A Principle of Least Action for the Training of Neural Networks

Skander Karkar, Ibrahim Ayed, Emmanuel de Bezenac, Patrick Gallinari

A Principle of Least Action for the Training of Neural Networks

Skander Karkar[1] [✉], Ibrahim Ayed[2], Emmanuel de Bézenac[2], and Patrick Gallinari[1,2]

[1] Criteo AI Lab, Criteo, Paris, France
as.karkar@criteo.com
[2] LIP6, Sorbonne Université, Paris, France
{ibrahim.ayed,emmanuel.de-bezenac,patrick.gallinari}@lip6.fr

Abstract. Neural networks have been achieving high generalization performance on many tasks despite being highly over-parameterized. Since classical statistical learning theory struggles to explain this behaviour, much effort has recently been focused on uncovering the mechanisms behind it, in the hope of developing a more adequate theoretical framework and having a better control over the trained models. In this work, we adopt an alternative perspective, viewing the neural network as a dynamical system displacing input particles over time. We conduct a series of experiments and, by analyzing the network's behaviour through its displacements, we show the presence of a low kinetic energy bias in the transport map of the network, and link this bias with generalization performance. From this observation, we reformulate the learning problem as follows: find neural networks that solve the task while transporting the data as efficiently as possible. This offers a novel formulation of the learning problem which allows us to provide regularity results for the solution network, based on Optimal Transport theory. From a practical viewpoint, this allows us to propose a new learning algorithm, which automatically adapts to the complexity of the task, and leads to networks with a high generalization ability even in low data regimes.

Keywords: Deep Learning · Optimal Transport · Dynamical Systems

1 Introduction

Deep neural networks (DNNs) have repeatedly shown their ability to solve a wide range of challenging tasks, while often having many more parameters than there are training samples. Such a performance of over-parametrized models is counter-intuitive. They seem to adapt their complexity to the given task, systematically achieving a low training error without suffering from over-fitting as could be expected [2,25,40]. This is in contradiction with the classical statistical practice of selecting a class of functions complex enough to represent the coherent patterns in the data, and simple enough to avoid spurious correlations [3,16]. Although this behavior has sparked much recent work towards explaining neural networks'

success [10,20,26,28], it still remains poorly understood. Among the factors to consider are the implicit biases present in the choices made for the parametrization, the architecture, the parameter initialization and the optimization algorithm, and that contribute all to this success. Our aim in this work is to uncover some of these hidden biases and highlight their link with generalization performance through the lens of dynamical systems.

We will focus on residual networks (ResNets) [18,19], now ubiquitous in applications. This family of models has made it possible to learn very complex non-linear functions by improving the trainability of very deep networks, and has thus improved generalization. Links have been derived between these networks and dynamical systems: a ResNet can be seen as a forward Euler scheme discretization of an associated ordinary differential equation (ODE) [35]:

$$\mathrm{x}_{k+1} = \mathrm{x}_k + v_k(\mathrm{x}_k) \iff \partial_t \mathrm{x}_t = v_t(\mathrm{x}_t) \tag{1}$$

This link has yielded many exciting results, *e.g.* new architectures [23] and reversible networks [7]. Here, we make use of this analogy and analyze the behavior of residual networks by studying their associated differential flows. Adopting this dynamical point of view allows us to leverage the theories and mathematical tools developed to study, approximate and apply differential equations.

More specifically, we conduct experiments to observe how neural networks displace their inputs–seen as particles–through time. We measure a strong empirical correlation between good test performance and neural networks with low kinetic energy along their transport flow. From this, we reformulate the training problem as follows: retrieve the network which solves the task using the principle of least action, *i.e.* expending as little kinetic energy as possible. This problem, in its probabilistic formulation, is tightly linked with and inspired by the well-known problem of finding an optimal transportation map [31]. This yields new insights into neural networks' generalization capabilities, and provides a novel algorithm that automatically adapts to the complexity of the data and robustly improves the network's performance, including in low data regimes, without slowing down the training. To summarize, our contributions are the following:

- Through the dynamic viewpoint, we highlight the *low-energy bias* of ResNets.
- We formulate a Least Action Principle for the training of Neural Networks.
- We prove existence and regularity results for networks with minimal energy.
- We provide an algorithm for retrieving minimal energy networks compatible with different architectures, which leads to better generalization performance on different classification tasks, without complexifying the architecture.

We introduce in Section 2 some background on Optimal Transport (OT) and highlight the link between the dynamical formulation of OT and ResNets. We describe in Section 3 the general setting of our analysis. Section 4 provides empirical evidence illustrating our point. The formal framework of networks trained with minimized energy and a practical algorithm are described in Section 5. Experiments on standard classification tasks are provided in Section 6. The code is available online at `github.com/skander-karkar/LAP`.

2 Background

This section outlines the main elements of the formalism and reasoning of our work. Supplementary Material A gives more details about Optimal Transport.

2.1 Optimal Transport

The principle of least action is central to many fields in physics, mathematics and economics. It is found in classical and relativistic mechanics, thermodynamics, quantum mechanics [11,13,14], etc.. It broadly states that the dynamical trajectory of a system between an initial and final configuration is one that makes a certain action associated with the system locally stationary [14]. One mathematical theory which can be associated with this general idea is the theory of Optimal Transport which was initially introduced as a way of finding a transportation map minimizing the cost of displacing mass from one configuration to another [31].

Formally, let α and β be absolutely continuous distributions compactly supported in $\mathbb{R}^d$, and $c : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ a cost function. Consider a transportation map $T : \mathbb{R}^d \to \mathbb{R}^d$ that satisfies $T_\sharp \alpha = \beta$, *i.e.* that pushes[1] α to β. The total cost of the transportation then depends on all the individual contributions of costs of transporting (infinitesimal) mass from each point x to $T(x)$, and finding the optimal transportation map amounts to solving:

$$\begin{aligned} \min_{T} \quad & \mathcal{C}^{\text{stat}}(T) = \int_{\mathbb{R}^d} c(x, T(x)) \mathrm{d}\alpha(x) \\ \text{s.t.} \quad & T_\sharp \alpha = \beta \end{aligned} \tag{2}$$

A standard choice for c is the p-th power of a norm of $\mathbb{R}^d$, *i.e.* $c(x, y) = \|x - y\|^p$, but other costs can be used, defining different variants of the problem. This cost induces, through the p-th root of the minimal value of (2), a distance W_p between any two distributions α and β of finite $p-$th moment, called the p-Wasserstein distance [27].

In [4], the link between Optimal Transport and the principle of least action was made by showing that the static transportation can equivalently be viewed as a dynamical one that minimizes an action as it gradually displaces particles of mass in time. In other words, instead of directly pushing samples of α to β in $\mathbb{R}^d$ using T, we can displace mass from α according to a continuous flow with velocity $v_t : \mathbb{R}^d \to \mathbb{R}^d$. This implies that the density μ_t at time t satisfies the *continuity equation* $\partial_t \mu_t + \nabla \cdot (\mu_t v_t) = 0$, assuming that initial and final conditions are given respectively by $\mu_0 = \alpha$ and $\mu_1 = \beta$. In this case, the optimal displacement is the one that minimizes the action $\|v_t\|^p_{L^p(\mu_t)}$:

$$\begin{aligned} \min_{v} \quad & \mathcal{C}^{\text{dyn}}(v) = \int_0^1 \|v_t\|^p_{L^p(\mu_t)} \mathrm{d}t \\ \text{s.t.} \quad & \partial_t \mu_t + \nabla \cdot (\mu_t v_t) = 0, \mu_0 = \alpha, \mu_1 = \beta \end{aligned} \tag{3}$$

[1] $T_\sharp \alpha$ is the *push-forward measure*: $T_\sharp \alpha(B) = \alpha(T^{-1}(B))$ for any measurable set B.

where $\|v_t\|^p_{L^p(\mu_t)} = \int_{\mathbb{R}^d} \|v_t(x)\|^p \mathrm{d}\mu_t(x)$ for costs $c(x,y) = \|x-y\|^p$ with $p > 1$. In this case, minimizers exist and the two transport costs are the same, *i.e.* $\mathcal{C}^{\mathrm{stat}}(T) = \mathcal{C}^{\mathrm{dyn}}(v)$ at the optimums. For $p = 2$ and the Euclidean norm, the dynamical cost $\mathcal{C}^{\mathrm{dyn}}(v)$ corresponds to the *kinetic energy*.

2.2 Link with Residual Networks

The dynamical formulation in (3) explicitly describes the evolution in time of the density μ_t, starting from an input distribution α. In this form, the link between deep residual networks and dynamical Optimal Transport is not clear. However, it is possible to adopt an alternate viewpoint which helps make it immediate. Instead of explicitly describing the density's evolution, we describe the paths $\phi^x \colon [0,1] \to \mathbb{R}^d$, $t \mapsto \phi^x_t$ taken by particles from α at position x, when displaced along the flow v. The continuity equations can then equivalently be written as:

$$\partial_t \phi^x_t = v_t(\phi^x_t) \tag{4}$$

See chapters 4 and 5 of [31] for details. We can now note the resemblance between the residual network (1) and equation (4). Rewriting the conditions as necessary, the dynamical formulation (3) can equivalently be represented by:

$$\begin{aligned} \min_v \quad & \mathcal{C}^{\mathrm{lag}}(v) = \int_0^1 \|v_t\|^p_{L^p((\phi^\cdot_t)_\sharp \alpha)} \, \mathrm{d}t \\ \text{s.t.} \quad & \partial_t \phi^x_t = v_t(\phi^x_t), \\ & \phi^\cdot_0 = \mathrm{id}, \\ & (\phi^\cdot_1)_\sharp \alpha = \beta \end{aligned} \tag{5}$$

where $\phi^\cdot_t : x \in \mathbb{R}^d \mapsto \phi^x_t \in \mathbb{R}^d$ corresponds to the transport map induced by the flow, up until time t. As both formulations are equivalent, we have that for any flow v, $\mathcal{C}^{\mathrm{lag}}(v) = \mathcal{C}^{\mathrm{dyn}}(v)$. Moreover, optimal transportation plans in the static (2) and dynamical (5) cases coincide: if T and $\phi^\cdot_t$, are respectively solutions to (2) and (5), we have that $T = \phi^\cdot_1$.

This link allows us to associate residual networks with a local action for each layer, which induces a global transportation cost $\mathcal{C}^{\mathrm{lag}}$, and taking $p = 2$ and the Euclidean norm allows us to refer to the network's kinetic energy.

3 General Setting

In order to better understand the inner workings of a DNN, it is essential to adopt a viewpoint in which the different driving mechanisms become apparent and are decoupled.

Decomposing a DNN We consider the following model of a deep neural network f where computations are separated into the three steps, *i.e.* $f = F \circ T \circ \varphi$ (this is similar to [22] and corresponds to the general structure of recent deep models or to the structure of components of a deep model [19,36,39]):

1. **Dimensionality change:** Starting from an input distribution $\mathcal{D}$ in $\mathbb{R}^n$, a transformation φ is applied, transforming it into $\alpha = \varphi_\sharp \mathcal{D}$, a distribution in $\mathbb{R}^d$. This corresponds to the first few layers present in most recent architectures and represents a change of dimensionality. φ is known as the *encoder*.
2. **Data Transport:** Then α is transformed by a mapping $T : \mathbb{R}^d \to \mathbb{R}^d$, which we see as a transport map. Here, the dimensionality doesn't change and, if this part of the network is a sequence of residual blocks, T can be written as the discretized flow of an ODE.
3. **Task-specific final layers:** A final function $F : \mathbb{R}^d \to \mathcal{Y}$ is applied to $T_\sharp \alpha$ in order to compute the loss $\mathcal{L}$ associated with the task at hand, *e.g.* F could be a perceptron classifier. Like φ, F is typically made up of a few layers.

The focus of this work is on analyzing the second phase, Data Transport, and we assume that the encoder φ is pretrained and fixed (this will be relaxed in some experiments later). To solve a complex non-linear task for which a DNN is needed, the data has to be transformed in a non-trivial way, meaning that this is an essential phase, *e.g.* in the case of classification, $T_\sharp \alpha$ needs to be linearly separable if F is linear. This model is quite general, as many ResNet-based architectures [36,39] alternate modules that change the dimensionality (step 1) and transport modules that keep the dimensionality fixed (step 2) and according to [21], the transport modules have similar behaviour. The model can then be considered as a simplified ResNet, sometimes called a *single representation* ResNet. Note that [30] finds that networks that keep the same resolution remain competitive.

The set of admissible targets As recent neural architectures have systematically achieved near-zero training error [2,3,20,40], we place ourselves in this regime, which makes it possible to model this as a hard constraint. For some tasks, this constraint over T is obvious: in a generative setting for example, $T_\sharp \alpha$ must be equal to some prescribed distribution β which is the target of the generation process. But in general, T is less strictly constrained and the condition depends on F and $\mathcal{L}$. This leads us to define a *set of admissible targets* for the task:

$$S_{F,\mathcal{L}} = \{\beta \in \mathcal{P}(\mathbb{R}^d) \mid \mathcal{L}(F, \beta) = 0\} \tag{6}$$

with $\beta = T_\sharp \alpha$. In general, $\mathcal{L}$ is fixed while F is learned jointly with T. This set is supposed to be non-empty for some F and, in general, it will contain many distributions. The goal of the learning task can then be reformulated as:

$$\text{Find } (T, F) \text{ such that } T_\sharp \alpha \in S_{F,\mathcal{L}} \tag{7}$$

An important observation is that, even when $S_{F,\mathcal{L}}$ is reduced to a singleton, the problem is still strongly under-constrained and it is possible to obtain many such (T, F) that lead to poor generalization. One can then ask why this is not the case in practice, as good generalization performance is usually achieved.

The case of classification Even though our framework is general, we focus our experiments on classification tasks, with $\mathcal{L}$ being the cross entropy loss. The task

consists in separating N classes. Let us denote α_i the class distributions which are supposed to be distributions in $\mathbb{R}^d$ of mutually disjoint supports, meaning that there is no ambiguity in the class of data points, and such that $\alpha = \sum_i \alpha_i/N$. One wants to find a transformation T of these distributions such that all transported distributions can be correctly classified by a classifier F. When F is linear, $S_{F,\mathcal{L}}$ is the set of distributions which have N components that are linearly separated by F. Note that we place ourselves in a noiseless ideal setting where perfect classification is possible. The question we examine in this work is then twofold:

- What are the properties characterizing mappings reached by standard residual architectures with common hyper-parameters?
- Can we find a criteria to *automatically select* mappings with desirable properties in order to improve performance and robustness?

4 Empirical Analysis of Transport Dynamics in ResNets

Before introducing our framework, we conduct an exploratory analysis of the impact of the network's inner dynamics on generalization. We present below two experiments. The first one highlights how good generalization performance is closely related to low transport cost for classification tasks on MNIST and CIFAR10. This cost therefore appears as a natural characterisation of the complexity and disorder of a network. The second experiment, performed on a toy 2D dataset, visualizes the transport induced by the blocks of a ResNet.

We consider ResNets where, after encoding, a data point x_0 is transported by applying $x_{k+1} = x_k + v_k(x_k)$ for K residual blocks and then classified using F. We measure the disorder/complexity of a network by its transport cost which is the sum of the displacements induced by its residual blocks: $\mathcal{C}(v) = \sum_k \|v_k(x_k)\|_2^2$. This quantity corresponds to the kinetic energy of the total displacement.

Transportation cost and generalization on MNIST and CIFAR10. In order to study the correlation between the transport cost of a residual network and its generalization ability on image data, we train convolutional 9-block ResNets with different initializations (orthogonal and normal with different gains), for 10-class classification tasks MNIST and CIFAR10. In Figure 2, each point represents a trained network and gives the transport cost $\mathcal{C}$ as a function of the test accuracy of the network. This experiment clearly highlights the strong negative correlation between transport cost and good generalization. This illustrates the importance of the implicit initialization bias and motivates initialization schemes which favour a low kinetic energy. We believe a number of factors contribute to this low energy bias: small initialization gains tend to bias $\|v_k(x_k)\|_2^2$ towards small values, and training using gradient descent does not change this much.

Visualizing network dynamics on 2D toy data. This experiment provides a 2D visualization of the transport dynamics inside a network. The task is 2-class classification of a non-linearly separable dataset (two concentric circles, from `sklearn`) that contains 1000 points with a train-test split of 80%-20%, see Figure

1 top left. The network is a ResNet containing 9 residual blocks, followed by a fixed linear classifier. Each residual block contains two fully connected layers separated by a batch normalization and a ReLU activation.

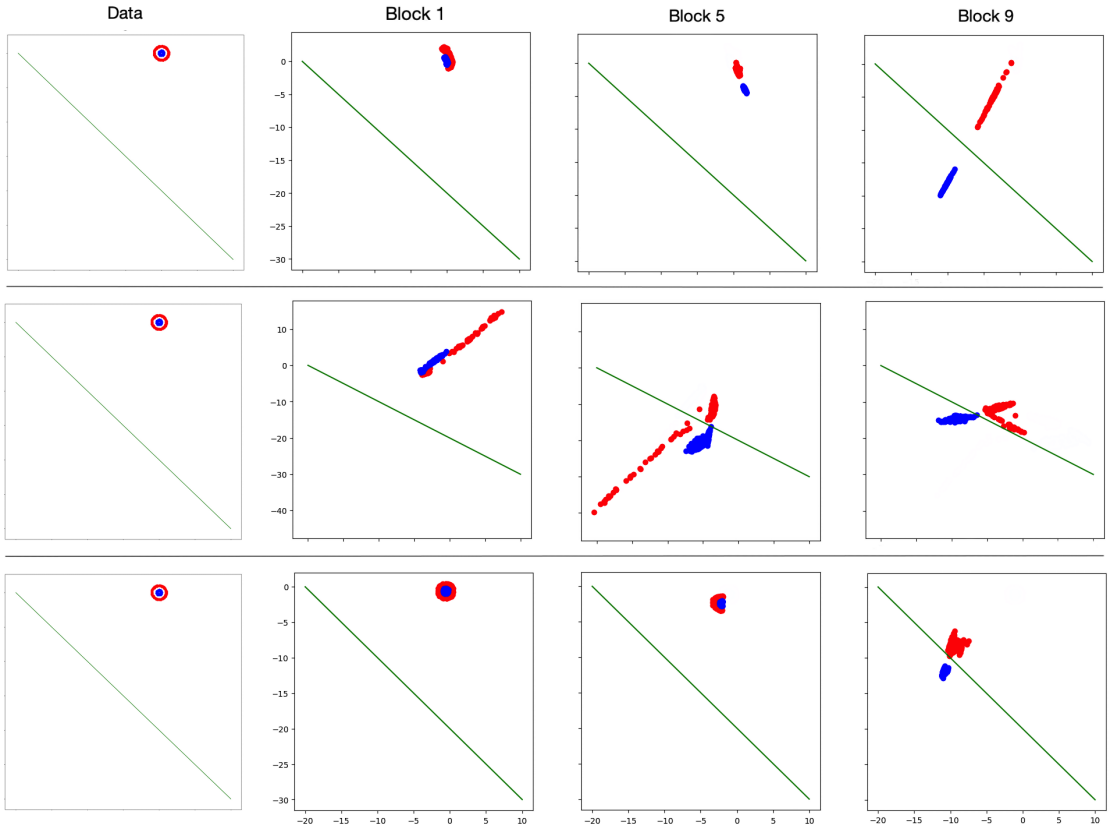

Fig. 1. Transformed circles test set by a ResNet9 after blocks 1, 5 and 9 after training; first row with good initialization; second row with a $\mathcal{N}(0, 5)$ initialization; third row with a $\mathcal{N}(0, 5)$ initialization and the transport cost added to the loss

With the cross-entropy loss alone, the behaviour of a well trained and carefully initialized network achieving 100% test accuracy is illustrated in the first row of Figure 1. With a $\mathcal{N}(0, 5)$ initialization, significantly bigger than an "optimal" initialization, the test accuracy drops to 98% (average of 100 runs) and the transport becomes chaotic (Figure 1, second row). Adding the transport cost to the loss improves the test accuracy (99.7% on average) of this badly initialized network and the movement becomes more controlled (third row of Figure 1). Thus, controlling transport improves the behavior and generalization ability of the network. This allows to explicitly control the network whereas implicit biases

such as "good" initialization rely on heuristics. In Supplementary Material C.4, more experiments show that in other situations that deviate from the ideal setting where the task is perfectly solved, e.g. when using a network which is too large or too small, or a small training set, controlling the transport cost also improves generalization.

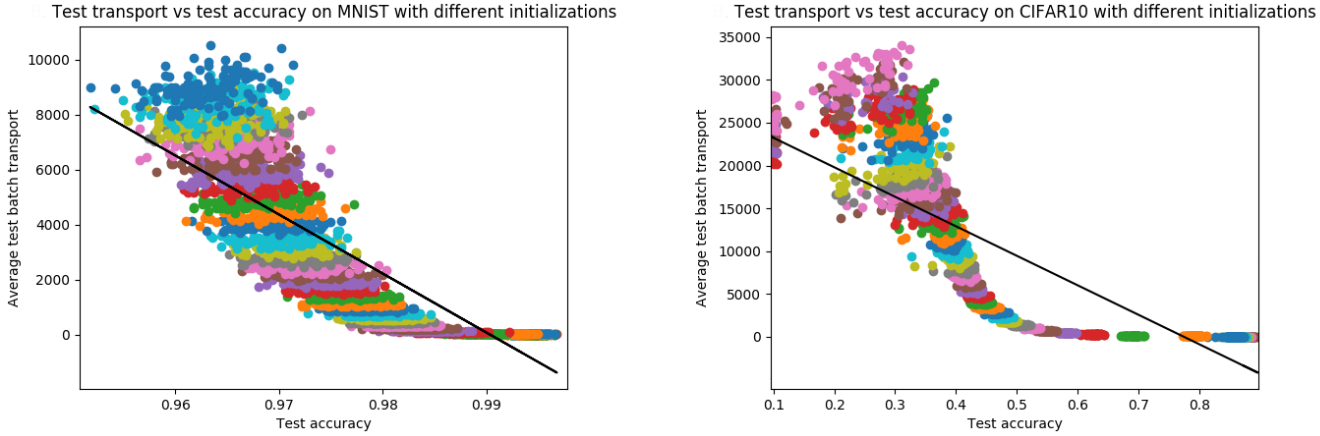

Fig. 2. Test transport against test accuracy of ResNet9 models on MNIST (left) and CIFAR10 (right) with fitted linear regressions, where each color indicates a different initialization (either orthogonal or normal with varying gains)

5 Least Action Principle for Training Neural Networks

The previous section has shed some light on the low energy bias of networks as well as on its potential benefits on test accuracy. In this section, we take a step further and make this implicit bias explicit by considering a formulation for training that enforces minimal kinetic energy, closely related to the problem of Optimal Transport. This allows us to prove the existence of minimizers, and exhibit interesting regularity properties of the minimal energy neural networks which may explain good generalization performance.

5.1 Formulation

We consider costs $c(x, y) = \|x - y\|^p$ (where $\|.\|$ is a norm of $\mathbb{R}^d$), with $p > 1$, and suppose that $\alpha \in \mathcal{P}_p(\mathbb{R}^d)$ (the set of absolutely continuous measures on $\mathbb{R}^d$ with finite p-th moment). We assume that the space of classifiers is compact, that the loss $\mathcal{L}$ is continuous, that the set $\cup_{F\in\mathcal{F}} S_{F,\mathcal{L}}$ is at a finite p-Wasserstein distance W_p from α (in particular, it is non-empty) and that all its bounded subsets are totally bounded (*i.e.* can be covered by finitely many subsets of any fixed size).

These properties depend on the choice of the loss $\mathcal{L}$ and of a class of functions $\mathcal{F}$ for the classifier F.

Returning to the transport problem as defined in Section 2.1, a natural way to select a robust model, given the empirical observations of Section 4, is to select, among the maps which transport α to $S_{F,\mathcal{L}}$ and thus solve the task, one with a minimal transport cost. This gives us the following optimization problem:

$$\begin{aligned} \inf_{T,F} \quad & \mathcal{C}(T) = \int_{\mathbb{R}^d} c(x, T(x)) \mathrm{d}\alpha(x) \\ \text{subject to} \quad & T_\sharp \alpha \in S_{F,\mathcal{L}} \end{aligned} \tag{8}$$

The equivalent dynamical version for $c(x, y) = \|x - y\|^p$ is, as per Section 2.2,

$$\begin{aligned} \inf_{v,F} \quad & \int_0^1 \|v_t\|^p_{L^p((\phi_t^\cdot)_\sharp \alpha)} \, \mathrm{d}t \\ \text{subject to} \quad & \partial_t \phi_t^x = v_t(\phi_t^x) \\ & \phi_0^\cdot = \mathrm{id} \\ & (\phi_1^\cdot)_\sharp \alpha \in S_{F,\mathcal{L}} \end{aligned} \tag{9}$$

where $\|v_t\|^p_{L^p((\phi_t^\cdot)_\sharp \alpha)} = \int_{\mathbb{R}^d} \|v_t\|^p \mathrm{d}(\phi_t^\cdot)_\sharp \alpha$. The result below shows that these two problems are equivalent and that the infima are realized as minima:

Theorem 1. *The infima of* (8) *and* (9) *are finite and are realized through a map T which is (or a velocity field v which induces) an optimal transportation map. When $c(x, y) = \|x - y\|^p$, then* (8) *and* (9) *are equivalent.*

Proof. From the hypothesis above, there exists $\beta \in S_{F,\mathcal{L}}$ at a finite distance from α. Taking any transport map between α and β, we see that the infima are finite.

Consider (8) and take a minimizing sequence $(T_i, F_i)_i$. Set $\beta_i = (T_i)_\sharp \alpha$. Then $(\mathcal{C}(T_i))_i$ converges to the infimum which is strictly bounded by $M > 0$. Then, by definition, for i large enough, $W_p^p(\alpha, \beta_i) \leq \mathcal{C}(T_i) \leq M$. So that $(\beta_i)_i$ is a bounded sequence in $\cup_F S_{F,\mathcal{L}}$. By the hypothesized total boundedness of bounded subsets and as $\mathcal{P}_p(\mathbb{R}^d)$ endowed with W_p is a complete metric space (see [6] for a proof), up to an extraction, $(\beta_i)_i$ converges to $\beta^\star$ in the closure of $\cup_F S_{F,\mathcal{L}}$. Moreover, up to an extraction, $(F_i)_i$ also converges to $F^\star$ by compactness of the class of classifiers. Taking $T^\star$ the OT map between α and $\beta^\star$ (see Supplementary Material A for existence of OT maps), we then have, by continuity of $\mathcal{L}$,

$$T^\star_\sharp \alpha = \beta^\star \in S_{F^\star,\mathcal{L}}$$

and $\mathcal{C}(T^\star) \leq \lim \mathcal{C}(T_i)$ by optimality of $T^\star$, which means, since $(\mathcal{C}(T_i))_i$ is a minimizing sequence, that $\mathcal{C}(T^\star)$ minimizes (8). So $(T^\star, F^\star)$ is a minimizer and $T^\star$ is an OT map.

Finally, there exists, by dynamical OT theory (Supplementary Material A), a velocity field $v_t^\star$ inducing the OT map between α and $\beta^\star$ which then gives a minimizer $(v^\star, F^\star)$ for (9). By the same reasoning, taking a minimizing sequence $(v^{(i)}, F_i)_i$ and the induced maps T_i shows that both problems are equivalent. □

Note that uniqueness doesn't hold anymore, as the constraint $T_\sharp\alpha \in S_{F,\mathcal{L}}$ in (9) is looser than in standard OT. However, as we show in the following section, the fact that the optimization problems are solved by OT maps will give regularity properties for the models induced by these optimization problems.

5.2 Regularity

Intuitively, the fact that we minimize the energy of the transport map transforming the data is akin to the core idea of Occam's razor: among all the possible networks that correctly solve the the task, the one transforming the data in the simplest way is selected. Moreover, it is possible to show that this optimal transformation is regular: our formulation provides an alternate view on generalization for modern deep learning architectures in the overparametrized regime.

Optimal maps can be as irregular as needed in order to fit the target distribution, however in much the same way as successfully trained DNNs, optimal maps are still surprisingly regular. In a way, they are as regular as possible given the constraints which is exactly the type of flexibility needed. However, the constraints in (8) and (9) are looser than in the standard definitions of Optimal Transport. Still, supposing that the input data distribution has a nicely behaved density, namely bounded and of compact support, with the same hypothesis as above, we have the following, which is mainly a corollary of Theorem 1:

Proposition 2. *Consider $T^\star$ the OT map induced by* (8) *(or* (9)*) given by Theorem 1. Take X, respectively Y, an open neighborhood of the support of α, respectively of $T^\star_\sharp\alpha$, then $T^\star$ is differentiable, except on a set of null α measure.*

Additionally, if $T^\star$ doesn't have singularities, there exists $\eta > 0$ and A, respectively B, relatively closed in X, respectively Y, such that $T^\star$ is η-Hölder continuous from $X \setminus A$ to $Y \setminus B$. Moreover, if the two densities are smooth, $T^\star$ is a diffeomorphism from $X \setminus A$ to $Y \setminus B$.

Proof. This is a consequence of Theorem 1, the hypothesis made in this section and the regularity theorems stated in Supplementary Material B. □

There are two main results in Proposition 2: the first gives α-*a.e.* differentiability. This is already as strong as might be expected from a classifier: there are necessarily discontinuities at the frontiers between different classes. The second is even more interesting: it gives Hölder continuity over as large a domain as possible, and even a diffeomorphism if the data distribution is well-behaved enough. We recall that a function f is η-Hölder continuous for $\eta \in]0, 1]$ if $\exists\ M > 0$ such that $\|f(x) - f(y)\| \leq M\|x - y\|^\eta$ for all x, y. η measures the smoothness of f, the higher its value, the better. In particular, in the case of classification, this means that the Hausdorff dimension along the frontiers between the different classes is scaled by less than a factor of $1/\eta$ in the transported domain. If the densities are smooth, the dimension even becomes provably smaller by this result.

Intuitively, this means that, in these models, the data is transported in a way that preserves and simplifies the patterns in the input distribution. In the following, we propose a practical algorithm implementing these models and use it for standard classification tasks, showing an improvement over standard models.

5.3 Practical Algorithm

We propose an algorithm for training ResNets using the least action principle by minimizing the kinetic energy. Starting from problem (9) with $p = 2$ and the Euclidean norm, we first discretize the differential equation via a forward Euler scheme, which yields $\phi_{k+1}^x = \phi_k^x + v_k(\phi_k^x)$. The discretized flow v_k is parameterized by a residual block, giving a standard residual architecture. The residual blocks, along with a classifier F, are parametrized by θ. Next, the constraint $(\phi_1^{\cdot})_\sharp \alpha \in S_{F,\mathcal{L}}$ is rewritten as $\mathcal{L}(F, (\phi_1^{\cdot})_\sharp \alpha) = 0$, denoted $\mathcal{L}(\theta) = 0$ below. Finally, as we only have access to a finite set $\mathcal{X}$ of samples x from α, we use a Monte-Carlo approximation of the integral $w.r.t$ the distributions $(\phi_t^{\cdot})_\sharp \alpha$, to obtain:

$$\begin{aligned} \min_{\theta} \quad & \mathcal{C}(\theta) = \sum_{x \in \mathcal{X}} \sum_{k=0}^{K-1} \|v_k(\phi_k^x)\|_2^2 \\ \text{s.t.} \quad & \phi_{k+1}^x = \phi_k^x + v_k(\phi_k^x), \\ & \phi_0^x = x, \ \forall \ x \in \mathcal{X}, \\ & \mathcal{L}(\theta) = 0 \end{aligned} \tag{10}$$

Is is easy to see that the min-max problem $\min_\theta \max_{\lambda > 0} \ \mathcal{C}(\theta) + \lambda \ \mathcal{L}(\theta)$ yields the same solution, as the first two constraints are satisfied trivially. If the constraint $\mathcal{L}(\theta) = 0$ corresponding to solving the task, which includes the classifier F, is not verified, this will cause the second term to grow unbounded, and the solution will thus be avoided by the minimization. This min-max problem can be solved using an iterative approach, starting from some initial λ_0 and θ_0:

$$\begin{cases} \theta_{i+1} = \arg\min_{\theta} \ \mathcal{C}(\theta) + \lambda_i \ \mathcal{L}(\theta) \\ \lambda_{i+1} = \lambda_i + \tau \ \mathcal{L}(\theta_{i+1}) \end{cases} \tag{11}$$

The minimization is done via SGD for a number of steps s, where a step means a batch, starting from the previous parameter value θ_i. This algorithm is similar to Uzawa's algorithm used in convex optimization [31]. In practice, it is more stable to divide the minimization objective in (11) by λ_i, yielding:

Algorithm: Training neural networks with Least Action Principle (LAP-Net)

Input: Training samples, step size τ, number of steps s, initial weight λ_0
Initialization: Initialize the parameters θ_0 and set $i = 0$
while *not converged* **do**
1. Starting from θ_i, perform s steps of stochastic gradient descent:
   1.1. $\theta_{i+1}^0 = \theta_i$
   1.2. $\theta_{i+1}^l = \theta_{i+1}^{l-1} - \epsilon(\nabla \mathcal{C}(\theta_{i+1}^{l-1})/\lambda_i + \nabla \mathcal{L}(\theta_{i+1}^{l-1}))$ for l from 1 to s
   1.3. $\theta_{i+1} = \theta_{i+1}^s$
2. Update the weight $\lambda_{i+1} = \lambda_i + \tau \ \mathcal{L}(\theta_{i+1})$ and increment $i \leftarrow i + 1$

Output: Learned parameters θ

While the high non-convexity makes it difficult to ensure exact optimality, we can still have some induced regularity when reaching a "good" local minimum:

Proposition 3. *Suppose* $(F^{\theta^\star}, T^{\theta^\star})$ *is reached by the optimization algorithm such that* $T^{\theta^\star}$ *is an* $\epsilon-OT$ *map between* α *and its push-forward*[2]*. Then we have, with the same notations as in Proposition 2,*

$$\forall x, y \in X \setminus A, \|T^{\theta^\star}(x) - T^{\theta^\star}(y)\| \leq O(\epsilon + \|x - y\|^\eta)$$

Proof. We simply write the decomposition:

$$T^{\theta^\star}(x) - T^{\theta^\star}(y) = T^{\theta^\star}(x) - T^\star(x) + T^\star(x) - T^\star(y) + T^\star(y) - T^{\theta^\star}(y)$$

and use the triangular inequality: the first and third terms are smaller than ϵ by hypothesis while Hölder continuity applies for the second by Proposition 2. □

This shows that minimizing the transport cost still endows the model with some regularity, even in situations where the global minimum is not reached.

6 Experiments

MNIST Experiments The base model is a ResNet with 9 residual blocks. Two convolutional layers first encode the image of shape $1 \times 28 \times 28$ into shape $32 \times 14 \times 14$. A residual block contains two convolutional layers, each preceded by a ReLU activation and batch normalization. The classifier is made up of two fully connected layers separated by batch normalization and a ReLU activation. We use an orthogonal initialization [32] with gain 0.01. This and all vanilla models and their training regimes are implemented by following closely the cited papers that first introduced them and our method is added over these training regimes. More implementation details are in Supplementary Material C.3.

When using the entire training set, the task is essentially solved (99.4% test accuracy). We penalize the transport cost as presented in Section 5.3, using $\lambda_0 = 5$, $\tau = 1$ and $s = 5$. The performance barely drops (99.3% test accuracy), and we can visualise the preservation of information from the point of view of a pretrained autoencoder (see Supplementary Material C.1). From the experiments in two dimensions, we suspect that adding the transport cost helps when the training set is small. For performance comparisons, we average the highest test accuracy achieved over 30 training epochs (over random orthogonal weight initializations and random subsets of the complete training set). We find that adding the transport cost improves generalization when the training set is very small (Table 1). We see that the improvement becomes more important as the training set becomes smaller and reaches an increase of almost 14 percentage points in the average test accuracy.

[2] By this, we mean that $\|T^{\theta^\star} - T^\star\|_\infty \leq \epsilon$ where $T^\star$ is the OT map.

Table 1: Average highest test accuracy and 95% confidence interval of ResNet9 over 50 instances on MNIST with training sets of different sizes (in %)

Training set size	ResNet	LAP-ResNet (Ours)
500	90.8, [90.4, 91.2]	**90.9**, [90.7, 91.1]
400	88.4, [88.0, 88.8]	**88.4**, [88.0, 88.8]
300	83.5, [83.0, 84.1]	**86.2**, [85.8, 86.6]
200	74.9, [73.9, 75.9]	**82.0**, [81.5, 82.5]
100	56.4, [54.9, 58.0]	**70.0**, [69.0, 71.0]

CIFAR10 Experiments We run the same experiments on CIFAR10. The architecture is exactly the same except that the encoder transforms the input which is of shape $3 \times 32 \times 32$ into shape $100 \times 16 \times 16$. For our method, we use $\lambda_0 = 0.1$, $\tau = 0.1$ and $s = 50$. We average the highest test accuracy achieved over 200 training epochs over random orthogonal weight initializations and random subsets of the complete train set. Here, we find that adding the transport cost helps for all sizes of the train set (which has 50 000 images in total). The increase in average precision becomes more important as the train set becomes smaller (Table 2).

Table 2: Average highest test accuracy and 95% confidence interval of ResNet9 over 20 instances on CIFAR10 with training sets of different sizes (in %)

Training set size	ResNet	LAP-ResNet (Ours)
50 000	91.49, [91.40, 91.59]	**91.94**, [91.84, 92.04]
30 000	88.61, [88.47, 88.75]	**89.41**, [89.31, 89.50]
20 000	85.73, [85.59, 85.87]	**86.74**, [86.61, 86.87]
10 000	79.25, [79.00, 79.49]	**80.90**, [80.74, 81.06]
5 000	70.32, [70.00, 70.63]	**72.58**, [72.36, 72.79]
4 000	67.80, [67.55, 68.07]	**70.12**, [69.81, 70.42]

CIFAR100 experiments On CIFAR100, results using a ResNet are in Supplementary Material C.2. We also used the ResNeXt [36] architecture: the residual block of a ResNeXt applies $x + \sum_i w_i(x)$ with the functions w_i having the same architecture but independent weights, followed by a ReLU activation. We used the ResNeXt-50-32×4d architecture detailed in [36]. This is a much bigger and state-of-the-art network, as compared with the single representation ResNet used so far. It also extends the experimental results beyond the theoretical framework in three ways: the embedding dimension changes between the residual blocks, a block applies $x_{k+1} = \texttt{ReLU}(x_k + \sum_i w_{k,i}(x_k))$ and the encoder is no longer fixed. We found that penalizing $\sum_i w_{k,i}(x_k)$ or $x_{k+1} - x_k$ is essentially equivalent. Table 3 shows consistent accuracy gains as our method (with $\lambda_0 = 1$, $\tau = 0.1$ and $s = 5$) corrects a slight overfitting of the bigger ResNeXt compared to ResNet.

Table 3: Average highest test accuracy and 95% confidence interval of ResNeXt50 over 10 instances on CIFAR100 with training sets of different sizes (in %)

Training set size	ResNeXt	LAP-ResNeXt (Ours)
50 000	72.97, [71.79, 74.14]	**76.11**, [75.32, 76.89]
25 000	62.55, [60.18, 64.92]	**64.11**, [62.25, 65.96]
12 500	45.90, [43.16, 48.67]	**48.23**, [46.39, 50.07]

An important observation is that adding the transport cost significantly reduces the variance in the results. This is expected as the model becomes more constrained and can be seen as an advantage, especially in cases where the results vary more with the initialization (*e.g.* transfer learning). This is illustrated by the width of the 95% confidence intervals in the tables above often becoming narrower when the transport cost is penalized. Finally, we could also have considered a relaxation of the optimization program by considering a fixed weight λ, which provides a simpler and quite competitive benchmark (see Supplementary Material C.2). The training's progress is shown there as well, and we see that the training is not slowed down by our method.

7 Related work

That ResNets [18,19] are naturally biased towards minimally transforming their input, especially for later blocks and deeper networks, is already shown in [21], which found that earlier blocks learn new representations while later blocks only slowly refine those representations. [17] found that the deeper the network the more its blocks minimally move their input. Both were inspirations for this work. The ODE point of view of ResNets has inspired new architectures [7,15,23,29]. Others were inspired by numerical schemes to improve stability, e.g. [7] add a penalty term that encourages the weights to vary smoothly from layer to layer and [41] replicate an Euler scheme and study the effect of diminishing the discretization step-size. More recently, [37] accelerate the training of [8]'s model for generative tasks using the link with dynamical transport. But most often, regularization is achieved by penalization of the weights (e.g. spectral norm regularization [38], smoothly varying weights [7]).

OT theory was used in [33] to analyse deep gaussian denoising autoencoders (not necessarily implemented through residual networks) as transport systems. In the continuous limit, they are shown to transport the data distribution so as to decrease its entropy. Closer to this work, the dynamical formulation of OT is used in [9] for the problem of unsupervised domain translation.

8 Discussion and Conclusion

In this work, we have studied the behavior of ResNets by adopting a dynamical systems perspective. This viewpoint leverages the vast literature in this field.

More specifically, we have analyzed ResNets' complexity through the lens of the transport cost induced by the data displacement across the model's blocks. We find that due to a certain number of factors, this transport cost is biased towards small values. Moreover, this cost is negatively correlated to test accuracy, which has brought us to consider explicitly minimizing it. This leads us to present a novel generic formulation for training neural networks, based on the least action principle, closely related to the problem of Optimal Transport: amongst all the neural networks that correctly solve the task, select the one that transforms the data with the lowest cost. Note that even though we have only considered residual networks as they induce an ODE flow, this framework can be applied to any architecture by considering the static formulation (8) of the problem.

We have proven general results of existence and regularity for models trained within our framework, studied their behaviour in low-dimensional settings when compared to vanilla models and shown their efficiency on standard classification tasks. We also found that the training is stabilized in an adaptive fashion without being slowed down.

An important property of our method which is yet to be tested and is hinted at by the regularity results and by the lower variance in the performances is the robustness of the models, more specifically in adversarial contexts. This will be one important venue of future work. Another interesting avenue of research would be to experiment with alternative transportation costs.

References

1. L. Ambrosio, N. Gigli, and G. Savare. *Gradient Flows in Metric Spaces and in the Space of Probability Measures*. Birkhäuser Basel, 2005.
2. M. Belkin, D. Hsu, S. Ma, and S. Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *PNAS*, 2019.
3. M. Belkin, S. Ma, and S. Mandal. To understand deep learning we need to understand kernel learning. In *ICML*, 2018.
4. J. Benamou and Y. Brenier. A computational fluid mechanics solution to the monge-kantorovich mass transfer problem. *Numerische Mathematik*, 2000.
5. N. Bjorck et al. Understanding batch normalization. In *NIPS*, 2018.
6. F. Bolley. Separability and completeness for the wasserstein distance. In *Séminaire de Probabilités XLI*. Springer, 2008.
7. B. Chang et al. Reversible architectures for arbitrarily deep residual neural networks. In *AAAI*, 2018.
8. R. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. In *NIPS*, 2018.
9. E. de Bézennac, I. Ayed, and P. Gallinari. Optimal unsupervised domain translation. *arXiv*, 2019.
10. G. De Palma, B. Kiani, and S. Lloyd. Random deep neural networks are biased towards simple functions. In *NIPS*, 2019.
11. R. P. Feynman. The principle of least action in quantum mechanics. In *Feynman's Thesis - A New Approach to Quantum Theory*. World Scientific Publishing, 2005.
12. A. Figalli. *The Monge-Ampère Equation and Its Applications*. Zurich lectures in advanced mathematics. European Mathematical Society, 2017.

13. V. Garcia-Morales, J. Pellicer, and J. Manzanares. Thermodynamics based on the principle of least abbreviated action. *Annals of Physics*, 2008.
14. C. G. Gray. Principle of least action. *Scholarpedia*, 2009.
15. E. Haber, K. Lensink, E. Treister, and L. Ruthotto. IMEXnet a forward stable deep neural network. In *ICML*, 2019.
16. T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer-Verlag, 2001.
17. M. Hauser. On residual networks learning a perturbation from identity. *arXiv*, 2019.
18. K. He, X. Zhan, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *ECCV*, 2016.
19. K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.
20. A. Jacot, F. Gabriel, and C. Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *NIPS*, 2018.
21. S. Jastrzebski et al. Residual connections encourage iterative inference. In *ICLR*, 2018.
22. Q. Li, L. Chen, C. Tai, and W. E. Maximum principle based algorithms for deep learning. *JMLR*, 2018.
23. Y. Lu, A. Zhong, Q. Li, and B. Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *ICML*, 2018.
24. X. Ma, N. Trudinger, and X. Wang. Regularity of potential functions of the optimal transportation problem. *Archive for Rational Mechanics and Analysis*, 2005.
25. P. Nakkiran et al. Deep double descent: Where bigger models and more data hurt. In *ICLR*, 2020.
26. R. Novak, Y. Bahri, D. A. Abolafia, J. Pennington, and J. Sohl-Dickstein. Sensitivity and generalization in neural networks: an empirical study. In *ICLR*, 2018.
27. G. Peyre and M. Cuturi. *Computational Optimal Transport*. Now Publishers, 2019.
28. N. Rahaman et al. On the spectral bias of neural networks. In *ICML*, 2019.
29. L. Ruthotto and E. Haber. Deep neural networks motivated by partial differential equations. *J Math Imaging Vis*, 2020.
30. M. Sandler et al. Non-discriminative data or weak model? on the relative importance of data and model resolution. In *ICCVW*, 2019.
31. F. Santambrogio. *Optimal transport for Applied Mathematicians*. Birkhäuser, 2015.
32. A. M. Saxe, J. L. Mcclelland, and S. Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural network. In *ICLR*, 2014.
33. S. Sonoda and N. Murata. Transport analysis of infinitely deep neural network. *JMLR*, 2019.
34. C. Villani. *Optimal Transport: Old and New*. Springer-Verlag, 2008.
35. E. Weinan. A proposal on machine learning via dynamical systems. *Commun. Math. Stat*, 2017.
36. S. Xie et al. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.
37. H. Yan, J. Du, V. Tan, and J. Feng. On robustness of neural ordinary differential equations. In *ICLR*, 2020.
38. Y. Yoshida and T. Miyato. Spectral norm regularization for improving the generalizability of deep learning. *arXiv*, 2017.
39. S. Zagoruyko and N. Komodakis. Wide residual networks. In *BMVC*, 2016.
40. C. Zhang et al. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017.
41. J. Zhang et al. Towards robust resnet: A small step but a giant leap. In *IJCAI*, 2019.

A Some Elements of Optimal Transport Theory

We state here the most important results of Optimal Transport theory and its dynamical formulation. Our main reference is [31]. [34] is another classical reference. The dynamical formulation of OT has been of great importance, both theoretically and practically. It stems mainly from the work of Benamou and Brenier [4].

A.1 Optimal Transport

OT studies the task of "transporting" mass from one configuration to another while minimizing the effort as described by a certain ground cost c. Let α and β be two absolutely continuous distributions. The Monge formulation of OT is:

$$\begin{aligned} \min_{T} \quad & \mathcal{C}(T) = \int_{\mathbb{R}^d} c(x, T(x)) \mathrm{d}\alpha(x) \\ \text{s.t.} \quad & T_\sharp \alpha = \beta \end{aligned} \tag{12}$$

We then have the following result, proven for example in Theorem 1.17 of [31], which gives a condition on the cost under which problem (12) has a unique minimum.

Theorem 4. *α, β absolutely continuous measures on $\mathbb{R}^d$. If $c(x, y) = h(x - y)$, with h strictly convex, then there exists a unique T such that $\mathcal{C}(T)$ is minimal.*

A.2 Dynamical Formulation

Instead of directly pushing samples of α to β in $\mathbb{R}^d$, we can view α and β as points in a space of measures, and consider trajectories from α to β in this space.A way to transport the probability mass from α to β is a curve between two points in this space. The curve corresponding to the optimal mapping is the *shortest* one, in other words it is the *geodesic curve* between α and β. More formally, we introduce the *Wasserstein metric space* $\mathbb{W}_p(\mathbb{R}^d)$, *i.e.* the space of absolutely continuous measures of $\mathbb{R}^d$ with finite p-th moment endowed with the Wasserstein distance:

$$W_p(\mu, \nu) = \min_{T_\sharp \mu = \nu} \mathcal{C}(T)^{\frac{1}{p}}$$

when costs $c(x, y) = \|x - y\|_q^p$ are considered, for $q, p > 1$. The OT map can then be seen as a trajectory of minimal length between α and β, in other words a *geodesic*. The following result (from Theorem 5.27 of [31]) motivates this approach:

Theorem 5. *$\mathbb{W}_p$ is a geodesic space, meaning that, for any measures $\mu, \nu \in \mathbb{W}_p$, there exists a geodesic curve $(\mu_t)_{t\in[0,1]}$ between μ and ν.*

Thus, according to this result, finding the optimal mapping between two distributions amounts to finding a curve of minimal length in a certain abstract measure space. However, it still does not provide much in the way of a practically useful algorithm. The following theorem makes a formal link with fluid dynamics and basically states that moving probability masses from one distribution to another is the same as moving fluid densities from one configuration to another under a certain velocity field [31]:

Theorem 6. *Given α and β absolutely continuous w.r.t. the Lebesgue measure and $(\mu_t)_{t\in[0,1]}$ the geodesic curve with $\mu_0 = \alpha$ and $\mu_1 = \beta$, we can associate a vector field $v_t \in L^p(\mu_t)$ that solves the continuity equation*[3]*:*

$$\partial_t \mu_t + \nabla \cdot (\mu_t v_t) = 0$$

with:

$$W_p^p(\alpha, \beta) = \int_0^1 \|v_t\|^p_{L^p(\mu_t)} \mathrm{dt}$$

In other words, the geodesic curve $(\mu_t)_{t\in[0,1]}$ between the two distributions and the minimal energy velocity vector field v solve the continuity equation. Moreover, the energy along this path is precisely equal to the Wasserstein distance $W_p^p(\alpha, \beta)$. If this vector field of minimal energy v could be obtained, probability mass could be displaced according to the flow defined by the continuity equation, and the geodesic curve could be retrieved. Thus, we can reformulate the problem as a problem of optimal control, where v is the control variate:

$$\begin{aligned} \min_v \quad & \mathcal{C}^{\mathrm{dyn}}(v) = \int_0^1 \|v_t\|^p_{L^p(\mu_t)} \mathrm{d}t \\ \text{s.t.} \quad & \partial_t \mu_t + \nabla \cdot (\mu_t v_t) = 0, \mu_0 = \alpha, \mu_1 = \beta \end{aligned} \tag{13}$$

B Regularity of Optimal Transport Maps

In this section, we recall some classical and more recent results of regularity for Optimal Transport mappings. This is an intricate subject and the problem was open for some time after OT theory had been established. The most important results have been established through the study of the Monge-Ampère equation by Caffarelli then De Philippis and Figalli. Extensions for larger families of costs were developed by Ma, Trudinger and Wang [24] but this is out of the scope of this work. In particular, Theorem 6.27 of [1] gives a classical almost-everywhere regularity result:

Theorem 7. *If $c(x, y) = \|x - y\|^p$ for $p > 1$, and α and β have compact supports with $d(supp(\alpha), supp(\beta)) > 0$, then the optimal transportation map T between α and β is $\alpha - a.e.$ differentiable and its Jacobian $\nabla T(x)$ has non-negative eigenvalues $\alpha - a.s.$*

[3] ∂_t is the partial derivative operator *w.r.t.* variable t, and $\nabla\cdot$ the divergence operator *w.r.t.* space.

More recently, results summarized below, which correspond to Theorems 4.23, 4.24 and Remark 4.25 of [12], state that the optimal transportation map has one degree of regularity more than the initial transported density:

Theorem 8. *Suppose there are X, Y, bounded open sets, such that the densities of α and β are null in their respective complements and bounded away from zero and infinity over them respectively. Then, if Y is convex, there exists $\eta > 0$ such that the OT map T between α and β is $C^{0,\eta}$ over X. If Y isn't convex, there exists two relatively closed sets A, B in X, Y respectively, such that $T \in C^{0,\eta}(X \setminus A, Y \setminus B)$, where A and B are of null Lebesgue measure.*

Moreover, if the densities are in $C^{k,\eta}$, then $C^{0,\eta}$ can be replaced by $C^{k+1,\eta}$ in the conclusions above. In particular, if the densities are smooth, then the transport map is a diffeomorphism (between the reduced input and target domains if the target support is not convex).

C Additional Results

C.1 Visualization of the Transport on MNIST

If we pretrain an autoencoder on MNIST, we can use its encoder as the encoder of the ResNet and freeze it during training. This makes it possible to visualize the transport of the data by decoding, using the pretrained decoder, the output of each residual block. We show this below on MNIST. In Figure 3, we see the decodings of a basic ResNet trained to achieve 99.4% test accuracy.

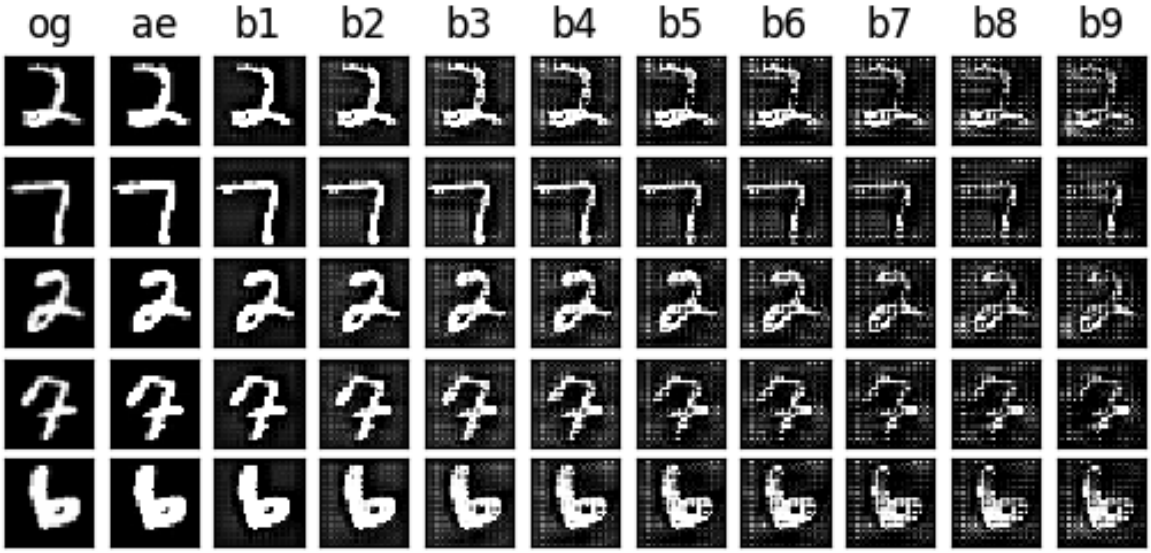

Fig. 3. Decodings of the internal representations (the outputs of the residual blocks) after training a ResNet9 on MNIST (og: original image, ae: encoding, b1: output of block 1...)

We add the transport cost with $\lambda_0 = 5$, $\tau = 1$ and $s = 5$. The performance barely drops (99.3% test accuracy) but we can see in Figure 4 the effect of the regularization as the decodings change much less.

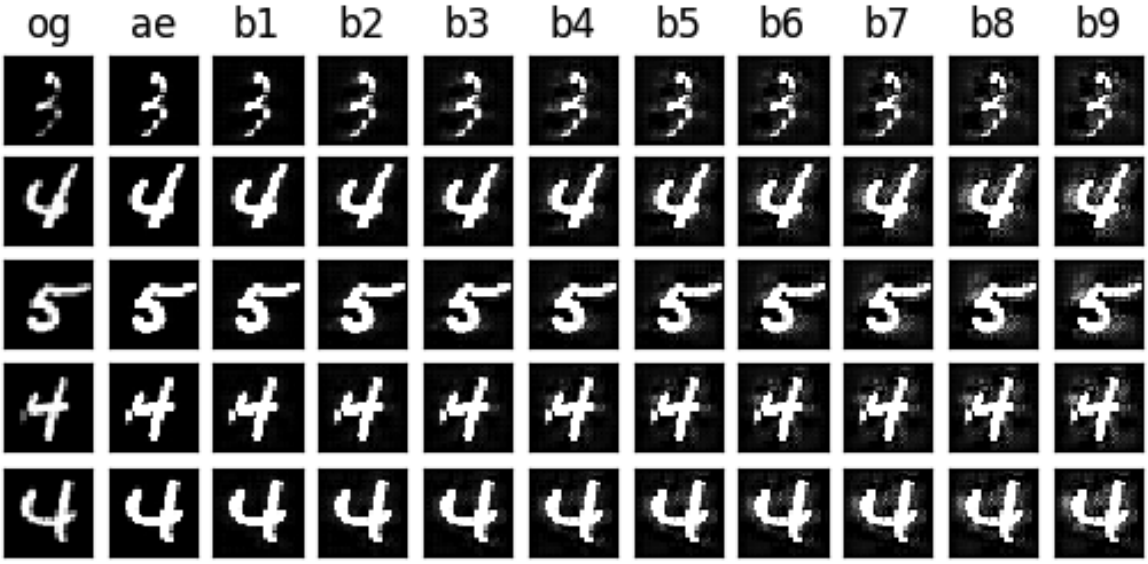

Fig. 4. Decodings of the internal representations (the outputs of the residual blocks) after training a LAP-ResNet9 on MNIST (og: original image, ae: encoding, b1: output of block 1...)

C.2 Additional Results with Fixed λ

In this section, we show some additional experimental results with a model where, instead of using an adaptive optimization algorithm, we simply take the transport cost as a regularizer, thus giving us a minimization objective:

$$\mathcal{L}(\theta) + \lambda \;\; \mathcal{C}(\theta)$$

This is an easier and more straightforward method, simply considering a relaxed constraint in the optimization problem. Aside from the advantage of simpler implementation, it allows for easier fine-tuning of the regularization hyper-parameter which is useful when the datasets and networks are big. The adaptivity is lost but this still leads to better test accuracy than the non-regularized networks. Results on the same tasks as in Section 6 are below.

Table 4: Average highest test accuracy and 95% confidence interval of ResNet9 over 20 instances on CIFAR10 with training sets of different sizes (in %)

Training set size	ResNet	LAP-ResNet	Regularized ResNet, $\lambda = 0.2$
50 000	91.49, [91.40, 91.59]	**91.94**, [91.84, 92.04]	91.36, [91.28, 91.44]
30 000	88.61, [88.47, 88.75]	**89.41**, [89.31, 89.50]	88.50, [88.38, 88.61]
20 000	85.73, [85.59, 85.87]	**86.74**, [86.61, 86.87]	85.82, [85.70, 85.93]
10 000	79.25, [79.00, 79.49]	**80.90**, [80.74, 81.06]	80.15, [80.02, 80.28]
5 000	70.32, [70.00, 70.63]	**72.58**, [72.36, 72.79]	72.03, [71.71, 72.34]
4 000	67.80, [67.55, 68.07]	**70.12**, [69.81, 70.42]	69.64, [69.35, 69.94]
1 000	49.22, [48.69, 49.74]	**51.14**, [50.69, 51.59]	50.38, [49.92, 50.82]
500	41.55, [41.14, 41.96]	**42.92**, [42.54, 43.29]	42.30, [41.88, 42.73]
100	26.98, [25.98, 27.97]	25.34, [24.63, 26.10]	**27.53**, [26.59, 28.47]

Table 5: Average highest test accuracy and 95% confidence interval of ResNet9 over 10 instances on CIFAR100 with training sets of different sizes (in %)

Training set size	ResNet	LAP-ResNet	Regularized ResNet, $\lambda \in \{0.05, 0.2\}$
50 000	72.32, [72.08, 72.56]	72.43, [72.25, 72.61]	**72.62**, [72.41, 72.83]
25 000	64.34, [64.10, 64.57]	64.34, [64.11, 64.58]	**64.76**, [64.52, 65.00]
10 000	49.27, [48.84, 49.69]	**50.57**, [50.34, 50.80]	50.46, [50.19, 50.72]
5 000	34.74, [33.90, 35.58]	37.97, [37.68, 38.27]	**38.44**, [37.99, 38.89]
1 000	15.66, [15.23, 16.08]	**16.42**, [16.10, 16.75]	16.03, [15.55, 16.52]

Table 6: Average highest test accuracy and 95% confidence interval of ResNeXt50 over 10 instances on CIFAR100 with training sets of different sizes (in %)

Training set size	ResNeXt	LAP-ResNeXt	Regularized ResNeXt, $\lambda = 0.01$
50 000	72.97, [71.79, 74.14]	**76.11**, [75.32, 76.89]	75.96, [74.92, 77.01]
25 000	62.55, [60.18,64.92]	**64.11**, [62.25, 65.96]	64.10, [62.36, 65.84]
12 500	45.90, [43.16, 48.67]	**48.23**, [46.39, 50.07]	47.77, [45.93, 49.62]

Finally, we point out that the least action principle acts by speeding up training in the first epochs as seen for the training of ResNeXt50 models on CIFAR100 in Figure 5. Batch training times are similar for the 3 models in Figure 5 on the same hardware (around 0.7 seconds).

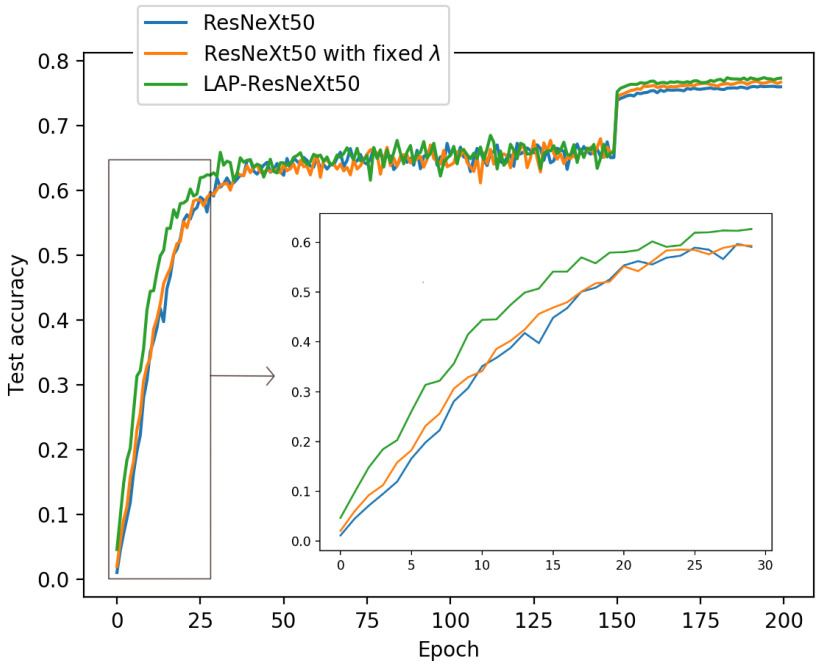

Fig. 5. Test accuracy during training of ResNeXt50 models on CIFAR100

C.3 Implementation Details

These are further implementation details about the experiments in Sections 6 and C.2. Orthogonal initialization with gain 0.01 is used for all ResNet models. Kaiming initialization is used for all ResNeXt models. SGD is used for training all models. The momentum is always set to 0.9 and weight decay to 0.0001. For ResNet models, the learning rate is 0.01 and is divided by 5 at epochs 120, 160 and 200 (when the training goes that far). For ResNeXt models, the learning rate is 0.1 and is divided by 10 at epochs 150, 225 and 250. Batch size is 128 for all experiments. Architectures of ResNet [18] and ResNeXt [36] blocks are standard and exactly as in the references. The ResNets used are single representation ResNets (i.e. containing one residual stage) with 9 blocks. The ResNeXt architecture used is the ResNeXt-50-32×4d from [36].

C.4 Additional Results on 2D Toy Data

Here is a comparison of our method with batch normalization (BN), which is known to impact the loss surface's geometry [5].We find that our method cooperates well with BN to improve test accuracy on the same 2D task as in Section 4 when the model is too small (1 block, Table 7), too big (100 blocks,

Table 8), badly initialized ($\mathcal{N}(0, 5)$ initialization, Table 9) and when the dataset is small (50 points, Table 10). LAP-ResNets use $\lambda_0 = 0.1$, $\tau = 0.1$ and $s = 5$.

Table 7: Average test accuracy and 95% confidence interval over 100 instances on the circles 2D dataset with 1000 points and 1 block (in %)

	No batch normalization	Batch normalization
ResNet	76.6, [73.1, 80.2]	75.4, [72.3, 78.6]
Regularized ResNet, $\lambda = 0.005$	76.5, [73.0, 80.0]	75.6, [72.2, 78.9]
LAP-ResNet	82.1, [79.5, 84.7]	**84.6**, [81.5, 87.6]

Table 8: Average test accuracy and 95% confidence interval over 100 instances on the circles 2D dataset with 1000 points and 100 blocks (in %)

	No batch normalization	Batch normalization
ResNet	89.1, [87.2, 91.00]	99.4, [99.0, 99.8]
Regularized ResNet, $\lambda = 0.09$	69.7, [65.6, 73.7]	99.5, [98.9, 1.00]
LAP-ResNet	75.7, [72.8, 78.6]	**99.8**, [99.7, 1.00]

Table 9: Average test accuracy and 95% confidence interval over 100 instances on the circles 2D dataset with a $\mathcal{N}(0, 5)$ initialization (in %)

	No batch normalization	Batch normalization
ResNet	90.2, [88.8, 91.5]	98.0, [97.2, 98.8]
Regularized ResNet, $\lambda = 0.04$	89.7, [88.2, 91.3]	**99.7**, [99.5, 99.9]
LAP-ResNet	79.1, [75.3, 83.0]	99.4, [99.0, 99.8]

Table 10: Average test accuracy and 95% confidence interval over 100 instances on the circles 2D dataset with 50 points and 9 blocks (in %)

	No batch normalization	Batch normalization
ResNet	88.2, [85.5, 90.1]	92.9, [90.9, 94.9]
Regularized ResNet, $\lambda = 0.04$	93.5, [91.4, 95.6]	94.4, [92.4, 96.3]
LAP-ResNet	95.8, [94.0, 97.6]	**96.0**, [94.6, 97.3]