Boltzmann machines

Machine learning club, University of Geneva

Jeremy Lane

UNIVERSITÉ
DE GENÈVE

April 16, 2019

Overview

1. Undirected graphical models (e.g. Ising, Hopfield, Boltzmann)
2. Max likelihood and gradient ascent
3. Gibbs sampling
4. Hinton's contrastive divergence algorithm
5. Other generative models

Undirected graphical models (UGMs)

Conditional independence:

$$x \perp y \mid z \iff p(x \mid z)p(y \mid z) = p(x, y \mid z)$$
$$\iff p(x \mid y, z) = p(x \mid z).$$

Consider pairs $(p, \mathcal{G})$ where:

- p is the joint probability distribution of a set of random variables, X,
- $\mathcal{G} = (V, E)$ is an undirected graph, and
- we have fixed a bijection $V \leftrightarrow X$.

Want edges of $\mathcal{G}$ to encode conditional independences.

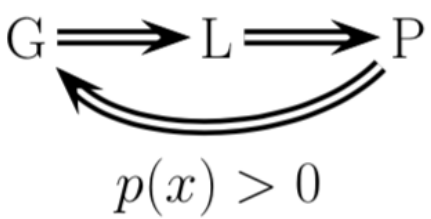
Definition

The *Markov blanket of* $v \in V$, $mb(v)$, is the intersection of all subsets $U \subseteq V \setminus \{v\}$ such that

$$v \perp V \setminus (\{v\} \cup U) \mid U.$$

Definition (global, local, and pairwise markov properties)

- (G) $\forall A, B, C \subseteq V$ disjoint, $A \perp B \mid C$ if C separates A and B.
- (L) $\forall v \in V$, $mb(v) =$ vertices adjacent to v
- (P) $\forall x, y \in V$, $x \perp y \mid V \setminus \{x, y\}$ iff $E_{x,y} = \emptyset$.



Definition

An *undirected graphical model* is a pair $(p, \mathcal{G})$ that satisfies the global Markov property (G).

Representability

Theorem (Hammersley and Clifford, 1971)

Assume $p > 0$. Then, p satisfies the conditional independence properties of an undirected graphical model on $\mathcal{G}$ iff p can be written in the form

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c),$$

where $\mathcal{C}$ is the set of maximal cliques in $\mathcal{G}$, and $\psi_c(\mathbf{x}_c) > 0$.

Letting $E_c = -\ln \psi_c$, we get a Gibbs distribution

$$p(\mathbf{x}) = \frac{1}{Z} \exp\left(-\sum_{\mathcal{C}} E_c(\mathbf{x}_c)\right).$$

The Ising model (Lenz and Ising, 1920)

Example

$\mathcal{G}$ a lattice (1D, 2D, 3D,...), $x_i \in \{\pm 1\}$ model particle spins.

W symmetric matrix of coupling strengths. $\mathbf{b}$ an external field.

$$p(\mathbf{x}) \propto \exp\left(-E(\mathbf{x})\right), \quad E(\mathbf{x}) = -\frac{1}{2}\mathbf{x}^T W \mathbf{x} - \mathbf{b}^T \mathbf{x}.$$

The Hopfield network (Hopfield, 1982)

Example

$\mathcal{G}$ a complete graph, $x_i \in \{-1, 1\}$ or $\{0, 1\}$, weights W, $\mathbf{b}$.

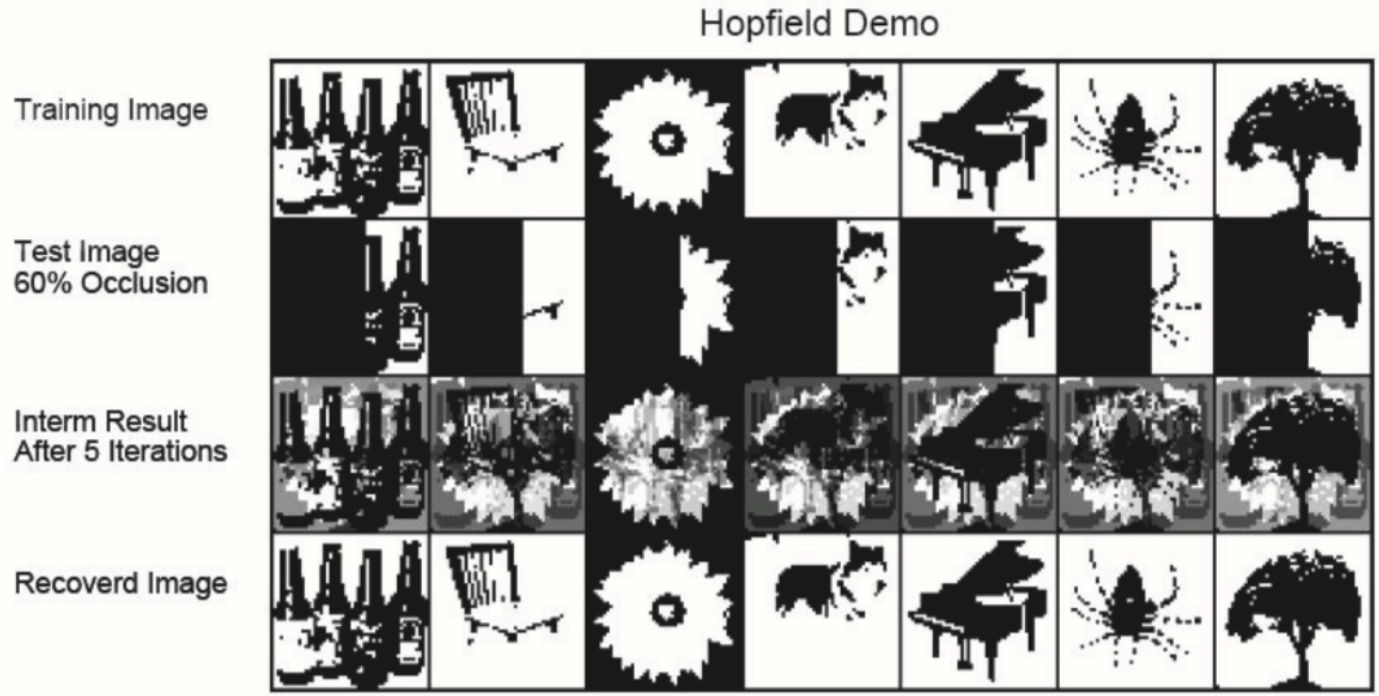


Figure 19.7 Examples of how an associative memory can reconstruct images. These are binary images of size 50×50 pixels. Top: training images. Row 2: partially visible test images. Row 3: estimate after 5 iterations. Bottom: final state estimate. Based on Figure 2.1 of Hertz et al. (1991). Figure generated by `hopfieldDemo`.

The Boltzmann machine (Hinton and Sejnowski, 1985)

For models with visible and hidden variables, introduce indices,

$$V = \{v_1, \ldots, v_n\}, \quad H = \{h_1, \ldots, h_m\}.$$

Useful to stack the variables as a vector,

$$\mathbf{x} = (\mathbf{v}, \mathbf{h})^T, \quad \mathbf{v} = (v_1, \ldots, v_n), \quad \mathbf{h} = (h_1, \ldots, h_n).$$

Example

Same as before, but not all vertices are visible: $\mathcal{G} = (V \cup H, E)$.

Observed data is modelled by the marginal distribution

$$p_v(\mathbf{v}) = \sum_h p(\mathbf{v}, \mathbf{h})$$

Hidden variables gives the model more "capacity."

Maximum likelihood

Let $p(\mathbf{v}, \mathbf{h} \mid \theta)$ be probability distribution with visible and hidden variables, and parameters θ.

Assume we are given i.i.d. samples $\{\mathbf{v}^{(i)}\}_i^N$ with $\mathbf{v}^{(i)} = (v_1^{(i)}, \ldots, v_n^{(i)})$.

Definition

The (normalized) *log-likelihood* of $\{\mathbf{v}^{(i)}\}_i^N$ is

$$\ell = \frac{1}{N} \ln p_v(\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(N)} \mid \theta) = \frac{1}{N} \sum_{i=1}^{N} \ln p_v(\mathbf{v}^{(i)} \mid \theta).$$

Maximum likelihood methods aim to find

$$\hat{\theta} \approx \theta^* = \operatorname{argmax}_\theta \ell.$$

Can add a regularization penalty term.

Log-linear models

Given $(p, \mathcal{G})$, with *feature functions* ϕ_s, and linear functions θ_s,

$$E_s(\mathbf{v}, \mathbf{h}, \theta) = \theta_s(\phi_s(\mathbf{v}, \mathbf{h})), \quad p(\mathbf{v}, \mathbf{h} \mid \theta) = \frac{1}{Z(\theta)} \exp\left(-\sum_S E_s(\mathbf{v}, \mathbf{h}, \theta)\right).$$

Model parameters are the linear functions θ_s. Then,

$$\begin{aligned}
\ell(\theta) &= \frac{1}{N} \sum_{i=1}^{N} \ln p_v(\mathbf{v}^{(i)} \mid \theta) \\
&= \frac{1}{N} \sum_{i=1}^{N} \left[\ln \left(\sum_h \exp\left(-\sum_S \theta_s(\phi_s(\mathbf{v}^{(i)}, \mathbf{h})) \right) \right) - \ln Z(\theta) \right]
\end{aligned}$$

We will try to optimize ℓ using gradient descent.

General theory of stochastic gradient ascent (SGD)

Given $f: E \to \mathbb{R}$, assume we can draw samples $v_\theta \in T_\theta E = E$ from a distribution such that $\mathbb{E}(v_\theta) \propto \nabla f(\theta)$.

Choose $\varepsilon > 0$, $M \in \mathbb{N}$; Randomly initialize θ;
for $k = 1$ *to* M **do**
  sample v_θ;
  $\theta_{k+1} \leftarrow \theta_k + \varepsilon v_\theta$;
end
return $\hat{\theta} = \frac{1}{M} \sum \theta_k$ *(or some other estimate)*

Theorem (Typical theorem for convergence of SGD)

Let $B, \rho > 0$. Assume f is concave and $\theta^* = \operatorname{argmax}_{||\theta|| \leq B} f$. Assume that $||v_\theta|| < \rho$. Let $\varepsilon = \sqrt{\frac{B^2}{\rho^2 M}}$. Then,

$$\mathbb{E}(f(\hat{\theta})) - f(\theta^*) \leq \frac{B\rho}{\sqrt{M}}.$$

Minibatch stochastic gradient ascent

Suppose $f(\theta) = \frac{1}{N}\sum_{i=1}^{N} L(x^{(i)}, \theta)$. Choosing $B \subseteq S = \{1, \ldots, N\}$ randomly,

$$\mathbb{E}\left(\nabla_\theta f_B(\theta)\right) \propto \nabla_\theta f(\theta), \quad \nabla_\theta f_B(\theta) = \frac{1}{|B|}\sum_{i \in B} \nabla_\theta L(x^{(i)}, \theta).$$

Choose $\varepsilon > 0$, $M \in \mathbb{N}$; Randomly initialize θ;
Randomly partition S into minibatches B;
for $k = 1$ *to* M **do**
  for *each minibatch* B **do**
    Compute $\nabla_\theta f_B(\theta)$;
    $\theta \leftarrow \theta + \varepsilon \nabla_\theta f_B(\theta)$;
  end
end
return θ.

Minibatch size chosen between 1 and $|S|$ to optimize convergence rate.

Returning to likelihoods...

Recall,

$$\ell(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left[\ln \left(\sum_{h} \exp \left(- \sum_{S} \theta_s(\phi_s(\mathbf{v}^{(i)}, \mathbf{h})) \right) \right) \right] - \ln Z(\theta).$$

For a single sample $\mathbf{v}^{(i)}$,

$$\nabla_{\theta_c} \ln \left(\sum_{h} \exp \left(\sum_{S} \theta_s(\phi_s(\mathbf{v}^{(i)}, \mathbf{h})) \right) \right) = \mathbb{E}(\phi_s(\mathbf{v}^{(i)}, \mathbf{h}) \mid \theta)$$

$$\nabla_{\theta_s} \ln Z(\theta) = \mathbb{E}(\phi_s(\mathbf{v}, \mathbf{h}) \mid \theta)$$

In the first expectation, $\mathbf{v}$ is clamped to $\mathbf{v}^{(i)}$ and expectation is computed with respect to $p(\mathbf{h} \mid \mathbf{v}^{(i)}, \theta)$. In the second both $\mathbf{v}$ and $\mathbf{h}$ are free, expectation with respect to $p(\mathbf{v}, \mathbf{h} \mid \theta)$.

Exercise:

$$\frac{\partial^2}{\partial\theta_{s,i}\partial\theta_{s',j}} \ln Z(\theta) = \dots$$

$$= \mathbb{E}(\phi_s(\mathbf{v},\mathbf{h})_i \phi_{s'}(\mathbf{v},\mathbf{h})_j \mid \theta)$$
$$- \mathbb{E}(\phi_s(\mathbf{v},\mathbf{h})_i \mid \theta)\mathbb{E}(\phi_{s'}(\mathbf{v},\mathbf{h})_j \mid \theta)$$
$$= \mathrm{Cov}(\phi_s(\mathbf{v},\mathbf{h})_i, \phi_{s'}(\mathbf{v},\mathbf{h})_j \mid \theta)$$

So $\mathrm{Hess} \ln Z(\theta) = \mathrm{Cov}(\phi_s)$ is positive semi-definite $\Rightarrow \ln Z(\theta)$ is convex.

Easy to see that the other term is concave in θ. Thus,

Theorem

For log-linear models, $\ell(\theta)$ is concave.

Although the function is concave, there is no analytic formula for a maximum. So we use gradient ascent.

Gradient ascent for log-linear models

The gradient of with respect to a single sample $\mathbf{v}^{(i)}$ is computed using

$$\nabla_{\theta_s} \ell^{(i)} = \mathbb{E}(\phi_s(\mathbf{v}^{(i)}, \mathbf{h}) \mid \theta) - \mathbb{E}(\phi_s(\mathbf{v}, \mathbf{h}) \mid \theta).$$

e.g. for the feature $\phi(\mathbf{v}, \mathbf{h}) = v_j h_k$,

$$\frac{\partial}{\partial w_{j,k}} \ell^{(i)} = \mathbb{E}(v_j^{(i)} h_k \mid \theta) - \mathbb{E}(v_j h_k \mid \theta).$$

Computational Problem

The expectations above are sums of 2^m and 2^{n+m} terms respectively...

Each step of gradient ascent is wildly computationally expensive...

Gibbs sampling

Let p be a joint distribution in random variables $X = \{x_1, \ldots, x_n\}$.

Randomly generate $\mathbf{x}^1$.

Given sample $\mathbf{x}^k = (x_1^k, \ldots, x_n^k)$, draw the next sample by computing

$$
\begin{aligned}
x_1^{k+1} &\sim p(x_1 \mid x_2^k, \ldots, x_n^k) \\
x_2^{k+1} &\sim p(x_2 \mid x_1^{k+1}, x_3^k, \ldots, x_n^k) \\
&\ldots \\
x_n^{k+1} &\sim p(x_n \mid x_1^{k+1}, x_2^{k+1}, \ldots, x_{n-1}^{k+1})
\end{aligned}
$$

This is still very slow.

Restricted Boltzmann Machines

Example

Let $\mathcal{G}$ be the complete bipartite graph on vertices (V, H). i.e. assume conditional independences:

$$v_i \perp v_j \mid H, \quad h_i \perp h_j \mid V \quad \forall i, j.$$

The Gibbs sampling algorithm simplifies:

$$v_i^{k+1} \sim p(v_i \mid \mathbf{h}^k), \quad \forall i = 1, \dots n$$
$$h_j^{k+1} \sim p(h_j \mid \mathbf{v}^{k+1}), \quad \forall j = 1, \dots m$$

Can update $\mathbf{v}$ and $\mathbf{h}$ in two steps instead of $n + m$.

Restricted Boltzmann Machines

Assuming $\mathcal{G}$ is bipartite, the probability distribution has the form

$$p(\mathbf{v}, \mathbf{h}) = \frac{1}{Z(\theta)} \exp\left(-E(\mathbf{v}, \mathbf{h})\right), \quad -E(\mathbf{v}, \mathbf{h}) = \frac{1}{2}\mathbf{v}^T W \mathbf{h} + \frac{1}{2}\mathbf{a}^T \mathbf{v} + \frac{1}{2}\mathbf{b}^T \mathbf{h}.$$

Then,

$$\begin{aligned} p(v_i = 1 \mid \mathbf{h}) &= \frac{p(v_i = 1, \hat{\mathbf{v}}, \mathbf{h})}{p(v_i = 1, \hat{\mathbf{v}}, \mathbf{h}) + p(v_i = -1, \hat{\mathbf{v}}, \mathbf{h})} \\ &= \frac{1}{1 + \exp(E(v_i = 1, \hat{\mathbf{v}}, \mathbf{h}) - E(v_i = -1, \hat{\mathbf{v}}, \mathbf{h}))} \\ &= \frac{1}{1 + \exp(-\mathbf{w}_i \mathbf{h} - a_i)} \\ \Rightarrow\ p(\mathbf{v} \mid \mathbf{h}) &= \sigma(W\mathbf{h} + \mathbf{a}^T) \end{aligned}$$

We recover the same formula as a sigmoid activation layer in a feed-forward neural network. Similarly,

$$p(\mathbf{h} \mid \mathbf{v}) = \sigma(W^T \mathbf{v} + \mathbf{b}^T).$$

Contrastive divergence (CD_1)

Burning in a MCMC is still slow. However, if you only run 1 step of blocked Gibbs sampling, it's faster, experimentally good results:

Contrastive divergence (CD_1) algorithm (Hinton, 2002)

For a batch of samples $B = \{\mathbf{v}_+\}$:
Sample $\mathbf{h}_+ \sim p(\mathbf{h} \mid \mathbf{v}_+)$.
Sample $\mathbf{v}_- \sim p(\mathbf{v} \mid \mathbf{h}_+)$.
Sample $\mathbf{h}_- \sim p(\mathbf{h} \mid \mathbf{v}_-)$.

Estimate the expected values of the features:

$$\mathbb{E}(v_{+,j} h_k \mid \theta) \approx \langle v_j h_k \rangle_{data} = \frac{1}{|B|} \sum_{\mathbf{v}_+ \in B} v_{+,j} \cdot h_{+,k}$$

$$\mathbb{E}(v_j h_k \mid \theta) \approx \langle v_j h_k \rangle_{recon} = \frac{1}{|B|} \sum_{\mathbf{v}_+ \in B} v_{-,j} \cdot h_{-,k}$$

$$W \leftarrow W + \varepsilon(\langle v_j h_k \rangle_{data} - \langle v_j h_k \rangle_{recon})$$

Contrastive divergence

The estimate of the expectation for the partition function produced by the CD_k is biased.

Theorem

Gibbs sampling produces a chain $\mathbf{v}^{(i)} = \mathbf{v}_0, \mathbf{h}_0, \mathbf{v}_1, \mathbf{h}_1, \mathbf{v}_2, \ldots$

$$\nabla_{\theta_s} \ell^{(i)} = -\sum_h p(\mathbf{h} \mid \mathbf{v}_0) \phi_s(\mathbf{v}_0, \mathbf{h})$$

$$+\mathbb{E}_{p(\mathbf{v}_k|\mathbf{v}_0)} \left[\sum_h p(\mathbf{h} \mid \mathbf{v}_k) \phi_s(\mathbf{v}_k, \mathbf{h}) \right] + \mathbb{E}_{p(\mathbf{v}_k|\mathbf{v}_0)} \left[\frac{\partial \ln p(\mathbf{v}_k)}{\partial \theta} \right]$$

The CD_k algorithm produces an unbiased estimate of the first two terms and

$$\mathbb{E}_{p(\mathbf{v}_k|\mathbf{v}_0)} \left[\frac{\partial \ln p(\mathbf{v}_k)}{\partial \theta} \right] \to 0 \text{ as } k \to \infty.$$

Applications

Restricted Boltzmann machines have been used for

- Generative modelling
- Collaborative filtering
- Pre-training deep models
- Dimension reduction
- Classification
- Topic modelling

Moving away from Boltzmann machines...

- VAEs and GANs are better at generative modelling
- Deep models no longer need pre-training
- Convolutional neural networks are better for computer vision

Generative models

Problem

Create a program that allows us to draw samples $x \sim p_{\text{model}}$ such that $p_{\text{model}} \approx p_{\text{true}}$.

Deep learning approach: create a neural network

$$G \colon \mathbb{R}^n \to \mathbb{R}^N$$

where $z \in \mathbb{R}^n$ is a low-dimension parameter sampled from some simple distribution p (e.g. uniform or Gaussian) and $\mathbb{R}^N$ is the data space.

The manifold hypothesis

$p_{\text{true}} = p + \varepsilon$ where p is a probability distribution supported on a high codimension submanifold of $\mathbb{R}^N$ and ε is noise.

The push-forward $p_{\text{model}} = G_* p$ is supported on the image $G(\mathbb{R}^n) \subset \mathbb{R}^N$.

Loss functions for generative models

Traditional wisdom in machine learning

Minimize the Kullback-Leibler divergence between probability densities

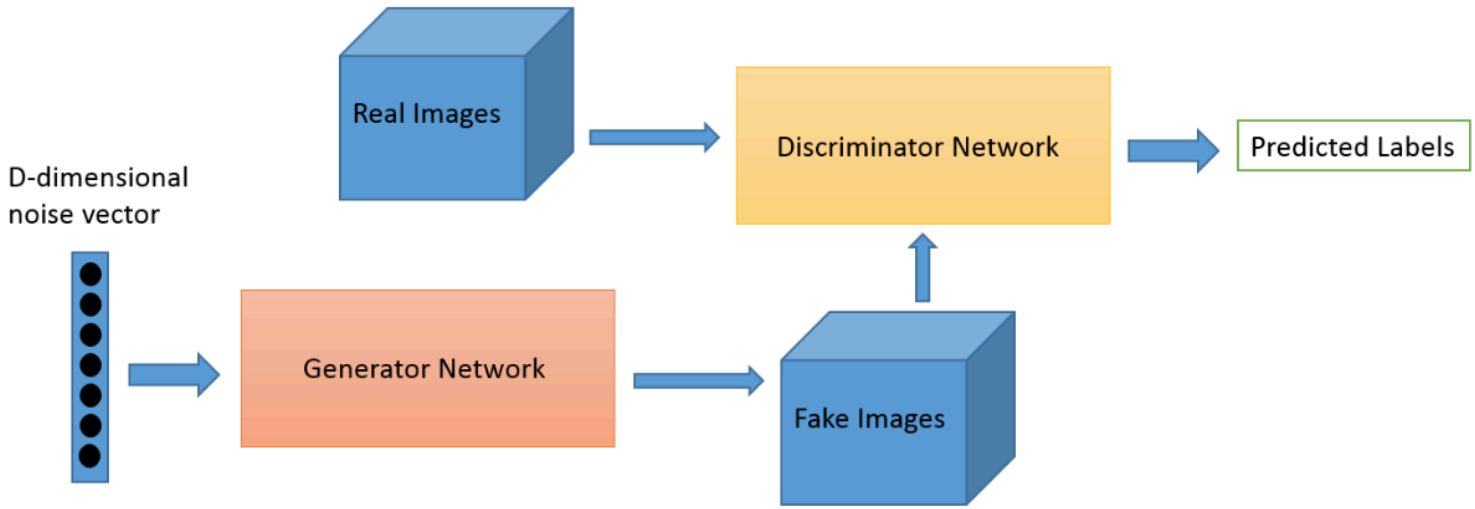
$$KL(p_{\text{true}}||p_{\text{model}}) = \mathbb{E}_{x\sim p_{\text{true}}}\left(\ln\left(\frac{p_{\text{true}}(x)}{p_{\text{model}}(x)}\right)\right).$$

Asymptotically equivalent to maximum likelihood.

Problem:

- If $p_{\text{model}}(x) = 0$ and $p_{\text{true}}(x) > 0$ on a set of non-zero measure with respect to dp_{true}, then $KL = \infty$.
- (assuming the manifold hypothesis) the supports of p_{true} and $p_{\text{model}} = G_* p$ probably don't intersect.

Generative Adversarial Networks (GANs)



$$L_D = \frac{1}{N}\sum^{N}\left[-\ln D(x^{(i)}) - \ln(1 - D \circ G(z^{(i)}))\right]$$

$$L_G = \frac{1}{N}\sum^{N}\ln(1 - D \circ G(z^{(i)}))$$

$$\min_G \max_D \left[\mathbb{E}_{x\sim p_{\text{emp}}} \ln D(x) + \mathbb{E}_{z\sim p} \ln(1 - D \circ G(z))\right]$$

Faces made with GANs

Figure 5: 1024 × 1024 images generated using the CELEBA-HQ dataset. See Appendix F for a larger set of results, and the accompanying video for latent space interpolations.

Karras et al, 2018.

Increasing resolution with GANs

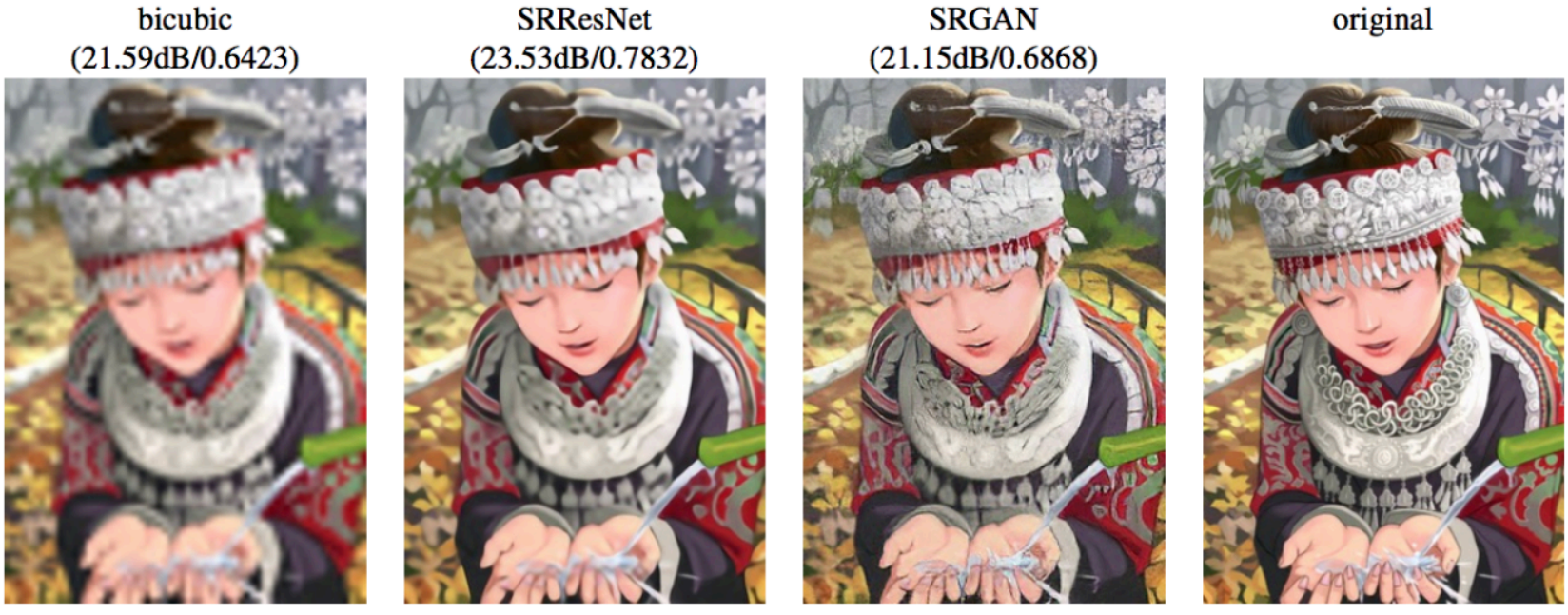


Figure 2: From left to right: bicubic interpolation, deep residual network optimized for MSE, deep residual generative adversarial network optimized for a loss more sensitive to human perception, original HR image. Corresponding PSNR and SSIM are shown in brackets. [4× upscaling]

Ledig et al, 2017.

Art made with GANs

https://github.com/robbiebarrat/art-DCGAN

Training GANs

GANs have various training issues that are hard to fix:

- Cyclic behaviour of gradients
- Mode collapse
- Gradients are often zero
- Discriminator gets too good too fast

(Arjovsky, Chintala, and Bottou, 2017)

GANs work better if you minimize the Wasserstein distance,

$$W(p_{\text{true}}, p_{\text{model}}) = \inf_{\gamma \in \Pi(p_{\text{true}}, p_{\text{model}})} \mathbb{E}_{(x,y)\sim\gamma} ||x - y||.$$

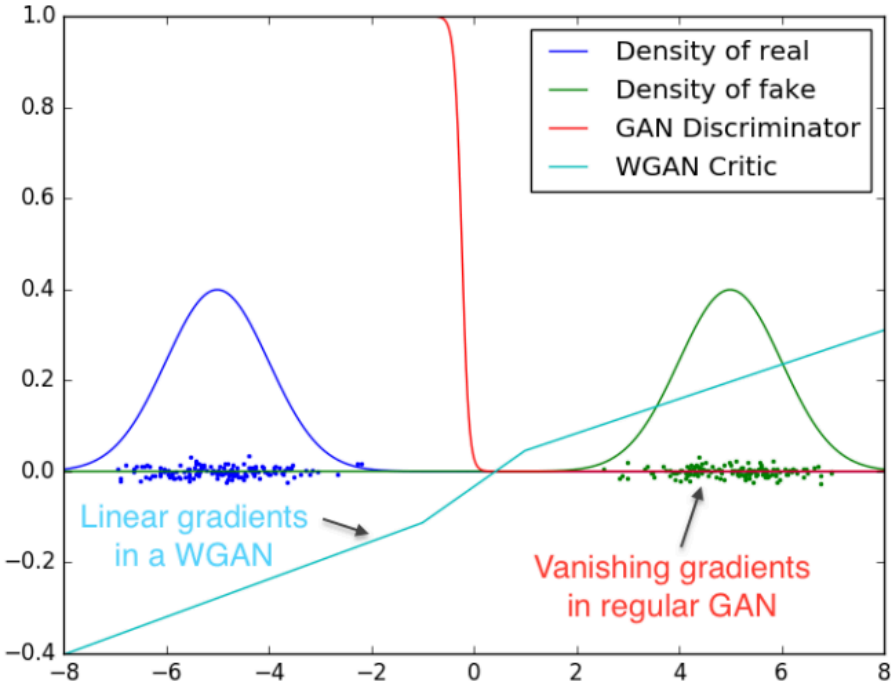


Figure 2: Optimal discriminator and critic when learning to differentiate two Gaussians. As we can see, the discriminator of a minimax GAN saturates and results in vanishing gradients. Our WGAN critic provides very clean gradients on all parts of the space.

Arjovsky, Chintala, and Bottou. 2017.

References (graphical models)

Murphy. Machine Learning: A Probabilistic Perspective. 2012.

Koller and Friedman. Probabilistic graphical models: Principals and techniques. 2009.

References (stochastic gradient descent)

Shalev-Shwartz and Ben-David. Understanding Machine Learning: From Theory to Algorithms. 2014.

Goodfellow, Bengio, and Courville. Deep Learning. 2017.

References (Boltzmann machines)

Hinton. Training products of experts by minimizing contrastive divergence. 2002.

Salakhutdinov and Hinton. Reducing the Dimensionality of Data with Neural Networks. 2006.

Salakhutdinov, Mihn, and Hinton. Restricted Boltzmann Machines for Collaborative Filtering. 2007.

Hinton. A Practical Guide to Training Restricted Boltzmann Machines. 2010.

Bengio and Delalleau. Justifying and generalizing contrastive divergence. 2009.

Goodfellow, Bengio, and Courville. Deep Learning. 2017.

References (VAEs and GANs)

Goodfellow et al. Generative adversarial nets. 2014.

Arjovsky, Chintala, and Bottou. Wasserstein GAN. 2017.

Balduzzi et al. The mechanics of n-player differentiable games. 2018.