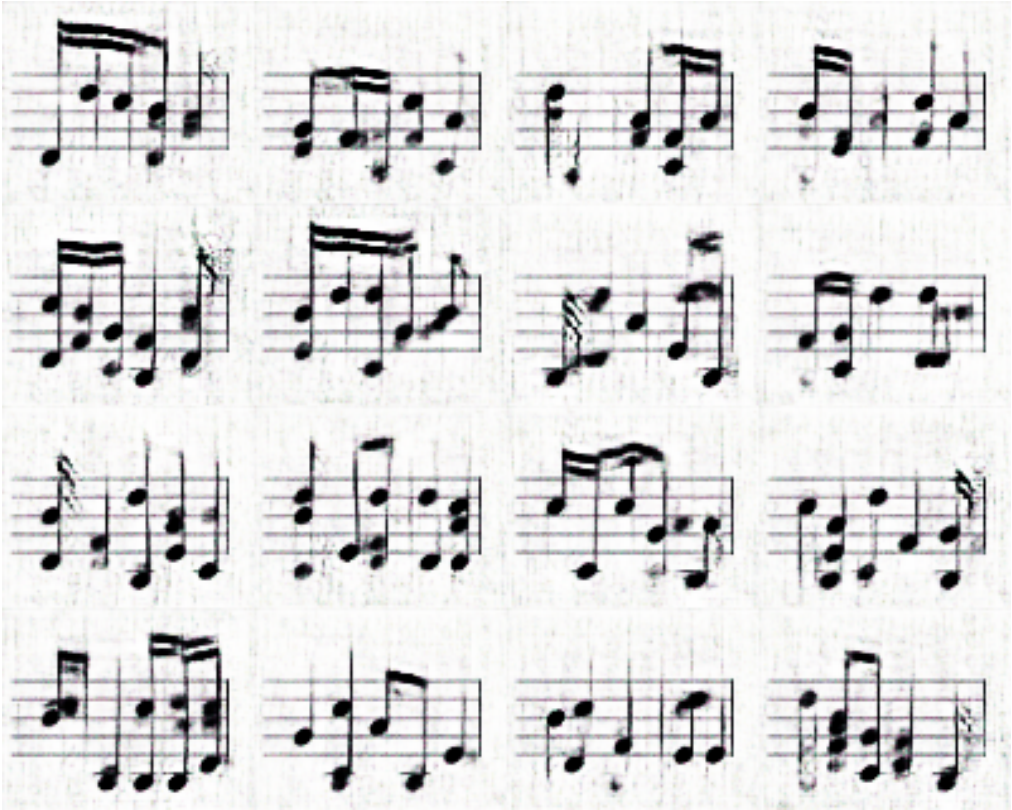
Symbolic Autoencoder

Oryan Barta

Oryan Barta © 2018 all rights reserved

Outline

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



Oryan Barta © 2018 all rights reserved

Problem and Motivation

Oryan Barta © 2018 all rights reserved

Problem: Unsupervised learning

- Given a set S of points sampled from distribution $\mathcal{D}$, we want to compute $\mathrm{p}_{model} \approx p_{\mathcal{D}}$
  - Explicitly or implicitly
- Formally, we want to minimize cross entropy:

$$H(p_{\mathcal{D}}, p_{model}) = -\,\mathrm{E}_{x\sim p_{\mathcal{D}}}(\log p_{model}) = -\sum_{x\sim\mathcal{D}} p_{\mathcal{D}}(x)\log\big(p_{model}(x)\big)$$

- Simplifies to minimization of KL-divergence:
  - $D_{\mathrm{KL}}(p \parallel q) = \int_{-\infty}^{\infty} p(x)\log\frac{p(x)}{q(x)}\,dx$

Continuous version:
$H(p,q) = -\,\mathrm{E}_{x\sim p}(\log q) = H(p) + D_{\mathrm{KL}}(p \parallel q)$
$D_{\mathrm{KL}}(p \parallel q) = \int_{-\infty}^{\infty} p(x)\log\frac{p(x)}{q(x)}\,dx$, *(KL divergence)*
$H(p) = -\int_{-\infty}^{\infty} p(x)\log p(x)\,dx$, *(Entropy)*

- Alternative formulation : maximize likelihood over model parameters:

$$\theta^* = \mathrm{argmax}_\theta\left(\mathbb{E}_{x\sim p_{\mathcal{D}}}(\log p_{model}(x|\theta))\right)$$

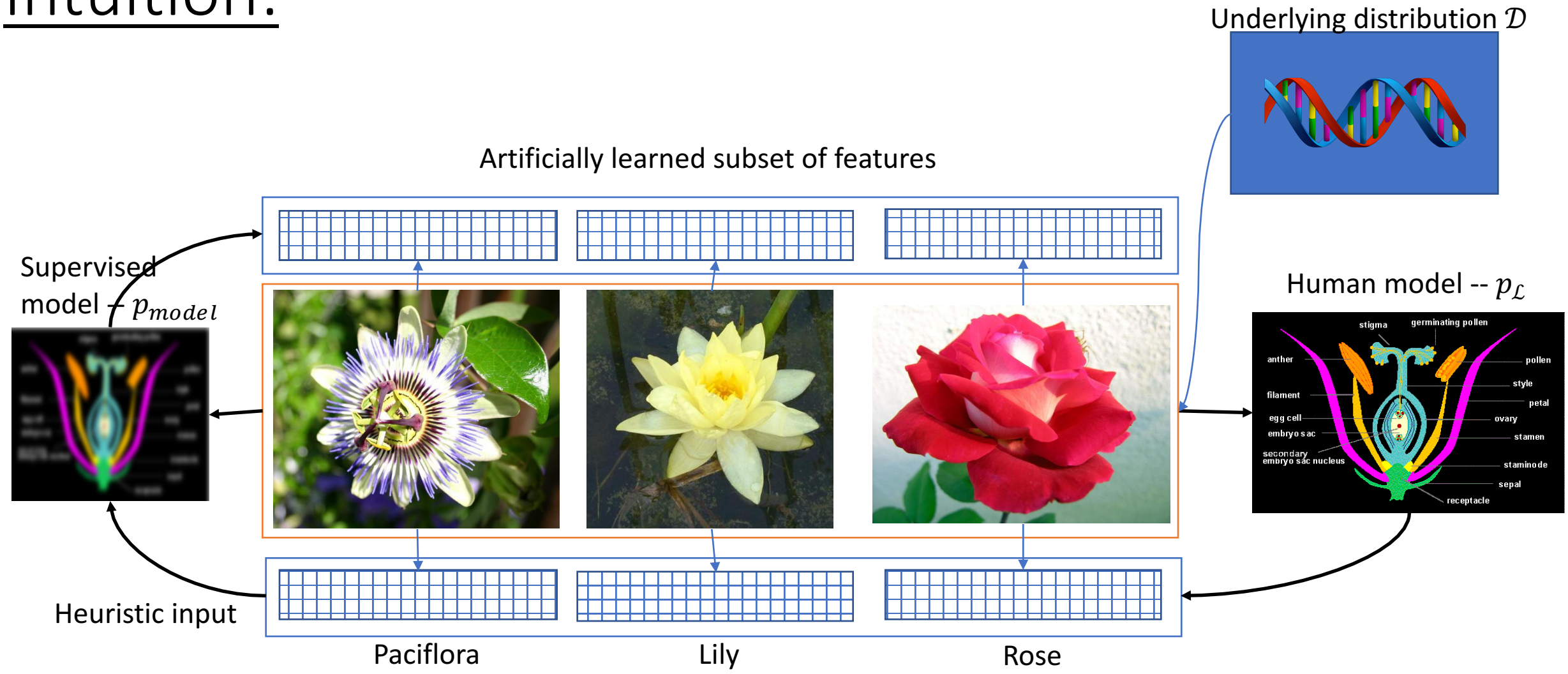
Oryan Barta © 2018 all rights reserved

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

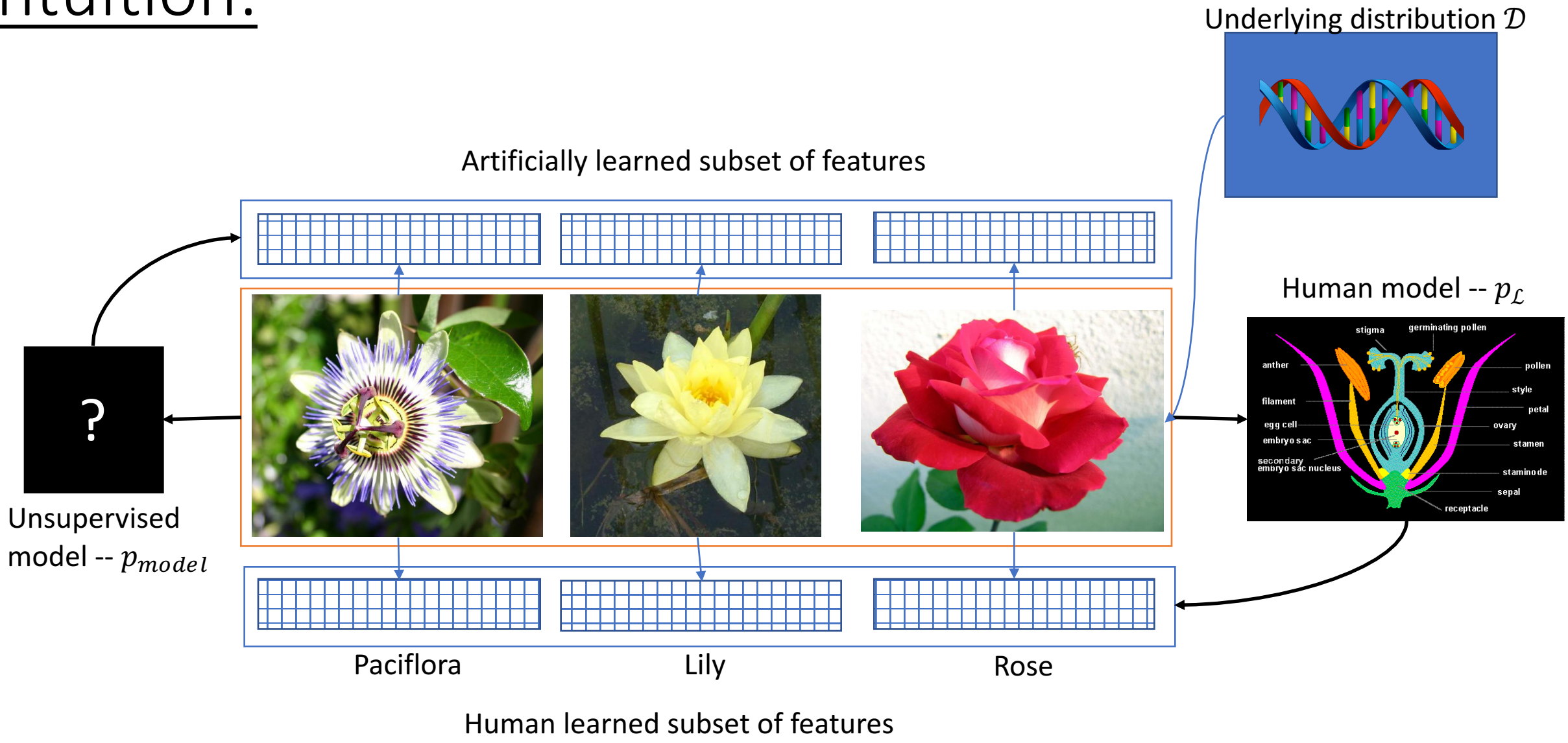
Oryan Barta © 2018 all rights reserved

Intuition:



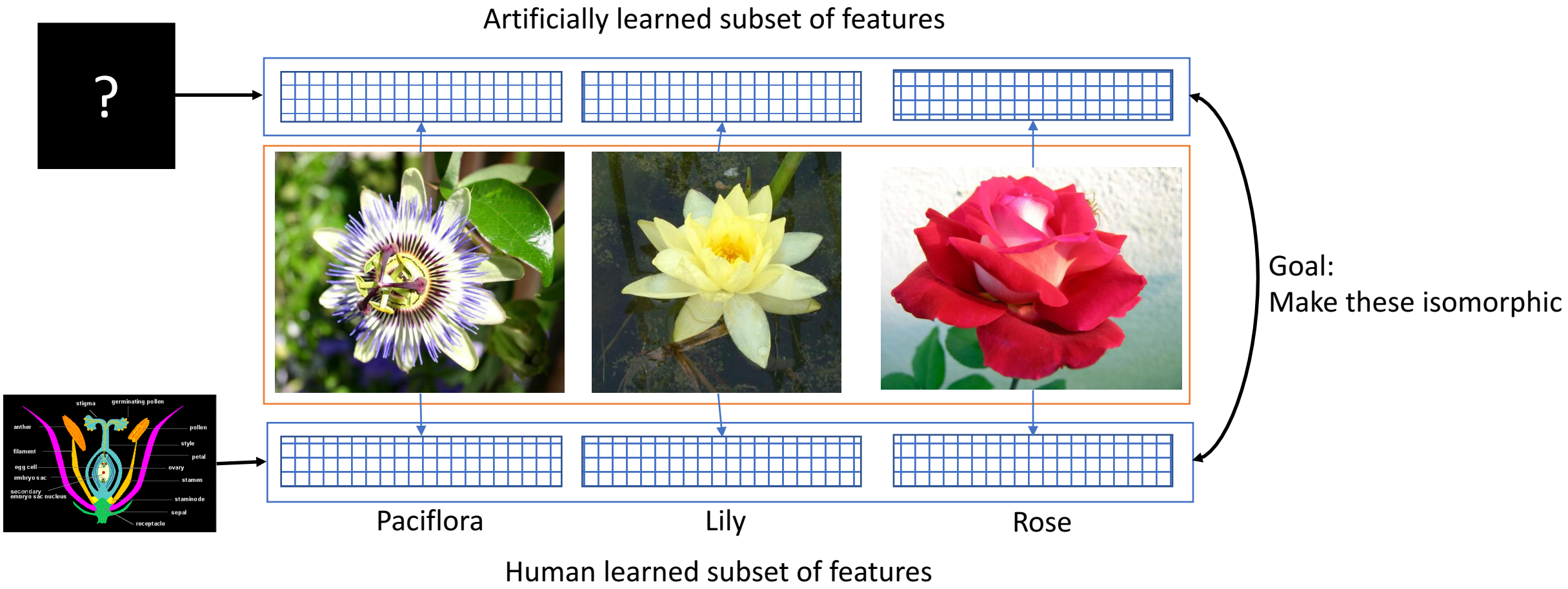
Oryan Barta © 2018 all rights reserved

Intuition:



Oryan Barta © 2018 all rights reserved

Intuition: “Implicit language”



Oryan Barta © 2018 all rights reserved

Project Goal: Language creation

- Want a system that can *automatically label* images, meaning:
  - Training only on images without labels, produce labeling $p'_{\mathcal{L}}$ which minimizes $H(p_{\mathcal{L}}, p'_{\mathcal{L}})$
  - Intuitively: Produced labels $\mathcal{L}'$ that are approximately isomorphic with human labels $\mathcal{L}$.
  - $\mathcal{L}'$ 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

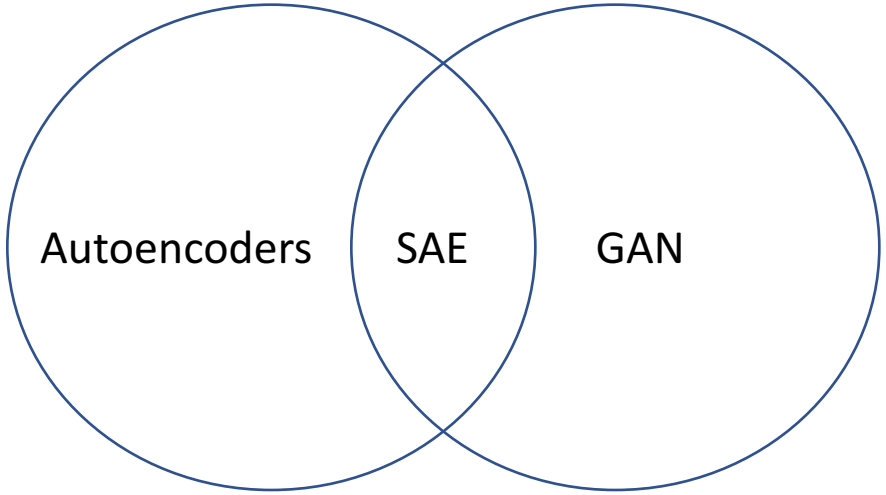
Oryan Barta © 2018 all rights reserved

Existing Solutions

Oryan Barta © 2018 all rights reserved

Existing solutions: Unsupervised models

- Explicit models:
  - PixelRNN, PixelCNN
  - Compute $p_{model}(x) = \Pi_{i=1}^{n} p(x_i | x_1, \dots, x_{i-1})$
- Implicit (Generative) models:
  - Naïve Autoencoders
  - Variational Autoencoders
  - Generative Adversarial Networks
    - Cutting edge currently



Oryan Barta © 2018 all rights reserved

GAN: Generative Adversarial Networks

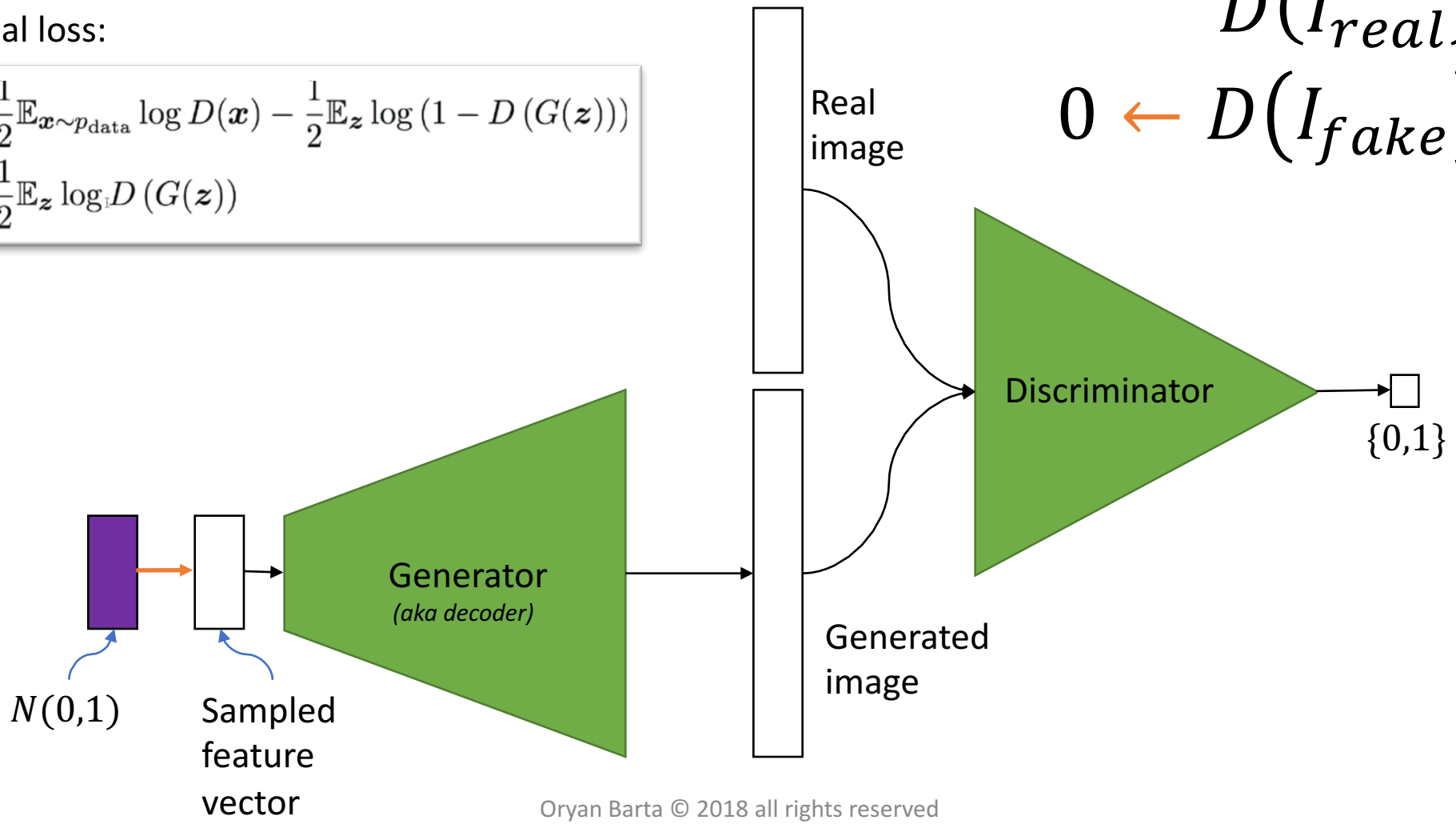
Adversarial loss:

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)$$

$$J^{(G)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log D\left(G(\boldsymbol{z})\right)$$

$$D(I_{real}) \rightarrow 1$$

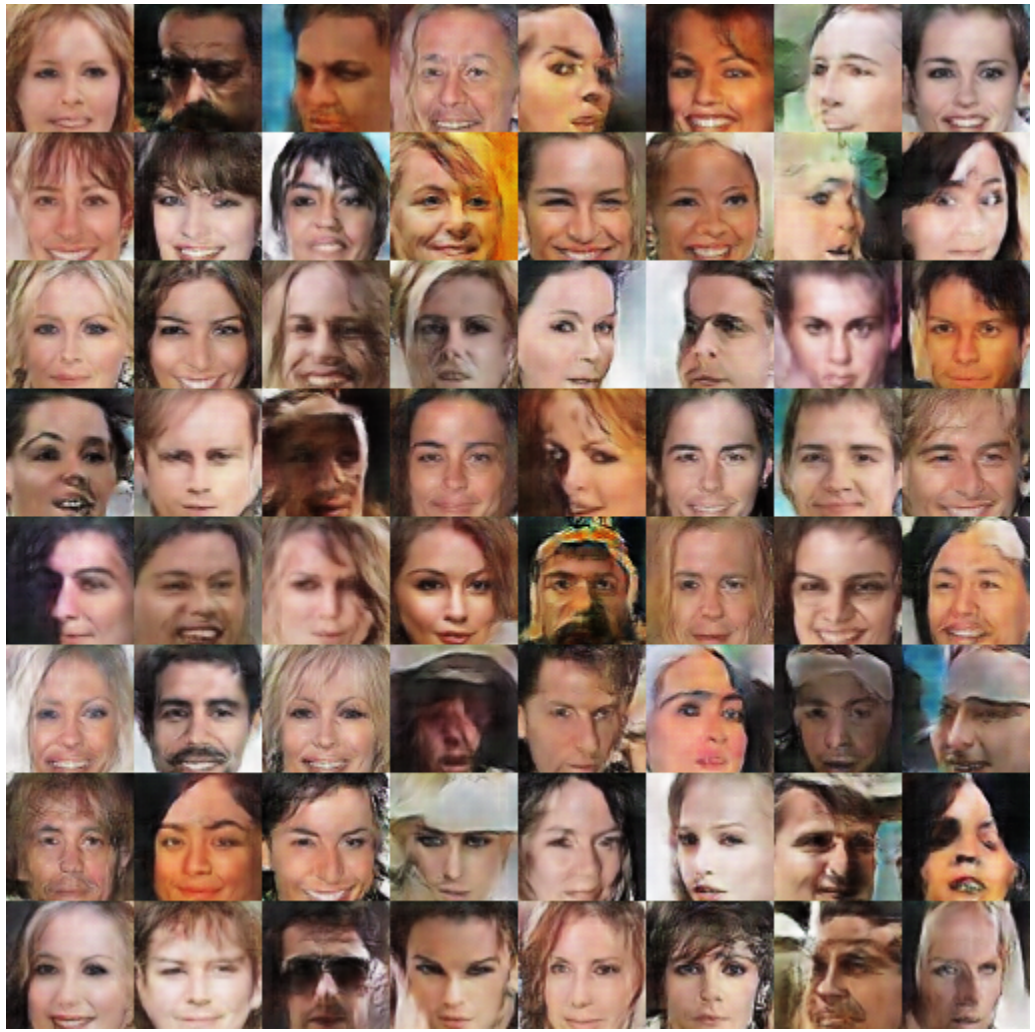
$$0 \leftarrow D(I_{fake}) \rightarrow 1$$



Oryan Barta © 2018 all rights reserved

DCGAN Results

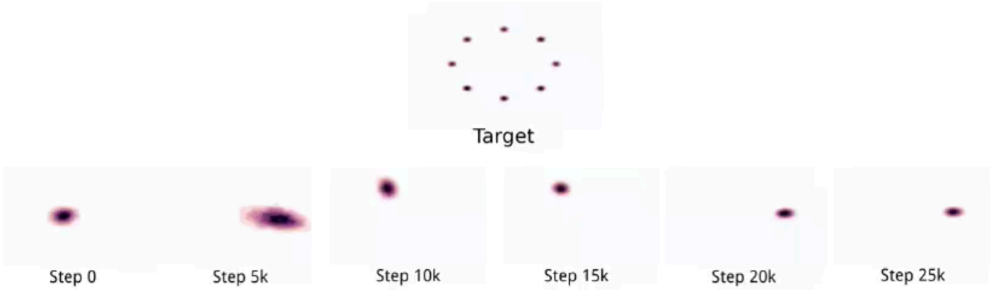
After 6 epochs



Oryan Barta © 2018 all rights reserved

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)

(Goodfellow 2016)

Oryan Barta © 2018 all rights reserved

The GAN Zoo

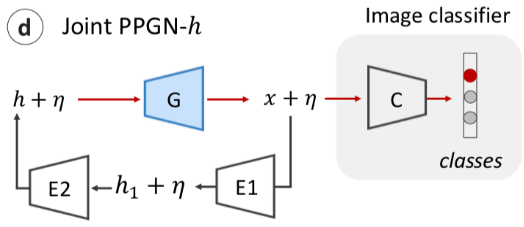
https://github.com/hindupuravinash/the-gan-zoo

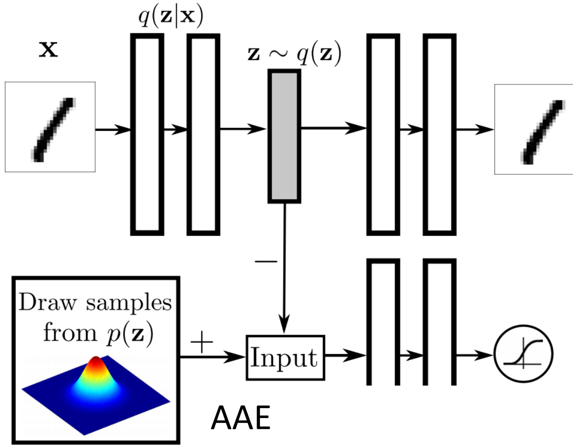
Year	Month	Abbr.	Title	Arxiv
2014	6	GAN	Generative Adversaria	https://arxiv.org/abs/1406.2661
2014	11	CGAN	Conditional Generativ	https://arxiv.org/abs/1411.1784
2015	6	LAPGAN	Deep Generative Imag	https://arxiv.org/abs/1506.05751
2016	6	ALI	Adversarially Learned	https://arxiv.org/abs/1606.00704
2016	6	CoGAN	Coupled Generative A	https://arxiv.org/abs/1606.07536v2
2016	6	Improved (	Improved Techniques	https://arxiv.org/abs/1606.03498
2016	6	InfoGAN	InfoGAN: Interpretabl	https://arxiv.org/abs/1606.03657v1
2016	11	GMAN	Generative Multi-Adv	http://arxiv.org/abs/1611.01673
2017	1	LAGAN	Learning Particle Phys	https://arxiv.org/abs/1701.05927

2016	12	textGAN	Generating Text via A	https://zhegan27.github.io/Papers/text
2017	1	AdaGAN	AdaGAN: Boosting Ge	https://arxiv.org/abs/1701.02386v1
2017	1	ID-CGAN	Image De-raining Usir	https://arxiv.org/abs/1701.05957v3
2017	1	LAGAN	Learning Particle Phys	https://arxiv.org/abs/1701.05927
2017	1	LS-GAN	Loss-Sensitive Genera	https://arxiv.org/abs/1701.06264
2017			erstein GAN	https://arxiv.org/abs/1701.07875v2
2017	2	MalGAN	Generating Adversaria	htt
2017	2	MaliGAN	Maximum-Likelihood	htt
2017	2	McGAN	McGan: Mean and Co	htt
2017	2	ST-GAN	Style Transfer Genera	htt
2017	6	crVAE-G		
	3	SEGAN	SEGAN: Speech Enhar	http
	3	SeGAN	SeGAN: Segmenting a	http
2017	3	SGAN	Steganographic Gene	http
2017	5	EGAN	Enhanced Experience	https
2017	5	Fisher GAN	Fisher GAN	https://arxiv.org/abs/1705.09675
2017	8	constrast-(	Generative Semantic	https://arxiv.org/abs/1708.00315

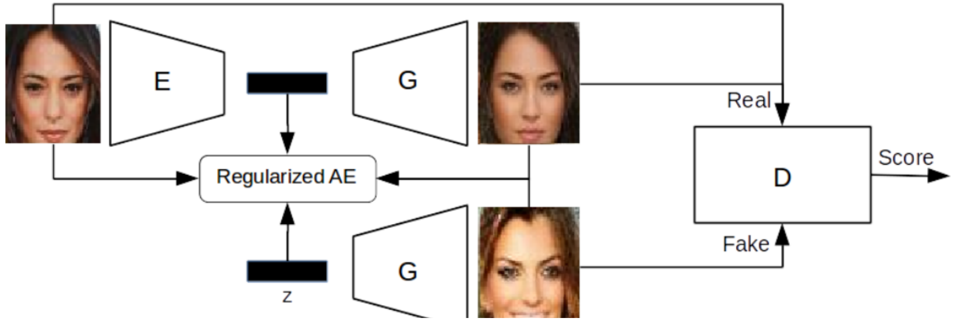
2017	5	Geometr
2017	5	MMD-G
2017	5	ORGAN
2017	5	PSGAN
2017	5	Pose-GA
2017	5	RankGA
2017	5	RPGAN
2017	5	RWGAN

2017	12	GAGAN
2017	12	GANG
2017	12	GANosaic
2017	12	GAN-RS
2017	12	G2-GAN
2017	12	IdCycleGAN
2017	12	manifold-WGAN
2017	12	MC-GAN
2017	12	MIL-GAN
2017	12	MS-GAN
2017	12	PacGAN
2017	12	PPAN
2017	12	PN-GAN
2017	12	RAN
2017	12	SGAN
2017	12	SRPGAN
2017	12	ST-CGAN
2017	12	Super-FAN
2017	12	TV-GAN
2017	12	UGACH
2017	12	UV-GAN
2017	12	VGAN
2017	12	weGAN
2018	1	AdvGAN
2018	1	CFG-GAN
2018	1	CipherGAN
2018	1	Cross-GAN
	1	dp-GAN
		ecGAN
		FusedGAN
		LAC-GAN
		SG-GAN
		SketchyGAN
		UGAN
		MaskGAN
2018	1	GLCA-GAN
2018	1	GeoGAN
2018	1	tempoGAN
2018	2	AmbientGAN
2018	2	C-GAN
2018	2	DP-GAN
2018	2	GC-GAN
2018	2	SCH-GAN
2018	2	WaveGAN
2018	2	First Order GAN
2018	2	CapsuleGAN
2018	2	RadialGAN
2018	2	DA-GAN
2018	2	DPGAN
2018	2	VoiceGAN
2018	2	LB-GAN
2018	2	Defense-GAN
2018	2	MAGAN
2018	2	ATA-GAN
2018	2	PGD-GAN
2018	2	SWGAN
2018	2	SAR-GAN
2018	2	ND-GAN
2018	2	StainGAN: Stain Style Trans
	2	Domain and Geometry Agn
	3	Spike-GAN
	3	E-GAN
	3	NetGAN
	3	memoryGAN
	3	OCAN
	3	ST-GAN
	3	GONet
	3	OT-GAN
	3	tiny-GAN
	3	Attention-GAN
	3	PGGAN
	3	BranchGAN
	3	Fictitious GAN
	3	GAAN
	3	MTGAN
	3	VOS-GAN
	3	Text2Shape
	3	BAGAN
	3	ELEGANT
	3	B-DCGAN
	3	D2IA-GAN
	3	NCE-GAN
	3	DBLRGAN
	3	Social GAN
	3	Sdf-GAN
	4	D-WCGAN
	4	SyncGAN
	4	SAGA
	4	CorrGAN
	4	FBGAN
	4	PS-GAN
	4	ProGanSR
	4	NAN
	4	MGGAN
	4	ModularGAN
	4	Graphical-GAN
	4	MelanoGAN
2018	4	sGAN
2018	4	AF-DCGAN

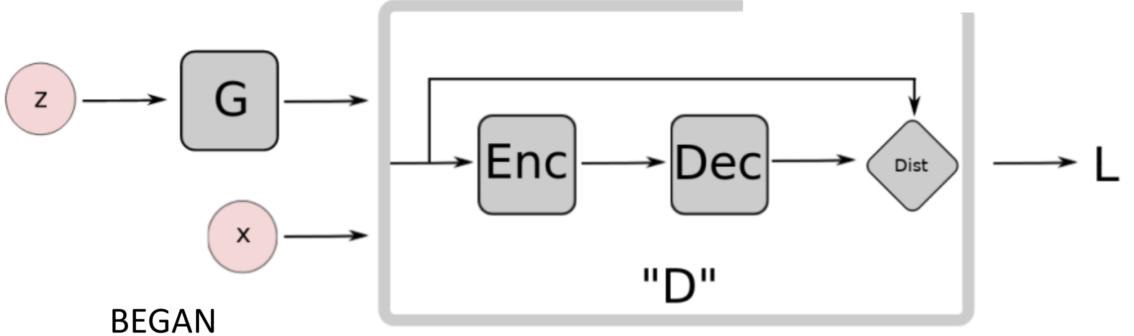




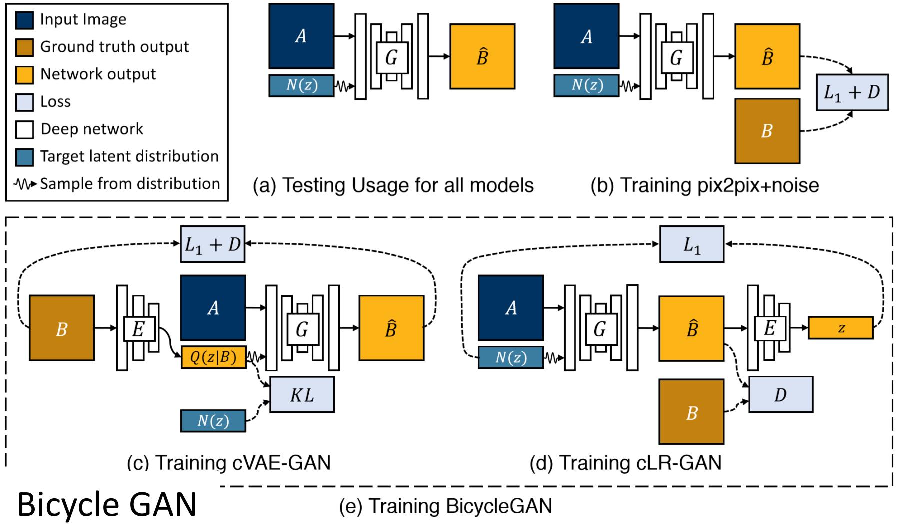
AAE (Adversarial Autoencoder)



Generative Adversarial Autoencoder Networks



BEGAN

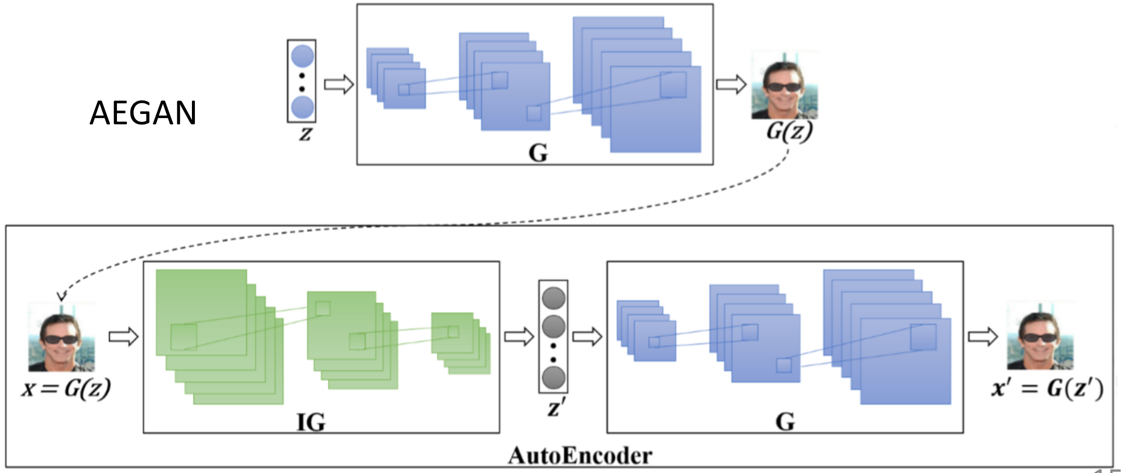


(a) Testing Usage for all models (b) Training pix2pix+noise

(c) Training cVAE-GAN (d) Training cLR-GAN

(e) Training BicycleGAN

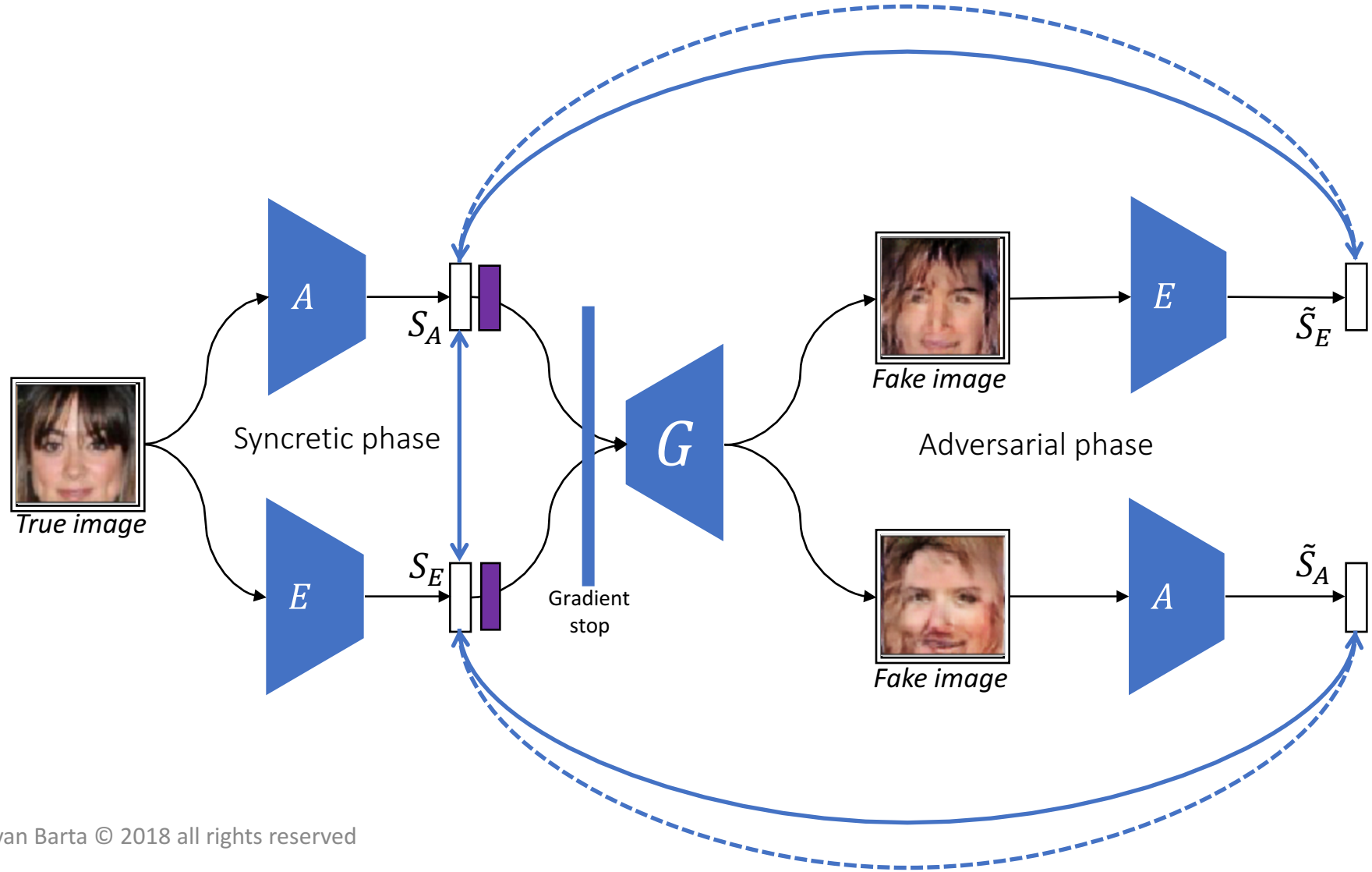
Bicycle GAN



Proposed Method

Oryan Barta © 2018 all rights reserved

SAE Computation

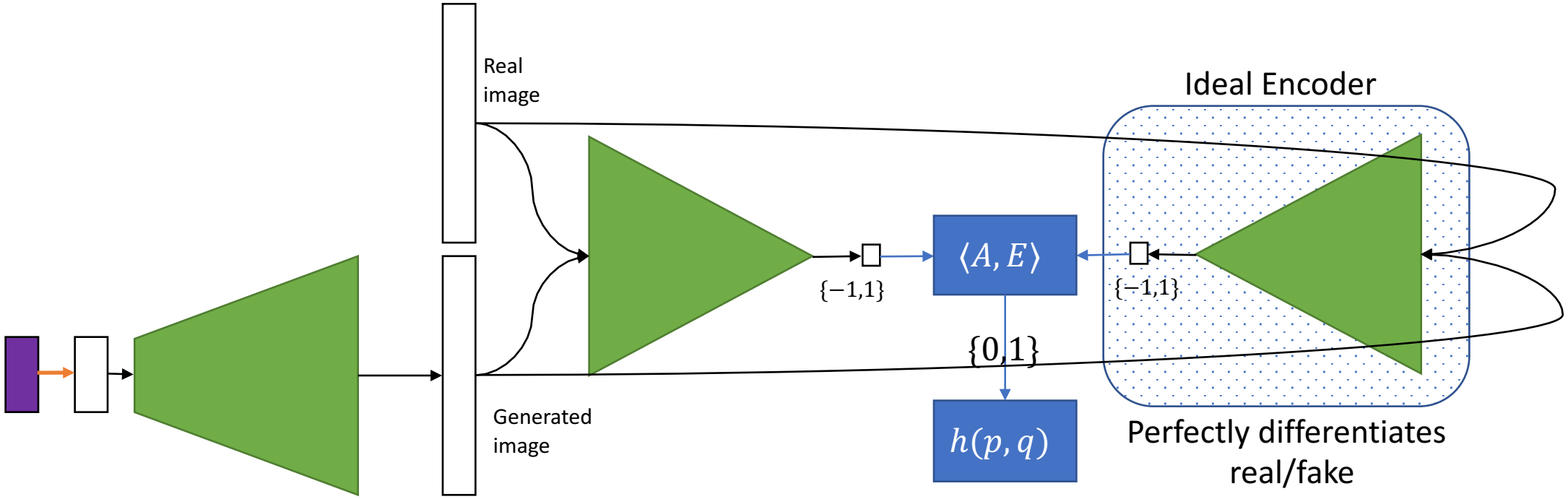


Oryan Barta © 2018 all rights reserved

SAE generalizes GAN:

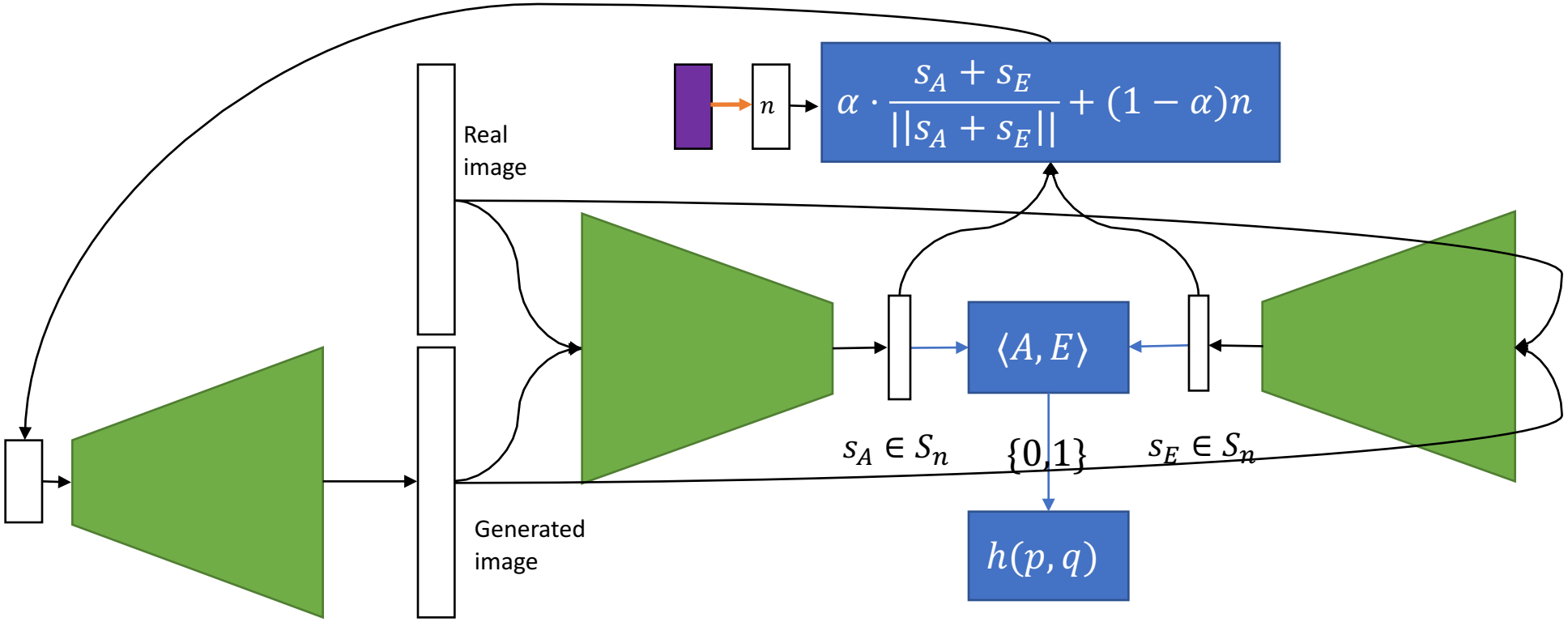
$$SAE(1,0) \equiv GAN$$

- 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 <u>each unit assumes the other is ideal</u>
  - 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. $l_{adv}^A(s_A, \tilde{s}_E) \leftarrow h(p(s_A, \tilde{s}_E), 0)$
  10. $l_{adv}^E(s_E, \tilde{s}_A) \leftarrow h(p(s_E, \tilde{s}_A), 0)$
  11. $\tilde{l}_{adv}^A(s_A, \tilde{s}_E) \leftarrow h(p(s_A, \tilde{s}_E), 1)$
  12. $\tilde{l}_{adv}^E(s_E, \tilde{s}_A) \leftarrow h(p(s_E, \tilde{s}_A), 1)$
  13. $l_D \leftarrow \tilde{l}_{adv}^A + \tilde{l}_{adv}^E$
  14. $l_G \leftarrow \tilde{l}_{adv}^A + \tilde{l}_{adv}^E$
- Backprop & update weights:
  8. $\theta_D \leftarrow \theta_D + \mu \frac{\partial l_D}{\partial \theta_D}$
  9. $\theta_G \leftarrow \theta_G + \mu \frac{\partial l_G}{\partial \theta_G}$

Oryan Barta © 2018 all rights reserved

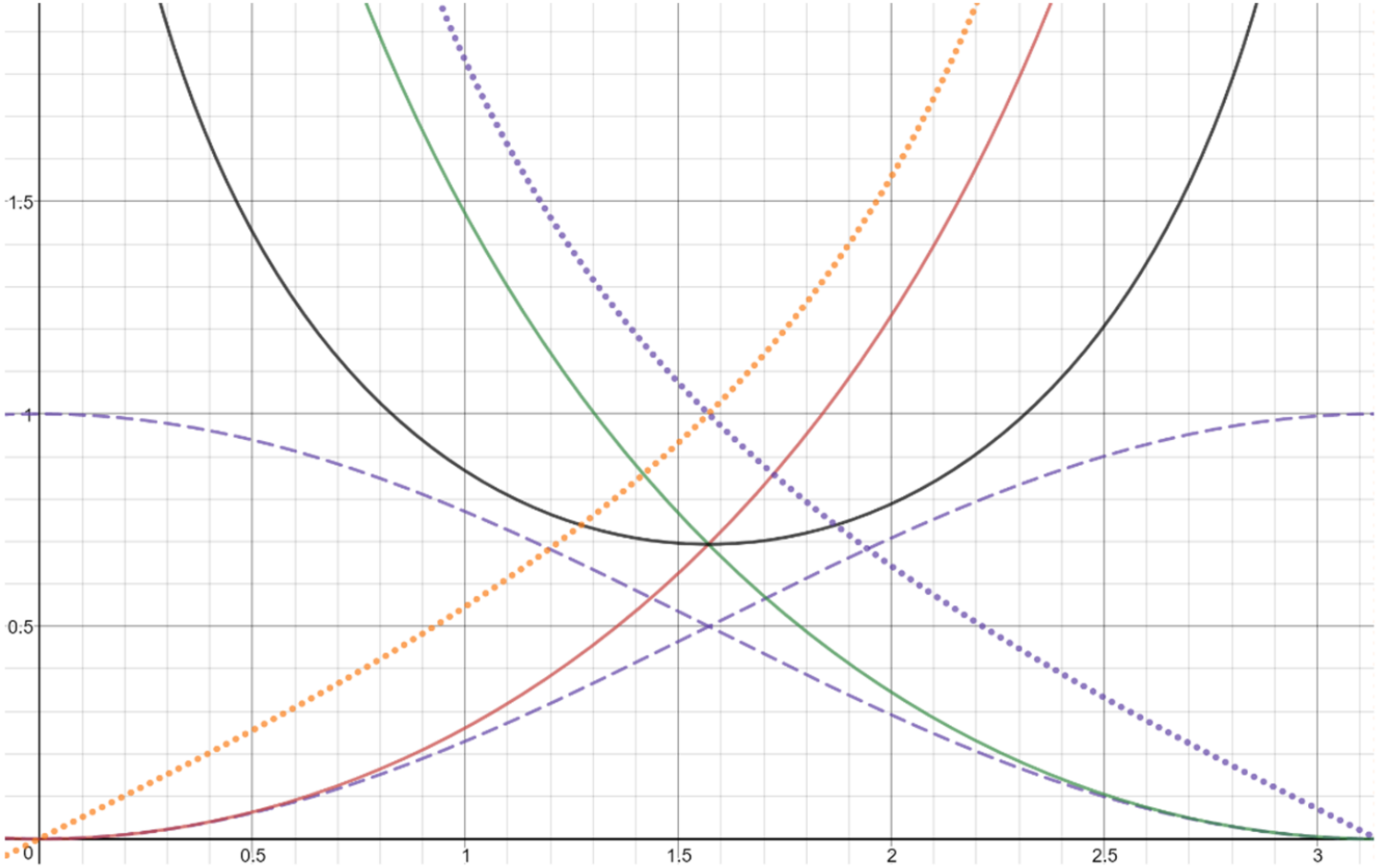
Distance & Loss functions

- 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
  - $h(p, q) = -q \cdot \ln(p) - (1 - q) \cdot \ln(1 - p)$

Oryan Barta © 2018 all rights reserved

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:

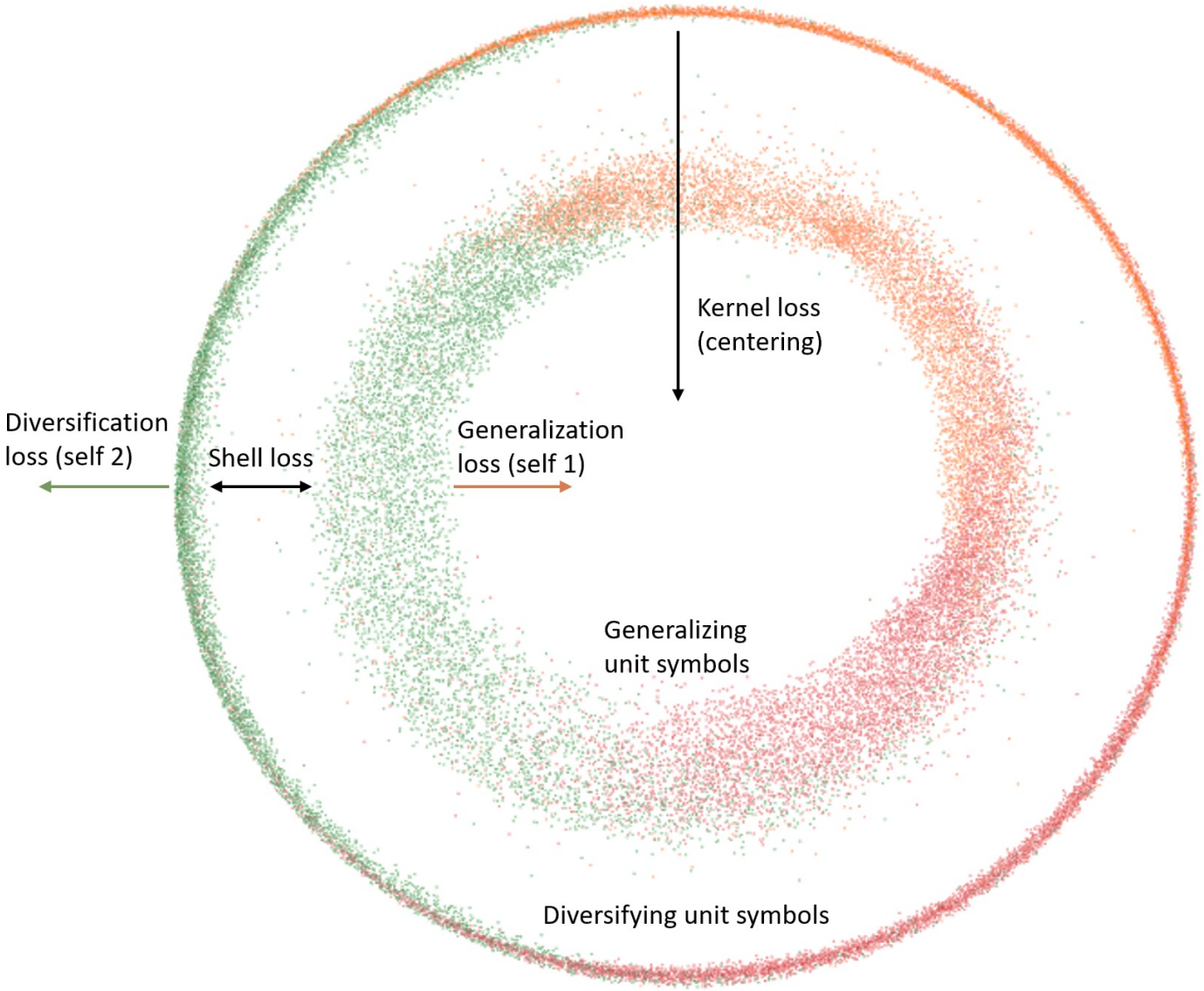
- $h(x, 1)$: attractive
- $h(x, 0)$: repulsive
- Dashed: distance functions (square half angle)
- Dotted: gradients of loss
  - $\tan\frac{x}{2}, \tan\frac{x+\pi}{2}$
- Equilibrium point: $\boldsymbol{x} = \frac{\boldsymbol{\pi}}{\mathbf{2}}$
  - When $\tan\frac{x}{2} + \tan\frac{x+\pi}{2} = 0$

Oryan Barta © 2018 all rights reserved

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

Symbolic sphere S

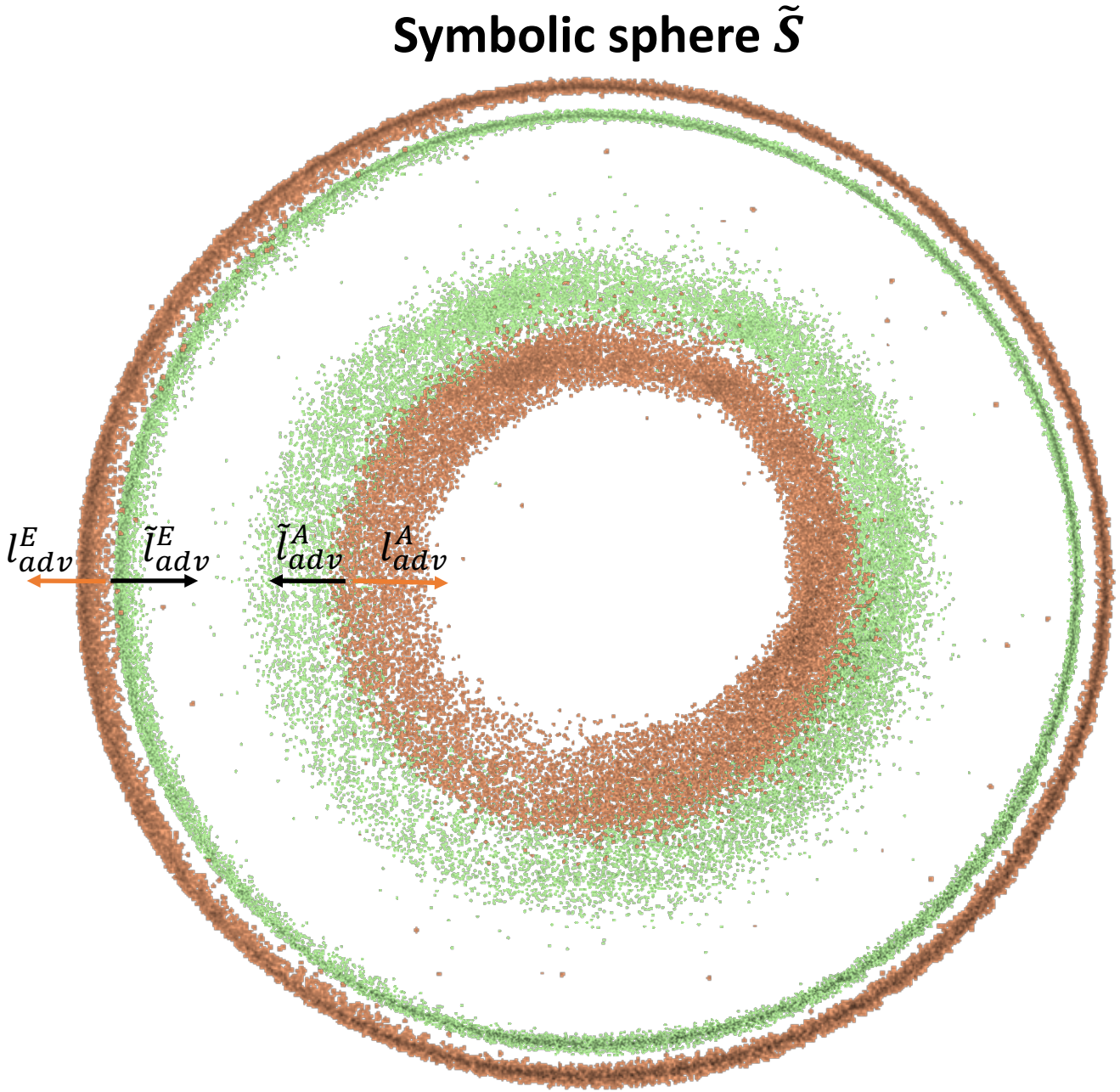


Mnist on 3 digits

Oryan Barta © 2018 all rights reserved

Adversarial loss:

- The adversarial loss provides the gradient direction for fake images.
- The crossed generator objective:
  - $A\big(G(s)\big) \rightarrow s_E$
  - $E\big(G(s)\big) \rightarrow s_A$
- The discriminator objectives are the opposite.



Oryan Barta © 2018 all rights reserved

Results

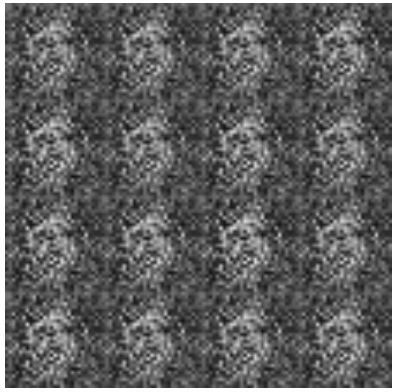
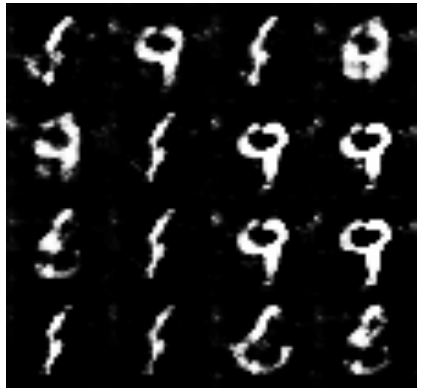
Oryan Barta © 2018 all rights reserved

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

Oryan Barta © 2018 all rights reserved

Compare Original DCGAN Results to SAE:
Mnist Unsupervised

	Iteration: 100	5 minutes Iteration: 1000	40 minutes Iteration: 6000	Iteration: 100,000
SAE				
Original				

Oryan Barta © 2018 all rights reserved

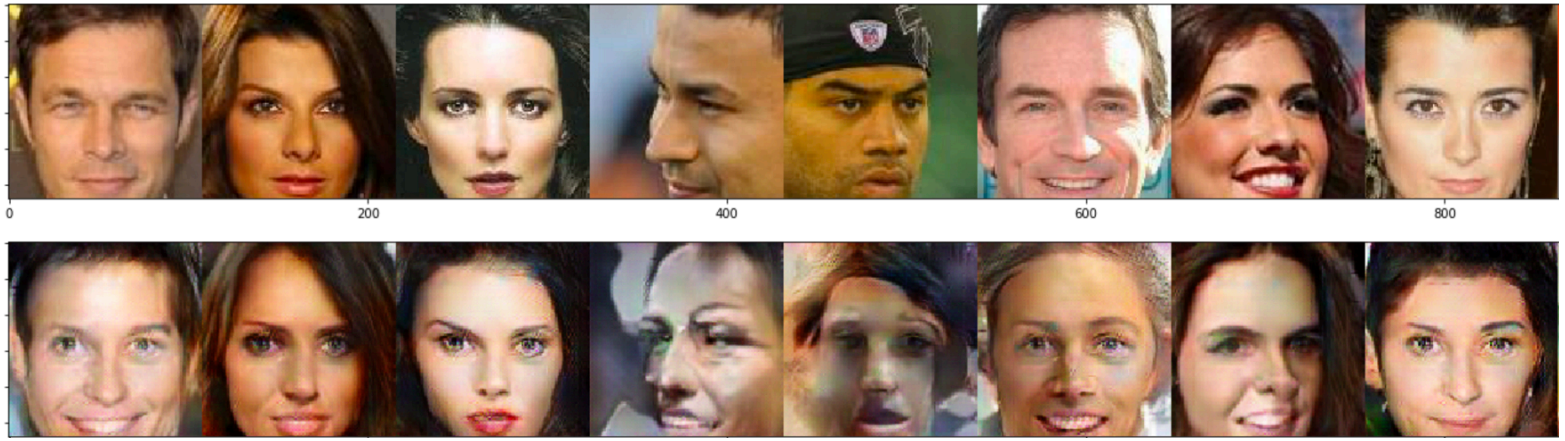
Semantic Reconstructions

Oryan Barta © 2018 all rights reserved

Semantic features learned

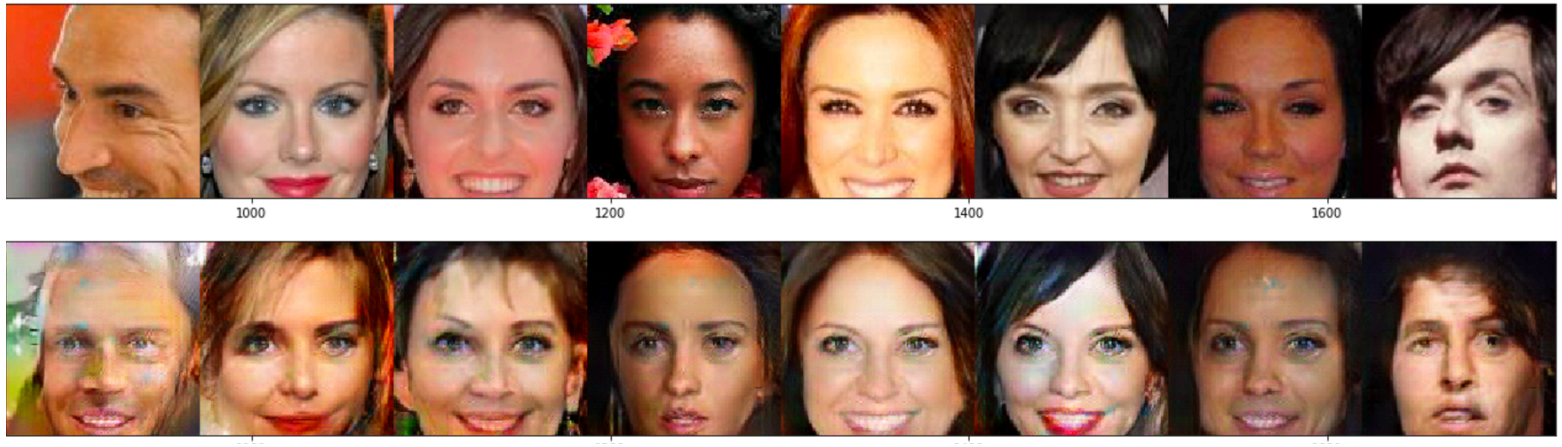
Original

Generated



Original

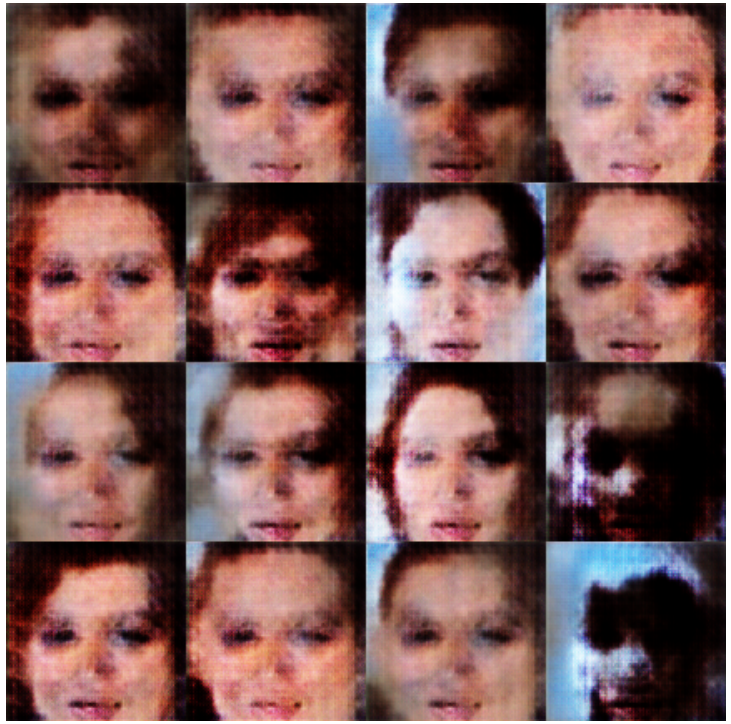
Generated

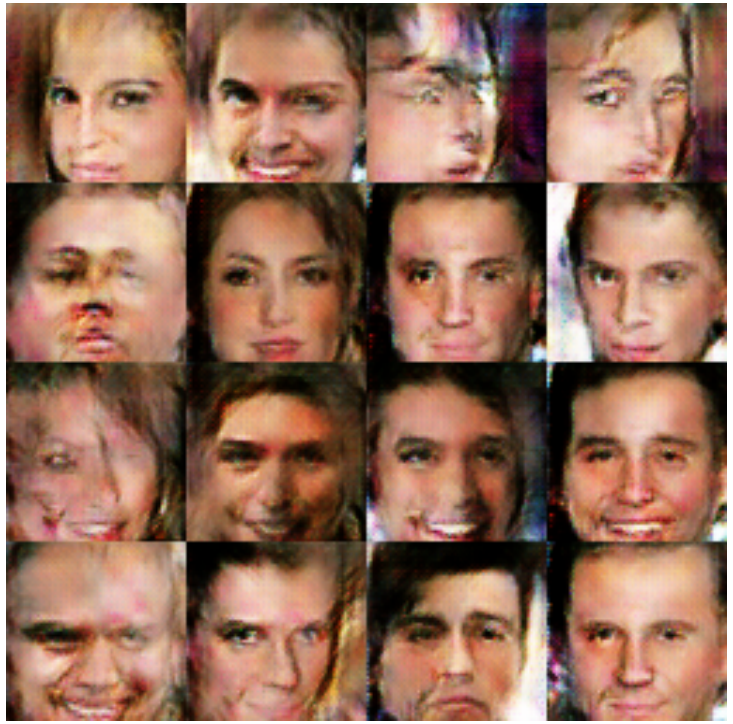


Oryan Barta © 2018 all rights reserved

DCGAN vs SAE

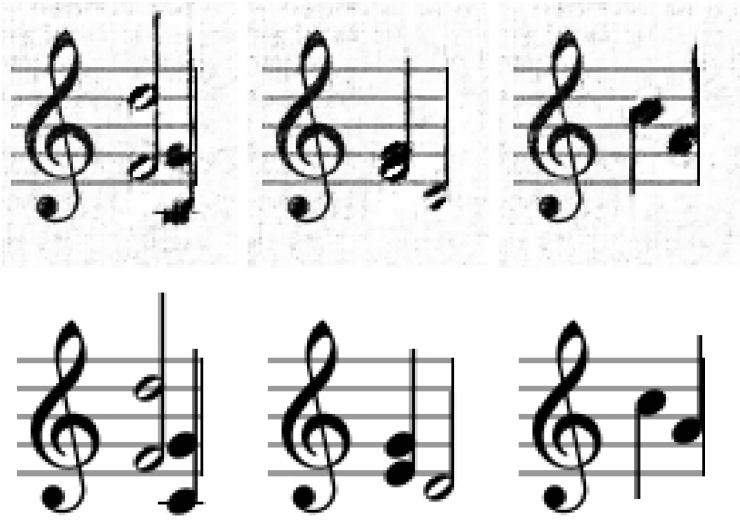
After ½ an epoch

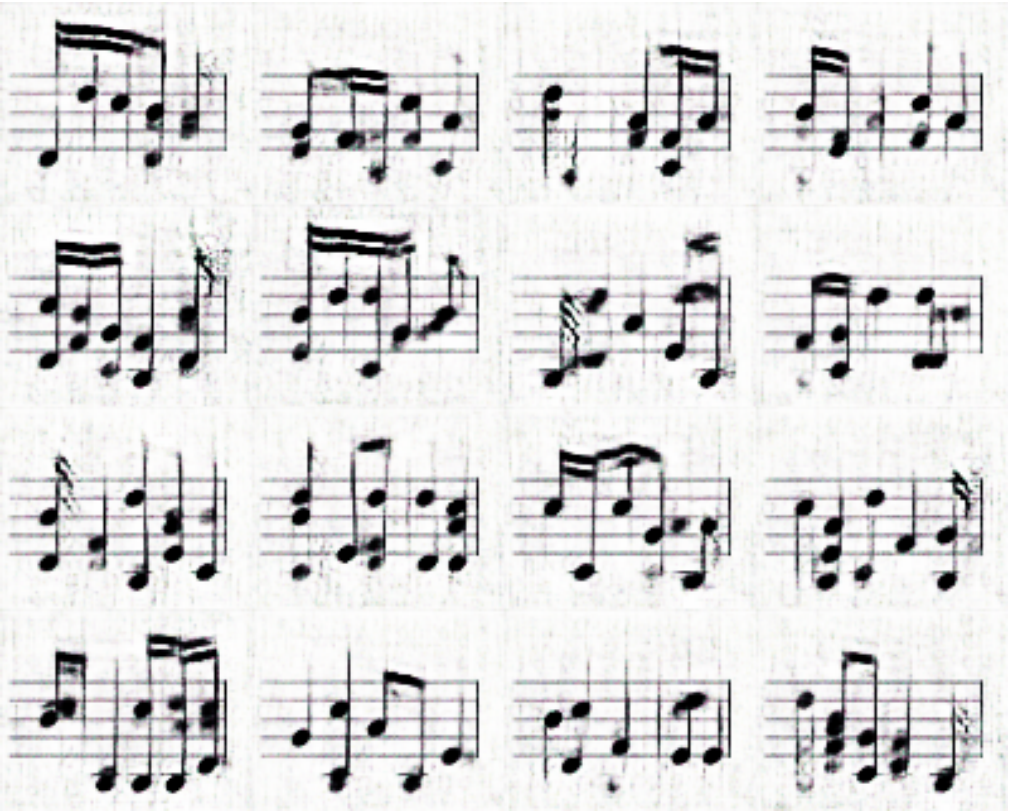




Oryan Barta © 2018 all rights reserved

Experiment on highly structured data: Music notation





Oryan Barta © 2018 all rights reserved

Conclusion

Oryan Barta © 2018 all rights reserved

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

Oryan Barta © 2018 all rights reserved

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)

Oryan Barta © 2018 all rights reserved

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, et al. **Generative adversarial nets**. In *NIPS*, 2014.

[2] Alec Radford, Luke Metz, Soumith Chintala, **Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks**, conference paper at ICLR 2016

[3] Anh Nguyen, Jeff Clune, Yoshua Bengio, Alexey Dosovitskiy, Jason Yosinski, **Plug & Play Generative Networks: Conditional Iterative Generation of Images in Latent Space**, IEEE 2017

[4] Martin Arjovsky, Soumith Chintala, Léon Bottou, **Wasserstein GAN**, arxiv 2017

[5] Junyu Luo, Yong Xu, Chenwei Tang, Jiancheng Lv, **Learning Inverse Mapping by Autoencoder based Generative Adversarial Nets**, arxiv 2017

[6] Jeff Donahue, Philipp Krähenbühl, Trevor Darrell, **Adversarial Feature Learning**, arxiv 2017

[7] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, Pieter Abbeel, **InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets**, arXiv 2016

[8] David Berthelot, Thomas Schumm, Luke Metz, **BEGAN: Boundary Equilibrium Generative Adversarial Networks**, arxiv.org, May 2017

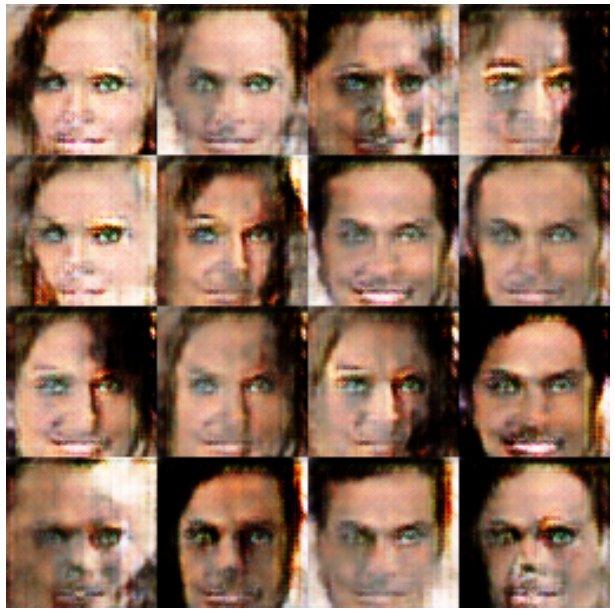
Oryan Barta © 2018 all rights reserved

Questions & Discussion

Oryan Barta © 2018 all rights reserved

Thank You!

Oryan Barta © 2018 all rights reserved



Oryan Barta © 2018 all rights reserved

Additional Material

Oryan Barta © 2018 all rights reserved

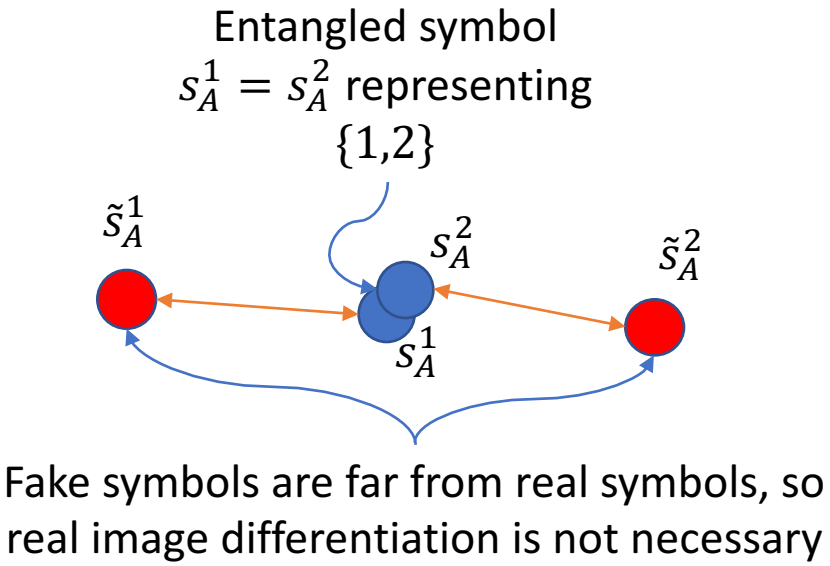
How it learns

- 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.

Oryan Barta © 2018 all rights reserved

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. $A(I_1) = A(I_2)$



Oryan Barta © 2018 all rights reserved

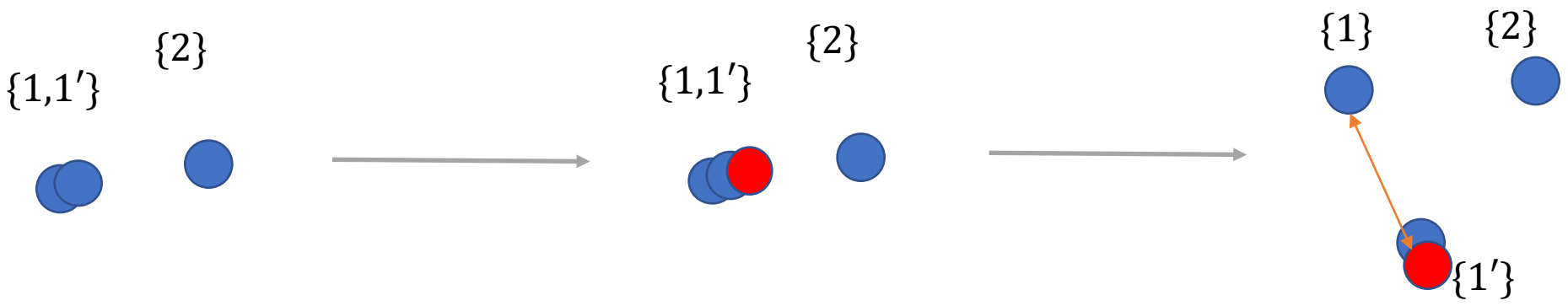
How it learns: Example

- 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.

Oryan Barta © 2018 all rights reserved

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.



Oryan Barta © 2018 all rights reserved

Problem: Orthogonality of symbols

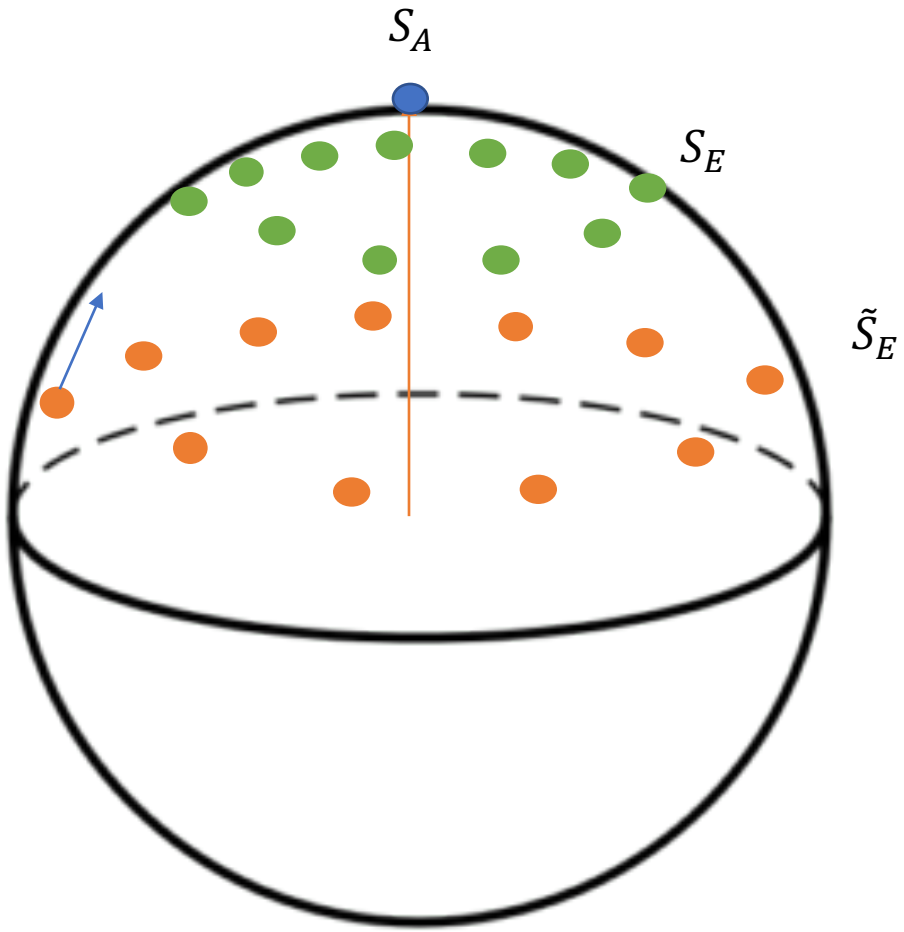
- Adversarial and syncretic forces reach equilibrium when symbols are orthogonal between the two units.
- After G trains, $\tilde{s}_E \approx s_E$:
  - If s_E is close to s_A (angle less than $\frac{\pi}{2}$) adversarial gradients are dominant pushing both $s_E, \tilde{s}_E$ away.
  - If s_E is far 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 $x = \frac{\pi}{2}$

Oryan Barta © 2018 all rights reserved

Orthogonality of symbols

For now, assume S_A constant.

G's turn:



Oryan Barta © 2018 all rights reserved

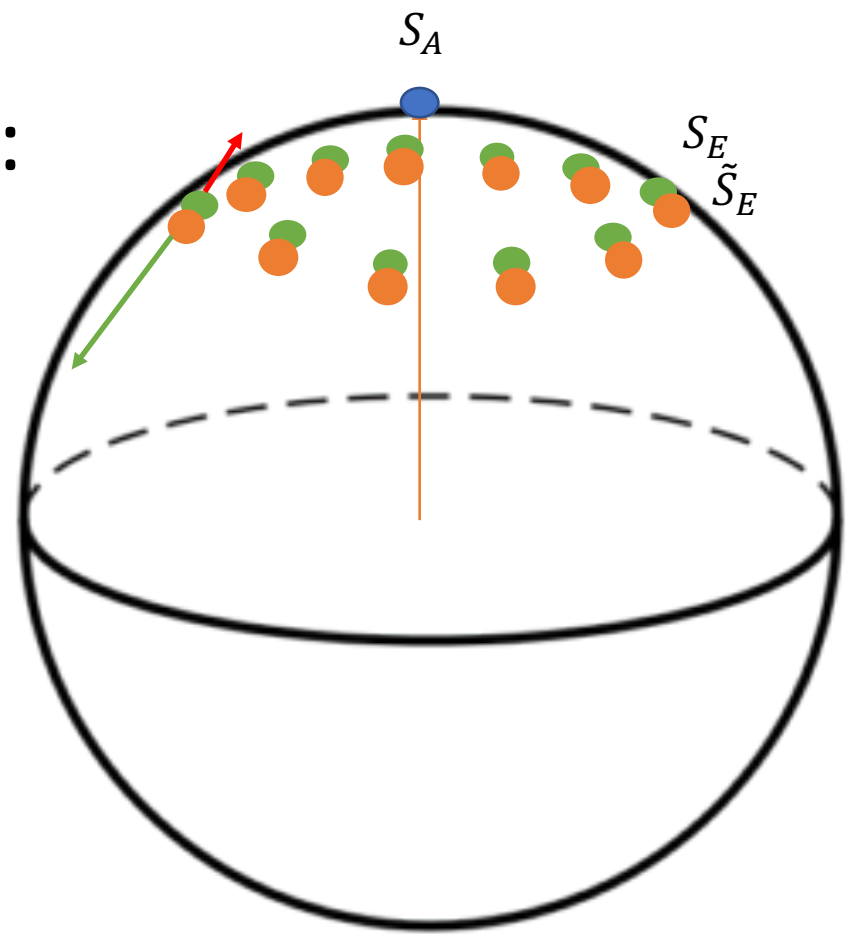
Orthogonality of symbols

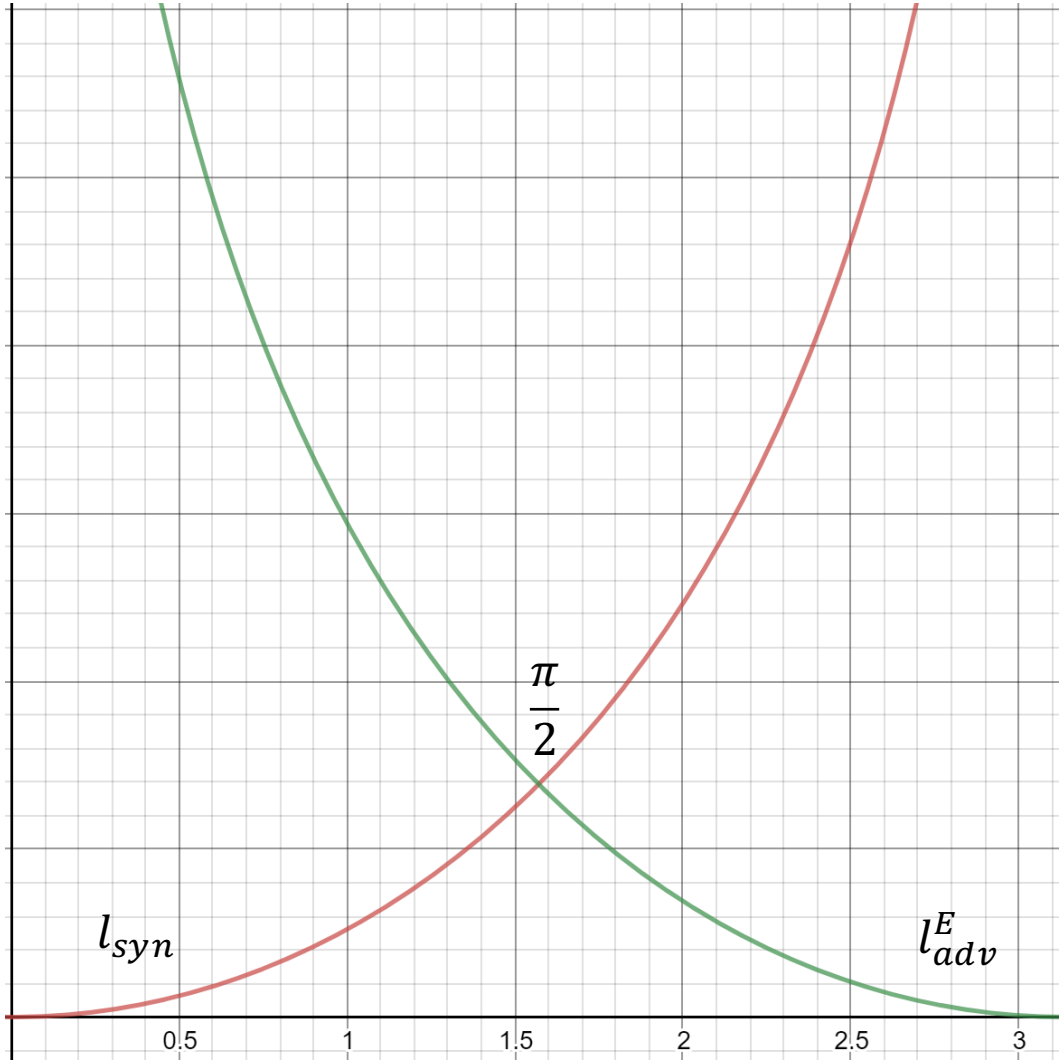
G finished:

$$\tilde{s}_E \approx s_E$$

E's turn:

Adversarial gradient stronger than syncretic gradient



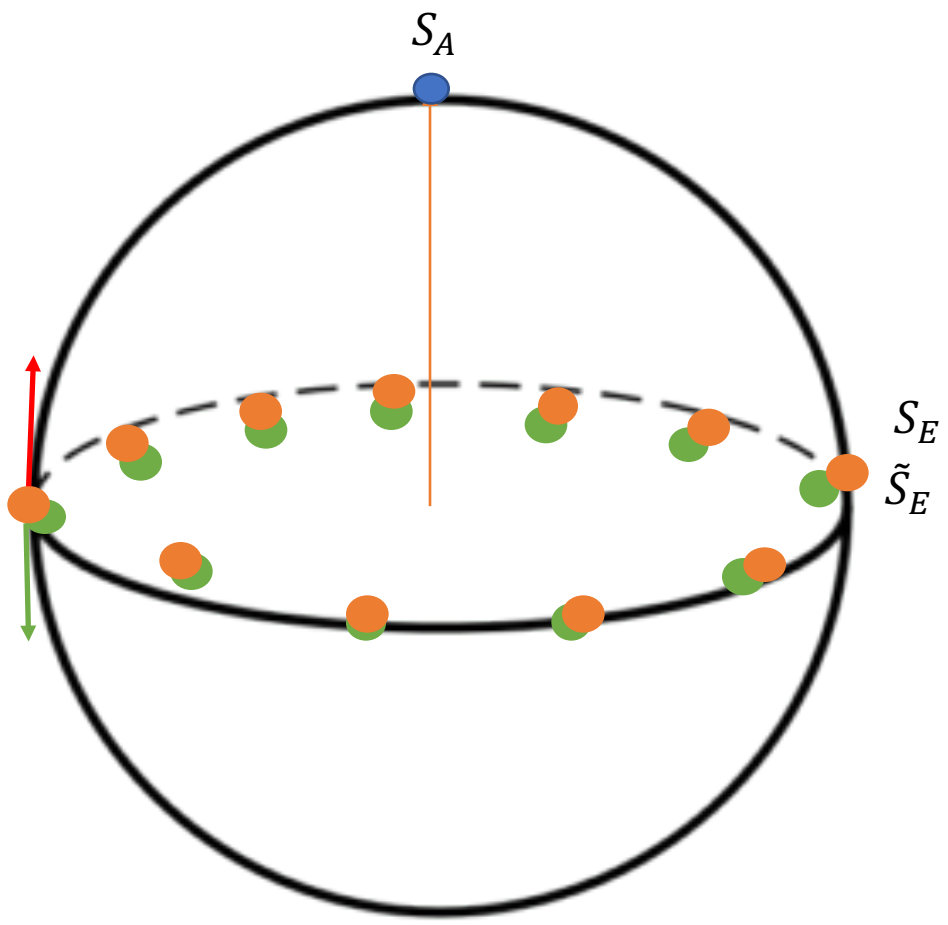


Oryan Barta © 2018 all rights reserved

Orthogonality of symbols

E's turn:

Equilibrium reached at $\frac{\pi}{2}$

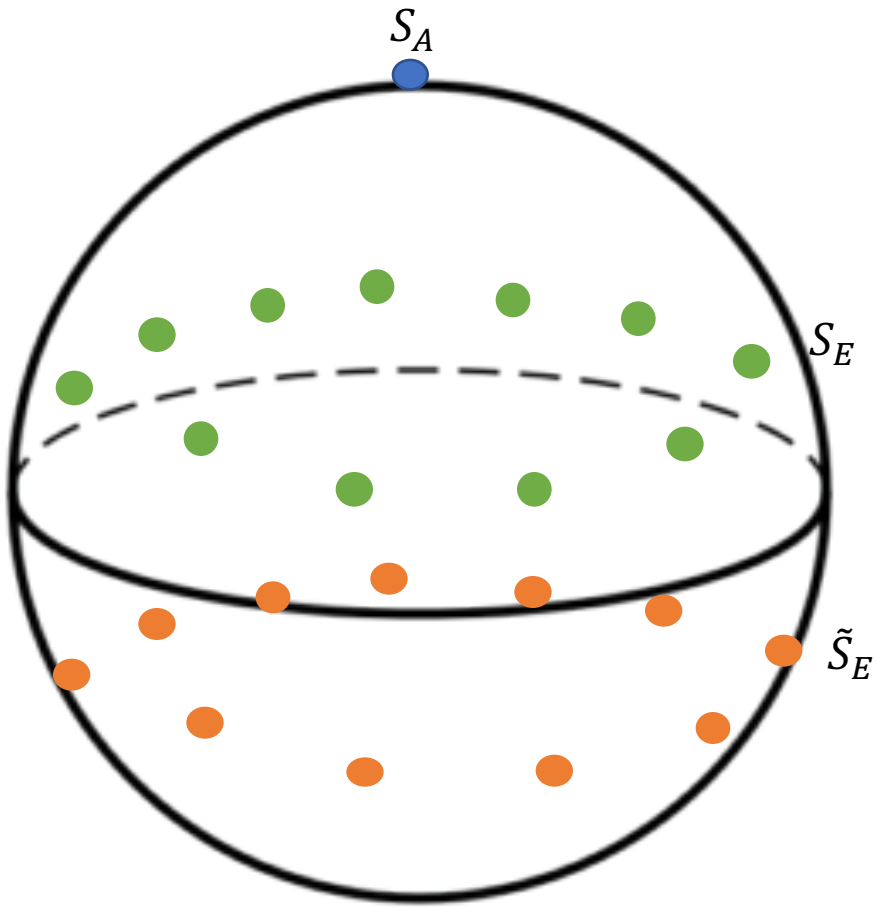


Oryan Barta © 2018 all rights reserved

Orthogonality of symbols

E's turn:

Most training is done around equilibrium point



Oryan Barta © 2018 all rights reserved

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

Oryan Barta © 2018 all rights reserved

Orthogonality: Unstable

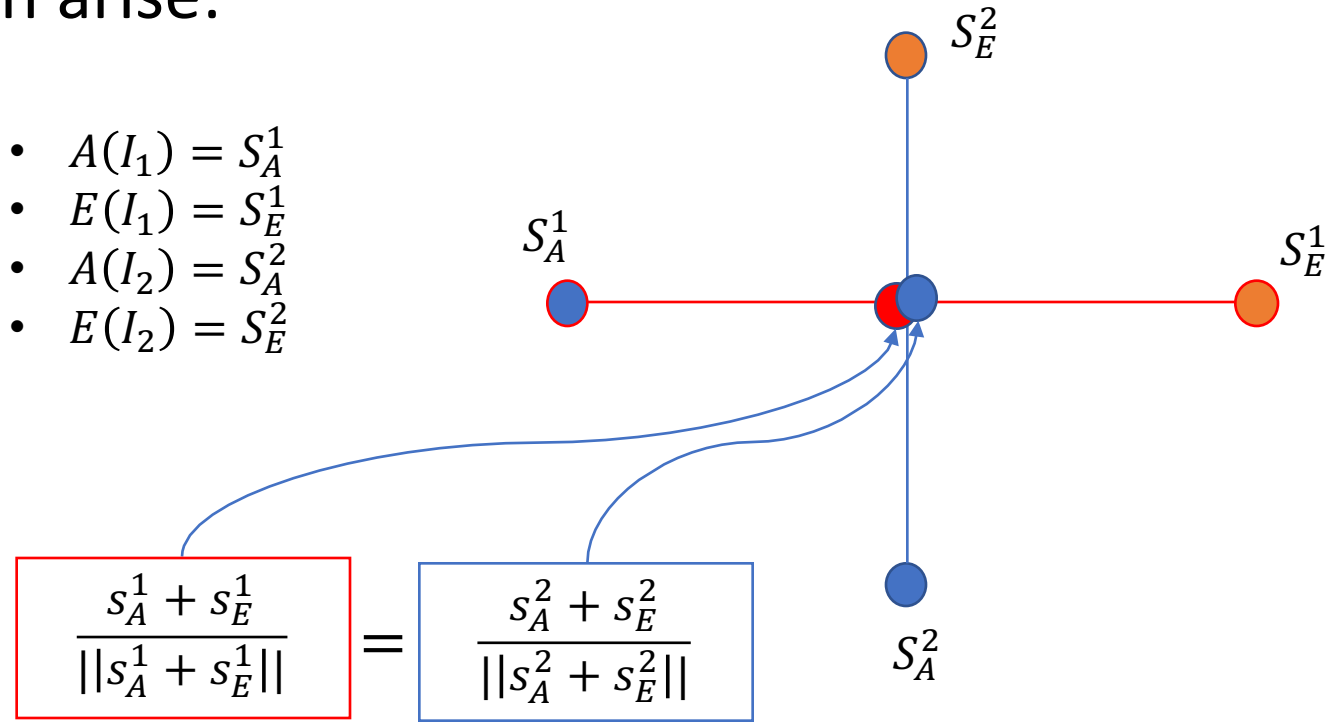
- In the example, A was constant. However as A is not constant there arise stability issues: Both units try to become orthogonal to the other.
- The orthogonality forming between the two units achieves the most stable form when one unit is the “central” unit and the other is the “peripheral” unit.
- 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.

Oryan Barta © 2018 all rights reserved

Orthogonality: Ambiguous symbols

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

  - $A(I_1) = S_A^1$
  - $E(I_1) = S_E^1$
  - $A(I_2) = S_A^2$
  - $E(I_2) = S_E^2$



Oryan Barta © 2018 all rights reserved

Solution: Centralization

- To stabilize symbol space we impose the central/peripheral structure:
  - $\min_{\mathrm{ij} \in batch} \mathrm{d}\left(s_A^i, s_A^j\right)$
  - $\max_{\mathrm{ij} \in batch} \mathrm{d}\left(s_E^i, s_E^j\right)$
- We call these the *self* losses of units A, E respectively.
- We also set as an objective, called the *kernel* loss:
  - $\min_{\mathrm{ij} \in batch} \mathrm{d}\left(s_A^i, s_E^j\right)$
- Note this is a *pairwise* computation, not elementwise as before.
- This causes the two units to maintain the same center of mass.

Oryan Barta © 2018 all rights reserved

Centralization: Central unit interpretation

- A is focused on finding *general* features of images
- Examples:
  - Spatial intelligence: understanding 3-dimenstionality 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

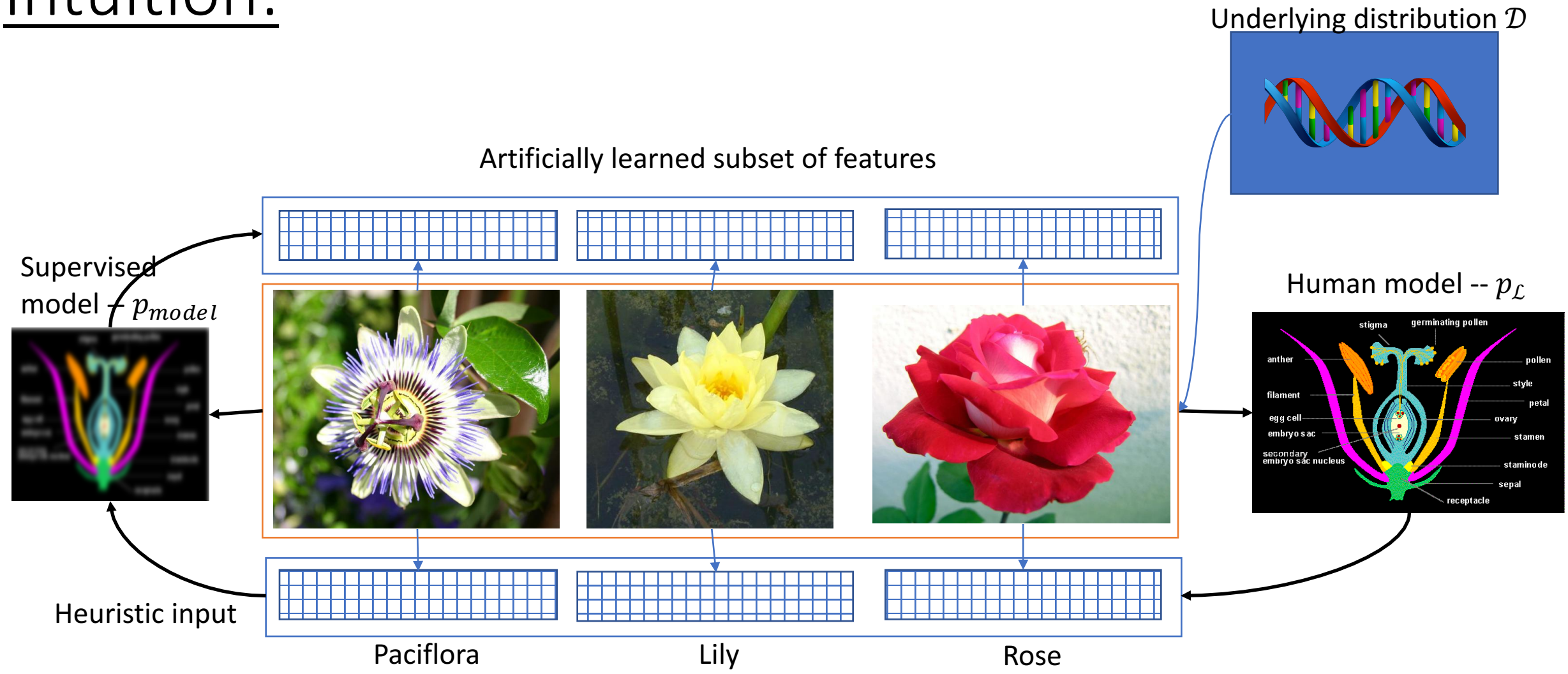
Oryan Barta © 2018 all rights reserved

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."

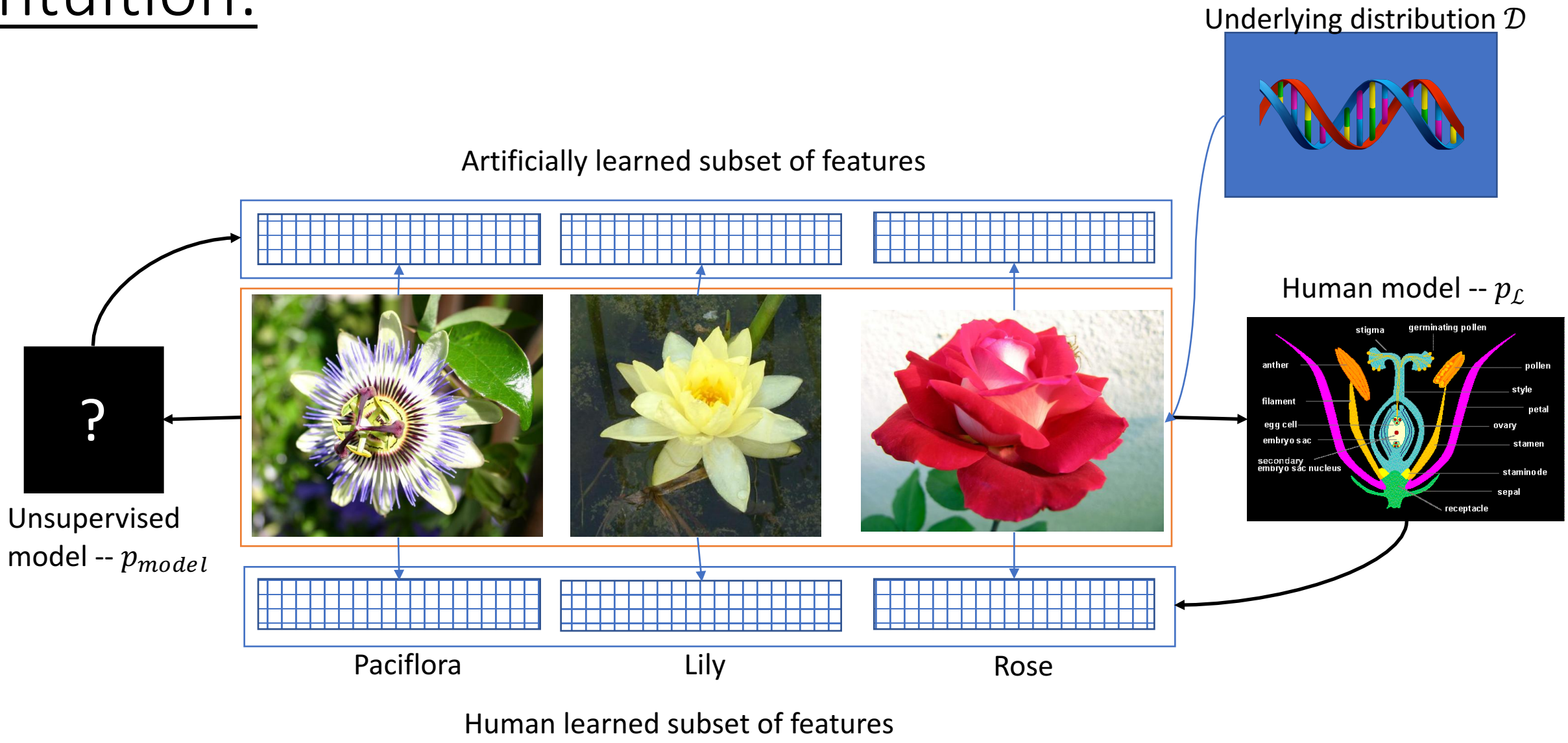
Oryan Barta © 2018 all rights reserved

Intuition:



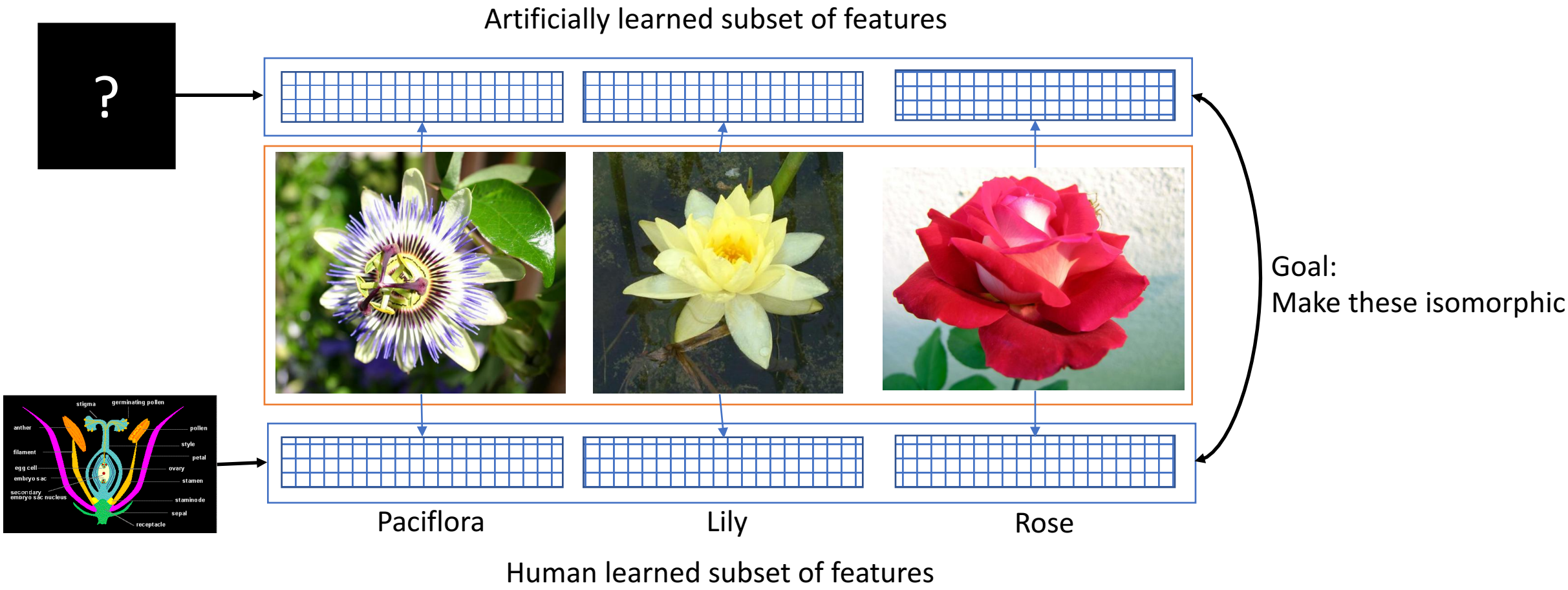
Oryan Barta © 2018 all rights reserved

Intuition:



Oryan Barta © 2018 all rights reserved

Intuition: “Implicit language”



Oryan Barta © 2018 all rights reserved

“Memories” vs “Perceptions”

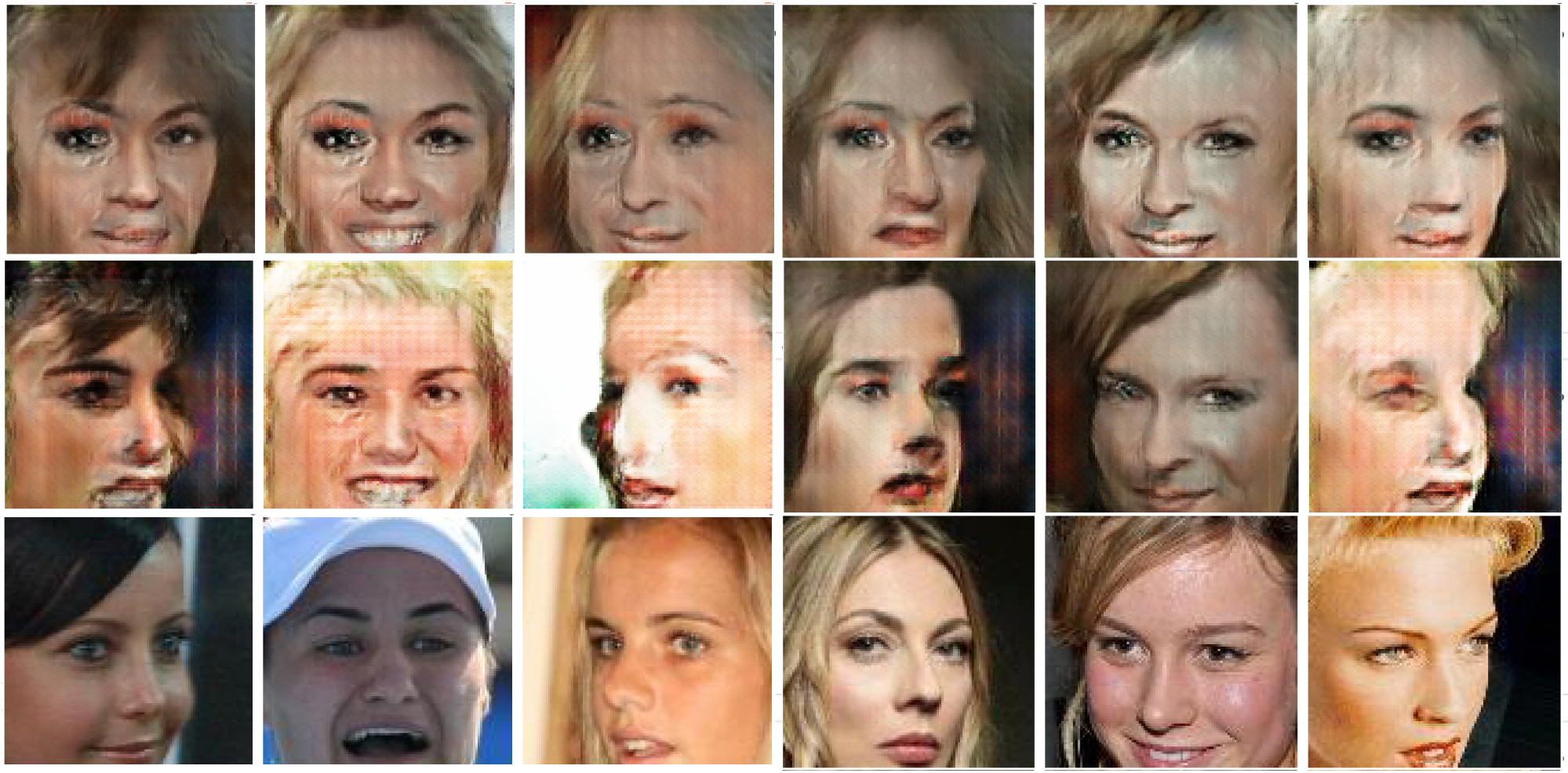
Memories:

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:



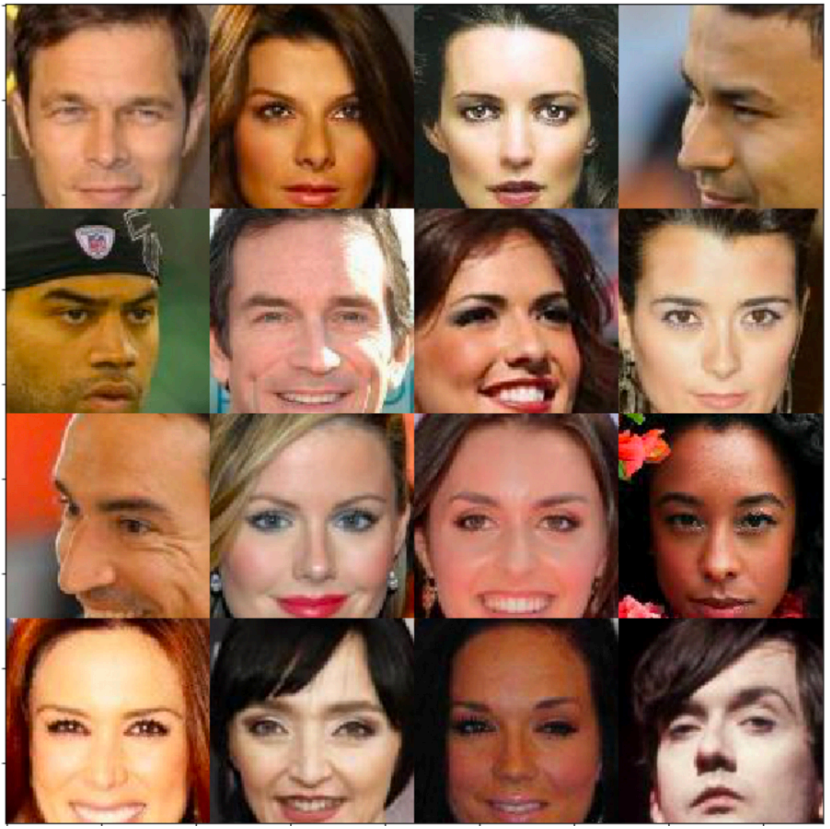
Oryan Barta © 2018 all rights reserved

Smooth mapping: Mnist

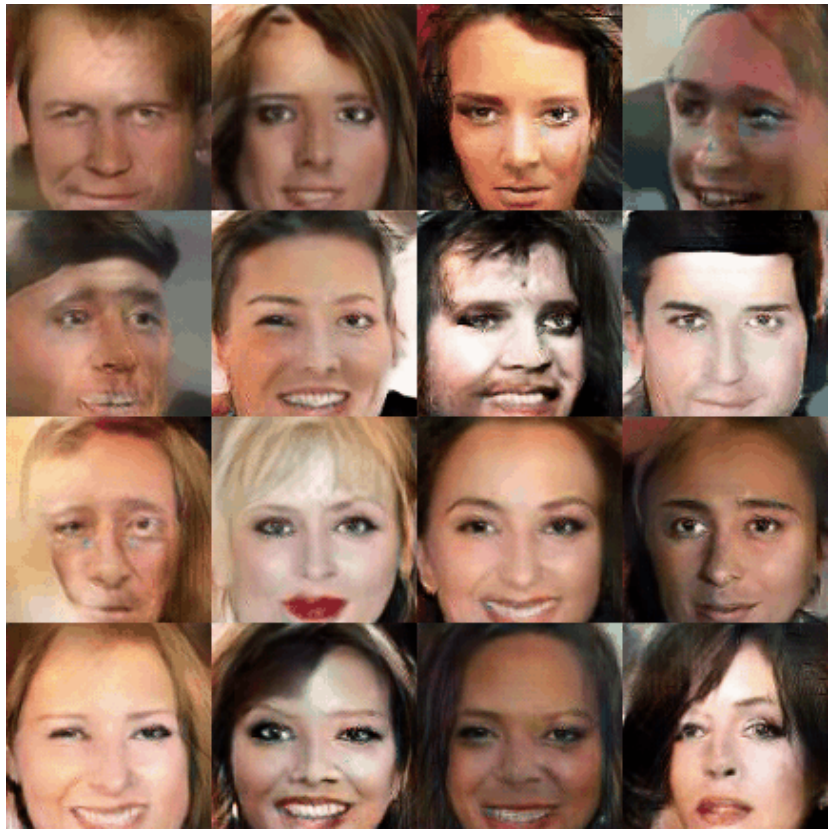
Oryan Barta © 2018 all rights reserved

Smooth mapping: CelebA

Original



Training 1

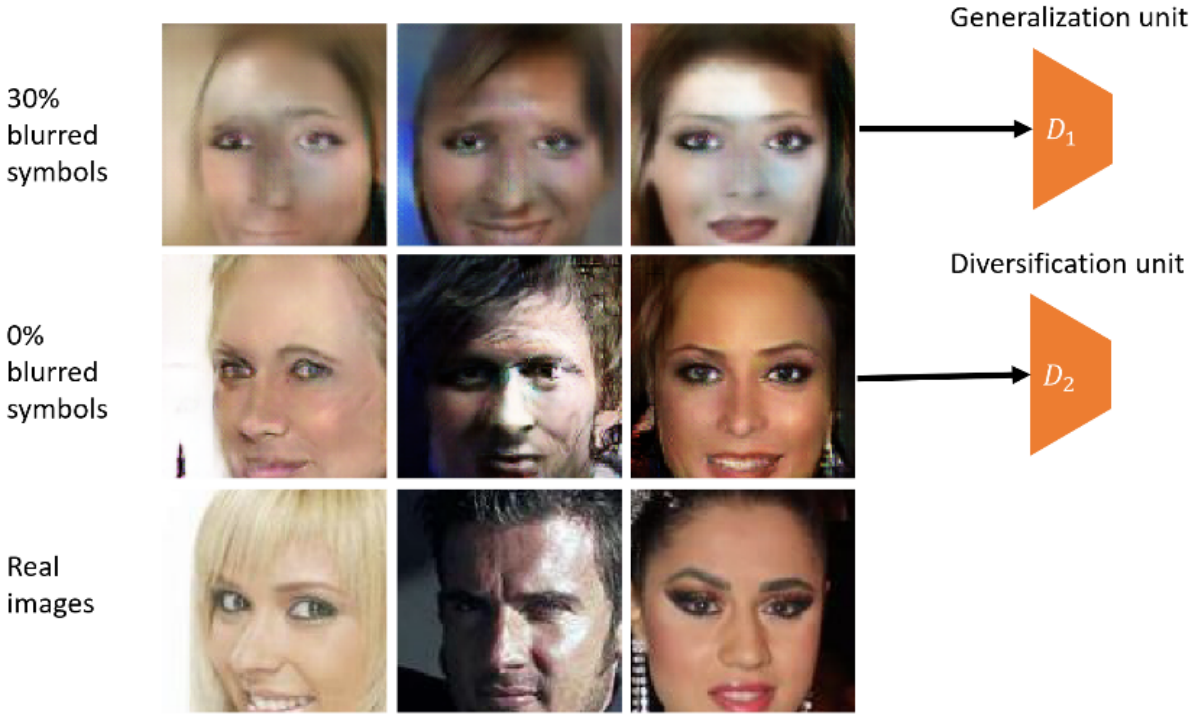


Training 2

Oryan Barta © 2018 all rights reserved

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.



Oryan Barta © 2018 all rights reserved

Oxford102 Flower dataset

Oryan Barta © 2018 all rights reserved

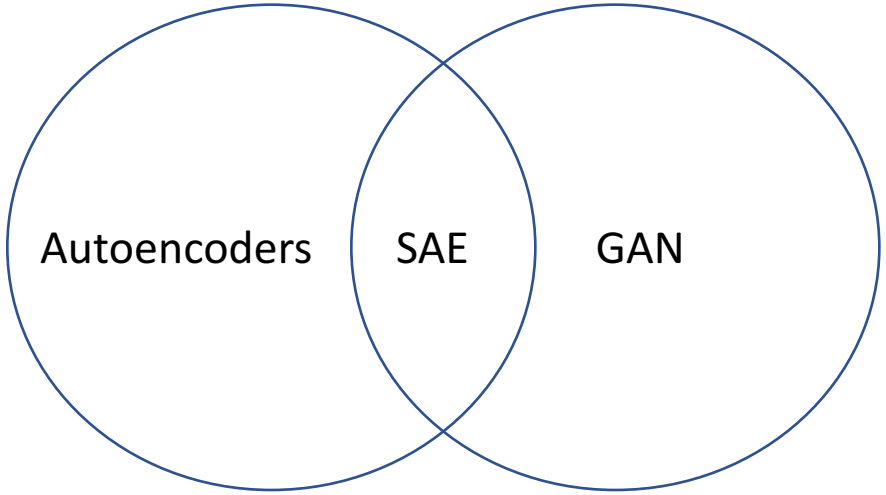
Existing Solutions in Detail

Autoencoders and VAEs

Oryan Barta © 2018 all rights reserved

Existing solutions: Unsupervised models

- Explicit models:
  - PixelRNN, PixelCNN
  - Compute $p_{model}(x) = \Pi_{i=1}^{n} p(x_i | x_1, \dots, x_{i-1})$
- Implicit (Generative) models:
  - Naïve Autoencoders
  - Variational Autoencoders
  - Generative Adversarial Networks
    - Cutting edge currently

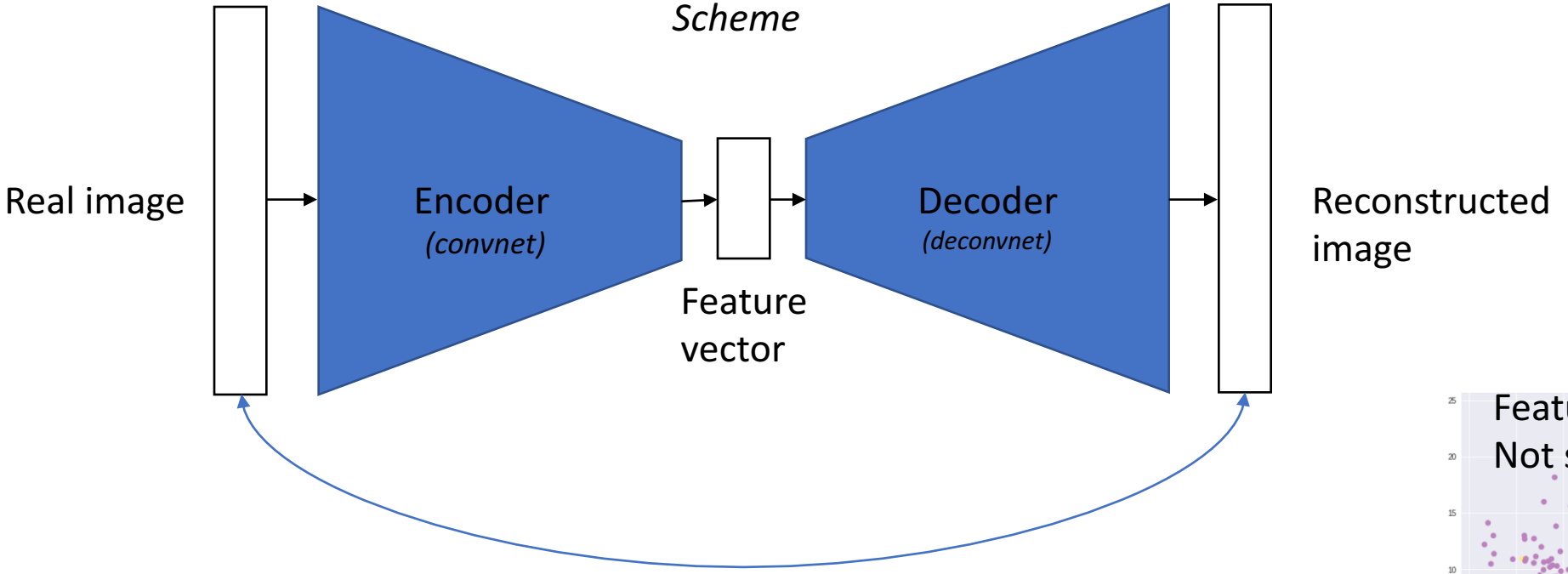


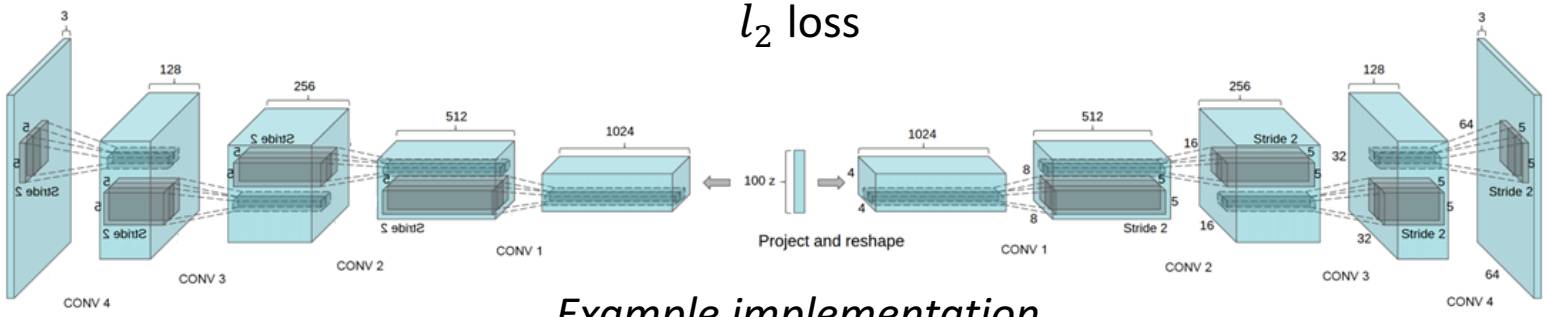
Oryan Barta © 2018 all rights reserved

Autoencoders

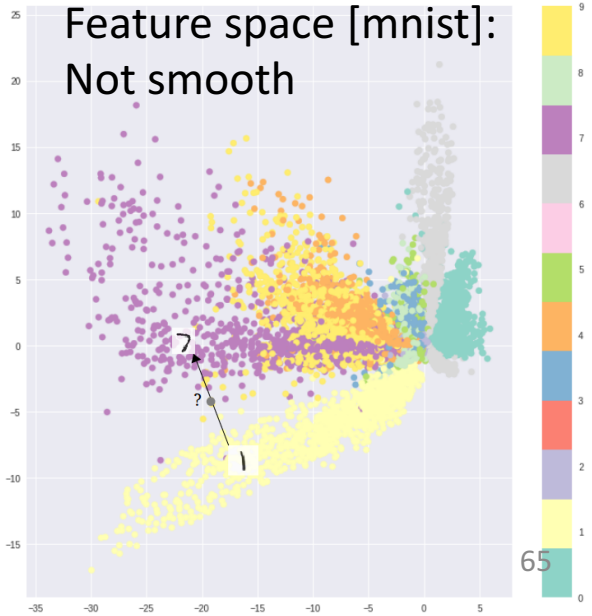
Deep "object level" features of data not effectively learned,
as l_2 loss between two images representing same object can be very large.

Scheme



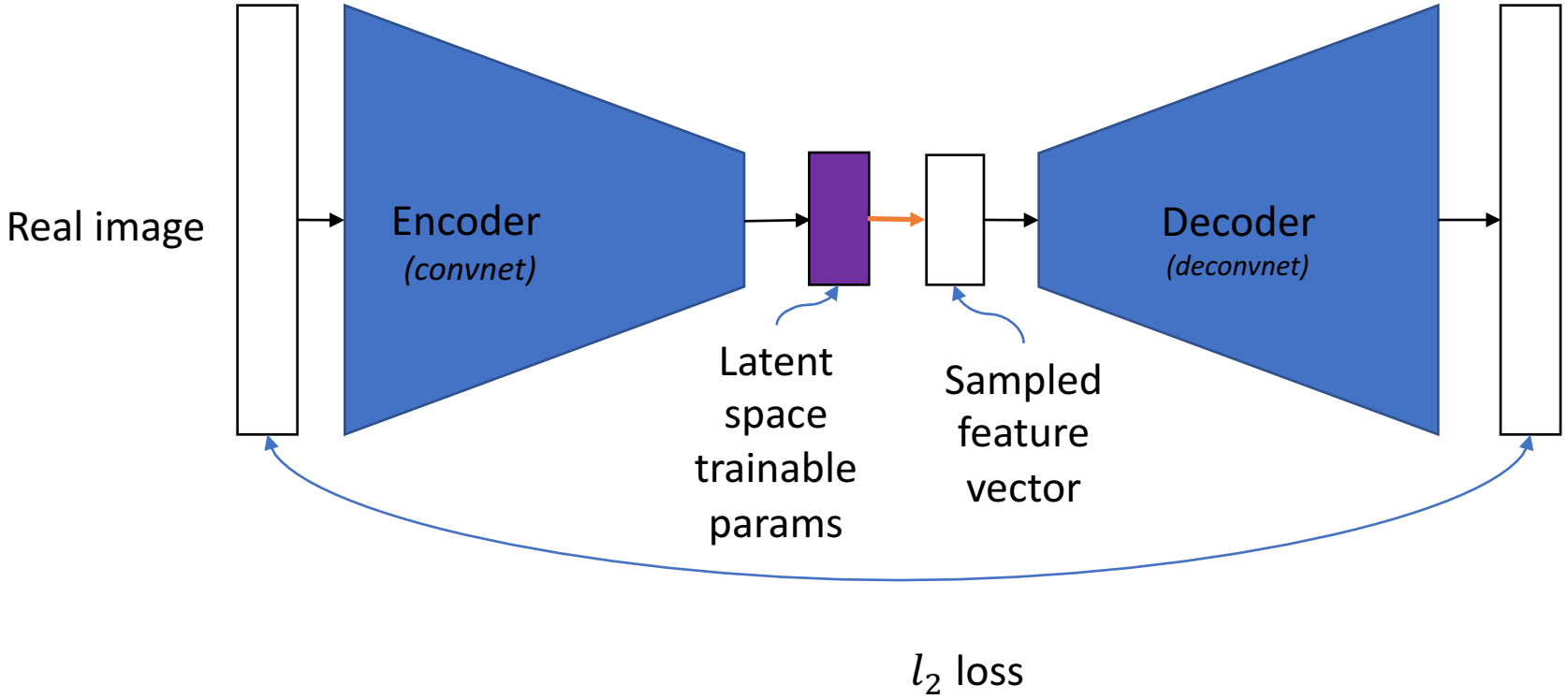


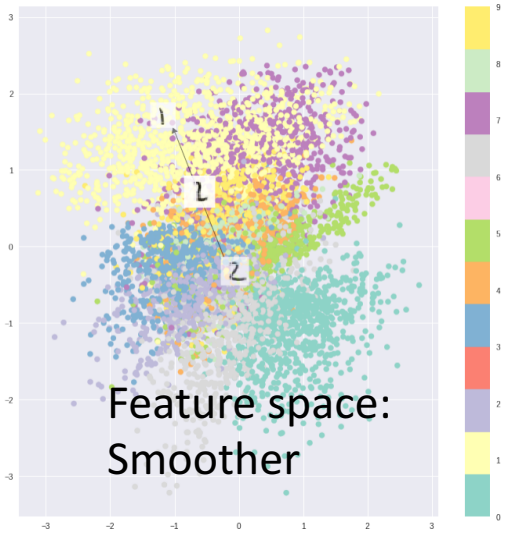
Example implementation



Oryan Barta © 2018 all rights reserved

Variational Autoencoders





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

Oryan Barta © 2018 all rights reserved