Fast, Effective and Interpretable Deep Learning

Frederik Hvilshøj

March 2, 2020

DEPARTMENT OF COMPUTER SCIENCE
AARHUS UNIVERSITY

Table of contents

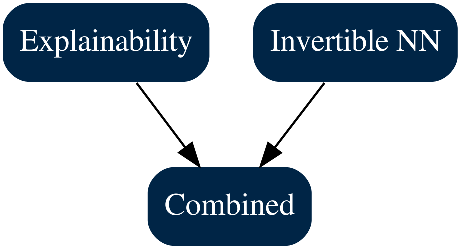
Introuction

Introduction

- Deep Learning is leading paradigm in Machine Learning
- Much effort put into network design and efficiency
- Models may be biased due to model constraints
- Limited insights – knowledge is based on intuition
- Explanations are important for sensitive tasks

This Project

- Explainability is based on heuristics
- Invertible functions give a higher level of insights



Normalizing Flows

Normalizing Flows

Goal: Learn a representation of a density $p(x)$ directly.

Model: Maximize log-likelihood of the data with respect to the parameters of an invertible Neural Network.

$$\log(p_\theta(x)) = \log(p_z(f(x))) + \log(det|\frac{\partial f(x)}{\partial x}|)$$

Constraints: invertible $f(x)$ and tractable Jacobian.

$$log\left(\det|\frac{\partial f_1(f_2(\ldots f_k(x)))}{\partial x}|\right) = \sum_{i=1}^{k} \log\left(\det|\frac{\partial f_i(z_{i-1})}{\partial z_{i-1}}|\right)$$

$$z_1 = x, \quad z_i = f_{i-1}(z_{i-1})$$

Normalizing Flow Layers

Coupling layer $c(x) = (x_1, x_2 + m(x_1))$, for $x = (x_1, x_2)$ [2]

Channel permutation Fixed channel permutations of 3D data [3]

1x1 convolutions Each spacial fiber multiplied by invertible matrix [5]

Periodic convolutions Circular convolutions over each channel [4]

What if Neural Networks had SVDs?

The Singular Value Decomposition

Definition

The Singular Value Decomposition of $W \in \mathbb{R}^{m \times n}$ is a factorization $W = U\Sigma V^T$, where U is an orthogonal $m \times m$ matrix, Σ is an $m \times n$ rectangular diagonal matrix, and V is an $n \times n$ orthogonal matrix.

Properties of the SVD

For out purposes, assume that W is a square, i.e., $W \in \mathbb{R}^{d \times d}$. Then:

$$U^{-1} = U^T \text{ and } V^{-1} = V^T \qquad O(d^2)$$

$$W^{-1} = (U\Sigma V^T)^{-1} = V\Sigma^{-1}U^T \qquad O(d^2)$$

$$det(W) = \prod_{i=1}^{d} \Sigma_{i,i} \qquad O(d)$$

Other fast computations are largest singular value, condition number, matrix exponential, and Cayley Map.

Use in Neural Networks

Neural Networks

Recall a simple fully-connected layer:

$$h_\ell = \sigma(W h_{\ell-1}) = \sigma\left(U \Sigma V^T h_{\ell-1}\right)$$

- Normalizing flows
- Recurrent Neural Networks
- W-GAN

Gradients

$$W^{t+1} = W^t - \eta \frac{\partial L}{\partial W}$$

$$\Sigma^{t+1} = \Sigma^t - \eta \frac{\partial L}{\partial \Sigma}$$

$$U^{t+1} = U^t - \eta \frac{\partial L}{\partial U}$$

The Householder Matrix

Definition

For a vector $v \in \mathbb{R}^d$, the corresponding Householder matrix is defined as

$$H = I - 2\frac{vv^T}{||v||_2^2} \tag{1}$$

- Note LHS of Eqn. (1) takes $O(d^2)$ but RHS takes $O(d)$.
- H stays orthogonal under gradient descent on v!

Lemma

The product of two orthogonal matrices is orthogonal.

Main Idea

Parametrize U and V as a product of Householder matrices [6].

$$U = H_1 \cdot H_2 \cdots H_d$$

When evaluating a NN, we need to compute

$$UX = H_1(H_2(\cdots(H_d X)))$$

with $X \in \mathbb{R}^{d \times m}$, where m is the batch size.

If we represent H_is as matrices, this takes $O(d^3 m)$ time (denoted parallel [6]). We can easily improve by representnig H_i by its vectors v_i instead, $O(d^2 m)$ (denoted sequential [6]).

Issues

Issues:

- The parallel algorithm is slow in theory.
- The sequential algorithm is not suited for GPUs.

We propose:

- Algorithm which is suited for GPUs and fast in theory.

Our Algorithm

Lemma

[1] For any m Householder matrices $H_1, \ldots, H_m$ there exists $W, Y \in \mathbb{R}^{d\times m}$ st.

$$I - 2WY^T = H_1 \cdots H_m.$$

Both W and Y can be computed by m sequential Householder multiplications in $O(dm^2)$ time.

H_1	$\cdots$	H_m	$\cdots$	H_{2m}	$\cdots$	H_d

$P_1 = I - 2W_1Y_1^T$ $\quad P_2 \quad \cdots \quad P_{\frac{d}{m}}$

Complexity

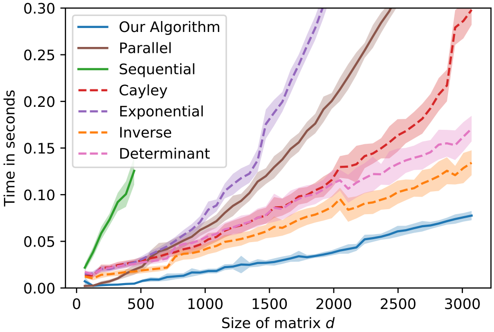
Time:

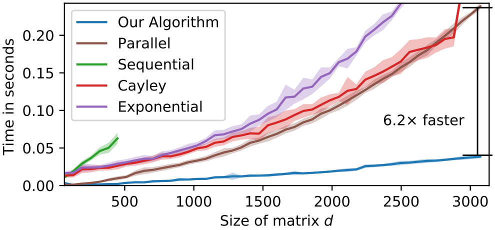
$$P_1 \cdot P_2 \cdots P_{\frac{d}{m}} X \qquad O(d^3 m) \Rightarrow O(d^2 m)$$

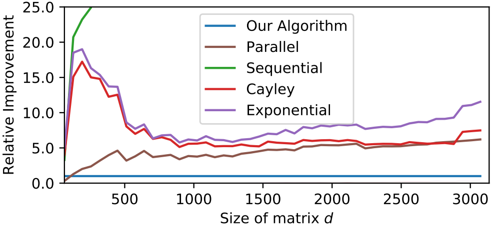
 # sequential opetations:

$$O(\text{compute } I - 2WY^T + \text{multiply } P_i\text{s}) = O(d/m + m)$$

Experiments

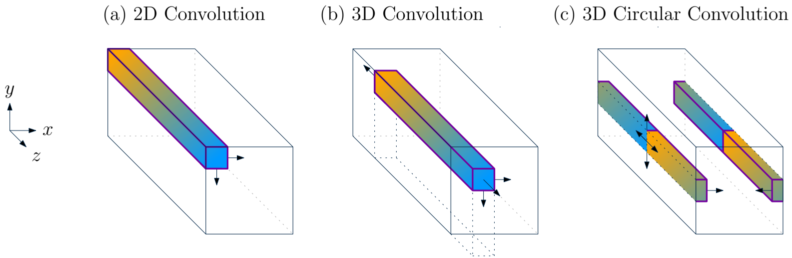






Circular 3D Convolutions

3D Circular Convolutions



Algorithm: Forward $O(hwc \log hwc)$, Determinant $O(hwc)$

Require: Image/activations $X \in \mathbb{R}^{h \times w \times c}$ and kernel $K \in \mathbb{R}^{h \times w \times c}$.

1: $X = \mathrm{DFT}_{3D}(X)$

2: $K = \mathrm{DFT}_{3D}(K)$

3: $X = X \odot K$

4: $X = \mathrm{DFT}_{3D}^{-1}(X)$

Comparison with Periodic Convolutions

For $X, K \in \mathbb{R}^{h \times w \times c}$ and $W \in \mathbb{R}^{c \times c \times h \times w}$

Circular 3D connvolutions

$$Z = F_{3D}^{-1}(F_{3D}(X) \odot F_{3D}(K))$$

Periodic convolutions

$$z_i = F_{2D}^{-1}(\sum_{j=1}^{c} F_{2D}(W_{i,j}) \odot F_{2D}(X_{:,:,j})),$$

Type	Params	Forward	Inverse
Periodic	$h \cdot w \cdot c^2$	$O(hwc^2)$	$O(hwc^3)$
3D-circular†	$h \cdot w \cdot c^2$	$O(hwc^2 \log(hwc))$	$O(hwc^2 \log(hwc))$

Improving Variational Auto-Encoders

Improving Variational Auto-Encoders

Variational lower-bound:

$$\log p(x) \geq \mathbb{E}_{z \sim q(z|x)} [\log p(x|z)] + D_{KL}(q(z|x)||p(z))$$

where $q(z|x) = \mathcal{N}(\mu(x), diag(\sigma(x)))$.

With formalizing flow f, the equation becomes

$$\log p(x) \geq \mathbb{E}_{z \sim p(z|x)} \left[\log p(x|f(z)) + \log |Det \frac{\partial f(z)}{\partial z}| \right] + D_{KL} \left(q(z|x)||p(f(z)) \right)$$

[7] let $f(z) = H_k H_{k-1} \cdots H_1 z$ with each H_i being parametrized by the encoder.

Improving Variational Auto-Encoders

Method	$\leq \ln p(\mathbf{x})$
VAE	-93.89 ± 0.09
VAE+HF(T=1)	-87.77 ± 0.05
VAE+HF(T=10)	-87.68 ± 0.06
VAE+NF (T=10) [17]	-87.5
VAE+NF (T=80) [17]	-85.1
VAE+NICE (T=10) [7]	-88.6
VAE+NICE (T=80) [7]	-87.2
VAE+HVI (T=1) [20]	-91.70
VAE+HVI (T=8) [20]	-88.30

Figure 1: Comparison of the lower bound of marginal log-likelihood measured in nats of the digits in the MNIST test set. For the first three methods the experiment was repeated 3 times. Direct copy from [7].

Explainability

What is explainability?

Improving Explanations with Probabilistic Saliency Estimation

Probabilistic Saliency Estimation

Area Under Perturbation Score

Attention Mechanisms in Gradient Explanations

Explanations Relies on Gradients

Issues

Generating Counterfactual Explanations

Construction

$$f(x) = \sigma\left(\beta^T NN(x)\right)$$

$$f^\dagger(y) = NN^{-1}\left(f(x) - \frac{f(x)^T\beta}{||\beta||}\right)$$

References i

[1] Christian Bischof and Charles Van Loan. The WY Representation for Products of Householder Matrices. *SIAM Journal on Scientific and Statistical Computing*, 1987.

[2] Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-Linear Independent Components Estimation. In *ICLR (Workshop)*, 2015.

[3] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *ICLR*, 2019.

[4] Emiel Hoogeboom, Rianne van den Berg, and Max Welling. Emerging Convolutions for Generative Normalizing Flows. In *ICML*, 2019.

[5] Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative Flow with Invertible 1x1 Convolutions. In *NeurIPS*, 2018.

[6] Zakaria Mhammedi, Andrew Hellicar, Ashfaqur Rahman, and James Bailey. Efficient Orthogonal Parametrisation of Recurrent Neural Networks Using Householder Reflections. In *ICML*, 2017.

[7] Jakub M. Tomczak and Max Welling. Improving Variational Auto-Encoders using Householder Flow. *arXiv preprint arXiv:1611.09630*, 2016.