Symbolic Autoencoder

Oryan Barta
Outline

• Problem & Motivation
• Existing solutions
  • Autoencoders & GAN
• Proposed method: SAE
  • Syncretic loss
• Empirical results
• Conclusion and next steps
Problem and Motivation
Problem: Unsupervised learning

- Given a set $S$ of points sampled from distribution $D$, we want to compute $p_{model} \approx p_D$
  - Explicitly or implicitly
- Formally, we want to minimize cross entropy:
  $$H(p_D, p_{model}) = -E_{x \sim p_D}(\log p_{model}) = - \sum_{x \sim D} p_D(x) \log(p_{model}(x))$$
- Simplifies to minimization of KL-divergence:
  $$D_{KL}(p \parallel q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx$$
- Alternative formulation: maximize likelihood over model parameters:
  $$\theta^* = \arg\max_{\theta} \left( \mathbb{E}_{x \sim p_D}(\log p_{model}(x|\theta)) \right)$$

Continuous version:
$$H(p, q) = -\mathbb{E}_{x \sim p}(\log q) = H(p) + D_{KL}(p \parallel q)$$
$$D_{KL}(p \parallel q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx, \text{ (KL divergence)}$$
$$H(p) = -\int_{-\infty}^{\infty} p(x) \log p(x) dx, \text{ (Entropy)}$$
Motivation: Supervised learning is flawed

• Given joint distribution $\mathcal{D} \times \mathcal{L}$

• Assume $H(p_\mathcal{D}, p_\mathcal{L})$ is small and minimize $H(p_\mathcal{L}, p_{model})$

• Problems:
  • Model useless on domains where labeled dataset is small or nonexistent
  • Relies on heuristic assumption that labels reflect structure of $p_\mathcal{D}$ -- limits computer intelligence by human intelligence

• Unsupervised learning would solve these problems, making neural networks more applicable & effective
Intuition:

Underlying distribution $\mathcal{D}$

Artificially learned subset of features

Supervised model $p_{\text{model}}$

Heuristic input

Human model -- $p_L$

Human learned subset of features

Paciflora  Lily  Rose

Paciflora  Lily  Rose

Paciflora  Lily  Rose

Paciflora  Lily  Rose
Intuition:

Human learned subset of features

Artificially learned subset of features

Unsupervised model -- $p_{model}$

Human model -- $p_L$

Underlying distribution $D$

Oryan Barta © 2018 all rights reserved
**Intuition: “Implicit language”**

Artificially learned subset of features

- Pacificflora
- Lily
- Rose

Human learned subset of features

Goal: Make these isomorphic
Project Goal: Language creation

• Want a system that can automatically label images, meaning:
  • Training only on images without labels, produce labeling $p'_L$ which minimizes $H(p_L, p'_L)$
  • Intuitively: Produced labels $L'$ that are approximately isomorphic with human labels $L$.
    • $L'$ is an “internal” language of the model, “implicitly” defined.

• Extracting all the deep features of image before supervised stage.

• Final step would consist in a thin translation layer, which would simply map the two domains (should require ideally one example per label).

• Inspiration: A natural system exists which implements unsupervised deep feature extraction, outputting language
Existing Solutions
Existing solutions: Unsupervised models

• Explicit models:
  • PixelRNN, PixelCNN
  • Compute $p_{model}(x) = \prod_{i=1}^{n} p(x_i|x_1, ..., x_{i-1})$

• Implicit (Generative) models:
  • Naïve Autoencoders
  • Variational Autoencoders
  • Generative Adversarial Networks
    • Cutting edge currently
GAN: Generative Adversarial Networks

Adversarial loss:

\[
\begin{align*}
J(D) &= -\frac{1}{2} \mathbb{E}_{x \sim p_{\text{data}}} \log D(x) - \frac{1}{2} \mathbb{E}_{z} \log (1 - D(G(z))) \\
J(G) &= -\frac{1}{2} \mathbb{E}_{z} \log D(G(z))
\end{align*}
\]

\[D(I_{\text{real}}) \to 1 \quad 0 \leftarrow D(I_{\text{fake}}) \to 1\]
DCGAN Results

After 6 epochs
GAN problems

- Half a model – no encoder produced
  - Discriminator is a “degenerate” encoder – only single feature extracted at top level (real/not real)
  - Feature extraction is inaccessible – cannot be used in semi-supervised setting to reduce need for labeled data in classification tasks

- Mode collapse:
  - Switches between modes

- Batch collapse:
  - Generator output becomes correlated only to batch statistics rather than latent code due to batch norm

- Batch norm dependence

- Need to heuristically balance D & G

(Metz et al 2016)
Generative Adversarial Autoencoder Networks

Bicycle GAN

BEGAN

AEGAN
Proposed Method
SAE Computation

Syncetic phase

Adversarial phase

Gradient stop

True image

Fake image

Fake image

Oryan Barta © 2018 all rights reserved
SAE generalizes GAN:

\[ SAE(1,0) \equiv GAN \]

- For \( d = 1, \alpha = 0 \) (100% noise input) SAE becomes GAN.
- In the unique case of \( d = 1 \), the output of the ideal encoder is known so we don’t have to explicitly train another encoder.
**SAE**($n, \alpha$)

- In the general case, a possible interpretation is each unit assumes the other is ideal
  - Each thinks the other has the label and trains accordingly.
- Also, noise is not necessary, and in some cases hinders convergence.
Basic Algorithm

For each iteration in train loop:

1. Sample $I \sim \mathcal{D}$ (a batch of $n$ images)
2. $s_A \leftarrow A(I)$
3. $s_E \leftarrow E(I)$
4. $s \leftarrow \frac{s_A + s_E}{||s_A + s_E||}$
5. $\tilde{I} \leftarrow G(s)$
6. $\tilde{s}_A \leftarrow A(\tilde{I})$
7. $\tilde{s}_E \leftarrow E(\tilde{I})$

Calculate losses:

8. $l_{syn}(s_A, s_E) \leftarrow h(p(s_A, s_E), 1)$
9. $l^A_{adv}(s_A, \tilde{s}_E) \leftarrow h(p(s_A, \tilde{s}_E), 0)$
10. $l^E_{adv}(s_E, \tilde{s}_A) \leftarrow h(p(s_E, \tilde{s}_A), 0)$
11. $\tilde{l}^A_{adv}(s_A, \tilde{s}_E) \leftarrow h(p(s_A, \tilde{s}_E), 1)$
12. $\tilde{l}^E_{adv}(s_E, \tilde{s}_A) \leftarrow h(p(s_E, \tilde{s}_A), 1)$
13. $l_D \leftarrow \tilde{l}^A_{adv} + \tilde{l}^E_{adv}$
14. $l_G \leftarrow \tilde{l}^A_{adv} + \tilde{l}^E_{adv}$

Backprop & update weights:

8. $\theta_D \leftarrow \theta_D + \mu \frac{\partial l_D}{\partial \theta_D}$
9. $\theta_G \leftarrow \theta_G + \mu \frac{\partial l_G}{\partial \theta_G}$
Distance & Loss functions

• Distance function:
  • \( p(s, s') = \frac{1 + \langle s, s' \rangle}{2} \in [0,1] \)
  • This is the square cosine of the angle to the bisector between \( s, s' \):
    • \( \cos^2 \left( \frac{x}{2} \right) = \frac{1 + \cos(x)}{2} \)
  • We can interpret this as “estimated probability that \( s, s' \) represent the same image”

• Loss function: sigmoid cross entropy distance \( p \) with target \( q \)
  • \( h(p, q) = -q \cdot \ln(p) - (1 - q) \cdot \ln(1 - p) \)
Training dynamics:

Loss function $h(p, q)$

Loss function. $0 < x < \pi$

- $x$ - axis denotes angle between symbols (inner product between vectors on $S_n$)
- $y$ - axis denotes loss:
  - $h(x, 1)$: attractive
  - $h(x, 0)$: repulsive
  - Dashed: distance functions (square half angle)
  - Dotted: gradients of loss
    - $\tan\frac{x}{2}, \tan\frac{x+\pi}{2}$
  - Equilibrium point: $x = \frac{\pi}{2}$
    - When $\tan\frac{x}{2} + \tan\frac{x+\pi}{2} = 0$
Syncretic loss:

- The syncretic loss provides the gradient direction for real images, using centralization to stabilize training.
- It has 3 components:
  - **Shell loss** – Drives formation of adversarial learning nodes along radii
  - **Self loss** – Centralization
  - **Kernel loss** – Centralization
Adversarial loss:

• The adversarial loss provides the gradient direction for fake images.
• The crossed generator objective:
  • $A(G(s)) \rightarrow s_E$
  • $E(G(s)) \rightarrow s_A$
• The discriminator objectives are the opposite.
Results
Results on 3 Datasets - Different Characteristics

• Mnist – 10 distinct modes
  • At least two orders of magnitude faster convergence than the unsupervised DCGAN – because no mode collapse
• CelebA – Single mode with continuous variations
  • Higher quality (deep features extraction) than original DCGAN
  • Ability to generate better pictures with less iterations, less filters, and larger picture dimensions
• Music Notation – Combinatorial multi modal
  • Original DCGAN not able to learn because of mode collapse
  • Ability to learn structured data
Compare Original DCGAN Results to SAE:
Mnist Unsupervised

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Time</th>
<th>SAE</th>
<th>Original</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>5 minutes</td>
<td><img src="image1" alt="SAE Image" /></td>
<td><img src="image2" alt="Original Image" /></td>
</tr>
<tr>
<td>1000</td>
<td>40 minutes</td>
<td><img src="image3" alt="SAE Image" /></td>
<td><img src="image4" alt="Original Image" /></td>
</tr>
<tr>
<td>6000</td>
<td></td>
<td><img src="image5" alt="SAE Image" /></td>
<td><img src="image6" alt="Original Image" /></td>
</tr>
<tr>
<td>100,000</td>
<td></td>
<td><img src="image7" alt="SAE Image" /></td>
<td><img src="image8" alt="Original Image" /></td>
</tr>
</tbody>
</table>
Semantic Reconstructions
Semantic features learned

Original

Generated

Original

Generated
DCGAN vs SAE

After ½ an epoch
Experiment on highly structured data: Music notation
Conclusion
SAE Overview

• Relatively simple architecture
• Generalization of GAN – resolves many issues simultaneously
  • Produces encoder: Is of more practical use then GAN
  • Resolves mode collapse problem
  • Resolves batch collapse problem & batch norm dependence
  • Resolves need for heuristically balancing D & G
  • Stabilizes convergence, improvement over DCGAN by several OOM
• Does not rely on latent variable, implying a change of paradigm from GAN models
• It also improves on VAE:
  • Produces sharp near-photorealistic images
  • Does not rely on pixelwise reconstruction loss which is ineffective for deep feature extraction
  • Does not precisely reproduce input image, but rather produces a similar image which shares deep features, showing ability to generalize
Next Steps

• The model has not yet been tested in a semi-supervised setting; this would be an immediate next step.

• There is much work yet to be done
  • Developing a mathematical understanding of training dynamics.
  • Arriving at a convergence measure.
  • Parameterization of degree of generalization – “semantic blurring”

• Currently, the “language” developed contains only nouns.
  • Incorporate RNN to allow communication of information beyond object-level (i.e. qualifiers, relationships between perceived objects)
References


Thank You!
Additional Material
How it learns

• Like GAN, comparison between real and fake images drives feature extraction
  • Gradients of $s, \tilde{s}$ face opposite directions, forcing $A, E$ to act as discriminators
• Unlike GAN, SAE *implicitly* differentiates between images in the real domain as a side effect of the explicit objective of differentiating real and fake images.
• Due to non-degenerate SAE encoder
  • Allows *real* symbols $s$ representing different modes to move away from each other.
  • Impossible in the GAN model which was “bound” to the one dimensional case $\{-1,1\}$ – has no freedom of movement.
How it learns: Example

• Take the simple case of a domain of 2 handwritten digits
• At first, modes are “entangled” as $G$ produces images very far from real domain. $A(I_1) = A(I_2)$
How it learns: Example

• At some point $G$ begins producing numbers resembling those from the real domain.

• If it produces an image with a 2 as the result of $G(s^1)$, this would cause implicit differentiation of the symbol for 1 and that for 2, as it tries to differentiate $\tilde{s}_A^1$ from $s_A^1$. 

\begin{align*}
\{1,2\} & \quad \rightarrow \quad \{1\} \\
\{1\} & \quad \rightarrow \quad \{2\}
\end{align*}
How it learns: Example

• Once the modes are differentiated, $G$ can specialize on each mode
  • As it gets symbolic input.
• This allows a smooth process where ever finer details are gradually disentangled.
Problem: Orthogonality of symbols

• Adversarial and syncretic forces reach equilibrium when symbols are orthogonal between the two units.

• After $G$ trains, $\tilde{S}_E \approx S_E$:
  • If $s_E$ is close to $s_A$ (angle less than $\frac{\pi}{2}$) adversarial gradients are dominant pushing both $s_E, \tilde{s}_E$ away.
  • If $s_E$ is far from $s_A$ (angle more than $\frac{\pi}{2}$) syncretic gradients are dominant pushing both $s_E, \tilde{s}_E$ closer.

• Side note: This is a simple consequence of the original GAN paper result that the optimal $D \equiv \frac{1}{2}$ (given optimal $G$)
  • In our case, we can think of $D$ as $\frac{1+\cos x}{2}$, so $x = \frac{\pi}{2}$
Orthogonality of symbols

For now, assume $S_A$ constant.

$G$’s turn:
Orthogonality of symbols

\( G \) finished:
\[ \tilde{S}_E \approx S_E \]

\( E \)'s turn:
Adversarial gradient stronger than syncretic gradient
Orthogonality of symbols

$E$’s turn:
Equilibrium reached at $\frac{\pi}{2}$
Orthogonality of symbols

$E$’s turn:

Most training is done around equilibrium point
Centralization: Natural solution to G/D Equilibrium

• Over generalization causes constant outputs, over diversification causes inability to produce realistic images

• System naturally corrects itself:
  • If $\tilde{I}$ is “overly diverse”, gradients from $A$ into $G$ are stronger.
  • If $\tilde{I}$ is “overly general”, gradients from $E$ into $G$ are stronger.

• BEGAN solved this problem using “proportional control”
  • adding artificial multipliers on the losses and correcting them using an “adaptive term” derived from ratios between losses

• SAE presents a simpler and more elegant solution as the self-correcting behavior naturally arises from the loss gradients
Orthogonality: Unstable

• In the example, $A$ was constant. However as $A$ is not constant there arise stability issues: Both units try to become orthogonal to the other.

• The orthogonality forming between the two units achieves the most stable form when one unit is the “central” unit and the other is the “peripheral” unit.

• Otherwise, the system is unstable: the units “flicker,” sometimes one is central and the other peripheral, or vice versa, and sometimes one is central on some modes and peripheral on others, etc.
Orthogonality: Ambiguous symbols

• Without this clean center/peripheral structure, “ambiguous symbols” can arise:

  • \( A(I_1) = S_A^1 \)
  • \( E(I_1) = S_E^1 \)
  • \( A(I_2) = S_A^2 \)
  • \( E(I_2) = S_E^2 \)

\[
\frac{s_A^1 + s_E^1}{||s_A^1 + s_E^1||} = \frac{s_A^2 + s_E^2}{||s_A^2 + s_E^2||}
\]
Solution: Centralization

• To stabilize symbol space we impose the central/peripheral structure:
  • $\min_{ij \in \text{batch}} d(s_A^i, s_A^j)$
  • $\max_{ij \in \text{batch}} d(s_E^i, s_E^j)$

• We call these the self losses of units $A, E$ respectively.
• We also set as an objective, called the kernel loss:
  • $\min_{ij \in \text{batch}} d(s_A^i, s_E^j)$

• Note this is a pairwise computation, not elementwise as before.
• This causes the two units to maintain the same center of mass.
Centralization: Central unit interpretation

• $A$ is focused on finding general features of images

• Examples:
  • Spatial intelligence: understanding 3-dimensionality of faces, shading, rotations, etc.
  • “Realism:” understanding the correct ratios between objects in the image: placement of facial features, etc.

• It is involved in forming ever deepening associations between the images in the domain.

• Symbols produced by $A$ can perhaps be interpreted as “memories,”
  • Tend to retain salient information while shedding non-salient
Centralization: Peripheral unit interpretation

• The peripheral unit $E$ is focused on finding *unique* features of images: features unique to the input image.

• Examples:
  • Hair style, eye color, skin color, etc.

• It is involved in trying to represent the perceived image as fully as possible.

• Symbols produced by $E$ can perhaps be interpreted as “perceptions.”
Intuition:

Underlying distribution $\mathcal{D}$

Artificially learned subset of features

Supervised model $P_{model}$

Heuristic input

Human learned subset of features

Paciflora  Lily  Rose
Intuition:

Unsupervised model -- $p_{\text{model}}$

Human learned subset of features

Artificially learned subset of features

Human model -- $p_{\mathcal{L}}$

Underlying distribution $\mathcal{D}$

Oryan Barta © 2018 all rights reserved
Intuition: “Implicit language”

Artificially learned subset of features

Goal: Make these isomorphic

Human learned subset of features

Paciflora

Lily

Rose
“Memories” vs “Perceptions”

Memories:
Images produced by $G$ optimizing towards highly associative symbols (central unit $A$)

Perceptions:
Images produced by $G$ optimizing towards highly dissociative symbols (peripheral unit $E$)

Real images:
Smooth mapping: Mnist
Smooth mapping: CelebA

Original

Training 1

Training 2
Semantic Blurring Using White Noise

• To support the generalization process, we add white noise to the symbol seeding the generalizing unit’s fake image.

• This causes an effect we call semantic blurring: the generator learns to associate higher amplitude frequencies in symbol space with more general features in image space.

• With faces, for example, blurring the symbol causes the face produced to lose shallower features, such as hairstyle, but retain deeper features, such as gender.

• This noise is also helpful to stabilize training. This feature is still in infancy – there is much here that needs to be developed.
Oxford102 Flower dataset
Existing Solutions in Detail
Autoencoders and VAEs
Existing solutions: Unsupervised models

• Explicit models:
  • PixelRNN, PixelCNN
  • Compute $p_{model}(x) = \prod_{i=1}^{n} p(x_i|x_1, ..., x_{i-1})$

• Implicit (Generative) models:
  • Naïve Autoencoders
  • Variational Autoencoders
  • Generative Adversarial Networks
    • Cutting edge currently
Autoencoders

Deep “object level” features of data not effectively learned, as $l_2$ loss between two images representing same object can be very large.

**Scheme**

Real image $\rightarrow$ Encoder *(convnet)* $\rightarrow$ Feature vector $\rightarrow$ Decoder *(deconvnet)* $\rightarrow$ Reconstructed image

$l_2$ loss

**Example implementation**

Feature space [mnist]:
Not smooth
Variational Autoencoders

- Introduction of latent sampling smooths the feature space resulting in more general features
- Images still blurry