Published as a conference paper at ICLR 2020

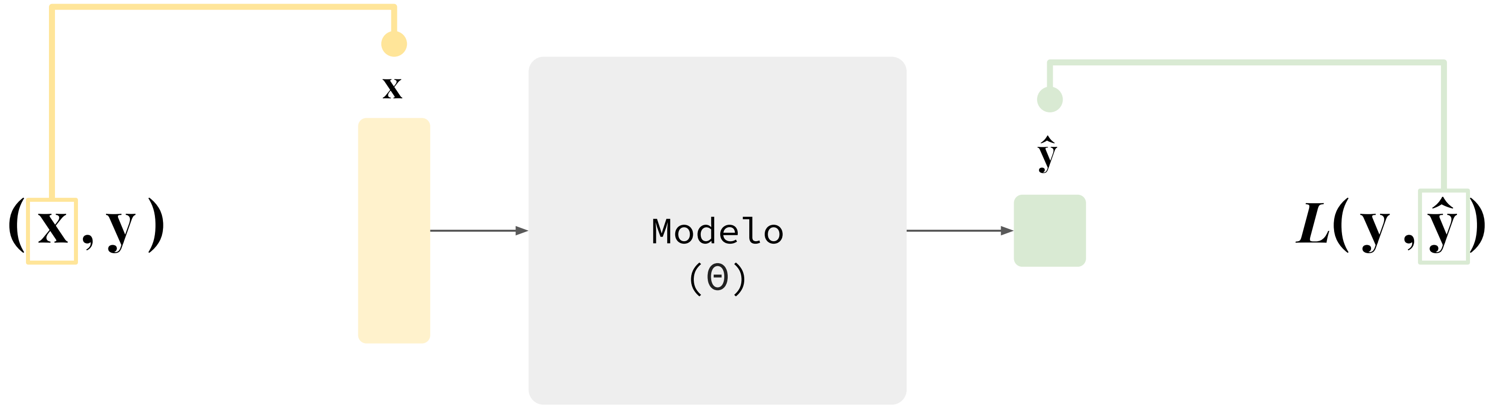
YOU ONLY TRAIN ONCE: LOSS-CONDITIONAL TRAINING OF DEEP NETWORKS

Alexey Dosovitskiy & Josip Djolonga
Google Research, Brain Team
{adosovitskiy, josipd}@google.com

Introdução

Intro

Supervised learning



Encontrar Θ que minimiza $L(\mathbf{y}, \hat{\mathbf{y}})$

Intro

Supervised learning - *Loss function*

Qual é a cara da $\boldsymbol{L}(\mathbf{y}, \hat{\mathbf{y}})$?

Intro

Supervised learning - *Loss function*

Qual é a cara da $\boldsymbol{L}(\mathbf{y}, \hat{\mathbf{y}})$?

RMSE $$\sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

Cross-Entropy $$-\frac{1}{m} \sum_{i=1}^{m} y_i \cdot \log(\hat{y}_i)$$

Intro

Supervised learning - *Loss function*

Qual é a cara da $\boldsymbol{L}(\mathbf{y}, \hat{\mathbf{y}})$?

RMSE $\sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$

Cross-Entropy $-\frac{1}{m} \sum_{i=1}^{m} y_i \cdot \log(\hat{y}_i)$

Custom Loss

$$L = \beta_a L_a + \beta_b L_b + \beta_c L_c$$

Intro

Supervised learning - *Loss function*

Qual é a cara da $\boldsymbol{L(\mathbf{y}, \hat{\mathbf{y}})}$?

RMSE $\sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$

Cross-Entropy $-\frac{1}{m} \sum_{i=1}^{m} y_i \cdot \log(\hat{y}_i)$

Custom Loss

$$\boxed{L = \beta_a L_a + \beta_b L_b + \beta_c L_c}$$

Intro

Parameterized loss function: Image Compression

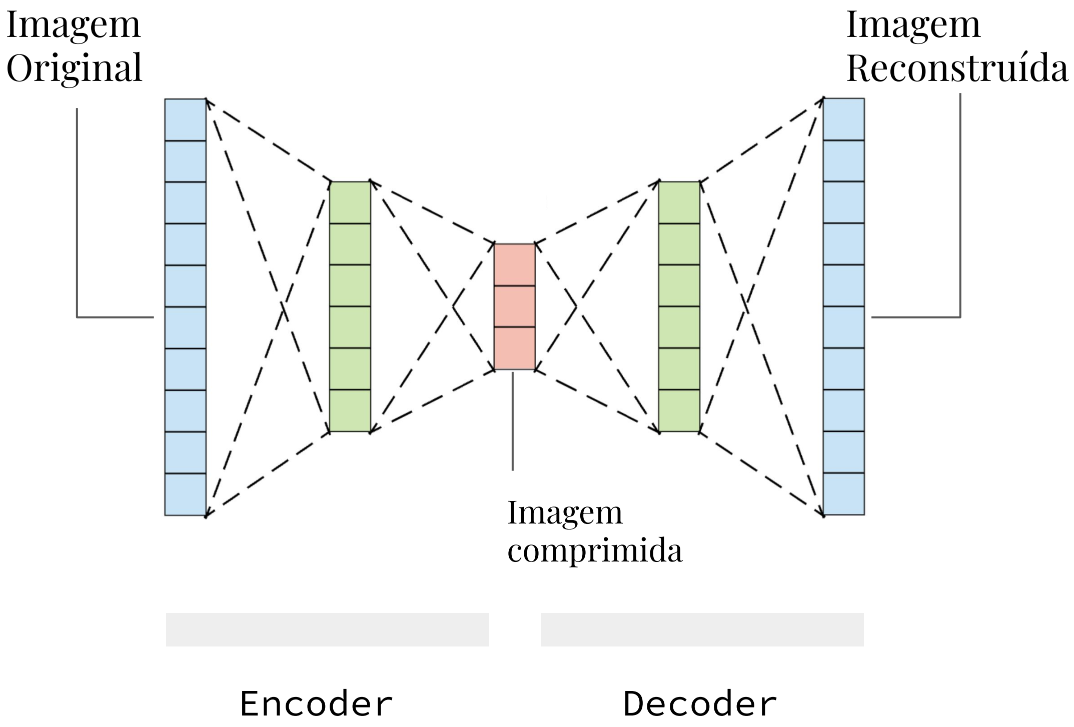


Imagem original = x

Imagem reconstruída = $\hat{x}$

Imagem comprimida = z

Distorção = $D = \mathrm{MSE}(x, \hat{x})$

Tamanho = $T = \mathrm{size}(z)$

$$L = \beta_d D + \beta_t T$$

Intro

Parameterized loss function: Image Compression

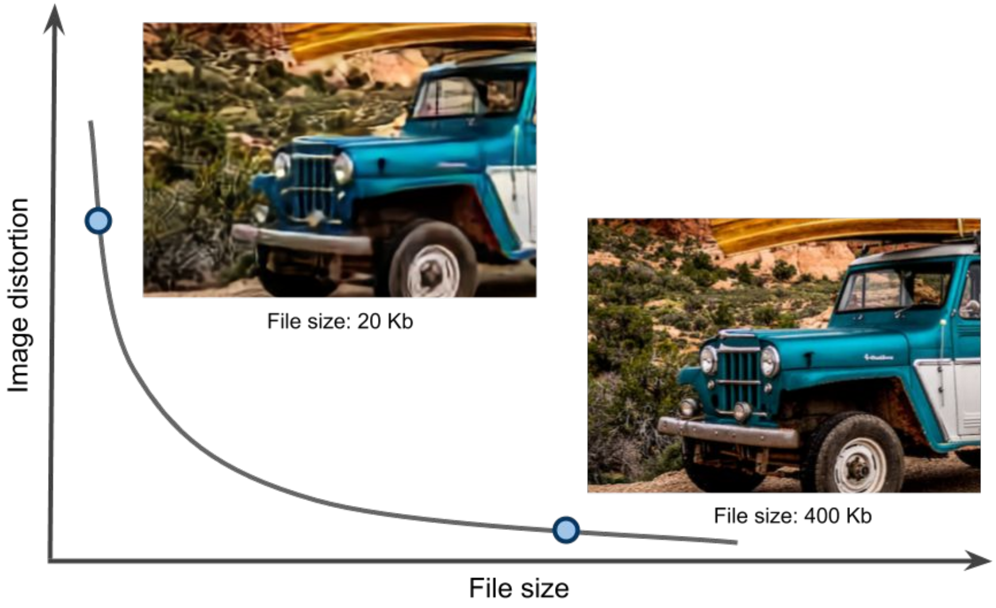


Imagem original = x
Imagem reconstruída = $\hat{x}$
Imagem comprimida = z
Distorção = $D = \mathrm{MSE}(x, \hat{x})$
Tamanho = $T = \mathrm{size}(z)$

$$L = \beta_d D + \beta_t T$$

Intro

Parameterized loss function: Image Compression

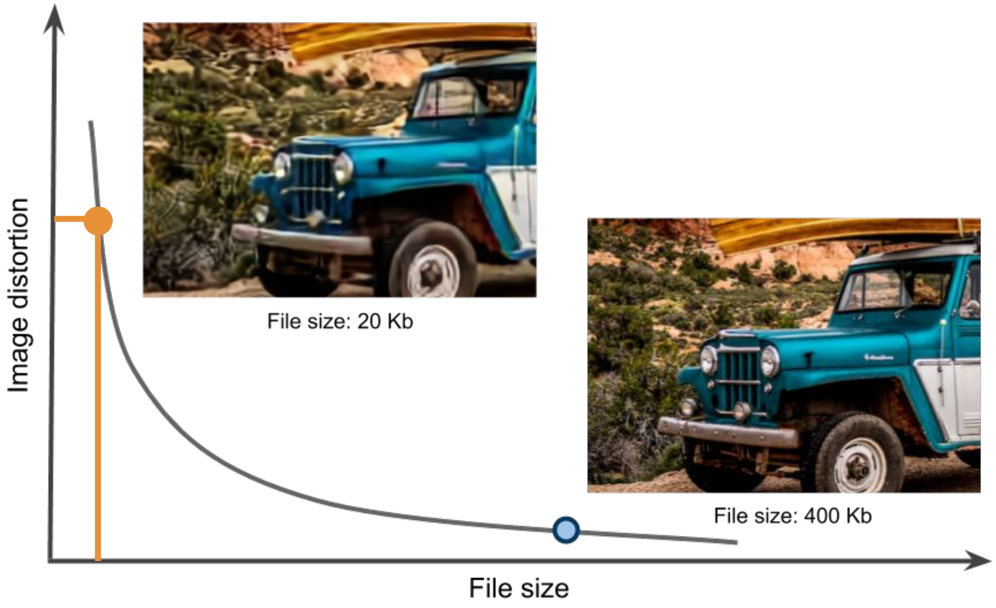


Imagem original = x

Imagem reconstruída = $\hat{x}$

Imagem comprimida = z

Distorção = $D = \mathrm{MSE}(x, \hat{x})$

Tamanho = $T = \mathrm{size}(z)$

$$L = \beta_d D + \beta_t T$$

Intro

Parameterized loss function: Image Compression

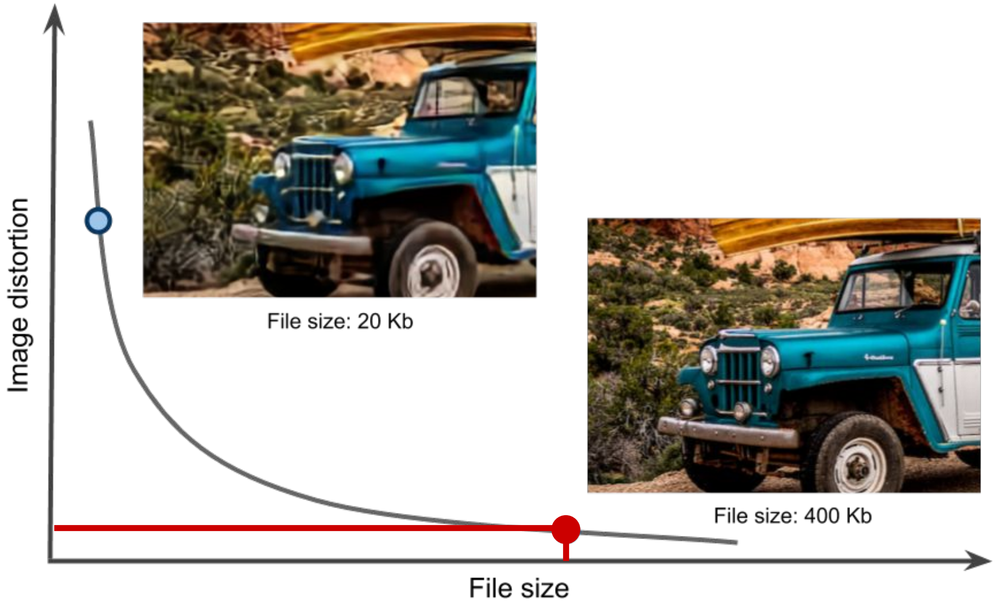


Imagem original = x

Imagem reconstruída = $\hat{x}$

Imagem comprimida = z

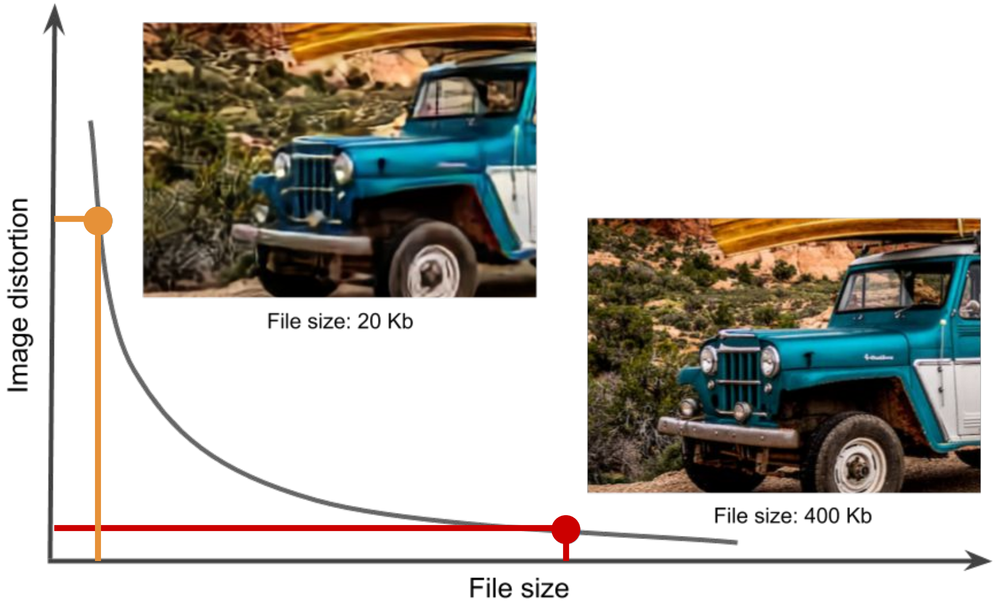
Distorção = $D = \mathrm{MSE}(x, \hat{x})$

Tamanho = $T = \mathrm{size}(z)$

$$L = \boldsymbol{\beta_d} D + \beta_t T$$

Intro

Parameterized loss function: Image Compression



$$\text{Modelo}_a: L = \beta_d D + \beta_t T$$

$$\text{Modelo}_b: L = \beta_d D + \beta_t T$$

Intro

Parameterized loss functions: multiple models limitations

Recursos computacionais

Se você tiver muitos termos na sua *loss function*, você terá muitos modelos independentes para treinar.

Dificuldade na seleção dos pesos

Muitas vezes não é trivial escolher quais as combinações de pesos nos termos das *loss* são as melhores para as diferentes aplicações. Pelo método tradicional, o ajuste desses parâmetros é um processo lento.

Proposta

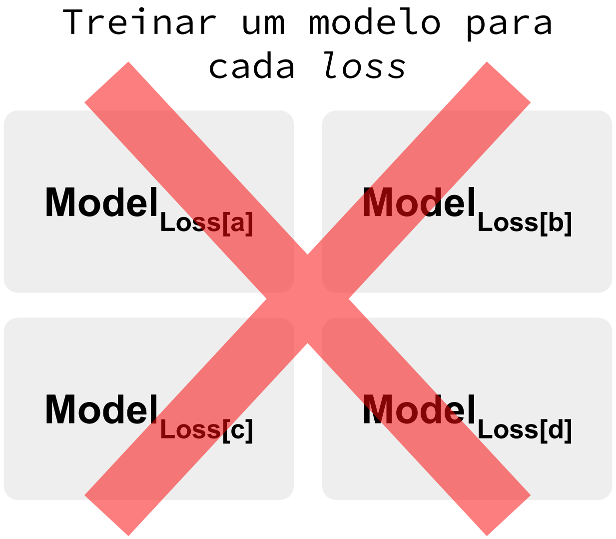
***Y**ou **O**nly **T**rain **O**nce*

Treinar um modelo para cada *loss*

$\mathbf{Model}_{\mathbf{Loss[a]}}$	$\mathbf{Model}_{\mathbf{Loss[b]}}$
$\mathbf{Model}_{\mathbf{Loss[c]}}$	$\mathbf{Model}_{\mathbf{Loss[d]}}$

Proposta

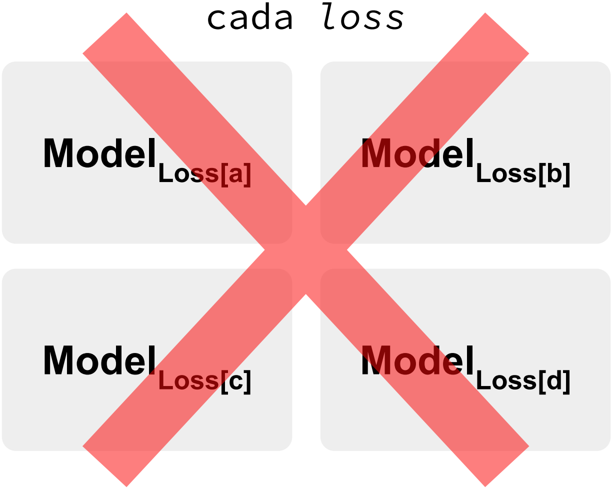
***Y**ou **O**nly **T**rain **O**nce*



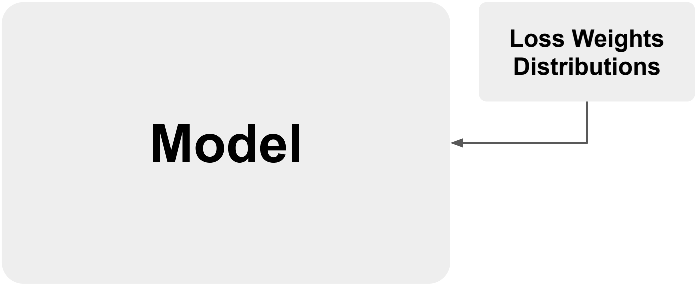
Proposta

***Y**ou **O**nly **T**rain **O**nce*

Treinar um modelo para cada *loss*

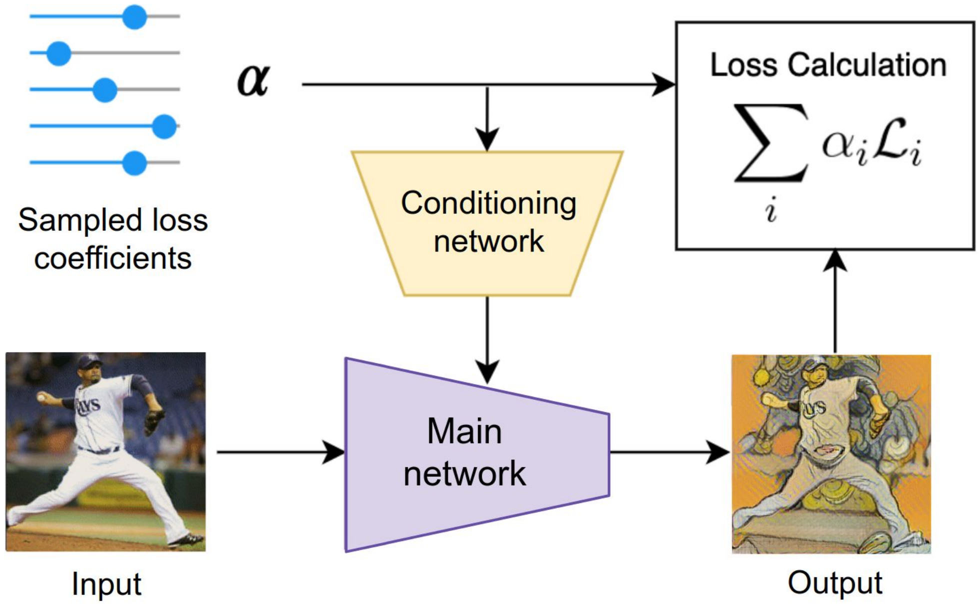


YOTO: You Only Train Once



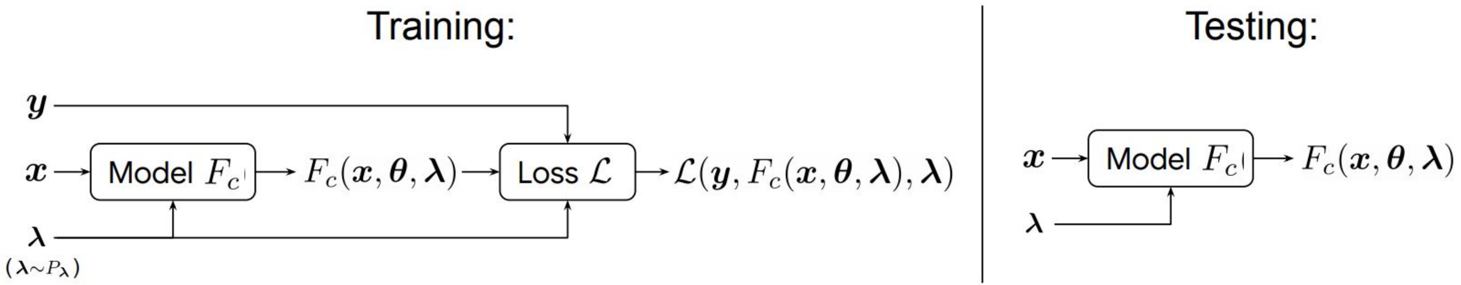
Método

Overview: ***You Only Train Once***



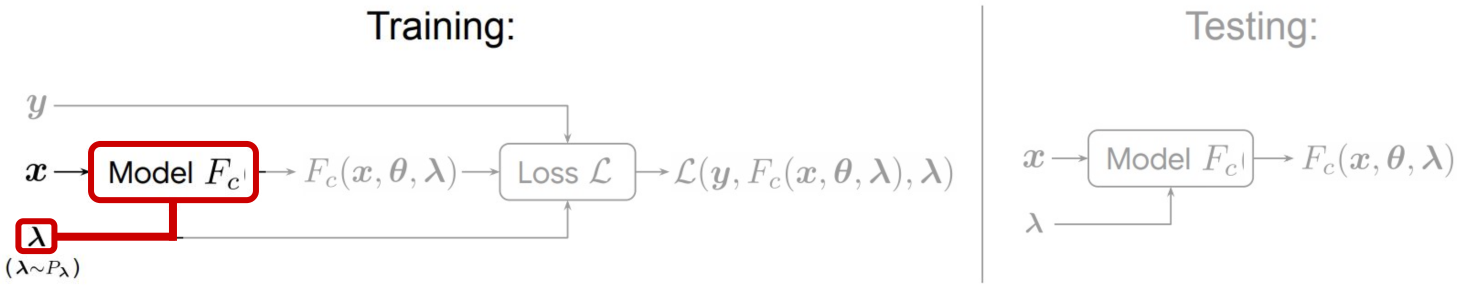
Método

Training vs. Testing



Método

Conditioning on loss weights



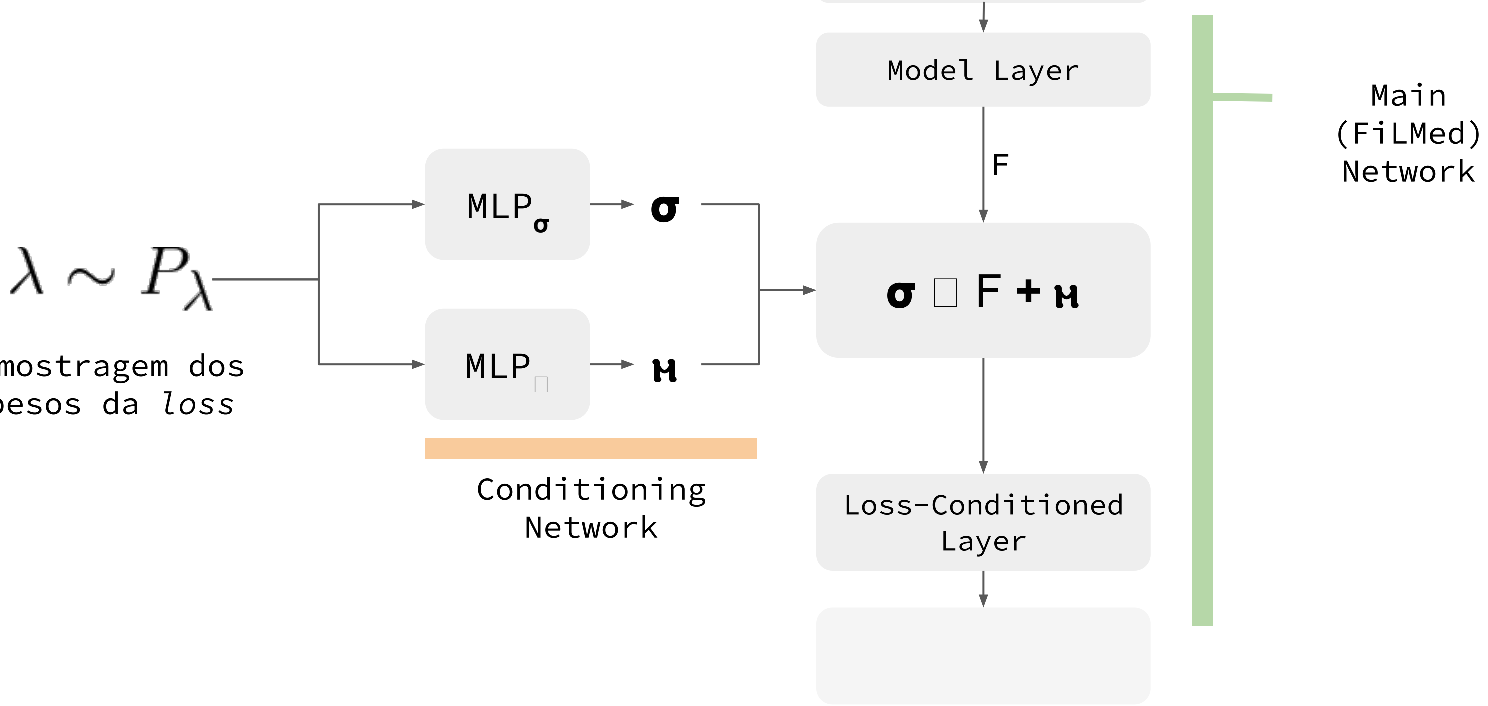
Método

Loss Conditioning

FiLM: Visual Reasoning with a General Conditioning Layer

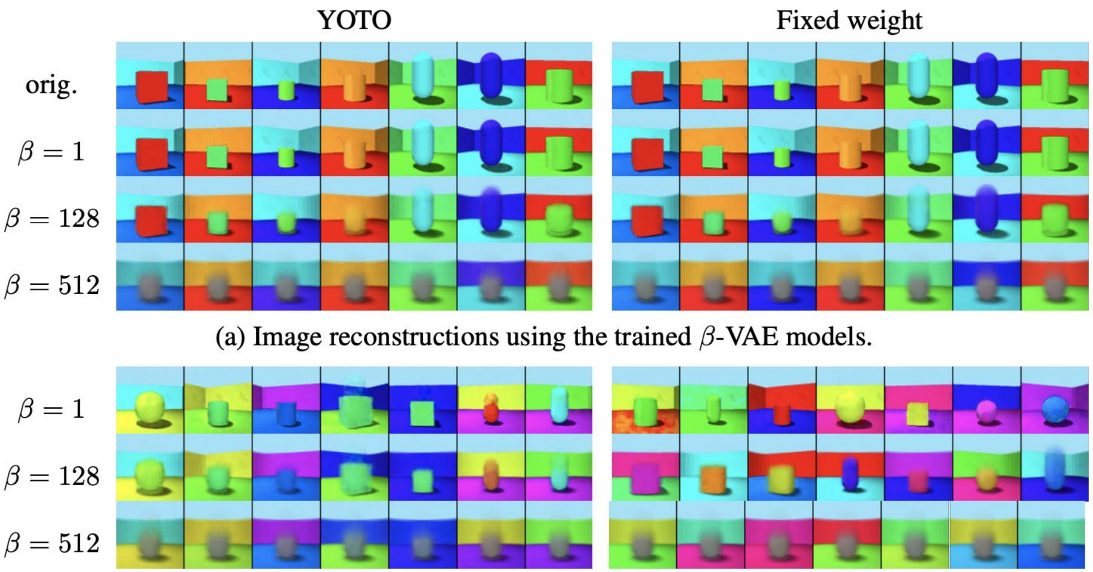
Ethan Perez[1,2], Florian Strub[4], Harm de Vries[1], Vincent Dumoulin[1], Aaron Courville[1,3]
[1]MILA, Université de Montréal, [2]Rice University, [3]CIFAR Fellow,
[4]Univ. Lille, CNRS, Centrale Lille, Inria, UMR 9189 CRIStAL France
ethanperez@rice.edu, florian.strub@inria.fr, mail@harmdevries.com,{dumouliv,courvila}@iro.umontreal.ca

Feature-wise Linear Modulation (FiLM)



Resultados

Experiment: β-Variational Autoencoders



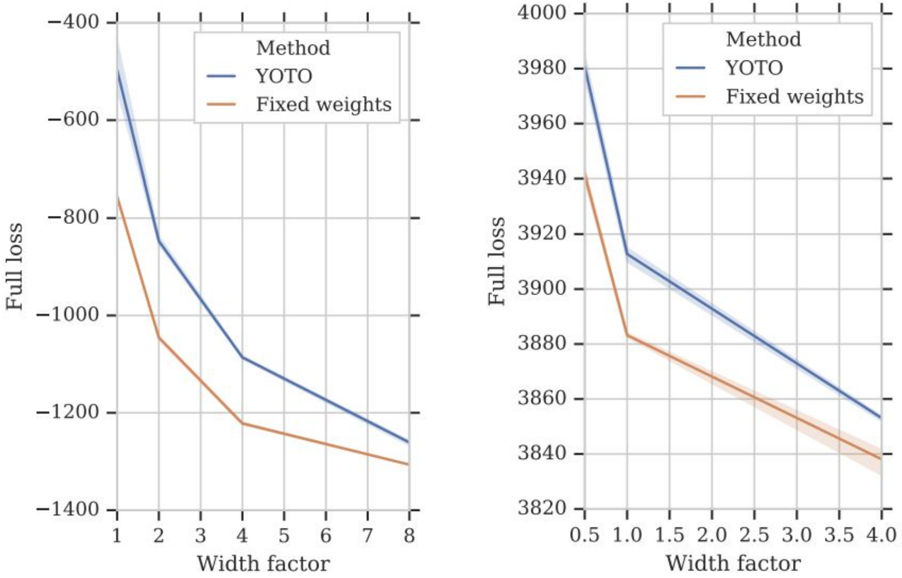
(a) Image reconstructions using the trained β-VAE models.

(b) Samples from the trained β-VAE models.

YOTO possui uma capacidade de reconstrução e geração de novas imagens muito similar a modelos com pesos fixos em β-VAEs

Resultados

Experiment: β-Variational Autoencoders

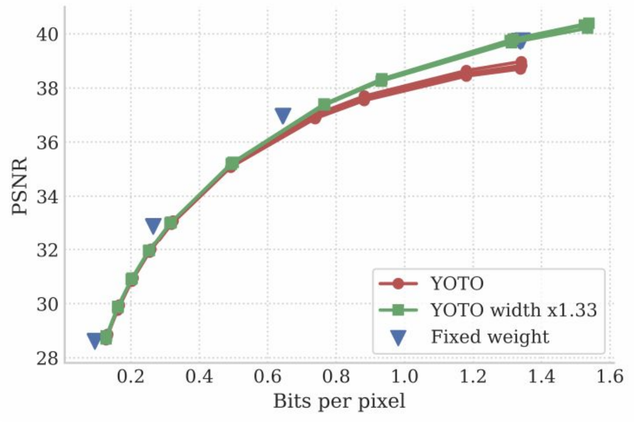


(a) CIFAR-10. (b) Shapes3D.

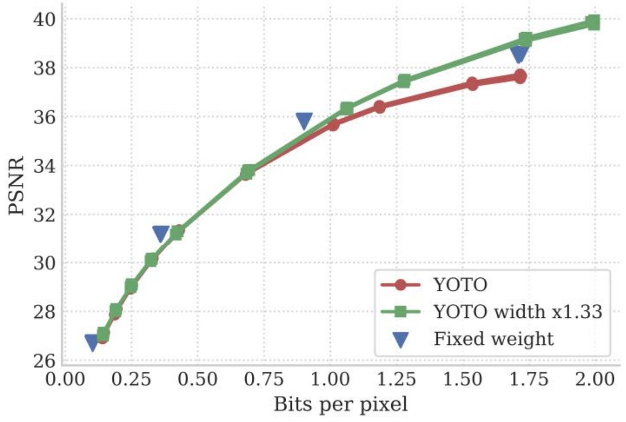
Um modelo **YOTO** precisa de redes com maior capacidade do que modelos com pesos fixos

Resultados

Experiment: Image Compression



(a) PSNR on Tecnick.



(b) PSNR on Kodak.

Na tarefa de compressão de imagens, o modelo **YOTO** se aproxima dos modelos com pesos fixos apenas quando sua capacidade é aumentada em 33%

Resultados

Experiment: Fast Style Transfer

(a) Varying the content coefficient λ_c

(b) Varying one of the style coefficients $\lambda_{s,3}$

Na tarefa de "transferência de estilo" em que a *loss* possui termos que regularizam o impacto do estilo, a rede treinada com **YOTO** foi capaz de reproduzir os resultados obtidos por modelos treinados independentemente.

Conclusões

- Bons resultados em tarefas de geração/compressão de imagens e transferência de estilo

- Estudos mais profundos precisam ser realizados a respeito do impacto da escolha das **distribuições dos pesos** nos termos da ***loss***.

- Abertura de possibilidades para aplicação do método em outros domínios além do processamento de imagens.

Dúvidas?