 for the latter. Pseudocode for the three types of feature-label gradient flow dynamics described in Section 6 is shown here in Algorithms 1 to 3.

For the parametrized flow mapping h_{flow} (§7.2), we use an autoencoder-type architecture with an encoder consisting of 2 convolutional and 5 fully-connected layers, and the decoder is a inverted copy of the encoder. It was trained for 20 epochs using ADAM with learning rate 1×10^{-3} , using ten different random restarts and choosing the best performing one in a held-out set. For transfer learning (§7.2), we use a LeNet-5 architecture with ReLU nonlinearities trained for 20 epochs using ADAM with learning rate 1×10^{-3} and weight decay 1×10^{-6} . It was fine-tuned for 10 epochs on the target domain(s) using the same optimization parameters. For the experiments in Table 1, we use 5K source (MNIST) and target (other *NIST datasets) samples. For both supervised and unsupervised flows, we use a flow step size of 1×10^{-1} , 1000 steps, and entropy regularization $\lambda = 1 \times 10^2$. For the unsupervised flow, we permute the values of the pseudo-labels obtained through clustering to match them to the indices of the target labels so as to allow accuracy computation.

All experiments were run on the same machine with an Intel Xeon 32-core 2.00GHz CPU with a single GeForce RTX 2080 Ti GPU. In this machine, the flows on synthetic datasets of Section 6 run at < 0.2 s per step, while the flows for the image classification datasets of Sections 6 and 7.2 run at ~ 5 s per step for 2K particles, for a total flow runtime of less than 5 minutes. Information about all the datasets used, including references, are provided in Table 2.

Algorithm 1 Gradient flow with feature-driven fixed-label (fd-f1) dynamics.

Input: Initial particle feature matrix $X_0 \in \mathbb{R}^{d \times n}$ and corresponding labels $\mathbf{y} \in \{0, \dots, k\}^n$.
requires_gradient(X_0) \leftarrow True
for time $t = 0$ **to** T **do**
 $\ell \leftarrow F(\mathbf{X}_t, \mathbf{y})$
 $\mathbf{X}_t \leftarrow \text{optim_step}(\nabla_{\mathbf{X}} \ell)$
 for every class $j = 1$ **to** k **do**
 $\mu_t^j, \Sigma_t^j \leftarrow \text{getstats}(\{\mathbf{x}_t^i \mid y_i = j\})$
 end for
 recompute_label_distances($\{\mu_t^j\}, \{\Sigma_t^j\}$) {subroutine in OTDD §3.3}
end for

Algorithm 2 Gradient flow with joint-driven fixed-label (jd-f1) dynamics.

Input: Initial particle feature matrix $X_0 \in \mathbb{R}^{d \times n}$ and corresponding labels $\mathbf{y} \in \{0, \dots, k\}^n$.
requires_gradient(X_0, Σ_0^j, μ_0^j) \leftarrow True
for time $t = 0$ **to** T **do**
 $\ell \leftarrow F(\mathbf{X}_t, \mathbf{y})$
 $\mathbf{X}_t \leftarrow \text{optim_step}(\nabla_{\mathbf{X}} \ell)$
 for every class $j = 1$ **to** k **do**
 $\mu_t^j \leftarrow \text{optim_step}(\nabla_{\mu^j} \ell)$
 $\Sigma_t^j \leftarrow \text{optim_step}(\nabla_{\Sigma^j} \ell)$
 end for
 recompute_label_distances($\{\mu_t^j\}, \{\Sigma_t^j\}$) {subroutine in OTDD §3.3}
end for

Algorithm 3 Gradient flow with joint-driven variable-label (jd-v1) dynamics.

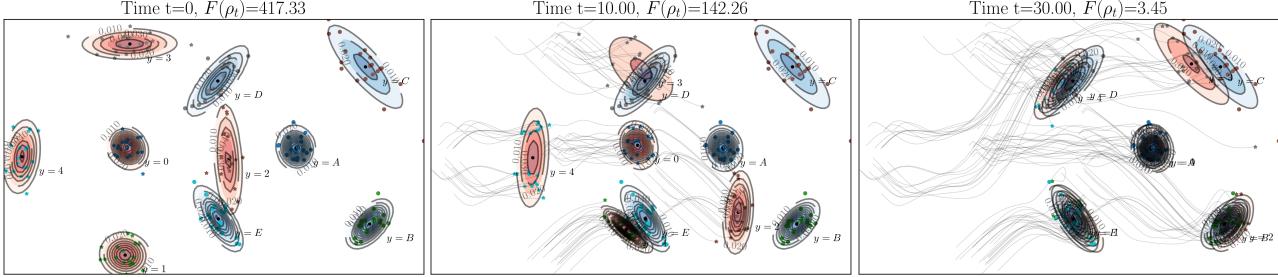
Input: Initial particle feature matrix $X_0 \in \mathbb{R}^{d \times n}$ and corresponding labels $\mathbf{y} \in \{0, \dots, k\}^n$.
requires_gradient(X_0, Σ_0^j, μ_0^j) \leftarrow True
for time $t = 0$ **to** T **do**
 $\ell \leftarrow F(\mathbf{X}_t, \mathbf{y})$
 $\mathbf{X}_t \leftarrow \text{optim_step}(\nabla_{\mathbf{X}} \ell)$
 for every particle $i = 1$ **to** n **do**
 $\mu_t^i \leftarrow \text{optim_step}(\nabla_{\mu^i} \ell)$
 $\Sigma_t^i \leftarrow \text{optim_step}(\nabla_{\Sigma^i} \ell)$
 end for
 $\mathbf{y}_t \leftarrow \text{clustering_method}(\{\mu_t\}, \{\Sigma_t\})$ {recompute discrete labels by clustering}
 recompute_label_distances($\{\mu_t^j\}, \{\Sigma_t^j\}$) {subroutine in OTDD §3.3}
end for

Table 2. Summary of datasets used. *: we rescale the USPS digits to 28×28 for comparison to the *NIST datasets, and the STL-10 and CAMELYON to 32×32 for comparison to CIFAR-10.

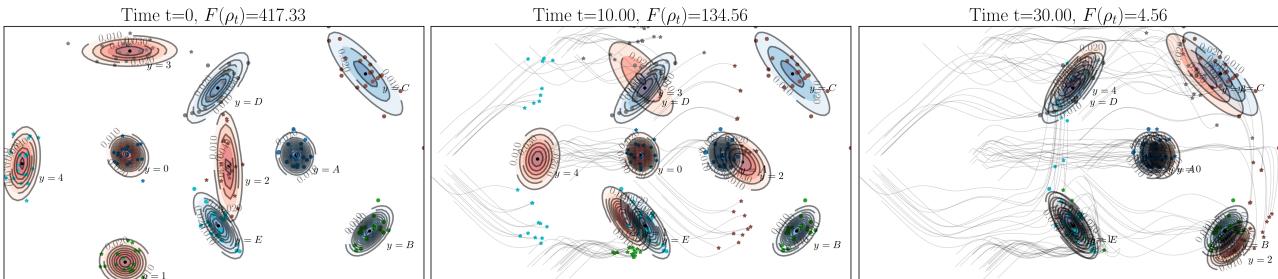
Dataset	Input Dimension	Number of Classes	Train Examples	Test Examples	Source
USPS	$16 \times 16^*$	10	7291	2007	(Hull, 1994)
MNIST	28×28	10	60K	10K	(LeCun et al., 2010)
KMNIST	28×28	10	60K	10K	(Clanuwat et al., 2018)
FASHION-MNIST	28×28	10	60K	10K	(Xiao et al., 2017)
CAMELYON	$128 \times 128^*$	2	262K	32K	(Litjens et al., 2018)
CIFAR-10	32×32	10	50K	10K	(Krizhevsky & Hinton, 2009)
STL-10	$96 \times 96^*$	10	5K	8K	(Coates et al., 2011)

D. Additional Experimental Results on Gaussian Flows

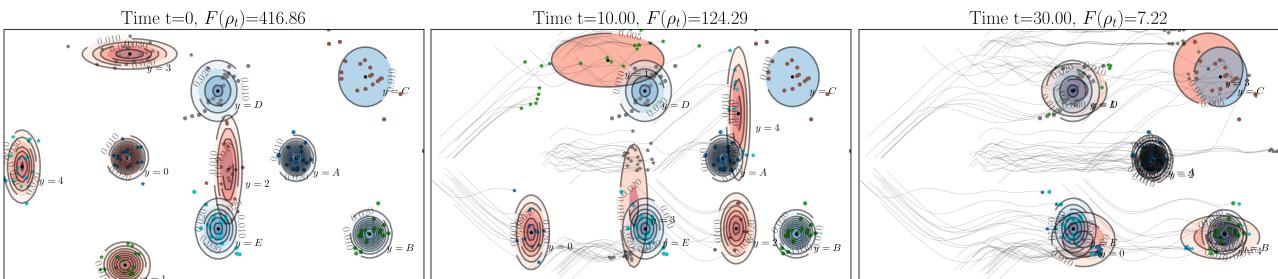
For the simple synthetic dataset example of Section 6, we show a comparison of the three types of flow dynamics in Figure 7, and experiments with various types of functionals in Figure 8.



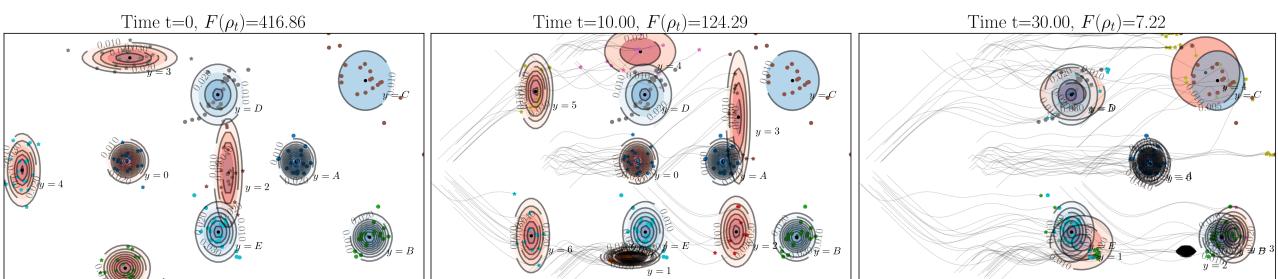
(a) Feature-driven (fcd) dynamics, ADAM optimizer.



(b) Joint-driven fixed-label (jcd-f1) dynamics, ADAM optimizer.

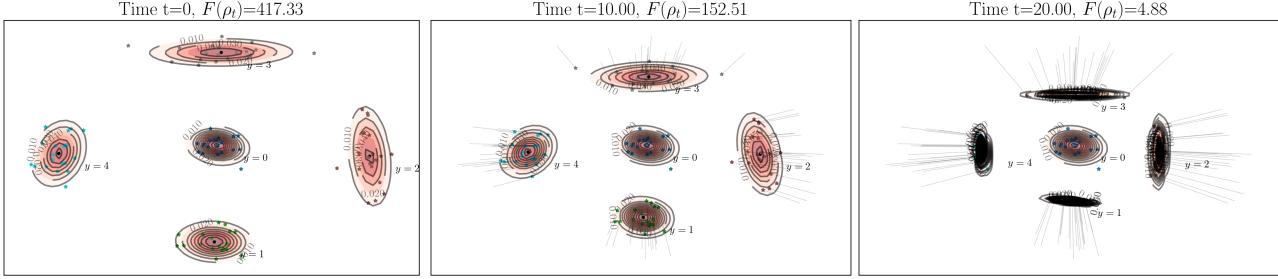


(c) Joint-driven variable-label (jcd-v1) dynamics, k-means clustering, ADAM optimizer.

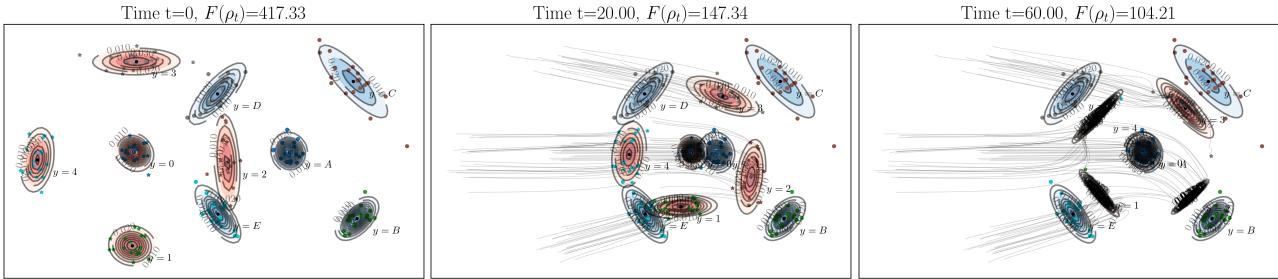


(d) Joint-driven variable-label (jcd-v1) dynamics, DBSCAN clustering, ADAM optimizer.

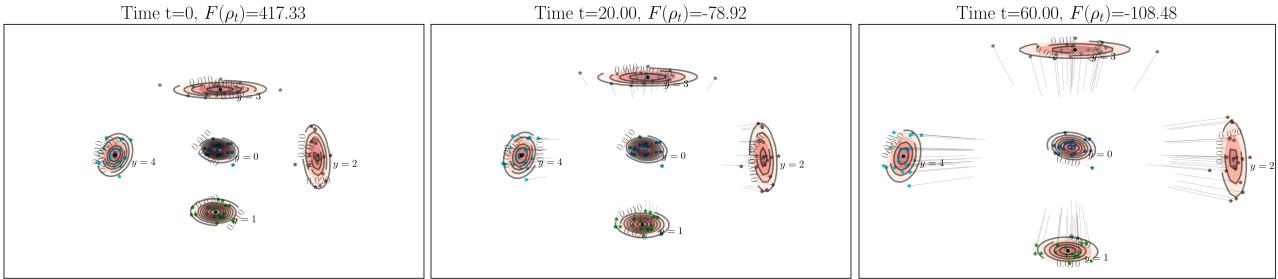
Figure 7. Gradient flows driven by functional $\mathcal{T}_\beta(\rho) = \text{OTDD}(\mathbf{D}_\rho, \mathbf{D}_\beta)$ starting from dataset ρ_0 (red) advecting towards β (blue) for various dynamic schemes (§6).



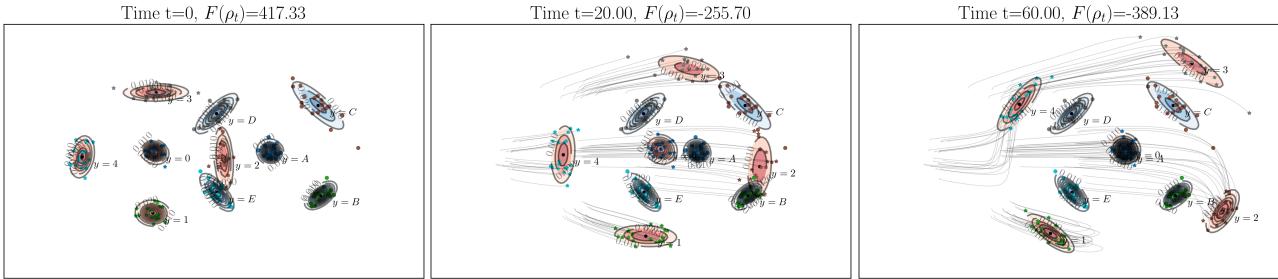
(a) Functional: $F(\rho) = \int (\|x - x_0\| - \tau)_+ d\rho(z)$, SGD optimizer.



(b) Functional: $F(\rho) = \text{OTDD}(\mathcal{D}_\rho, \mathcal{D}_\beta) + \lambda \int (\|x - x_0\| - \tau)_+ d\rho(z)$, SGD optimizer.



(c) Functional: $F(\rho) = \iint -\|x - x'\|^2 \mathbb{1}_{y \neq y'} d\rho(z) d\rho(z')$, SGD optimizer.



(d) Functional: $F(\rho) = \text{OTDD}(\mathcal{D}_\rho, \mathcal{D}_\beta) + \lambda \iint -\|x - x'\|^2 \mathbb{1}_{y \neq y'} d\rho(z) d\rho(z')$, SGD optimizer.

Figure 8. Gradient flows starting from dataset ρ_0 (red) advecting towards β (blue) driven by different functionals, using SGD+jd-vl dynamics in all cases.

E. Additional Experimental Results on *NIST Flows

In Figure 9 we show the effect of the transformation h_θ parametrized as a neural network and learnt from data to mimic the effect of the flow mapping $h_{\text{flow}} : \mathbf{x}_0 \mapsto \mathbf{x}_T$.



Figure 9. **Left:** initial particles \mathbf{x}_0 taken from USPS. **Center:** intermediate state of particles \mathbf{x}_t after gradient flow driven by similarity-seeking functional $T_\beta(\rho) = \text{OTDD}(D_\rho, D_\beta)$ for $D_\beta : \text{MNIST}$. **Right:** particles mapped by using a parametric approximation of h_{flow} learnt from data.

As described in Section 7.2, we use gradient flows to approach to problem of transfer learning. Figure 10 shows results on the 5- and 10-shot tasks on the NIST datasets. Notably, the results follow a similar trend as Figure 3e, although, as expected by the smaller target datasets, the classification errors are higher.

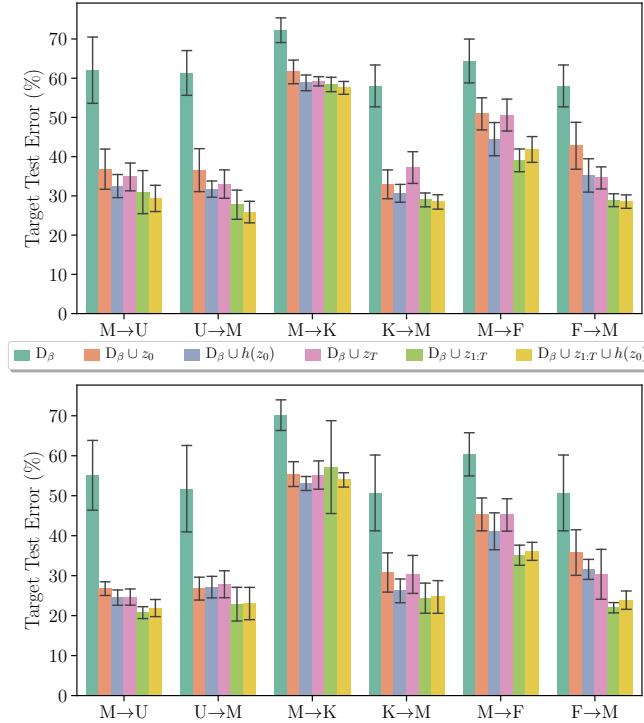


Figure 10. Transfer learning results on image datasets for 5-shot (top) and 10-shot (bottom) tasks. See Section 7.2 for symbol key.

F. Additional Miscellaneous Experimental Results

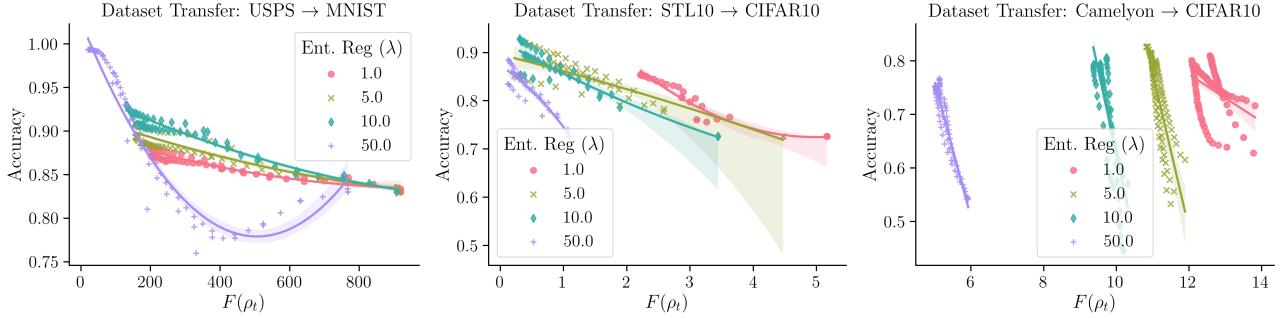


Figure 11. Qualitative evaluation of flows: for the same setting as in Figure 6, we plot here the flow objective against the accuracy. The strong correlation between these quantities shows that the OTDD functional provides a good proxy for dataset transfer quality. The relation between these quantities is mostly linear and monotonic, except in configurations with an over-regularized OTDD objective, as shown for the $\lambda = 50$ curve in the first plot.

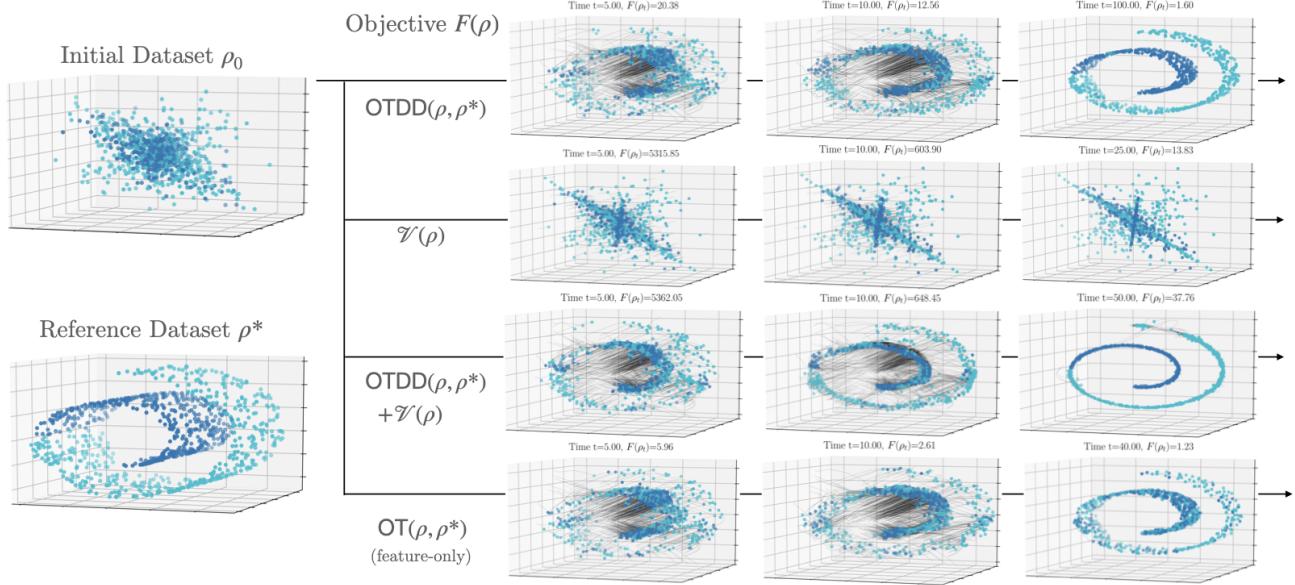


Figure 12. Shaping datasets via flows: our framework allows for simple and principled transformation of classification datasets by following the gradient flow of a functional objective, such as: similarity to a reference dataset (shown here as $\text{OTDD}(\cdot, \rho^*)$), a function enforcing collapse along a given dimension (shown here as $\mathcal{V}(\rho)$), or a combination thereof. In the bottom row we show the flow obtained with a vanilla (feature-only) optimal transport distance functional, which unsurprisingly misses the important class structure of the dataset.