Lifted Regression/Reconstruction Networks

Lifted Regression/Reconstruction Networks

Rasmus Kjær Høier
hier@chalmers.se
Christopher Zach
zach@chalmers.se
Chalmers University of Technology
Gothenburg, Sweden

Abstract

In this work we propose *lifted regression/reconstruction networks* (LRRNs), which combine lifted neural networks with a guaranteed Lipschitz continuity property for the output layer. Lifted neural networks explicitly optimize an energy model to infer the unit activations and therefore—in contrast to standard feed-forward neural networks—allow bidirectional feedback between layers. So far lifted neural networks have been modelled around standard feed-forward architectures. We propose to take further advantage of the feedback property by letting the layers simultaneously perform regression and reconstruction. The resulting lifted network architecture allows to control the desired amount of Lipschitz continuity, which is an important feature to obtain adversarially robust regression and classification methods. We analyse and numerically demonstrate applications for unsupervised and supervised learning.

1 Introduction

Deep neural networks (DNNs) are very powerful and expressive tools in machine learning to solve many classification and regression tasks. Due to their expressiveness and highly non-linear properties, DNNs are generally very sensitive to minor perturbations of the input, which makes them unreliable in e.g. safety-critical applications. A quickly growing body of works aims to obtain robust DNNs either by design or by a dedicated training approach such as adversarial training. In this work the main goal is to obtain powerful regression methods that are robust to input perturbations by design. This is achieved by controlling the Lipschitz continuity of the input-to-output mapping, which puts limits on the sensitivity of a mapping with respect to bounded input perturbations (w.r.t. the Euclidean norm). Thus, robustness of the prediction is guaranteed even for unseen test samples and need not to be established via an expensive verification procedure.

We further step away from pure feed-forward architectures for neural networks, but base our classification and regression approach on layered energy-based models, which enable bidirectional feedback between layers when determining the internal network activations. Feed-forward DNNs can be obtained from such layered energy models as a limit case, hence energy-based models can be considered as powerful as regular DNNs. To our knowledge the Lipschitz properties of such energy-based models have not been considered in the literature. We propose a simple energy model that guarantees non-expansiveness of the mapping from input to output activations essentially by symmetrizing an energy model. Consequently each layer in the underlying energy has a regression and a reconstruction component. In contrast

to existing literature on Lipschitz continuous DNNs, no difficult-to-enforce constraints on the weight matrices (such as orthonormality) are needed in our framework.

2 Related Work

Lifted DNNs Lifted neural networks introduce an explicit set of unknowns for the internal network activations, and inference is performed by minimization (or marginalization) w.r.t. the network activations. Hence, they are based on a different computational model than regular feed-forward networks. Lifted neural networks can be traced back to two somewhat different origins. First, they can be seen as instances of more general undirected energy-based models rooted in (restricted) Boltzmann machines [1, 13, 29] and a contrastive learning paradigm, where the learning signal is induced by the energy difference between fully and partially clamped visible units (e.g. [23]). It has been demonstrated, that back-propagation is a limit case of contrastive learning [28, 37, 39] for appropriate layered energy models.

A more recent motivation for lifted networks is the ability of highly parallel training procedures [5], which proposes a quadratic relaxation for the feedforward computation in a DNN, and using modern optimization methods such as ADMM [31]. The ability to use convex energy models for e.g. ReLU networks is connected with re-interpreting the ReLU operation as projection to the non-negative real line [2, 40]. The convex energy models used in our work imitate feedforward networks only in so called weak feedback setting. [11, 19] propose (block-convex but not jointly convex) lifted network models that aim to replicate the standard feed-forward pass in a DNN for a general class of activation functions.

DNN Robustness The discovery of the intrinsic brittleness of predictions made by deep neural networks [30] has led to a significant amount of research on better constructing adversarial inputs (e.g. [4, 9, 17, 20, 22]) and making neural-network based classifiers more robust with respect to adversarial perturbations (e.g. by using a robust training loss [20, 30, 35] or Lipschitz regularization [6, 24, 32, 38]). Determining adversarial perturbation amounts to minimizing a highly non-linear and non-convex optimization problem, and therefore an explicit search for adversarial examples cannot certify robustness of the network. Recently, it was empirically shown, that adversarial training is insufficient by using computationally expensive attacks [24, 33] and therefore leads to a false sense of robustness. These results strongly motivate the design of intrinsically robust neural networks architectures. Scattering networks [3, 21] are wavelet-based, non-trainable feature representations combining Lipschitz continuity with transformation invariance. Parseval networks [6] aim for intrinsic 1-Lipschitz continuity of a trained DNN by favoring orthonormal weight matrices. Non-expansive networks [24] propose to utilize (approximately) distance preserving network layers, and Lipschitz margin training [32] estimates the Lipschitz constant during the training phase and uses it to adjust a required classification margin.

A complementary approach is to verify robustness of a DNN for a particular input sample via robust optimization techniques. Networks with general piece-wise linear activation functions can be verified using linear programming relaxations [34, 35], which emerge immediately from exact, but not scalable, mixed integer-linear programs [8, 15]. Stronger (but computationally demanding) relaxations can be obtained via semi-definite programming [25, 26].

3 Lifted Regression/Reconstruction Networks (LRRN)

In this section we propose a lifted network energy that is by construction Lipschitz continuous with a user-specified Lipschitz constant. In contrast to lifted networks proposed in [11, 19, 39, 40] aiming to mimic the behavior of feed-forward DNNs, we add a reconstructive term to the network energy model. Thus, we propose to use a network energy of the form

$$E(z;x) = \frac{1}{2}\sum_{k=0}^{L-1}\left(\|z_{k+1} - W_k z_k - b_k\|^2 + \beta_k \left\|W_k^T z_{k+1} - z_k - c_k\right\|^2\right) \tag{1}$$

subject to $z_0 = x$ and $z_k \in \mathcal{C}_k$ for $k = 1,\dots,L$. Each $\mathcal{C}_k \subseteq \mathbb{R}^{d_k}$, $k = 1,\dots,L$, is a closed convex set, which can be used to introduce non-linear behaviour. The choices $\mathcal{C}_k = \mathbb{R}^{d_k}$ (linear activation function) and $\mathcal{C}_k = \mathbb{R}^{d_k}_{\geq 0}$ (ReLU-like activation function) are of particular interest.

Obtaining a prediction from the energy models requires computing the activations $z^*(x) = \arg\min_z E(z;x)$ by solving a (strictly) convex program. The last layer z_L is considered the output layer, hence the mapping $x \mapsto z_L^*(x)$ corresponds to the network's prediction function.

The parameters $\beta_k \geq 0$, $k = 1,\dots,L$, control the Lipschitz constant of $z_L^*(x)$ as we will see shortly. Setting $\beta_k = 0$ for all k yields a pure forward regression network resembling standard feed-forward DNNs [39]. We also tie the forward (regression) weights W_k and the reconstructive weights W_k^T, but not the biases b_k and c_k as tying them has no impact on Lipschitz continuity.

3.1 Motivation: Lipschitz continuity of linear 1-layer LRRNs

We consider first an LRRN with a single layer and no constraint on the latent variables. Thus, the energy model in Eq. 1 reduces to

$$E(z;x) = \tfrac{1}{2}\|z - Wx - b\|^2 + \tfrac{\beta}{2}\left\|W^T z - x - c\right\|^2. \tag{2}$$

The first order optimality condition for z^* for given x is

$$(\mathtt{I} + \beta WW^T)z^* = Wx + b + \beta Wx + \beta Wc = (1+\beta)Wx + b + \beta Wc.$$

Thus, $z^*(x)$ is explicitly given by

$$z^*(x) := (1+\beta)(\mathtt{I} + \beta WW^T)^{-1}Wx + \underbrace{(\mathtt{I} + \beta WW^T)^{-1}(b + \beta Wc)}_{=:\tilde{c}}. \tag{3}$$

Using the singular value decomposition of $W = U\Sigma V^T$, and therefore $\mathtt{I} + \beta WW^T = U(\mathtt{I} + \beta\Sigma^2)U^T$, this translates to

$$z^*(x) = U(1+\beta)(\mathtt{I} + \beta\Sigma^2)^{-1}\Sigma V^T x + b =: A_\beta x + \tilde{c}$$

The diagonal matrix $(1+\beta)(\mathtt{I} + \beta\Sigma^2)^{-1}\Sigma$ has the elements $(1+\beta)\sigma_i/(1+\beta\sigma_i^2) \geq 0$ on its main diagonal. The mapping $f_\beta : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ with

$$f_\beta(\sigma) := \frac{(1+\beta)\sigma}{1+\beta\sigma^2}$$

has a single maximum at $\sigma = \sqrt{1/\beta}$. Thus,

$$f_\beta(\sqrt{1/\beta}) = \frac{(1+\beta)\beta^{-1/2}}{1+\beta\beta^{-1}} = \frac{(1+\beta)\beta^{-1/2}}{2} = \frac{\beta^{1/2}+\beta^{-1/2}}{2}$$

This means that the singular values of A_β are in $[0, (\beta^{1/2}+\beta^{-1/2})/2]$, and the mapping $x \mapsto A_\beta x + \tilde{c}$ is Lipschitz continuous with constant $(\beta^{1/2}+\beta^{-1/2})/2$ (which is an upper bound on the operator norm of A_β). With $\beta = 1$ one has $\|A_\beta\|_2 \leq 1$.

Remark 1. The operator norm of A_β can be explicitly stated as $\max_i\{\frac{(1+\beta)\sigma_i}{1+\beta\sigma_i^2}\}$, where $(\sigma_i)_i$ are the singular values of W. The maximum is attained for the singular value σ_i that is "closest" (in a certain sense) to $1/\beta$. If $\beta \to 0$, then the largest of $\{\sigma_i\}$ yields the operator norm. Therefore in this setting the stated Lipschitz constant is explicitly dependent on β and on the singular values $(\sigma_i)_i$.

3.2 Lipschitz continuity of proximal-like operators

In the previous section the Lipschitz continuity of the mapping $x \mapsto \arg\min_z \|z - Wx - b\|^2/2 + \beta\|W^T z - x - b\|^2/2 = (1+\beta)(\mathtt{I} + \beta W^T W)^{-1} Wx + \tilde{c}$ was established. In order to add constraints on z (such as non-negativity constraints to obtain a non-linear mapping) and to analyse deeper LRRNs we need a more general approach. We are intersted in the Lipschitz properties of the function

$$\mathcal{P}_{\beta,W,G} : x \mapsto \arg\min_z \tfrac{1}{2}\|z - Wx\|^2 + \tfrac{\beta}{2}\|W^T z - x\|^2 + G(z), \tag{4}$$

where $G(z)$ is essentially an arbitrary convex function (not necessarily differentiable). Since $\|z - Wx\|^2/2$ is strictly convex in z, the minimizer in the l.h.s. of Eq. 4 is unique and therefore $\mathcal{P}_{\beta,W,G}$ is a proper function. If $W = \mathtt{I}$, then the above mapping reduces to

$$x \mapsto \arg\min_z (1+\beta)\|z - x\|^2 + G(z), \tag{5}$$

which is known as proximal operator $x \mapsto \mathrm{prox}_{G/(1+\beta)}(x)$ in the convex optimization literature. Proximal operators are firmly non-expansive and therefore 1-Lipschitz continuous. This property is extended in a suitable way to $\mathcal{P}_{\beta,W,G}$:

Lemma 1. *Let G be any proper l.s.c. convex function, $W \in \mathbb{R}^{n\times m}$ a matrix with compatible dimensions, and $\beta \geq 0$. Then $\mathcal{P}_{\beta,W,G}$ is $\frac{\beta^{1/2}+\beta^{-1/2}}{2}$-Lipschitz.*

Proof. The optimal $z^* = z^*(x)$ of $\|z - Wx\|^2/2 + \beta\|W^T z - x\|^2/2 + G(z)$ is determined by the optimality condition

$$0 = (\mathtt{I} + \beta WW^T)z^* - (1+\beta)Wx + g, \tag{6}$$

where $g \in \partial G(z^*)$ is a subgradient of G at z^*. Since the subgradient is a monotone operator, it satisfies $(g_1 - g_2)^T(z_1 - z_2)$ for all $g_i \in \partial G(z_i)$, $i = 1,2$. Choosing $z_i = z_i^* = z^*(x_i)$ and inserting $g_i = (1+\beta)Wx_i - (\mathtt{I} + \beta WW^T)z_i^*$ yields

$$\big((1+\beta)W(x_1 - x_2) - (\mathtt{I} + \beta WW^T)(z_1^* - z_2^*)\big)(z_1^* - z_2^*) \geq 0 \tag{7}$$

or

$$(1+\beta)(x_1-x_2)^T W^T (z_1^*-z_2^*) \geq (z_1^*-z_2^*)^T (\mathtt{I}+\beta W W^T)(z_1^*-z_2^*). \tag{8}$$

We introduce $u := x_1 - x_2$ and $v := z_1^* - z_2^*$. Among all u with a fixed Euclidean norm $\delta \geq 0$, the vector u leading to the largest l.h.s. is given by $u = \alpha W^T v$ (using the Cauchy-Schwarz inequality), where $\alpha \geq 0$ satisfies $\alpha \|W^T v\| = \delta$. Hence, the above constraint can be restated as

$$(1+\beta)\alpha v^T W W^T v \geq v^T(\mathtt{I}+\beta W W^T)v \qquad \text{or} \qquad \|v\|^2 \leq \big((1+\beta)\alpha - \beta\big)\|W^T v\|^2. \tag{9}$$

This induces the constraint $\alpha \geq \beta/(1+\beta)$ for the solution to be feasible. Inserting $u = W^T v/\alpha$ and rearranging yields

$$\frac{\|v\|}{\|u\|} = \frac{\|v\|}{\alpha\|W^T v\|} \leq \frac{\sqrt{(1+\beta)\alpha-\beta}}{\alpha} \tag{10}$$

for all feasible $\alpha \geq \beta/(1+\beta)$. It is straightforward to verify that the mapping $f_\beta(\alpha) := \frac{\sqrt{(1+\beta)\alpha-\beta}}{\alpha}$ with domain $[\beta/(1+\beta), \infty)$ has range $[0, (\beta^{1/2}+\beta^{-1/2})/2]$, where the maximum is attained at $\alpha^* = 2\beta/(1+\beta)$. Hence,

$$\frac{\|z_1^*-z_2^*\|}{\|x_1-x_2\|} \leq \frac{\beta^{1/2}+\beta^{-1/2}}{2} \qquad \text{i.e.} \qquad \|z_1^*-z_2^*\| \leq \frac{\beta^{1/2}+\beta^{-1/2}}{2}\|x_1-x_2\|, \tag{11}$$

which completes the proof. □

Consistent with Section 3.1 the smallest Lipschitz constant is obtained by setting $\beta = 1$, which yields the following corollary:

Corollary 1. *$\mathcal{P}_{1,W,G}$ is 1-Lipschitz.*

Unlike in Section 3.1 (cf. Remark 1) the provided Lipschitz constant only depends on β, and for $\beta \to 0$ the Lemma above yields a vacuous bound. Nevertheless, one has the following simple lemma:

Lemma 2. *$\mathcal{P}_{0,W,G}$ is $\|W\|_2$-Lipschitz.*

Proof. We have

$$\mathcal{P}_{0,W,G}(x) = \arg\min_z \tfrac{1}{2}\|z - Wx\|^2 + G(z) = \operatorname{prox}_G(Wx) = (\operatorname{prox}_G \circ (W\cdot))(x), \tag{12}$$

i.e. $\mathcal{P}_{0,W,G}(x)$ is the composition of a linear mapping with a proximal step. Since the Lipschitz constant of the mapping $x \mapsto Wx$ is $\|W\|_2$ and the proximal operator is 1-Lipschitz, we deduce that the Lipschitz constant of $\mathcal{P}_{0,W,G}$ is at most $\|W\|_2$. □

In practice we are mostly interested in the choice of $\beta = 1$ (both regression and full reconstructive terms) and $\beta = 0$ (pure regression term).

3.3 General LRRNs

Using Lemma 1 the analysis of the energy underlying the lifted regression/reconstruction networks (Eq. 1) is relatively straightforward. We define for $k = 1, \dots, L$

$$\rho_k := \begin{cases} \frac{\beta_k^{1/2} + \beta_k^{-1/2}}{2} & \text{if } \beta_k > 0 \\ \|W_k\|_2 & \text{if } \beta_k = 0. \end{cases} \tag{13}$$

Theorem 1. *For the layered energy model given in Eq. 1 let $z^*(x) = \arg\min_z E(z,x)$ be the minimizer for given input x. Then $x \mapsto z_L^*(x)$ (i.e. the mapping from the input to the last layer latent variables) is Lipschitz continuous with constant $\prod_{k=1}^L \rho_k$.*

Proof. Let $z_1^*(x)$ be given by

$$\begin{aligned} z_1^*(x) &= \arg\min_{z_1} \min_{z_2,\dots,z_L} E((z_1,\dots,z_L),x) \\ &= \arg\min_{z_1 \in \mathcal{C}_1} \min_{z_2 \in \mathcal{C}_2,\dots,z_L \in \mathcal{C}_L} \tfrac{1}{2} \sum \|z_{k+1} - W_k z_k - b_k\|^2 + \tfrac{\beta_k}{2} \sum \left\| z_k - W_k^T z_{k+1} - c_k \right\|^2. \end{aligned} \tag{14}$$

Since minimizing out variables in a jointly convex function yields a convex function in the remaining unknowns, we can write the above as

$$z_1^*(x) = \arg\min_{z_1 \in \mathcal{C}_1} \tfrac{1}{2} \|z_1 - W_0 x\|^2 + \tfrac{\beta_1}{2} \left\| W_0^T z_1 - x \right\|^2 + G_1(z_1) = P_{\beta_1, W_0, G_1}(x). \tag{15}$$

Hence, $z_1^*(x)$ is ρ_1-Lipschitz due to Lemmas 1 and 2. Due to the layered structure z_2^* only depends on $z_1^* = z_1^*(x)$, therefore

$$z_2^*(z_1^*) = \arg\min_{z_2 \in \mathcal{C}_2} \tfrac{1}{2} \|z_2 - W_1 z_1^*\|^2 + \tfrac{\beta_2}{2} \left\| W_1^T z_2 - z_1^* \right\|^2 + G_2(z_2) = P_{\beta_2, W_1, G_2}(z_1^*) \tag{16}$$

for a suitable convex function G_2. Hence, $z_2^*(z_1^*)$ is ρ_2-Lipschitz. Applying this argument iteratively on the remaining layers, we find that $z_k^*(z_{k-1}^*)$ is ρ_k-Lipschitz. Further, $z_L^*(x) = \left(z_L^* \circ z_{L-1}^* \circ \cdots \circ z_1^*\right)(x)$, the Lipschitz constant of $z_L^*(x)$ is at most $\prod_{k=1}^L \rho_k$. □

Corollary 2. *Let $\beta_1 = \cdots = \beta_{L-1} = 1$ and $\beta_L = 0$ (i.e. the output layer is a pure regression layer). Then the Lipschitz constant of $x \mapsto z_L^*(x)$ is at most $\|W_{L-1}\|_2$.*

Networks with such a choice for $(\beta_k)_{k=1}^L$ have a clear interpretation: the first $L-1$ layers extract non-expansive feature representations, and the last layer is an arbitrary linear regression layer to generate the target output. Hence, the Lipschitz properties of the network (and therefore the robustness with respect to input perturbations) can be assessed by inspecting the last layer matrix W_{L-1}.

Remark 2. The network energy in Eq. 1 allows direct feedback from a subsequent layer to the previous one (and therefore indirect feedback to all earlier layers). This feedback from later layers can be essentially suppressed by using discounted terms [37, 40],

$$E(z;x) = \frac{1}{2} \sum_{k=0}^{L-1} \gamma^k \left(\|z_{k+1} - W_k z_k - b_k\|^2 + \beta_k \left\| W_k^T z_{k+1} - z_k - c_k \right\|^2 \right) \tag{17}$$

for a *feedback parameter* $\gamma > 0$. With $\gamma \to 0$ one recovers a feed-forward DNN, and the contrastive learning approach for supervised training (see Section 4.2) is equivalent to back-propagation. In that sense energy-based models such as Eqs. 1 and 17 are more general

than feed-forward networks. Observe that Eq. 1 and Eq. 17 are actually equivalent, since the feedback weight γ^k can be absorbed by reparametrizing the weights W_k, biases b_k and c_k, and the activations z_k. Nevertheless, the feedback parameter still influences initialization of the network parameters and any weight regularization term.

Implementing LRRNs Determining $z^*(x)$ requires minimizing a strictly (even strongly) convex, possibly constrained, optimization problem Eq. 1. The easiest method to solve such a task is coordinate descent, which minimizes sequentially the scalar network activations $\{z_{kj}\}$ (with k iterating over all layers and j traversing units in the current layer). For many relevant constraint sets $\mathcal{C}_k$ the optimal solution for z_{kj} after fixing all other activations can be stated in closed form. Hence, we employ coordinate descent in our implementation.

Feed-forward DNNs with RR layers Since the ReLU activation function is 1-Lipschitz, it is possible to stack *linear* recognition+reconstruction (RR) layers followed by ReLU non-linearities to obtain 1-Lipschitz feed-forward DNNs. Unfortunately, back-propagation in this setting requires expensive matrix inversions. As shown in Section 4.2, using lifted NNs avoids such explicit matrix inverses and further allows *non-linear* RR layers.

4 Learning with LRRNs

In this section we briefly discuss the application of LRRNs for unsupervised and supervised learning. In the unsupervised setting LRRNs generalize subspace learning, and supervised learning requires a non-standard approach since back-propagation is not directly applicable for energy-based network models. We show results for the MNIST [18], Fashion-MNIST (FMNIST, [36]), Kuzushiji-MNIST (KMNIST, [7]) and CIFAR [16] datasets. Training of the models is performed by stochastic gradient descent, where the activations z^* are first inferred using coordinate descent, and the contribution of a training sample to the gradient is based on these activations, e.g. $\nabla_{W_k} E(z^*;x) = (W_k z_k^* - z_{k-1}^* + b_k)(z_k^*)^T + \beta_k z_{k+1}^* (W_k z_{k+1}^* - z_k^* - c_k)^T$.

4.1 Unsupervised setting

Let $\{x_i\}_{i=1}^N$ be a set of unlabelled training samples. One question is whether a sensible energy model can be obtained by solely minimizing the average energy of the training samples, i.e.

$$\min_\theta J(\theta) = \min_\theta \tfrac{1}{N} \sum_i \min_z E(z;x_i) = \min_\theta \tfrac{1}{N} \sum_i E(z^*(x_i);x_i), \tag{18}$$

where θ contains all the weights and biases in the energy model (Eq. 1). Since in this setting we are not interested in the output layer, $E(z;x)$ reduces to

$$E(z_1;x) = \tfrac{1}{2} \|z_1 - W_0 x\|^2 + \tfrac{1}{2} \left\|W_0^T z_1 - x\right\|^2 + G_1(z_1), \tag{19}$$

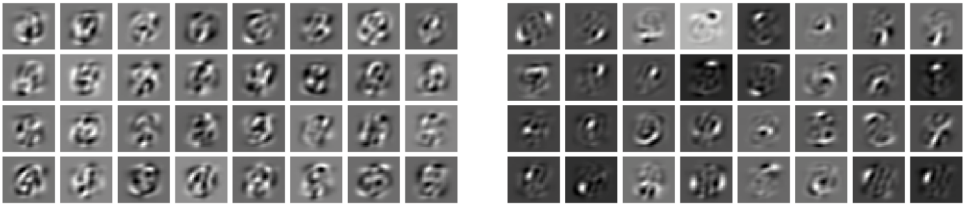
where G_1 is a convex function obtained by minimizing out all subsequent layers $z_2, \dots, z_L$, and G_1 acts therefore as a (learnable) prior on z_1. We also absorbed the bias terms into G_1.

Since $E(z;x)$ induces an *unnormalized* energy model $E(x) = \min_z E(z;x)$, it is not immediately clear that the loss in Eq. 18 (which can also be seen as maximizing an unnormalized probability) leads to any desired energy model. It can be shown that if G_1 is coercive (i.e. $G_1(z_1) \to \infty$ as $\|z_1\| \to \infty$), then $E(x)$ is also coercive, and therefore samples with small

energy (thus highly likely ones) are concentrated to bounded (convex) sets. Hence, proper shaping of G_1 is somewhat analogous to the "bottleneck" in standard auto-encoders.

Note that Eq. 19 resembles an auto-encoder with a single hidden layer: the first term defines the encoder, the second term is a reconstruction error and therefore corresponds to the decoder, and the last term represents the prior on the latent variables. If $G_1(z_1) \equiv 0$ (which means that the underlying LRRN has exactly one linear layer), it can be shown that Eq. 18 essentially performs an eigen-decomposition of the scatter matrix $\sum_i x_i x_i^T$, and is therefore strongly connected to PCA and subspace learning. If all training points $\{x_i\}$ lie in an r-dimensional subspace, and $\dim(z_1) = r$, then W_0^T is an orthonormal matrix satisfying $W_0 W_0^T = \mathtt{I}$, and $E(z_1^*(x); x) = 0$ for all points lying in that subspace.

Fig. 1 depicts the filter obtained from such unsupervised training on the MNIST dataset. Using linear activations leads to PCA-like modes for the filter (Fig. 1(a)), whereas ReLU-like non-negative activations yield filters that resemble dictionary elements learned via sparse coding (Fig. 1(b)). Unlike PCA, the filters in Fig. 1(a) describe only a subspace and are therefore not necessarily aligned with the PCA basis. Further visual results are shown in the supplementary material.



(a) Linear (b) ReLU-type

Figure 1: First layer filters of unsupervised 784-32-32 LRRNs using different activations.

Table 1 demonstrates the ability of a trained energy-based model to distinguish between different datasets. Test data from the same dataset has on average consistently smaller energies (along the main diagonal) than samples from different datasets. Further, horizontally flipped (mirrored) test data from the same dataset has on average also a higher energy than the original test data. Finally, samples from a Gaussian fitted to the training set with diagonal covariance matrix have significantly higher energy values. Note that the characters in KMNIST have generally larger variability than e.g. MNIST, which explains the higher overall energies for this dataset. In summary, unsupervised learning of LRRNs allows to train *unnormalized* energy models capturing the training distribution *even without* explicitly addressing the lack of normalization (as opposed to contrastive divergence [13], noise-contrastive estimation [12] or score matching [14]).

		Test data				
	$\min_z E(z;x)$	MNIST	KMNIST	FMNIST	Mirrored	Fitted Gaussian
Training	MNIST	13.7±5.1	26.4±6.0	35.9±8.3	21.5±6.8	48.0±3.8
	KMNIST	59.5±24.1	36.5±12.9	52.1±16.0	43.8±15.6	78.9±4.5
	FMNIST	55.4±30.8	26.8±11.5	12.6±6.7	17.4±11.0	65.0±3.9

Table 1: Avg. energies (and std. deviations) of unsupervised 784-32-32 ReLU-models trained on MNIST, KMNIST and FMNIST (rows), evaluated on different test sets (rows).

Remark 3. Since $E(z_1;x)$ is jointly convex in x and z_1, $E(z_1^*(x);x)$ reduces to a convex function of x. Due to the connection of $\mathcal{P}_{1,W_0,G_1}$ with proximal operators (which are themselves generalizations of projection steps to convex sets), the mapping $x \mapsto E(z_1^*(x);x)$ can be interpreted as generalization of the squared distance to a convex set (whose exact shape is learned from data). Hence, $E(z_1^*(x);x)$ cannot directly represent e.g. non-convex manifolds. In order to increase the expressive power of LRRNs in the unsupervised setting, one can e.g. train class-specific energy models with weights shared across classes, $\frac{1}{N}\sum_i \min_{z:z_L=y_i} E(z;x_i) \to \min_\theta$, where y_i is the label associated with x_i.

4.2 Supervised learning

Let $\{(x_i,y_i)\}_{i=1}^N$ be labelled training data, and the aim is to estimate network parameters such that $z_L^*(x_i) \approx y_i$, where $z_L^*(x)$ is obtained by minimizing the energy model in Eq. 1. Since the mapping $x_i \mapsto z_L^*(x_i)$ has generally no closed form expression in terms of the model parameters $\theta = (W_k, c_k, b_k)_{k=0}^{L-1}$, we employ a contrastive learning approach [28, 37, 39] in the supervised setting. Let us denote the free and the so called clamped solution by $z^*(x)$ and $\hat{z}(x,y)$, respectively,

$$z^*(x) = \arg\min_z E(z;x) \qquad \hat{z}(x,y) = \arg\min_{z:z_L=y} E(z;x). \tag{20}$$

Thus, the clamped solution is obtained by minimizing the network energy with the additional constraint of fixing the output layer. By construction $E(z^*(x);x) \leq E(\hat{z}(x,y);x)$, and the aim of contrastive learning is to close the gap between these two energies by adjusting the model parameters θ:

$$\ell(\theta) = \tfrac{1}{N}\sum_i \left(E(\hat{z}(x_i,y_i);x_i) - E(z^*(x_i);x_i)\right) \to \min_\theta . \tag{21}$$

This loss can be interpreted as an approximation of the cross-entropy loss [39]. Since E is strongly convex, $E(\hat{z}(x,y);x) \approx E(z^*(x);x)$ implies that $z_L^*(x) \approx y$. We apply weight decay regularization on the last layer matrix W_{L-1} to favor non-contracting, distance-preserving feature representations in the first $L-1$ layers.

Fig. 2 illustrates the evolution of the training loss and test accuracies w.r.t. the number of epochs for the MNIST, FMNIST and KMNIST datasets. By inspecting the spectral norm of the last layer weight matrix, the mappings $x \mapsto z_L^*(x)$ have Lipschitz constants of at most 0.94, 0.95, and 1.07, respectively, for MNIST, KMNIST and FMNIST trained models. It can be easily derived (see e.g. [32]), that the classifier output is unaffected by any input perturbation Δx with $\|\Delta x\|_2 \leq m_x/(\sqrt{2}\rho)$, where ρ is a Lipschitz constant of $x \mapsto z_L^*(x)$ and $m_x = z_{L,j(x)}^*(x) - \max_{j\neq j(x)} z_{L,j}^*(x)$ is the margin for the predicted label $j(x) = \arg\max_j z_{L,j}^*(x)$. Given the median margins for the corresponding test data, this translates to median norms of 0.70 (MNIST), 0.51 (KMNIST), and 0.46 (FMNIST), for provably safe perturbations. For comparison, [32] reports a value of 1.02 for an MNIST-trained small-scale CNN (but does not state its test accuracy). We also explore the impact of unsupervised pretraining on supervised learning in the supplementary material (yielding slightly higher accuracies).

The analogous results for the CIFAR-10 dataset are shown in Fig. 3, where a $\{1024, 3072\}$-256-256-10 ReLU-type LRRN is trained for 100 epochs on grayscale and RGB images, respectively, without any data augmentation. The obtained networks are almost 1-Lipschitz with Lipschitz constants of 1.0011 and 1.007, respectively. Using the observed classification margins this translates to median radii of 0.07 (grayscale) and 0.12 (RGB) for the guarded

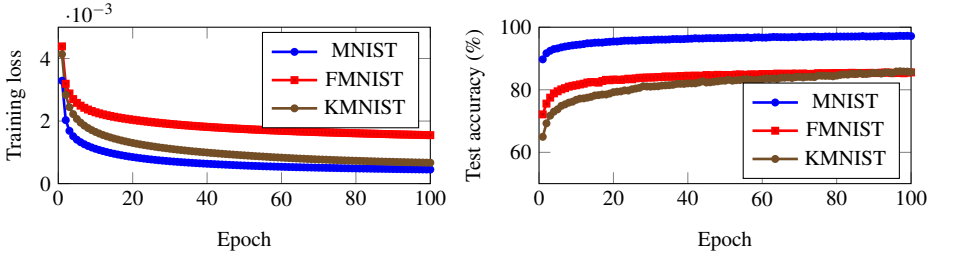


Figure 2: Loss and classification accuracies for a 784-64-64-10 ReLU-type LRRN. A final accuracy of 97.2% is achieved for MNIST, 85.6% for FMNIST and 85.7% for KMNIST.

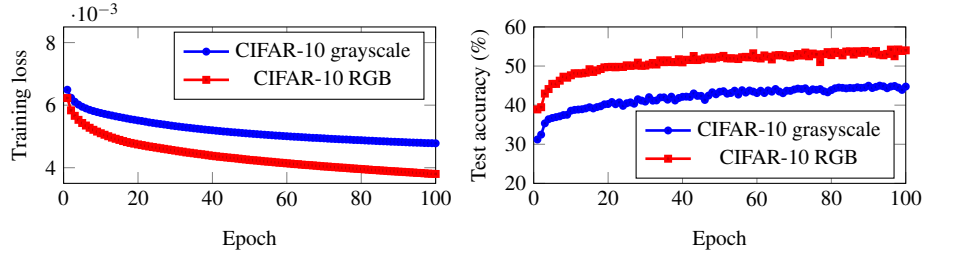


Figure 3: Loss and classification accuracies for a $\{1024, 3072\}$-256-256-10 ReLU-type LRRN. An accuracy of 44.75% and 54.02% is achieved for grayscale and RGB versions of CIFAR-10 respectively.

region, which indicates that the network is significantly more uncertain in its predictions than the MNIST models above (which is also reflected in the lower test accuracy). Unfortunately, [32] does not report any result for CIFAR-10. To put these numbers somewhat into perspective, the median norms of successful empirical L_2-attacks reported in [27] are ≈ 0.13 for a baseline CNN model and ≈ 0.8 for an adversarially trained CNN.

5 Conclusion

We propose lifted regression/reconstruction networks (LRRNs), that guarantee controlled Lipschitz continuity with easy-to-evaluate constants in layered energy-based models. This is achieved by essentially symmetrizing the terms in the underlying energy model, and therefore explicit penalizers on the model parameters (e.g. weight matrices) are not required to achieve a target Lipschitz property. We demonstrate how LRRNs can be used for both supervised and unsupervised learning. Future work includes exploration of the semi-supervised setting by combining the discriminative (contrastive) loss with unsupervised training. One goal is to obtain a unified DNN architecture for regression (and classification) that is further able to detect out-of-distribution samples (in the spirit of [10]), but at the same time explicitly allow regression tasks, aim for robustness by design, and target a unified training method.

Acknowledgements This work was supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

[1] David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169, 1985.

[2] Armin Askari, Geoffrey Negiar, Rajiv Sambharya, and Laurent El Ghaoui. Lifted neural networks. *arXiv preprint arXiv:1805.01532*, 2018.

[3] Joan Bruna and Stéphane Mallat. Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1872–1886, 2013.

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017.

[5] Miguel Carreira-Perpinan and Weiran Wang. Distributed optimization of deeply nested systems. In *Artificial Intelligence and Statistics*, pages 10–19, 2014.

[6] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 854–863, 2017.

[7] Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

[8] Ruediger Ehlers. Formal verification of piece-wise linear feed-forward neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pages 269–286. Springer, 2017.

[9] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[10] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. *arXiv preprint arXiv:1912.03263*, 2019.

[11] Fangda Gu, Armin Askari, and Laurent El Ghaoui. Fenchel lifted networks: A lagrange relaxation of neural network training. *arXiv preprint arXiv:1811.08039*, 2018.

[12] Michael U Gutmann and Aapo Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(Feb):307–361, 2012.

[13] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[14] Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(Apr):695–709, 2005.

[15] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, pages 97–117. Springer, 2017.

[16] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[17] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[18] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[19] Jia Li, Cong Fang, and Zhouchen Lin. Lifted proximal operator machines. *arXiv preprint arXiv:1811.01501*, 2018.

[20] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[21] Stéphane Mallat. Group invariant scattering. *Communications on Pure and Applied Mathematics*, 65(10):1331–1398, 2012.

[22] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

[23] Javier R Movellan. Contrastive hebbian learning in the continuous hopfield model. In *Connectionist Models*, pages 10–17. Elsevier, 1991.

[24] Haifeng Qian and Mark N Wegman. L2-nonexpansive neural networks. *arXiv preprint arXiv:1802.07896*, 2018.

[25] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*, 2018.

[26] Aditi Raghunathan, Jacob Steinhardt, and Percy S Liang. Semidefinite relaxations for certifying robustness to adversarial examples. In *Advances in Neural Information Processing Systems*, pages 10877–10887, 2018.

[27] Jérôme Rony, Luiz G Hafemann, Luiz S Oliveira, Ismail Ben Ayed, Robert Sabourin, and Eric Granger. Decoupling direction and norm for efficient gradient-based l2 adversarial attacks and defenses. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4322–4330, 2019.

[28] Benjamin Scellier and Yoshua Bengio. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in computational neuroscience*, 11:24, 2017.

[29] Paul Smolensky. *Information processing in dynamical systems: Foundations of harmony theory*, volume 1, chapter 6, pages 194–281. MIT Press, Cambridge, 1986.

[30] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *In International Conference on Learning Representations*, 2014.

[31] Gavin Taylor, Ryan Burmeister, Zheng Xu, Bharat Singh, Ankit Patel, and Tom Goldstein. Training neural networks without gradients: A scalable admm approach. In *International Conference on Machine Learning*, pages 2722–2731, 2016.

[32] Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. In *Advances in Neural Information Processing Systems*, pages 6541–6550, 2018.

[33] Shiqi Wang, Yizheng Chen, Ahmed Abdou, and Suman Jana. Mixtrain: Scalable training of formally robust neural networks. *arXiv preprint arXiv:1811.02625*, 2018.

[34] Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane Boning, and Inderjit Dhillon. Towards fast computation of certified robustness for relu networks. In *International Conference on Machine Learning*, pages 5273–5282, 2018.

[35] Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5283–5292, 2018.

[36] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[37] Xiaohui Xie and H Sebastian Seung. Equivalence of backpropagation and contrastive hebbian learning in a layered network. *Neural computation*, 15(2):441–454, 2003.

[38] Yuichi Yoshida and Takeru Miyato. Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*, 2017.

[39] Christopher Zach and Virginia Estellers. Contrastive learning for lifted networks. In *British Machine Vision Conference*, 2019.

[40] Ziming Zhang and Matthew Brand. Convergent block coordinate descent for training tikhonov regularized deep neural networks. In *Advances in Neural Information Processing Systems*, pages 1721–1730, 2017.