Google AI

Image understanding and image-to-image translation through the lens of information loss

A. Zhmoginov

April 2021

Google AI

Outline

Overview

Talk Overview

1. **MobileNetV2** and the role of invertibility in its design
2. **CycleGAN** and how invertibility can make models sensitive to perturbations
3. **Information Bottleneck** and how it can be used to find salient regions

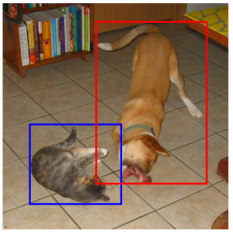
Group and Collaborators

Mark Sandler, Andrew Howard, Casey Chu, Ian Fischer, Menglong Zhu, Liang-Chieh Chen, Alec Go, Benoit Jacob, Bo Chen, Grace Chu, Dmitry Kalenichenko, Hartwig Adam, Marco Andreetto, Matthew Tang, Mingxing Tan, Quoc Le, Skirmantas Kligys, Tien-Ju Yang, Tobias Weyand, Vivienne Sze, Weijun Wang, Xiao Zhang

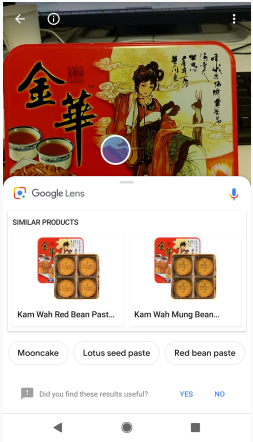
MobileNetV2

Applications

- Classification
- Recognition
- Object Detection



- Semantic Segmentation



History of the MobileNet Family

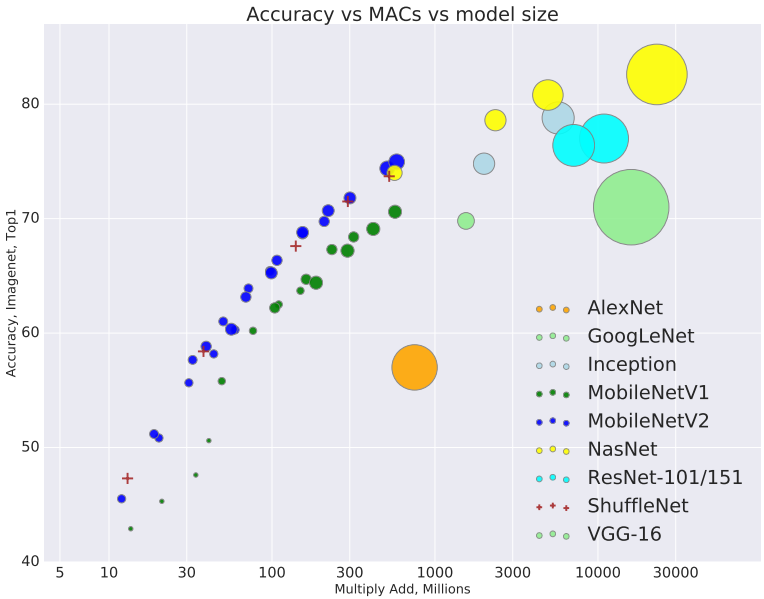
- **Mobilenet V1** – A. Howard, et al., *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications* (2017)
  - Separable convolutions
  - Depth and resolution multipliers
- **Mobilenet V2** – M. Sandler, et al., *MobileNetV2: Inverted Residuals and Linear Bottlenecks* (2018)
  - Inverted residuals and linear bottlenecks
- **Mobilenet V3** – A. Howard, et al., *Searching for MobileNetV3* (2019)
  - Architecture search on direct device latencies
  - Using better non-linearities
  - Using "Squeeze and Excite" blocks [J. Hu, et al., *Squeeze-and-Excitation Networks* (2017)]
  - Mixing 3×3 and 5×5 kernels

History of the MobileNet Family

- **Mobilenet V1** – A. Howard, et al., *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications* (2017)
  - Separable convolutions
  - Depth and resolution multipliers
- **Mobilenet V2** – M. Sandler, et al., *MobileNetV2: Inverted Residuals and Linear Bottlenecks* (2018)
  - Inverted residuals and linear bottlenecks
- **Mobilenet V3** – A. Howard, et al., *Searching for MobileNetV3* (2019)
  - Architecture search on direct device latencies
  - Using better non-linearities
  - Using "Squeeze and Excite" blocks [J. Hu, et al., *Squeeze-and-Excitation Networks* (2017)]
  - Mixing 3×3 and 5×5 kernels

Evaluation Metrics

- How to compare different models?
- Number of operations is not the best metric



- **Width Multiplier**
  - Scaling the number of channels by the same factor
  - αn_1 input channels and αn_2 output channels $\Rightarrow$ Complexity $\sim \alpha^2$
- **Resolution Multiplier**
  - Scale resolution
  - $(\beta W) \times (\beta H)$ image $\Rightarrow$ Complexity $\sim \beta^2$
- **Quantization**
- **Separable convolutions**

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

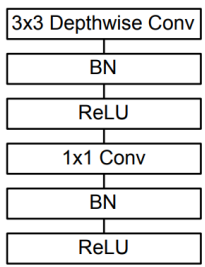
MobileNetV1: Separable Convolution

- Convolution:

$$o(x, y, k) = \sum_{\Delta_x, \Delta_y, m} w(\Delta_x, \Delta_y, k, m)\ i(x - \Delta_x, y - \Delta_y, m)$$

- Separable kernel $(X \times Y \times K \times M \to X \times Y \times M + K \times M)$:

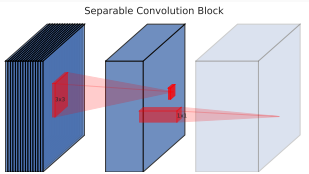
$$w(\Delta_x, \Delta_y, k, m) = w^{\mathrm{PW}}(k, m)\, w^{\mathrm{DW}}(\Delta_x, \Delta_y, m)$$



- Add intermediate normalization and activation σ

$$o(x, y, k) = \sum_{m} w^{\mathrm{PW}}(k, m)\ \sigma \left[\sum_{\Delta_x, \Delta_y} w^{\mathrm{DW}}(\Delta_x, \Delta_y, m)\ i(x - \Delta_x, y - \Delta_y, m) \right]$$

- $WHXYKM$ operations $\to WHXYM + WHKM = WHXYKM[K^{-1} + (XY)^{-1}]$
- Computational cost $\sim 1/8$ of full, top-1 IMAGENET accuracy $71.7\% \to 70.6\%$



- Built using lessons from **MobileNetV1**
- **Same overall structure**: sequence of similar blocks
- Main component: **inverted residual with linear bottleneck**

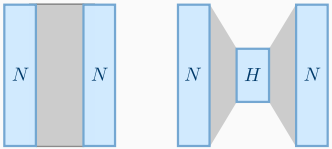
Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

- Fully-connected layer: weights w – $N \times N$ matrix
- **Linear bottleneck**:

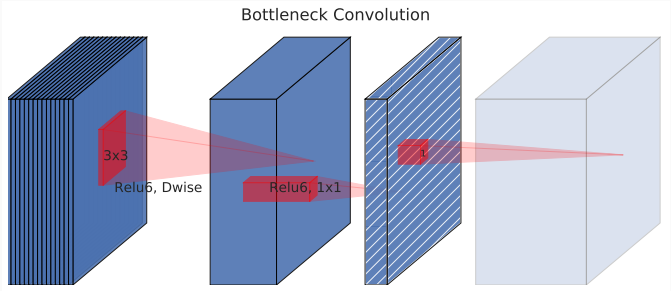
$$w(k,m) = \sum_{h=1}^{H} w^{\mathrm{A}}(k,h) w^{\mathrm{B}}(h,m)$$

with $H < N$

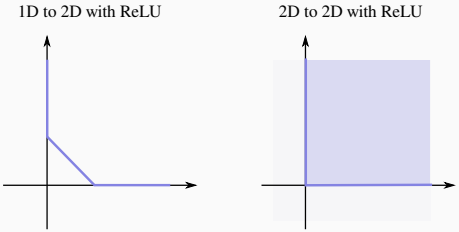
  - Number of parameters and operations: $N^2 \to 2NH$
  - Bottleneck puts a limit on processed information

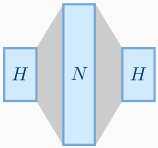


- Combining with **Separable Convolution**:

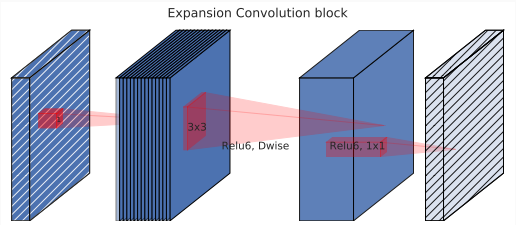


- Different perspective:
  1. **bottleneck** limits information flow
  2. **expansion** followed by **nonlinearity** for complex computation
- Reducing bottleneck size → nonlinearity can cause irreversible information loss



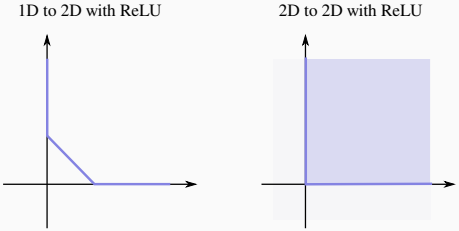


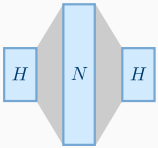
- Combining with **Separable Convolution**:



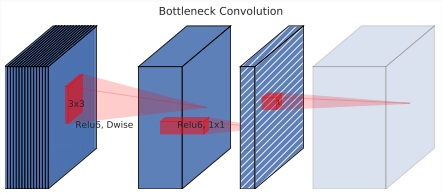
MobileNetV2: Information Flow Interpretation – Expansion Layer

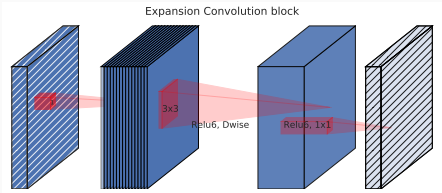
- Different perspective:
  1. **bottleneck** limits information flow
  2. **expansion** followed by **nonlinearity** for complex computation
- Reducing bottleneck size → nonlinearity can cause irreversible information loss



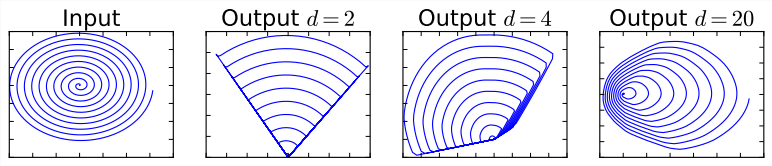


- Combining with **Separable Convolution**:





- Example of the manifold transformation:



- Which inputs "collapse" for a given weight matrix W?
  - Are there solutions for x different from $x_0 \in \mathbb{R}^n$?

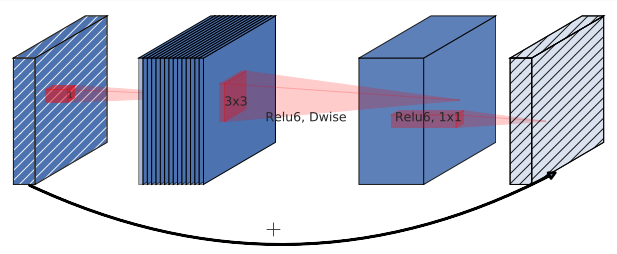
$$y_0 = \mathrm{ReLU}(Wx_0) = \mathrm{ReLU}(Wx)$$

  - **At initialization** each input is not "collapsed" with probability

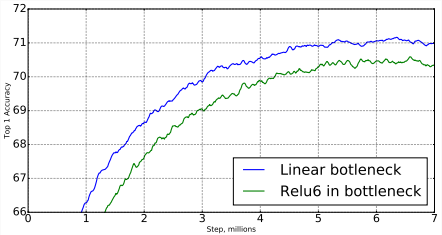
$$\frac{1}{2^m} \sum_{k=0}^{m-n} \begin{pmatrix} m \\ k \end{pmatrix}$$

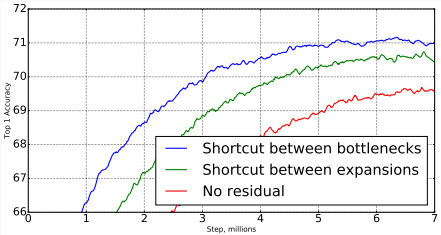
    - For $m \gg n$ bounded by $1 - 2^{-m/2}$

- **Final block design**: adding residual connection:



- Design decisions driven by empirical results:





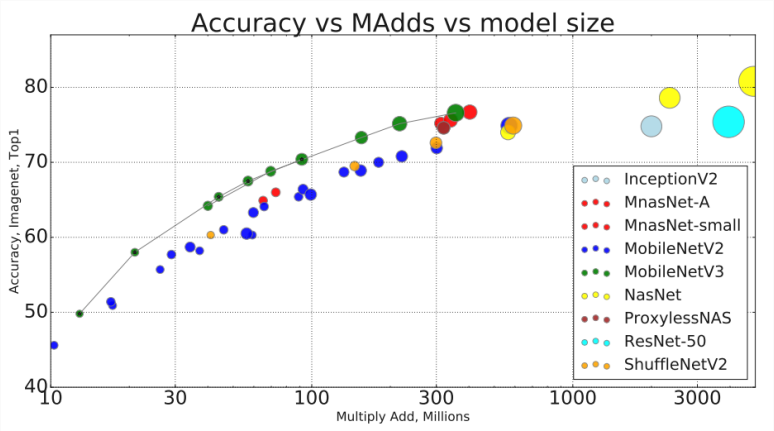
Architectures

- NasNet – B. Zoph, et al., *Learning Transferable Architectures for Scalable Image Recognition* (2017)
- ShuffleNetV2 – N. Ma, et al., *ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design* (2018)
  - Latency better than MobileNetV2 despite having comparable # of operations
- NetAdapt – T.-J. Yang, et al., *NetAdapt: Platform-Aware Neural Network Adaptation for Mobile Applications* (2018)
- MnasNet – M. Tan., et al., *MnasNet: Platform-Aware Neural Architecture Search for Mobile* (2018)
  - About 40% faster for the same accuracy as MobileNetV2
- MobileNetV3 – A. Howard, et al., *Searching for MobileNetV3* (2019)

- Image Transformers – A. Dosovitsky, et al., *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale* (2020)
  - H. Touvron, et al., *Training data-efficient image transformers & distillation through attention* (2020)

Architectures

- NasNet – B. Zoph, et al., 2017
- ShuffleNetV2 – N. Ma, et al., 2018
- NetAdapt – T.-J. Yang, et al., 2018
- MNasNet – M. Tan., et al., 2018
- MobileNetV3 – A. Howard, et al., 2019

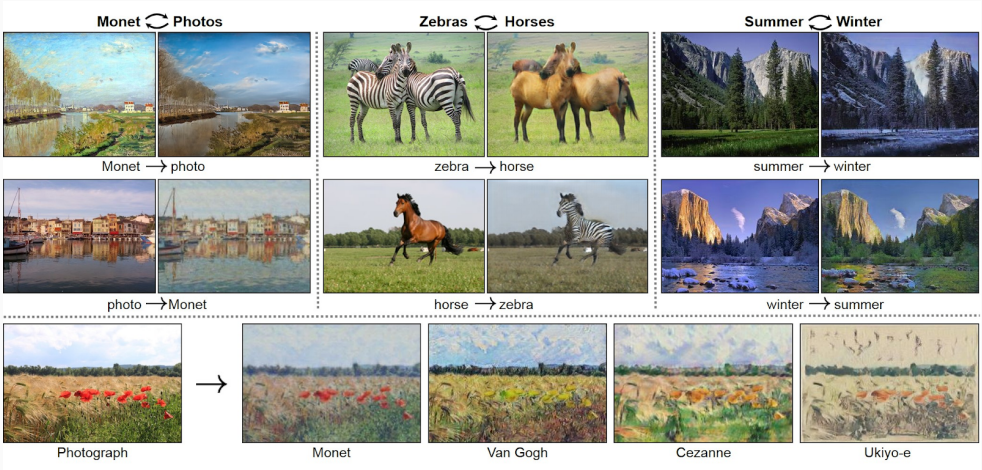


Takeaways

- MobileNetV2 architecture is similar to MobileNetV1 – sequence of blocks
- Blocks are inverted residuals with linear bottlenecks
- Design intuition based on information flow
- Design has to take real latencies into account
- Modern architectures are freq. designed using NAS

CycleGAN and Information Loss

- **Reversible Generative Models**: RealNVP, NICE, Glow, ...
- **CycleGAN** is a model for unpaired image-to-image translation
- J.-Y. Zhu, T. Park, P. Isola, A. A. Efros, *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*, **ICCV 2017**
- Given samples from $p(a)$ and $p(b)$ learn $f_{a\to b}$, $f_{b\to a}$ s.t. $p(f_{a\to b}(a)) \sim p(b)$ and $p(f_{b\to a}(b)) \sim p(a)$



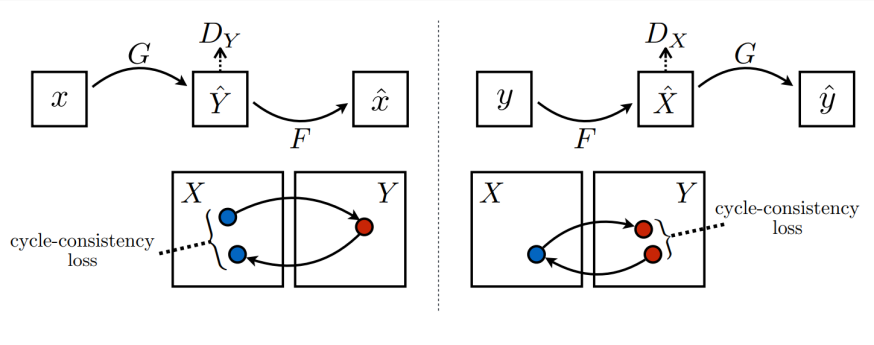
- The model consists of two **GANs** on both domains and the **cycle consistency loss**
- The model attempts to find maps $F : Y \to X$ and $G : X \to Y$ such that:
  - **Discriminator component**: $G(x)$ and $F(y)$ are realistic for any real samples $x \in X$ and $y \in Y$

$$\mathcal{L}_{\text{GAN}}(G, D_Y) = \mathbb{E}_{y \sim p_{\text{data}}(y)} \log D_Y(y) + \mathbb{E}_{x \sim p_{\text{data}}(x)} \log[1 - D_Y(G(x))]$$

$$\mathcal{L}_{\text{GAN}}(F, D_X) = \mathbb{E}_{x \sim p_{\text{data}}(x)} \log D_X(x) + \mathbb{E}_{y \sim p_{\text{data}}(y)} \log[1 - D_X(F(y))]$$

  - **Cyclic consistency**: $F \circ G$ and $G \circ F$ are identity operators on X and Y correspondingly

$$\mathcal{L}_{\text{cyc}}(F, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)} \|F(G(x)) - x\|_1 + \mathbb{E}_{y \sim p_{\text{data}}(y)} \|G(F(y)) - y\|_1$$

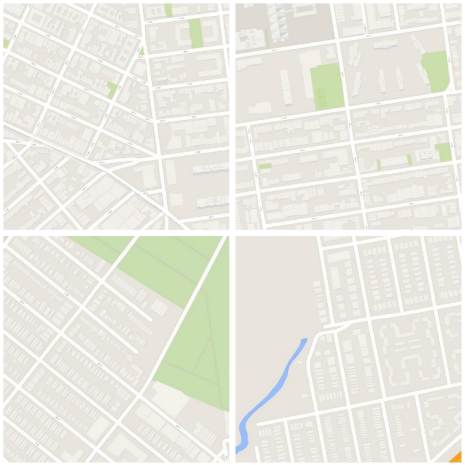


Preserving Information

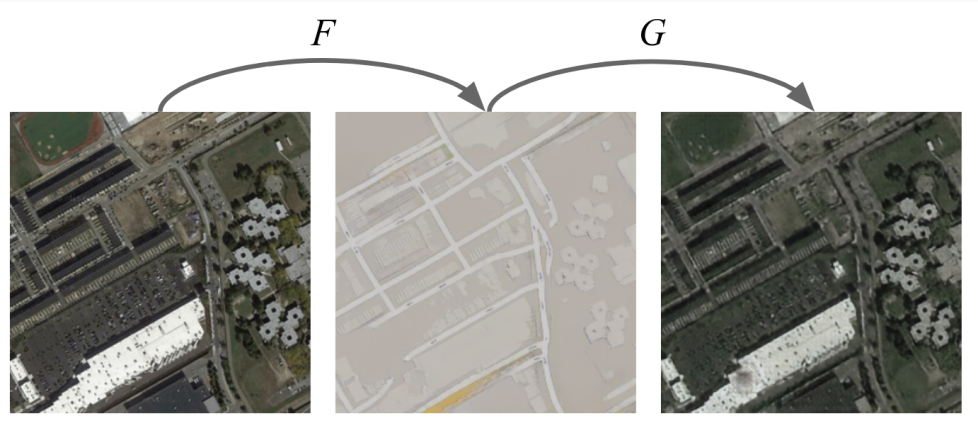
- Cyclic consistency has to **preserve information**, but what happens when we have **high- and low-entropy image domains**?
  - **Example**: Google Maps images
  - C. Chu, A. Zhmoginov, M. Sandler, *CycleGAN, a Master of Steganography* (2017)

Domain A: **aerial imagery**

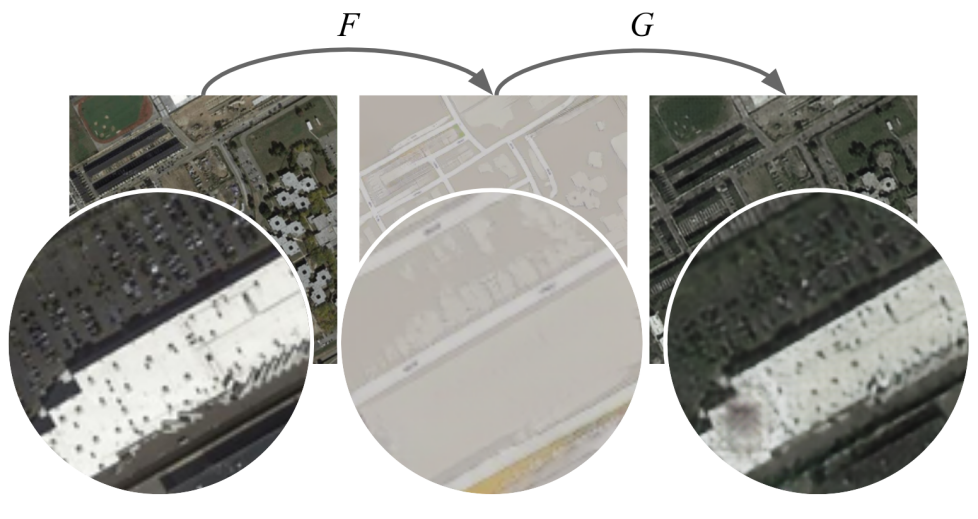
Domain B: **maps**



- **CycleGAN** works very well, but **perhaps too well?**

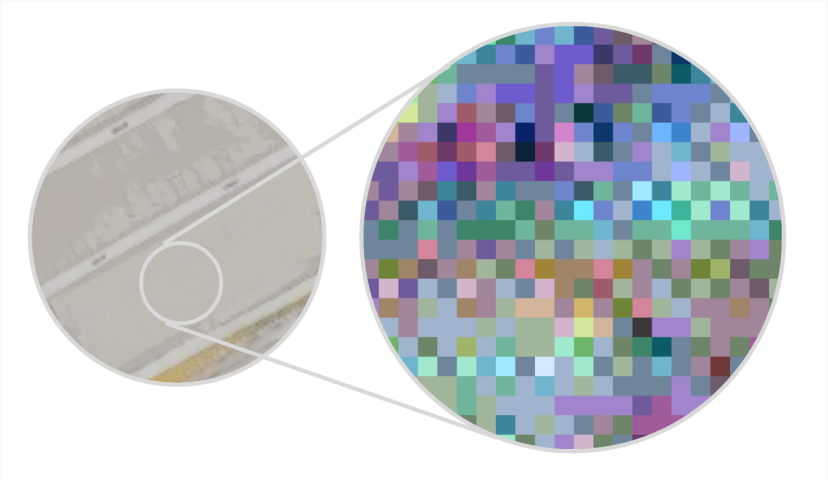


- **CycleGAN** works very well, but **perhaps too well?**

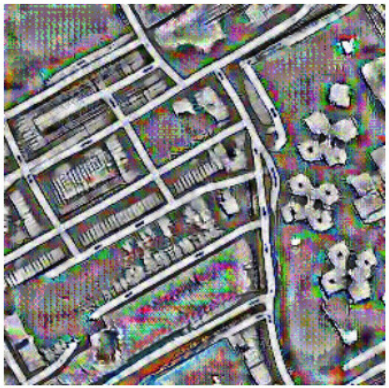


Preserving Information

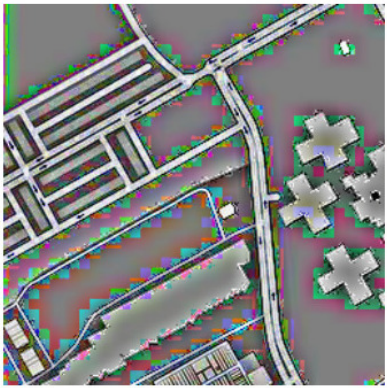
- It turns out that the information necessary for reconstruction is **encoded in the high-frequency noise**



- It turns out that the information necessary for reconstruction is **encoded in the high-frequency noise**

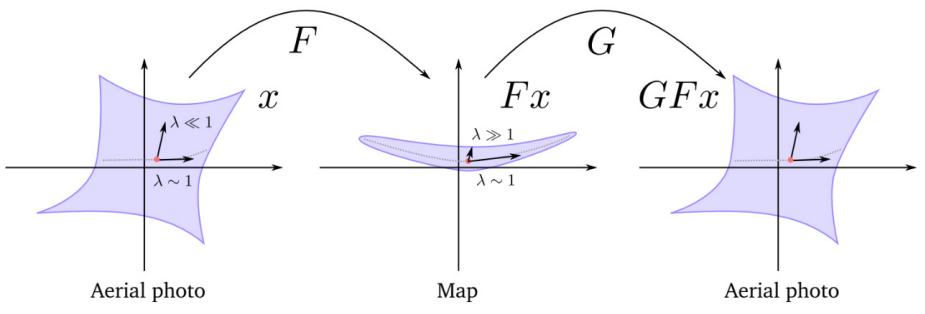


(a) Generated map.



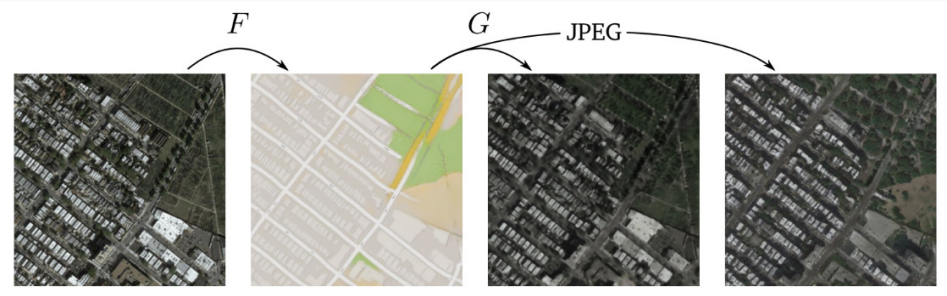
(b) Training map, for comparison.

- In the case when one image space has more entropy than the other (*photos vs maps*), one transformation **contracts** degrees of freedom ($\lambda \ll 1$), while another one **expands** them ($\lambda \gg 1$)



- Contraction rate is expected to be **forced down** by the discriminator

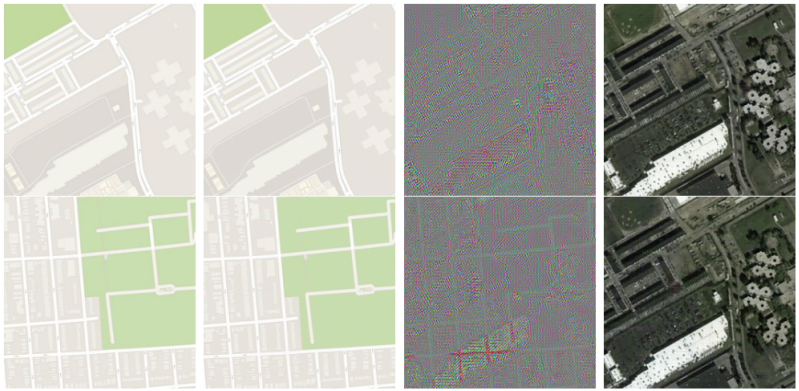
- One of two transformations is very sensitive to small image perturbations – **is this so bad?**
- Image perturbations (like those caused by quantization or JPEG compression) destroy reconstructed detail:



- Training with noise forces the model to hide information in the higher-amplitude signal

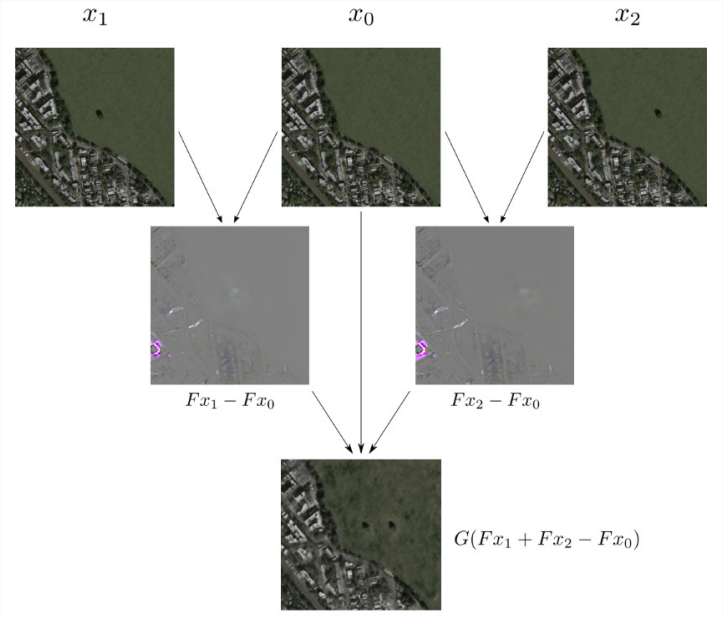
- The fact that CycleGAN encodes details in low-amplitude noise can be exploited by finding **adversarial examples**

$$y^*(x) = \underset{y}{\operatorname{argmin}} \|G(y) - x\|_1, \quad y \approx y_0$$



Implications of Sensitivity

- G is **very sensitive** in a vicinity of the data manifold
- G is **almost linear** in a vicinity of the data manifold



Takeaways

- If image domains have very different "complexity" $\rightarrow$ cyclic consistency may lead to extreme sensitivity of learned maps
- One-to-one mapping – manifolds should be similar
- Large eigenvalues of J – susceptible to adversarial attacks
- A. Almahairi, et al., *Augmented CycleGAN: Learning Many-to-Many Mappings from Unpaired Data* (2018).

Information Bottleneck and Salient Regions

- Supervised task: $p(x, y)$
- Deep neural network mapping input source X to a stochastic encoding Z via $p(z|x;\theta)$
- N. Tishby, et al., *The information bottleneck method* (1999)
- Information bottleneck objective:

$$\max_{\theta} \mathbb{I}(Z, Y; \theta) \quad \text{s.t.} \quad \mathbb{I}(X, Z; \theta) \leq I_c$$

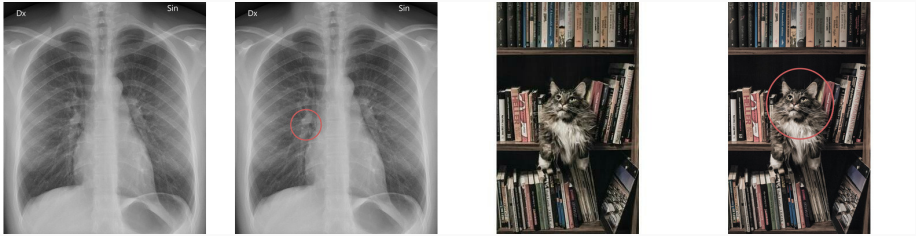
or introducing a Lagrange multiplier β:

$$\boxed{\max_{\theta} \left[\mathbb{I}(Z, Y; \theta) - \beta \mathbb{I}(X, Z; \theta)\right]}$$

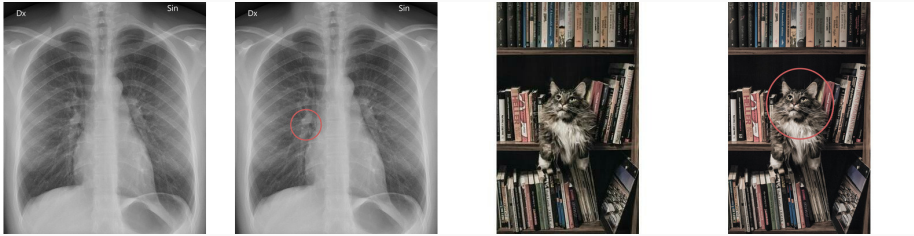
- **Variational Information Bottleneck**
  - A. Alemi, et al., *Deep Variational Information Bottleneck* (2017)

Saliency Maps

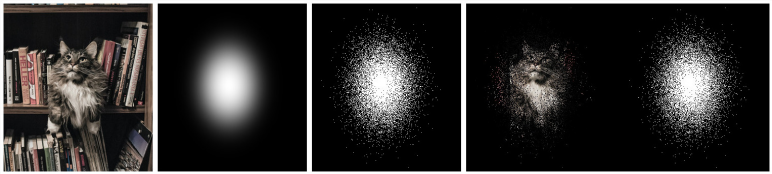
- *Information-Bottleneck Approach to Salient Region Discovery* (2019)
- **Input**: dataset of labeled samples $\mathcal{D} = \{(x_i, y_i)|i = 1, \ldots, N\}$
- Learn to identify **salient regions** in each sample



- *Information-Bottleneck Approach to Salient Region Discovery* (2019)
- **Input**: dataset of labeled samples $\mathcal{D} = \{(x_i, y_i)|i = 1, \ldots, N\}$
- Learn to identify **salient regions** in each sample



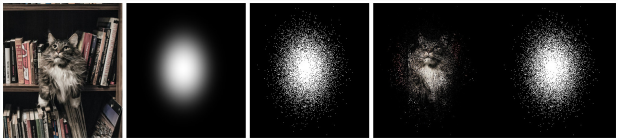
- **Masking model** $\rho(i; \zeta)$ parametrized by ζ



$i_{x,y}$ $\rho_{x,y}$ $m_{x,y}$ $(i \odot m)_{x,y}$

I M $I \odot M$

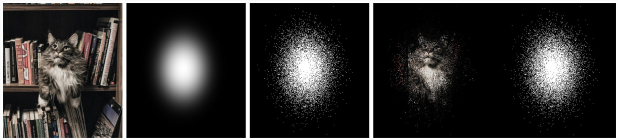
- **Masking model** $\rho(i;\zeta)$ parametrized by ζ



$i_{x,y}$	$\rho_{x,y}$	$m_{x,y}$	$(i \odot m)_{x,y}$
I		M	$I \odot M$

- **What properties should the salient region** $I \odot M$ **have?**:
  - $I \odot M$ contains enough information to infer the image label C
  - M is sufficiently small: minimizing information entropy of $I \odot M$

- **Masking model** $\rho(i;\zeta)$ parametrized by ζ



$i_{x,y}$ $\rho_{x,y}$ $m_{x,y}$ $(i \odot m)_{x,y}$

I M $I \odot M$

- **What properties should the salient region** $I \odot M$ **have?**:
  - $I \odot M$ contains enough information to infer the image label C
  - M is sufficiently small: minimizing information entropy of $I \odot M$
- **Information-Bottleneck objective**:
  - Naïvely, masking model should optimize:

$$\underset{\zeta}{\operatorname{argmax}} \left[\mathbb{I}(I \odot M; C) - \beta \mathbb{I}(I; I \odot M)\right]$$

  - Masking model learns to encode image label in the mask itself $\rightarrow$ need to adjust the optimization objective

- Consider a definition based on the **Information-Bottleneck objective**

$$\underset{\zeta}{\operatorname{argmax}} \left[\mathbb{I}(I \odot M; C) - \beta \mathbb{I}(I; I \odot M) \right] \tag{1}$$

- Consider a definition based on the **Information-Bottleneck objective**

$$\underset{\zeta}{\operatorname{argmax}} \left[\mathbb{I}(I \odot M; C) - \beta \mathbb{I}(I; I \odot M) \right] \tag{1}$$

- **Expected salient regions** for a simple example of **2-digit classification**

- Consider a definition based on the **Information-Bottleneck objective**

$$\underset{\zeta}{\operatorname{argmax}} \left[\mathbb{I}(I \odot M; C) - \beta \mathbb{I}(I; I \odot M) \right] \tag{1}$$

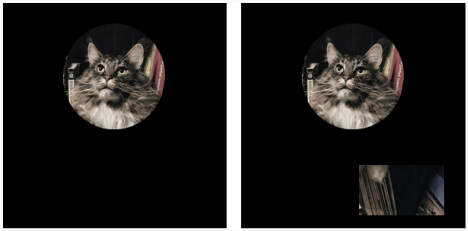
- **Expected salient regions** for a simple example of **2-digit classification**

- **Actual solution optimizing Eq.** (1):

- **Why does this happen?**
  - Image class is **encoded in the mask itself**
  - Possible because ζ is fixed in $I \xrightarrow{\zeta} I \odot M \to C$
  - Calculating $\mathbb{I}(I \odot M; C)$ we should ideally treat ζ as a random variable

Another simple solution

- $M \to$ revealing more pixels $\to M'$
- Should be even easier to predict the label: $\mathbb{I}(I \odot M'; C) \geq \mathbb{I}(I \odot M; C)$



- **Why does this happen?**
  - Image class is **encoded in the mask itself**
  - Possible because ζ is fixed in $I \xrightarrow{\zeta} I \odot M \to C$
  - Calculating $\mathbb{I}(I \odot M; C)$ we should ideally treat ζ as a random variable

Another simple solution

- $M \to$ revealing more pixels $\to M'$
- Should be even easier to predict the label: $\mathbb{I}(I \odot M'; C) \geq \mathbb{I}(I \odot M; C)$

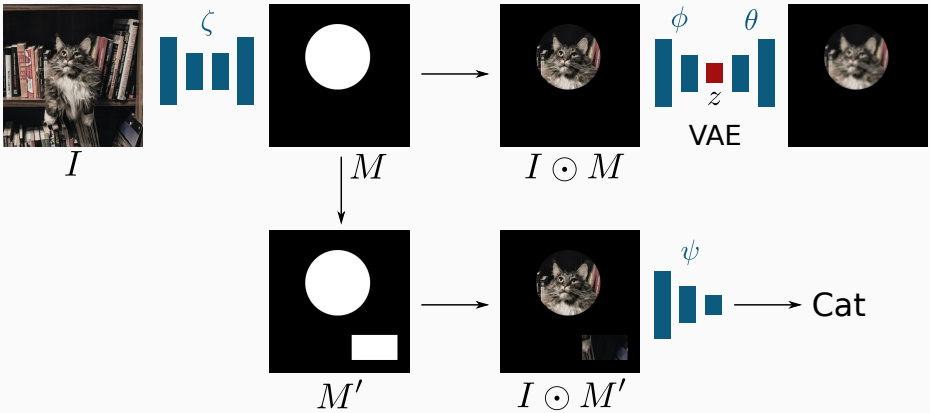
Modified Information-Bottleneck Objective

- M' is a stochastically "grown" mask M
- New objective on M:

$$\underset{\zeta}{\operatorname{argmax}} \left[\mathbb{I}(I \odot M'; C) - \beta \mathbb{I}(I; I \odot M) \right]$$

- $I \odot M'$ uses a perturbed mask $\to$ impossible to communicate the image class through M reliably

- In our final model we train ζ, ϕ, θ and ψ simulatenously:

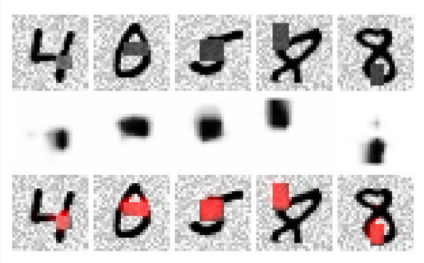


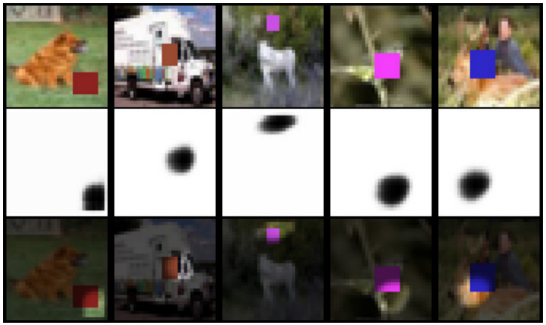
- Train simultaneously:
  (a) mask generator $I \to M$ (learning ζ)
  (b) VAE on the masked image $I \odot M \leftrightarrow Z$ (learning ϕ and θ)
  (c) classifier on the aug. masked image $I \odot M' \to C$ (learning ψ)

Experiments with “Anomalies”

- **Dataset**: Using a set of images, we generate a dataset
  - unperturbed images – **class 0**
  - images with an added “anomaly” – **class 1**

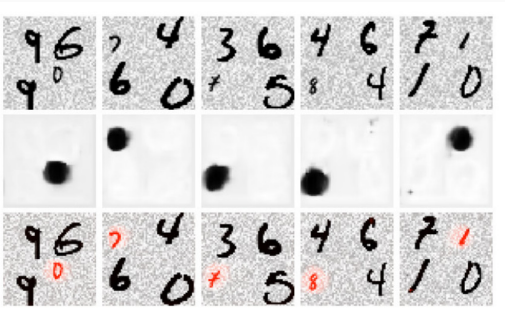
- **Dataset**: Using a set of images, we generate a dataset
  - unperturbed images – **class 0**
  - images with an added "anomaly" – **class 1**
- **Examples**:
  - Classification accuracy: $\sim 98\%$
  - No anomaly $\rightarrow$ empty mask M
  - With anomaly:



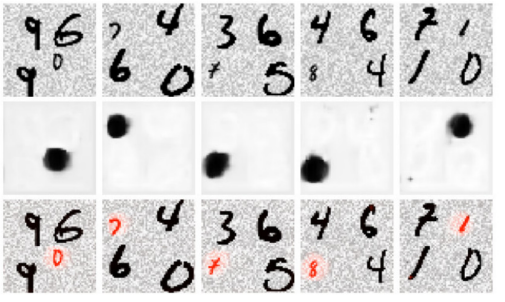


Experiments with “Distractors” and SVHN

- **Dataset**: Using 4 MNIST digits with the smallest one defining the image class (92%)



- **Dataset**: Using 4 MNIST digits with the smallest one defining the image class (92%)



- **SVHN dataset**: Label = number of digits

Takeaways

- Information bottleneck principle provides a rigorous framework for the information flow intuition
- Information theoretic objectives can be useful for:
  - finding good data representations
  - finding salient regions in images or sentences
  - explaining pre-trained models
- A. Oord, et al., *Representation Learning with Contrastive Predictive Coding* (2018)
- K. Schulz, et al., *Restricting the Flow: Information Bottlenecks for Attribution* (2020)
- S. Taghanaki, et al., *InfoMask: Masked Variational Latent Representation to Localize Chest Disease* (2019)

Question

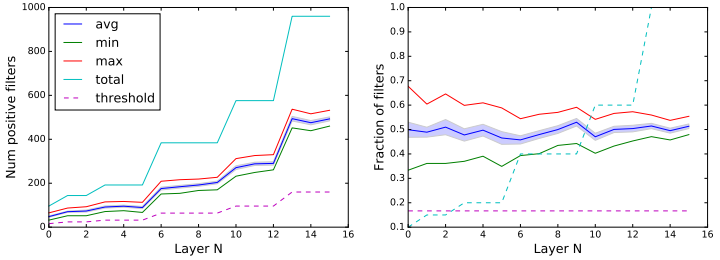
If I understood correctly, the question was about the possibility of detecting salient regions if there is a "correlation" between different objects appearing in the same image.

I like a specific example with "airplane in the sky" vs. "dog on a lawn" classification task. If all airplanes are only seen with a sky background (same for dogs), a sufficiently general statistical method probably should not be able to "discover" that the sky is not a "part" of an "airplane object". Furthermore, any small part of the sky (or a lawn) should count as a *salient region* because its presence is a "strong predictor" of the label.

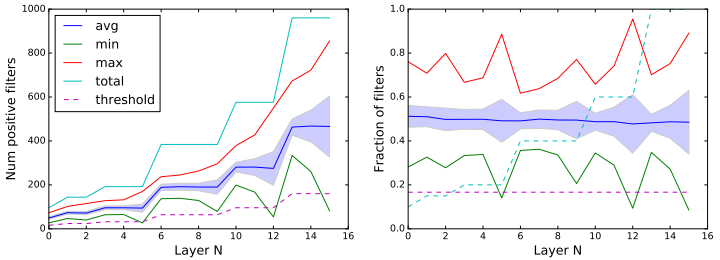
Appendix

Number of Positive Activations

- At initialization, the number of positive activations is almost certainly $> 1/6$ (in expansion)



- After training, the average number of positive activations does not change, but the variation grwos dramatically (mostly still invertible)



Derivation

- Modified Information-Bottleneck objective:

$$\underset{\zeta}{\operatorname{argmax}} \left[\mathbb{I}(I \odot M'; C) - \beta \mathbb{I}(I; I \odot M) \right]$$

- Rewriting our objective as:

$$\underset{\zeta}{\operatorname{argmin}} \left[\beta \mathbb{H}(I \odot M) - \beta \mathbb{H}(I \odot M | I) + \mathbb{H}(C | I \odot M') \right]$$

- Using **variational upper bound** for $\mathbb{H}$

$$\mathbb{H}(A) \leq -\mathbb{E}_a \log p_\phi(a)$$

How to parametrize $p_\theta(i \odot m)$?

- Following VAE approach, we use a latent variable model
- $p_\theta(i \odot m) = \mathbb{E}_z p_\theta(i \odot m | z) p(z)$

$$-\log p_\theta(i \odot m) \leq -\mathbb{E}_{z \sim q_\phi} \log p_\theta(i \odot m | z) + D_{\mathrm{KL}}[q_\phi(z | i \odot m) \| p(z)]$$

- Using **Gumbel-Softmax trick** as a way to draw samples from a categorical distribution