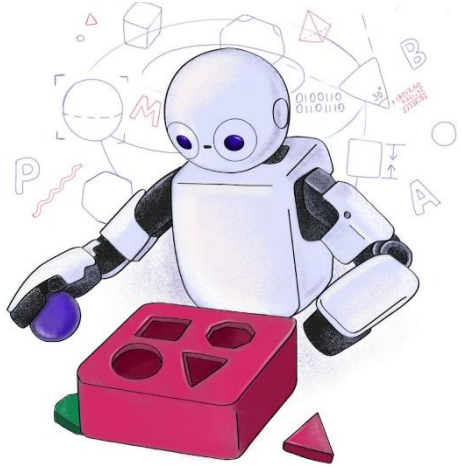
TP558 - Tópicos avançados em Machine Learning:

Adicione aqui seu tema



Inatel

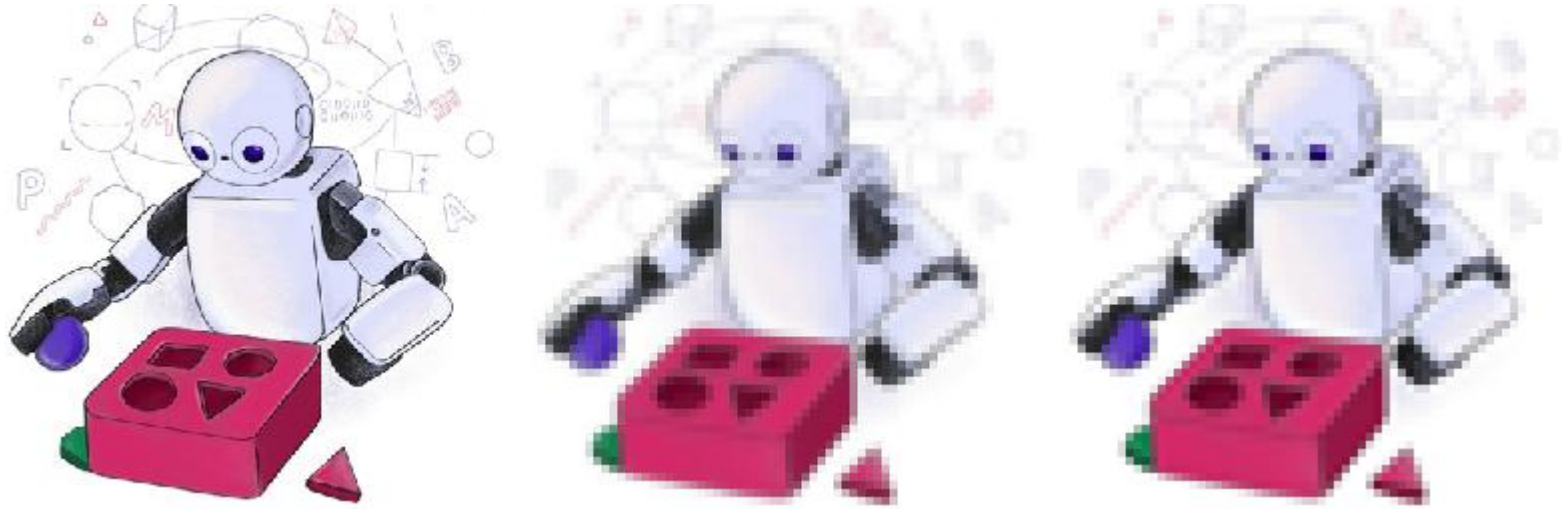
LUIS ENRIQUE CARDOZO RAMIREZ

Introdução

- Já pensaram como **reduzir** uma imagem de **alta resolução**, como uma foto para uma IA classificar objetos **sem perder detalhes cruciais**, como os contornos que diferenciam um carro de um caminhão?
- Imagine uma fotografia de alta resolução que precisa ser exibida em uma tela pequena ou usá-la em um aplicativo, classificação de objetos.

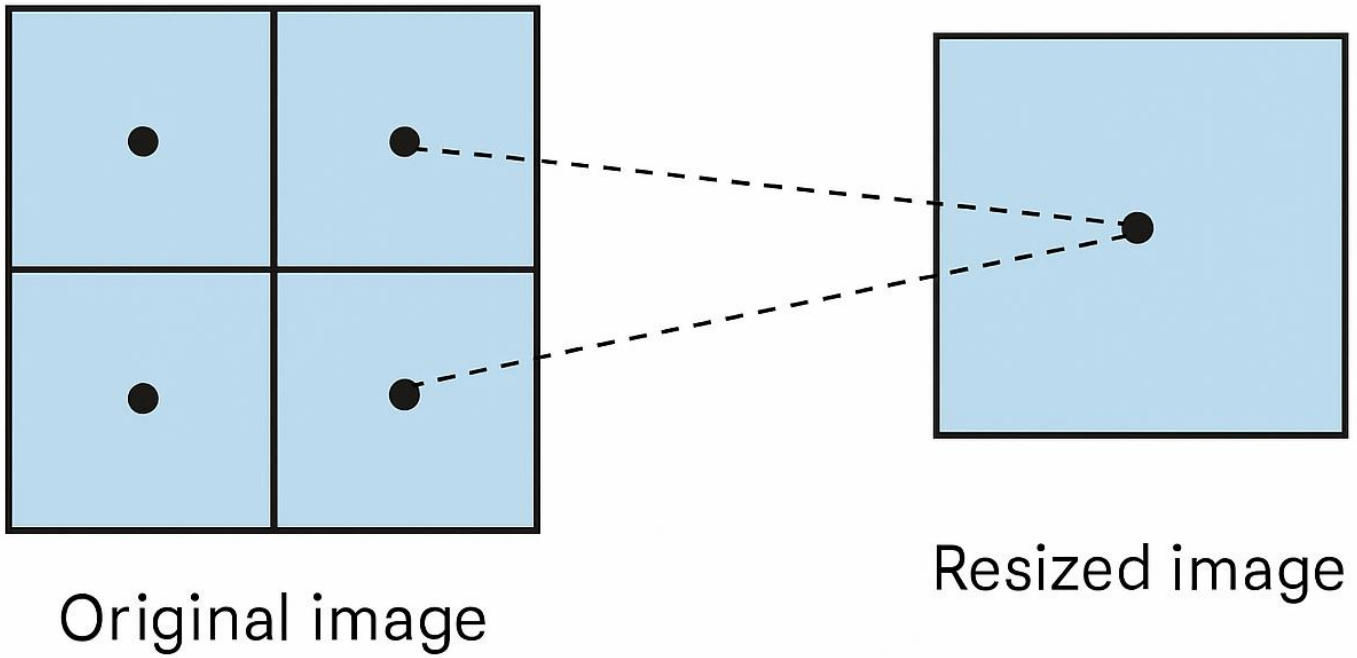
Introdução

- Você precisa torná-la menor. Esse processo é chamado de **redimensionamento**.
- Tradicionalmente, os programas de computador utilizam **métodos padronizados** para isso, baseados em fórmulas matemáticas pré-definidas.



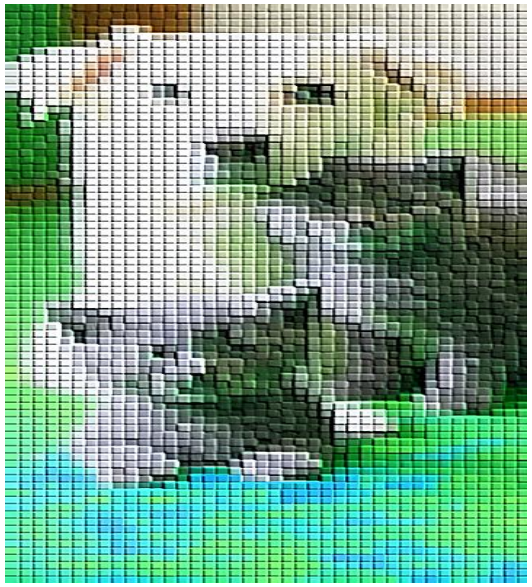
Introdução

- Um método bastante utilizado é a **Interpolação Bilinear**, que considera os quatro pixels vizinhos mais próximos na imagem original e calcula uma média ponderada para gerar um novo pixel na imagem redimensionada.



Introdução

- O problema central reside no fato de que esses resizers aplicam a mesma **lógica de escalonamento em toda a imagem**, sem levar em conta quais detalhes são cruciais para o reconhecimento
- Ele não distingue, por exemplo, entre o gramado ao fundo. Ao calcular a média das cores, ao mesmo tempo em que **preserva sem dificuldade áreas de menor relevância**, como uma parede uniforme.



Introdução

Solução Inteligente:

Um Redimensionador que "**Adapta**" a Imagem para **Máquinas**

E se, em vez de usar um **método fixo** como o redimensionamento bilineal, tivéssemos um "**especialista digital**" que analisa a imagem e a redimensiona de forma a **destacar** as características mais importantes para uma tarefa específica de **visão computacional**?

Introdução

Proposed learned image resizer

- O método utiliza uma **rede neural convolucional (CNN)** que **aprende, junto com o modelo** de visão, a redimensionar imagens de maneira otimizada para a tarefa-alvo.
- O treinamento é conjunto, utilizando a **mesma função de perda** (ex.: entropia cruzada para classificação).

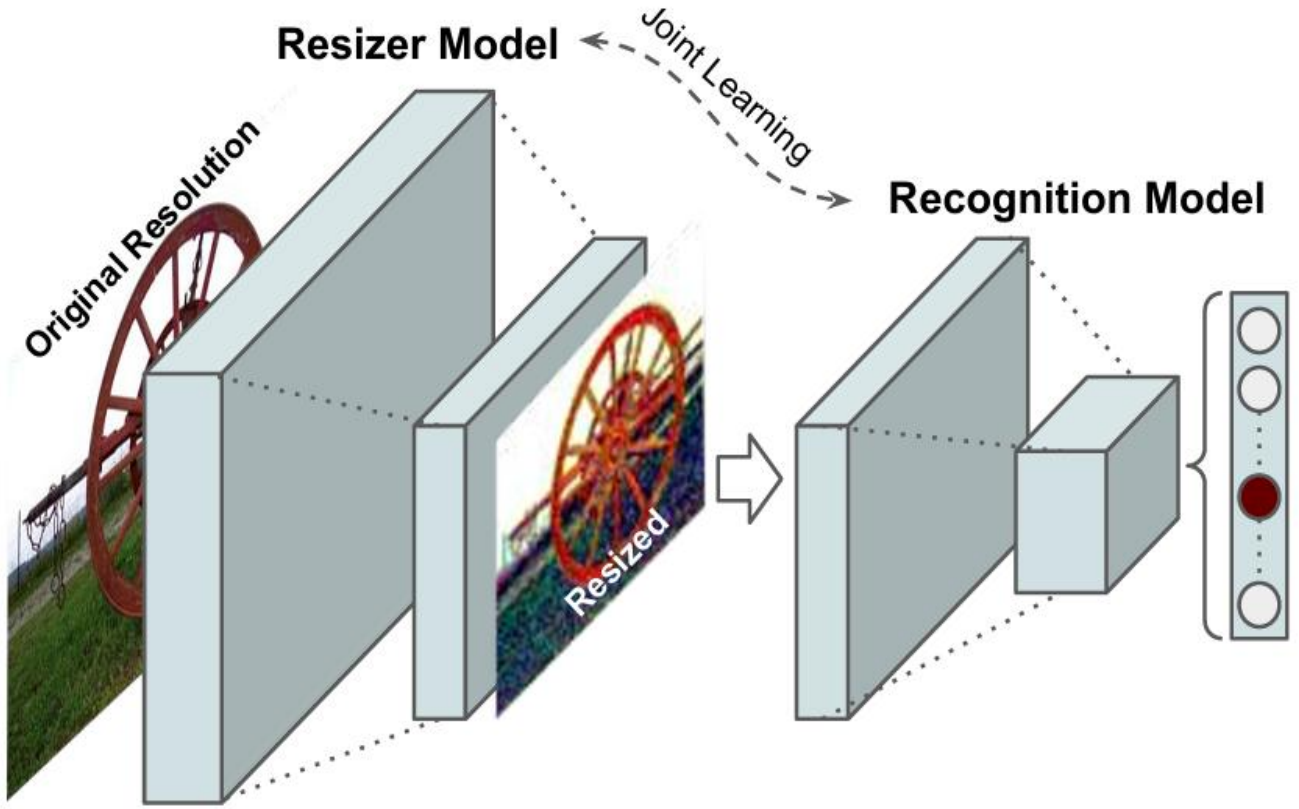


Figure 1. Our proposed framework for joint learning of the image resizer and recognition models.

Introdução

Proposed learned image resizer

Característica	Descrição
Integração adaptativa	Adapta-se a diferentes modelos, melhorando consistentemente os resultados.
Independência de perdas perceptuais	Gera efeitos visuais otimizados para máquinas, não para humanos.
Flexibilidade de escala	Permite redimensionamento em fatores arbitrários, buscando a resolução ideal.
Expansão para IQA	Adapta-se com sucesso à avaliação de qualidade de imagem.
Aplicação em inferência remota	Reduz latência em sistemas cliente-servidor, mantendo precisão.

Introdução

- A **Tabela 1** apresenta as **métricas de avaliação** de desempenho e qualidade para **quatro modelos** de redes neurais amplamente utilizados em **visão computacional**.

Métrica	Resumo
Top-1 Error	Percentual de vezes que a classe correta não é a 1ª predição. (**Menor é melhor** ↓)
IQA	Tarefa de avaliar a qualidade da imagem comparando com julgamentos humanos.
PLCC	Correlação linear entre predições do modelo e avaliações humanas. (**Maior é melhor** ↑)

Introdução

- A **Tabela 1** apresenta as **métricas de avaliação** de desempenho e qualidade para **quatro modelos** de redes neurais amplamente utilizados em **visão computacional**.

Task	Model	Top-1 Error ↓	
		Bilinear Resizer	Proposed Resizer
Classification	Inception-v2 [34]	26.7%	24.0%
	DenseNet-121 [9]	33.1%	29.8%
	ResNet-50 [8]	24.7%	23.0%
	MobileNet-v2 [28]	29.5%	28.4%
		PLCC ↑	
		Bicubic Resizer	Proposed Resizer
IQA	Inception-v2 [34]	0.662	0.686
	DenseNet-121 [9]	0.662	0.683
	EfficientNet-b0 [38]	0.642	0.671

- Inception-v2 [34]: desenvolvida por Szegedy et al. (2016)
- DenseNet-121 [9]: Proposta por Huang et al. (2017)
- ResNet-50 [8]: Desenvolvida por He et al. (2016),
- MobileNet-v2 [28]: Proposta por Sandler et al. (2018)
- EfficientNet-b0 [38]: Desenvolvida por Tan y Le (2019)

Introdução

- Inception-v2: Conhecida por usar blocos inception, que **capturam características em múltiplas escalas**. Foi testada para classificação e IQA.
- DenseNet-121: Utiliza conexões densas para promover a **reutilização de características e melhorar a eficiência**. Foi testada em ambas as tarefas.
- ResNet-50: Uma rede residual que usa "**conexões de salto**" para facilitar o **treinamento de redes muito profundas**. Foi testada para classificação.
- MobileNet-v2: Uma arquitetura leve, **otimizada para dispositivos móveis**, que emprega convoluções separáveis. Foi testada para classificação.
- EfficientNet-b0: Uma rede que equilibra **profundidade, largura e resolução** de forma eficiente. Foi testada apenas para IQA.

Introdução

- A **Tabela 1** apresenta as **métricas de avaliação** de desempenho e qualidade para **quatro modelos** de redes neurais amplamente utilizados em **visão computacional**.

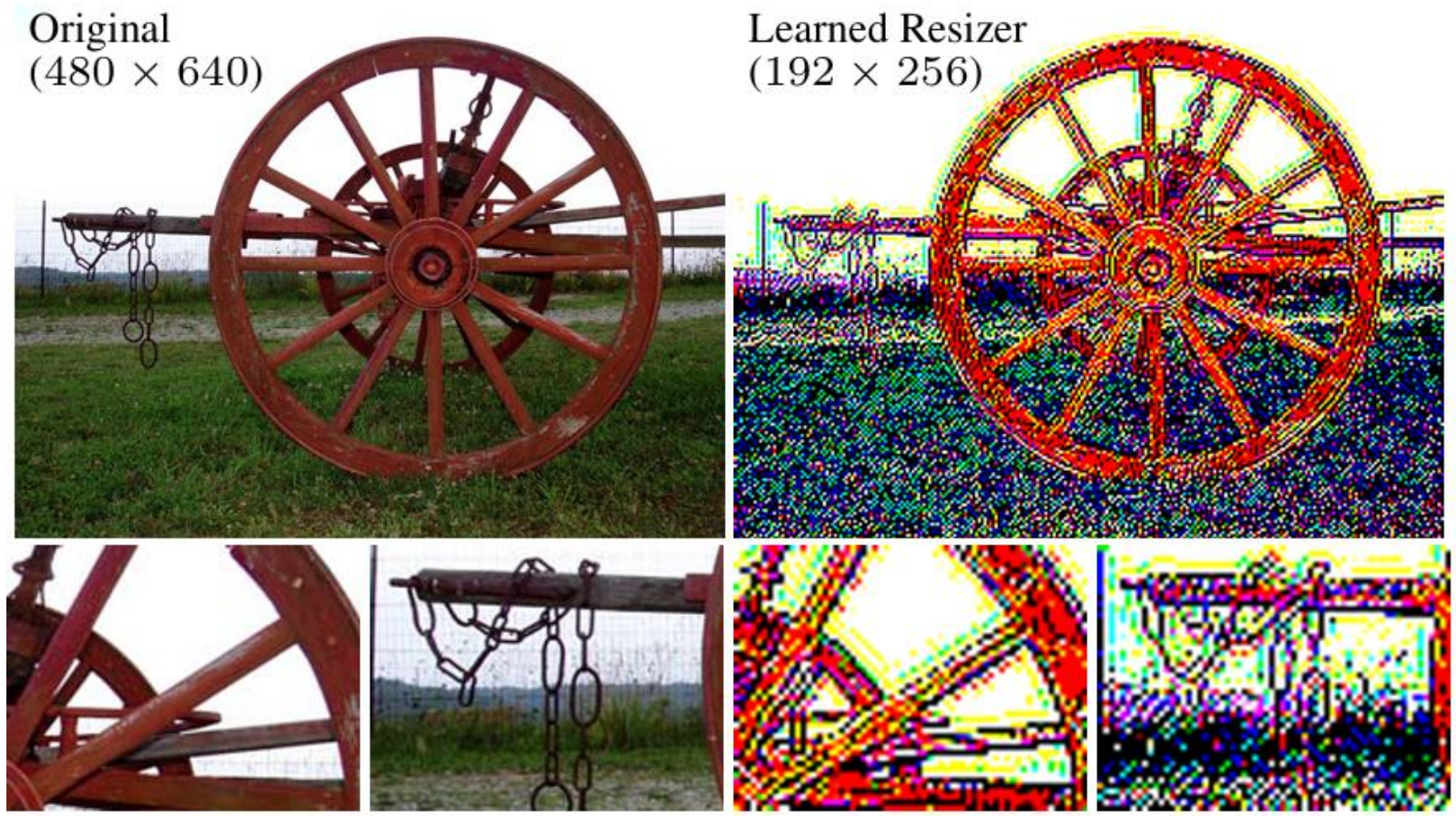
Task	Model	Top-1 Error ↓	
		Bilinear Resizer	Proposed Resizer
Classification	Inception-v2 [34]	26.7%	24.0%
	DenseNet-121 [9]	33.1%	29.8%
	ResNet-50 [8]	24.7%	23.0%
	MobileNet-v2 [28]	29.5%	28.4%
		PLCC ↑	
		Bicubic Resizer	Proposed Resizer
IQA	Inception-v2 [34]	0.662	0.686
	DenseNet-121 [9]	0.662	0.683
	EfficientNet-b0 [38]	0.642	0.671

- Inception-v2 [34]: desenvolvida por Szegedy et al. (2016)
- DenseNet-121 [9]: Proposta por Huang et al. (2017)
- ResNet-50 [8]: Desenvolvida por He et al. (2016),
- MobileNet-v2 [28]: Proposta por Sandler et al. (2018)
- EfficientNet-b0 [38]: Desenvolvida por Tan y Le (2019)

Introdução

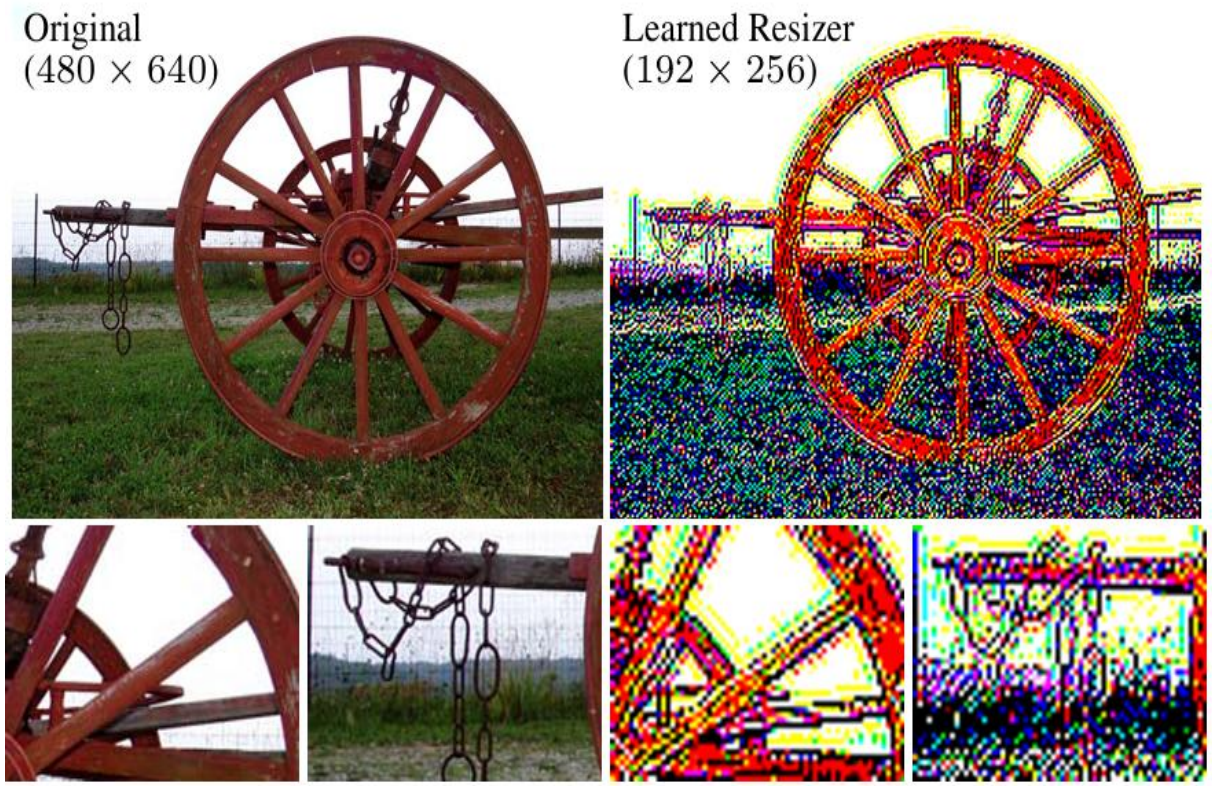
Solução Inteligente: Redimensionamento **Otimizado para Máquinas**

- A Figura 2 ilustra um exemplo prático do redimensionador aprendido, treinado para a classificação no conjunto de dados ImageNet.



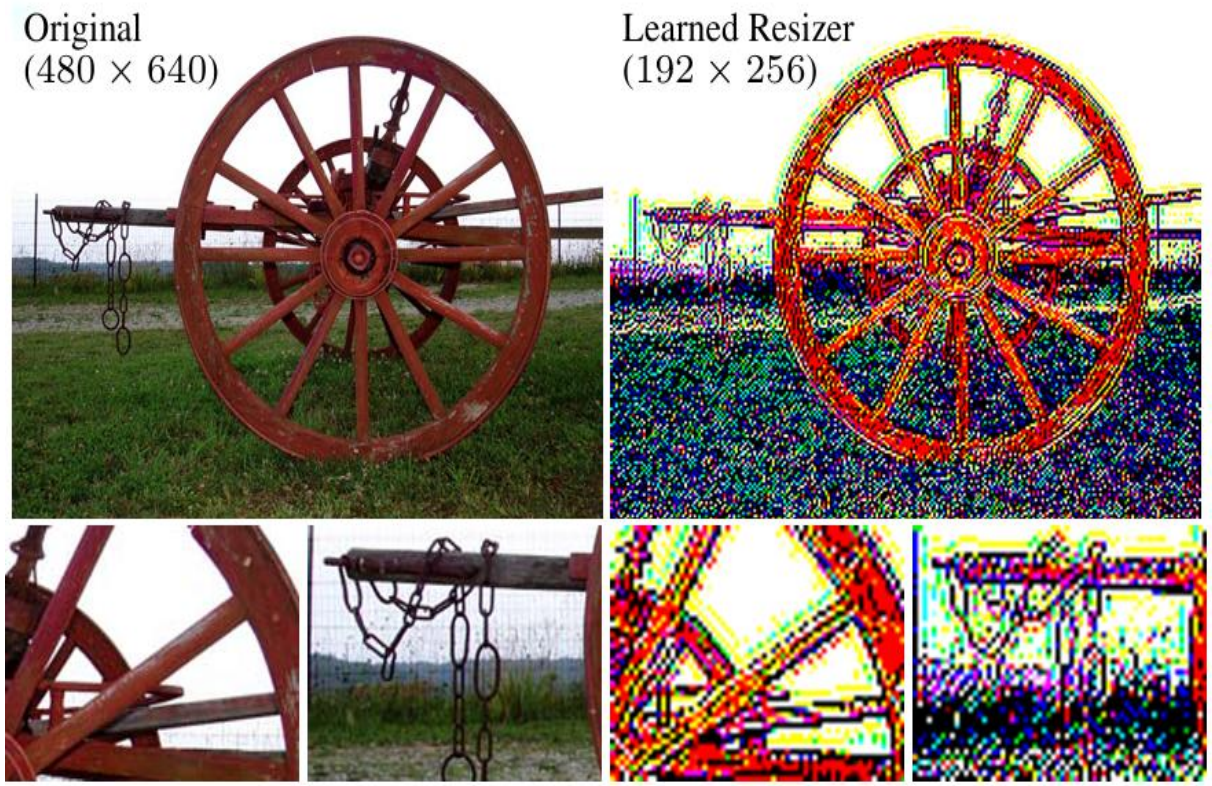
Introdução

- **Redimensionamento Otimizado para Máquinas**
- A imagem original é redimensionada por um modelo aprendido em conjunto com o **Inception-v2**, com uma redução de 480 × 640 para 192 × 256 pixel.
- Diferente dos métodos tradicionais, a **imagem resultante não foca na qualidade visual humana**, mas sim em realçar características "amigáveis para máquinas", como detalhes de alta frequência.



Introdução

- **Redimensionamento Otimizado para Máquinas**
- Essa abordagem melhora o desempenho da tarefa de classificação, alinhando-se com os resultados da Tabela 1, onde o redimensionador reduz o erro Top-1 do Inception-v2 de 26,7% para 24,0%.



Fundamentação teórica

1. Interpolação Bilinear (Métodos Tradicionais de Redimensionamento)

A interpolação Bilinear é usada como baseline no artigo (Tabela 1, Figura 3), redimensionando imagens de forma suave para uniformidade em mini-batches. No entanto, não é otimizada para percepção de máquinas, causando perda uniforme de detalhes ao contrário do redimensionador aprendido.

O valor do píxel interpolado Q(x, y) é calculado como:

$$Q(x, y) = (1 - a)(1 - b)P_{11} + a(1 - b)P_{21} + (1 - a)bP_{12} + abP_{22}$$

Fundamentação teórica

1. Interpolação Bilinear (Métodos Tradicionais de Redimensionamento)

$$Q(x, y) = (1 - a)(1 - b)P_{11} + a(1 - b)P_{21} + (1 - a)bP_{12} + abP_{22}$$

Análise dos Componentes

- $P_{11}, P_{12}, P_{21}, P_{22}$: Valores dos quatro píxeles vizinhos mais próximos na imagem original, em coordenadas de grade.
- $a = x - \lfloor x \rfloor$: Fração decimal da coordenada horizontal, indicando a proximidade relativa aos píxeles.
- $b = y - \lfloor y \rfloor$: Fração decimal da coordenada vertical.
- $(1-a)(1-b), a(1-b), (1-a)b, ab$: Pesos que ponderam a contribuição de cada píxel vizinho, somando 1.

Fundamentação teórica

2. Interpolação Bicúbica(Métodos Tradicionais de Redimensionamento)

É uma técnica amplamente utilizada para **redimensionar imagens de forma suave e precisa.** No artigo, a interpolação bicúbica é usada como um método de referência para comparar o desempenho do modelo proposto de redimensionamento aprendido (learned resizer).

O valor do píxel interpolado Q(x, y) na posição (x, y) é calculado usando uma função polinomial cúbica que considera os 16 píxeis vizinhos.

$$Q(x,y) = \sum_{i=-1}^{2} \sum_{j=-1}^{2} P(x_i, y_j) \cdot w(x - x_i) \cdot w(y - y_j)$$

Fundamentação teórica

2. Interpolação Bicúbica(Métodos Tradicionais de Redimensionamento)

$$Q(x,y) = \sum_{i=-1}^{2} \sum_{j=-1}^{2} P(x_i, y_j) \cdot w(x - x_i) \cdot w(y - y_j)$$

$P(x_i, y_j)$ São os valores dos **16 píxeis vizinhos** ao ponto (x, y) na imagem original.

x e y : São as coordenadas normalizadas do ponto onde o valor interpolado deve ser calculado.

x_i, y_j : São as coordenadas dos píxeis vizinhos em relação ao ponto (x, y).

w(u) : É a função de peso cúbico que determina a contribuição de cada píxel ao resultado final. A função de peso típica é:

Fundamentação teórica

2. Interpolação Bicúbica(Métodos Tradicionais de Redimensionamento)

$$Q(x,y) = \sum_{i=-1}^{2} \sum_{j=-1}^{2} P(x_i, y_j) \cdot w(x - x_i) \cdot w(y - y_j)$$

w(u) : É a função de peso cúbico que determina a contribuição de cada píxel ao resultado final. A função de peso típica é:

$$w(u) = \begin{cases} (1.5|u|^3 - 2.5|u|^2 + 1), & \text{se } |u| \leq 1 \\ (-0.5|u|^3 + 2.5|u|^2 - 4|u| + 2), & \text{se } 1 < |u| \leq 2 \\ 0, & \text{se } |u| > 2 \end{cases}$$

Fundamentação teórica

Uma Receita Adaptável

Este aprendizado é guiado por uma **formulação que otimiza o redimensionador** e o modelo base juntos, como descrito na Figura 1. A base teórica é o aprendizado conjunto, onde a imagem redimensionada é otimizada para **maximizar o desempenho da tarefa, não a qualidade visual humana.**

Formulação Matemática

O objetivo é encontrar os parâmetros ótimos do redimensionador (θ) e do modelo base (ϕ) que **minimizem o erro da tarefa final**, conforme a equação:

$$(\theta^*, \phi^*) = \arg\min_{\theta,\phi} E_{(x,y)\sim D}\left[\mathcal{L}\left(g_\phi\left(f_\theta(x)\right), y\right)\right]$$

Fundamentação teórica

$$(\theta^*, \phi^*) = \arg\min_{\theta,\phi} E_{(x,y)\sim D} \left[\mathcal{L} \left(g_\phi \left(f_\theta(x) \right), y \right) \right]$$

Análise dos Componentes

- $f_\theta(x)$: O redimensionador (CNN, Figura 3), que transforma a imagem de entrada x (e.g., 480×640) em uma versão redimensionada (e.g., 192×256). θ são seus pesos (11k–93k parâmetros, Tabela 2).
- $g_\phi(...)$: O modelo base (e.g., Inception-v2, ResNet-50) que processa a imagem redimensionada para a tarefa. ϕ são seus pesos.
- $\mathcal{L}(..., y)$: Função de perda da tarefa: entropia cruzada com *label-smoothing* para classificação (ImageNet, Equação 1) ou Earth Mover's Distance (EMD) para IQA (AVA, Equação 2).
- $E_{(x,y)\sim D}$: Média sobre o dataset D (e.g., pares imagem-etiqueta do ImageNet ou AVA).
- $\arg\min_{\theta,\phi}$: Otimização conjunta via gradiente descendente (e.g., *momentum optimizer*).

Fundamentação teórica

Entropia Cruzada con Label Smoothing

A função de perda é definida como

$$L = -\sum_{k=1}^{K} q'_k \log(p_k)$$

Entropia Cruzada com Label Smoothing ajusta os parâmetros do modelo de classificação (ϕ) para **minimizar o erro de previsão**, ao mesmo tempo que evita que o modelo se torne excessivamente confiante em suas predições.

Fundamentação teórica

Entropia Cruzada con Label Smoothing

$$L = -\sum_{k=1}^{K} q'_k \log(p_k)$$

Análise dos Componentes

q'_k: As etiquetas suavizadas são calculadas como $q'_k = (1-\epsilon)\delta_{k,y} + \frac{\epsilon}{K}$, onde:

- $\delta_{k,y}$: É 1 se $k = y$ (classe verdadeira) e 0 caso contrário.
- ϵ: Parâmetro de suavização ($\epsilon = 0.1$).
- K: Número total de classes (e.g., $K = 1000$ para ImageNet). Essa suavização distribui parte da probabilidade entre todas as classes, reduzindo a confiança excessiva do modelo na classe correta.
- p_k: Probabilidades preditas pelo modelo para cada classe k. Essas probabilidades são obtidas pela saída da camada Softmax no modelo base (e.g., Inception-v2, ResNet-50).

Fundamentação teórica

Entropia Cruzada con Label Smoothing

$$L = -\sum_{k=1}^{K} q'_k \log(p_k)$$

Análise dos Componentes

- $\log(p_k)$: O logaritmo das probabilidades preditas, que penaliza fortemente predições incorretas ou incertas.
- $\sum_{k=1}^{K}$: A soma é realizada sobre todas as classes K, garantindo que o modelo seja avaliado em relação a todas as possíveis classes.

Fundamentação teórica

Earth Mover’s Distance (EMD) (para IQA)

O objetivo da (Earth Mover's Distance - EMD) é medir a **diferença entre duas distribuições de probabilidades**, especialmente no contexto de avaliação de qualidade de imagens (Image Quality Assessment - IQA). A fórmula da **EMD** é definida como:

$$\mathrm{EMD}(P, Q) = \left(\frac{1}{K} \sum_{k=1}^{K} |\mathrm{CDF}(P_k) - \mathrm{CDF}(Q_k)|^d \right)^{1/d}$$

EMD permite uma comparação mais precisa e robusta. No artigo, essa **métrica foi usada com sucesso para treinar modelos de avaliação de qualidade** em conjuntos de dados como o AVA, resultando em melhorias consistentes na correlação entre as pontuações preditas e as reais.

Fundamentação teórica

$$\mathrm{EMD}(P, Q) = \left(\frac{1}{K} \sum_{k=1}^{K} |\mathrm{CDF}(P_k) - \mathrm{CDF}(Q_k)|^d \right)^{1/d}$$

Earth Mover's Distance (EMD) (para IQA)

P **e** Q: Representam as distribuições de probabilidade das avaliações humanas e as predições do modelo, respectivamente.

- P_k: Probabilidade acumulada da k-ésima classe na distribuição de referência (avaliações humanas).
- Q_k: Probabilidade acumulada da k-ésima classe na distribuição predita pelo modelo.
- **CDF**(P_k) **e** **CDF**(Q_k): Funções de distribuição cumulativa (*Cumulative Distribution Function - CDF*) das distribuições P e Q, respectivamente. A CDF é usada para capturar a diferença entre as distribuições em cada ponto.

Fundamentação teórica

$$\mathrm{EMD}(P, Q) = \left(\frac{1}{K} \sum_{k=1}^{K} |\mathrm{CDF}(P_k) - \mathrm{CDF}(Q_k)|^d \right)^{1/d}$$

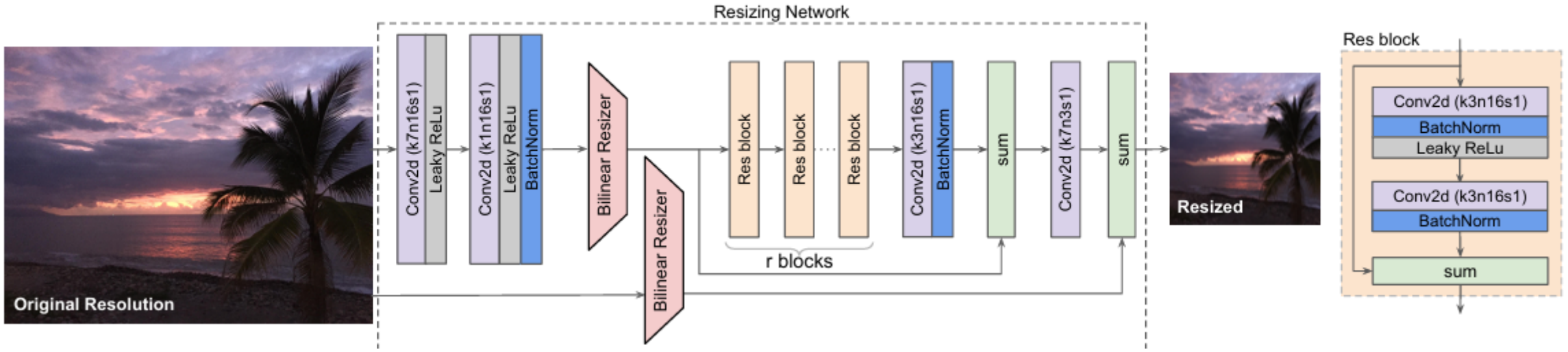
Earth Mover's Distance (EMD) (para IQA)

- K: Número total de classes ou bins na distribuição (e.g., $K = 10$ para o conjunto de dados AVA, onde as pontuações variam de 1 a 10).
- d: Parâmetro que controla a sensibilidade à diferença entre as distribuições. No artigo, $d = 2$ foi considerado o valor mais eficaz.
- $\frac{1}{K}$: Normalização para garantir que a distância seja calculada em média sobre todas as classes.

EMD permite que o **redimensionador (Figura 3)** adapte imagens para prever distribuições de qualidade melhorando PLCC (e.g., 0,662 para 0,686 com Inception-v2, **Tabela 1**). É mais robusta que regressão, alinhando-se com a percepção humana no AVA.

Arquitetura e funcionamento

A Figura 3 apresenta a **arquitetura da CNN** usada como redimensionador aprendido, que transforma imagens em resoluções otimizadas para tarefas como classificação e IQA. Seus componentes são:



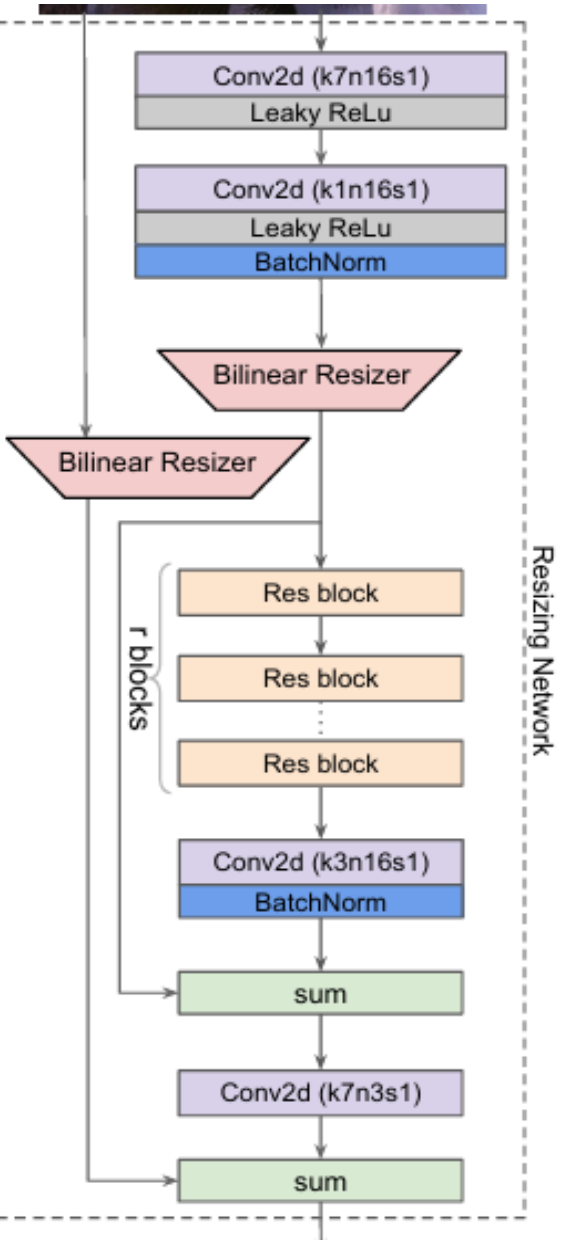
***Entrada**: Imagem em resolução original (e.g., 480 × 640). ***Convolução inicial**: Kernel 7×7 extrai características iniciais.

***Blocos residuais** (*r*=1 ou 2, *n*=16 filtros): Convoluções 3×3, normalização de batch e LeakyReLU.

***Redimensionamento bilinear**: Ajusta características para a resolução desejada (e.g., 192 × 256).

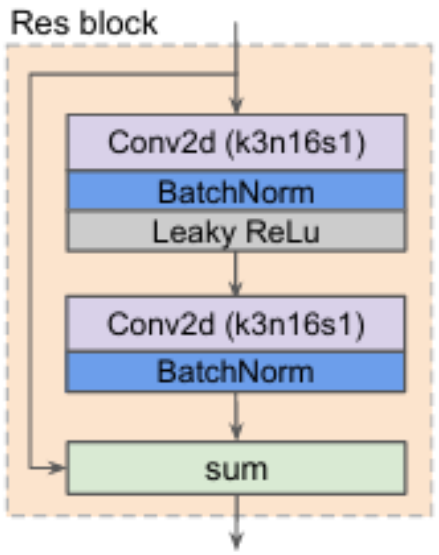
***Conexão de salto**: Soma a imagem bilinear à saída da CNN. ***Convolução final**: Kernel 7×7 gera a imagem redimensionada.

Arquitetura e funcionamento



Componente	Descrição	Propósito
Imagem Original	Entrada inicial ao modelo.	Fornecer os dados de entrada para o modelo.
Conv2d(k7n16s1)	Camada convolucional inicial.	Extrair características iniciais da imagem.
Bilinear Resizer 1	Redimensionador bilinear após BatchNorm.	Facilitar o fluxo de informações diretamente para Sum1.
Conexão Direta 1	Conexão paralela do Bilinear Resizer 1 ao Sum1.	Combinar características aprendidas com a imagem redimensionada bilinearmente.
Bloques Residuais (r blocks)	Blocos residuais para aprender características complexas.	Aprender características específicas para a tarefa.
Sum1	Combinação de características aprendidas e bilineares.	Integrar informações de diferentes fontes.
Conv2d(k7n3s1)	Camada convolucional final.	Gerar a imagem redimensionada final.
Bilinear Resizer 2	Redimensionador bilinear da imagem original.	Proporcionar uma base sólida para o redimensionamento.
Conexão Direta 2	Conexão paralela do Bilinear Resizer 2 ao Sum2.	Garantir que não se perca informação importante da imagem original.
Sum2	Combinação final das características.	Produzir a saída final do modelo.

Arquitetura e funcionamento



Componente	Descrição	Impacto no Modelo
Filtros (n)	Número de filtros nas camadas convolucionais.	Quanto maior o número de filtros (n), maior a capacidade de aprender características complexas, mas também aumenta o número total de parâmetros.
Blocos Residuais (r)	Número de blocos residuais no modelo.	Cada bloco residual contém duas camadas convolucionais. Quanto maior o número de blocos, maior a profundidade e capacidade do modelo, mas também aumenta a complexidade.
Configurações Chave	$n = 16, r = 1$: Modelo mais leve. $n = 32, r = 2$: Modelo mais pesado.	Ideal para tarefas com limitações de memória. Melhora o desempenho em troca de maior consumo computacional.
Número de Parâmetros	Varia entre 11.87K e 93.37K.	O modelo é significativamente mais leve que arquiteturas como ResNet-50 (23M parâmetros).

- O modelo proposto é relativamente **leve** e pode ser **facilmente integrado** em diferentes tarefas de visão computacional.

Arquitetura e funcionamento

A **Tabela 2** apresenta o número de **parâmetros treináveis (em milhares) do redimensionador**, variando o número de blocos residuais (r) e filtros (n):

Filters \ Blocks	$r = 1$	$r = 2$	$r = 3$	$r = 4$
n=16	11.87	16.48	21.08	25.69
n=32	38.08	56.51	74.94	93.37

Treinamento e otimização

A Tabela apresenta os **principais parâmetros** e configurações adotados nos experimentos. Esses elementos incluem framework de treinamento, otimizador, taxas de aprendizado, e a métrica utilizada para avaliação.

Parâmetro	Configuração
Framework	TensorFlow
Otimizador	Momentum (decaimento 0.9)
Taxa de aprendizado	0.05 ou 0.005
Decaimento da taxa	Exponencial (0.94 a cada 2 épocas)
Resoluções de entrada	224×224 a 448×448
Tamanho do lote	Ajustado para otimizar uso de memória
Inicialização	Aleatória (redimensionador), pré-treinada (modelos base)

Treinamento e otimização

```
import tensorflow as tf
from keras import layers
from keras.optimizers import SGD
LEARNING_RATE = 0.05
DECAY_STEPS = 2
DECAY_RATE = 0.94
INPUT_RESOLUTIONS = [(224, 224), (448, 448)]
optimizer = SGD(learning_rate=LEARNING_RATE, momentum=0.9)
# O Decaimento Exponencial pode ser implementado com callbacks.
lr_schedule = tf.keras.optimizers.schedules.ExponentialDecay(
    initial_learning_rate=LEARNING_RATE,
    decay_steps=DECAY_STEPS,
    decay_rate=DECAY_RATE
)
# Convolução inicial para extrair características da imagem de entrada.
    x = layers.Conv2D(filters=filters, kernel_size=7, strides=1, padding="same")(inputs)
    x = layers.LeakyReLU(0.2)(x)
 # Segunda convolução, seguida de normalização.
    x = layers.Conv2D(filters=filters, kernel_size=1, strides=1, padding="same")(x)
    x = layers.LeakyReLU(0.2)(x)
    x = layers.BatchNormalization()(x)
 # Convolução final para projetar as características refinadas de volta para o espaço da imagem.
    x = layers.Conv2D(
        filters=filters, kernel_size=3, strides=1, padding="same", use_bias=False
    )(x)
    x = layers.BatchNormalization()(x)
model.compile(
    loss=keras.losses.CategoricalCrossentropy(label_smoothing=0.1),
    optimizer="sgd",
    # `metrics`: Métrica de desempenho, neste caso, a acurácia.
    metrics=["accuracy"],)
```

Vantagens e desvantagens

- O redimensionamento aprendido reflete um avanço em relação aos métodos tradicionais, pois é projetado especificamente para **melhorar o desempenho em tarefas de visão computacional**, em vez de focar na qualidade perceptual da imagem.

Vantagens	Desvantagens
Melhoria no desempenho em tarefas de visão computacional (ex.: redução de erro top-1 de 26,7% para 24,0% em ImageNet).	Complexidade adicional no treinamento debido al entrenamiento conjunto con el modelo de base.
Flexibilidad para redimensionar a cualquier resolución objetivo, ajustándose a diferentes tareas.	Qualidade perceptual no es priorizada, resultando en imágenes menos atractivas visualmente.
Arquitectura ligera (11,87 mil parámetros para n=16, r=1) con ganancias significativas en rendimiento.	Dependencia del modelo base, exigiendo entrenamiento específico para cada arquitectura.
Adaptable a diferentes tareas, como clasificación y IQA, con ganancias consistentes.	Aumento en el costo computacional de inferencia (ex.: aumento de FLOPS de 3,88 para 5,07 billones).
Ideal para inferencia remota, minimizando la pérdida de rendimiento en imágenes redimensionadas.	—

Exemplo(s) de aplicação

Contexto	Tarefa de Visão Computacional	Impacto Esperado
Reconhecimento Facial em Sistemas de Segurança	Classificação de Identidades	Menos falsos positivos/negativos, maior confiabilidade em ambientes críticos
Diagnóstico Médico por Imagens	Classificação de Anomalias	Maior precisão na detecção de condições médicas, como tumores ou fraturas
Veículos Autônomos	Classificação de Objetos e Cenas	Decisões mais seguras, com menor erro em detecção de pedestres e sinais
Avaliação de Qualidade em Plataformas de Streaming	Avaliação de Qualidade de Imagem (IQA)	Avaliação mais precisa da qualidade de vídeos, melhorando a experiência do usuário
Monitoramento Ambiental por Drones	Classificação de Padrões Ambientais	Identificação mais precisa de áreas degradadas, apoiando ações de conservação

Comparação com outros algoritmos

Método	Desempenho em Tarefas	Qualidade Perceptual	Complexidade Computacional	Flexibilidade
CNN Resizer	Alto (ex.: erro top-1 de 24,0% vs. 26,7% com Inception-v2)	Baixa (não priorizada)	Moderada (11,87 mil parâmetros, 5,07B FLOPS)	Alta (resoluções arbitrárias)
Bilinear	Moderado (erro top-1 de 26,7%)	Moderada	Baixa (não treinável, 3,88B FLOPS)	Moderada (fixo para downscaling)
Bicúbica	Moderado (correlação de 0,642 em IQA)	Alta	Baixa (não treinável)	Moderada (fixo para downscaling)
Superresolução (SRResNet, EDSR)	Baixo para downscaling	Muito alta	Alta (milhões de parâmetros)	Baixa (otimizado para upscaling)
Pré-processamento com Perdas Perceptivas	Moderado a alto	Alta	Alta (treinamento separado)	Moderada (depende da tarefa)

Validação e Refutação

O método de redimensionamento aprendido é validado pelos resultados do artigo, que mostram **melhorias consistentes** no desempenho de tarefas de visão computacional (**Tabela 1**), como **redução de erro top-1** em classificação e aumento da correlação de Pearson em IQA. Sua flexibilidade para resoluções arbitrárias e sua arquitetura leve (Tabela 2) o tornam superior a métodos tradicionais e de superresolução em cenários com restrições computacionais e foco na precisão da tarefa. No entanto, a dependência de treinamento conjunto e a falta de priorização da qualidade perceptual podem limitar sua aplicação em contextos onde a estética visual é crucial, como interfaces de usuário. Assim, o método é validado para aplicações técnicas, mas pode ser refutado em cenários que exigem alta qualidade visual sem treinamento adicional.

QUIZ

QUIZ SEMINARIO

Referências

[1] H. Talebi e P. Milanfar, *Aprendendo a Redimensionar Imagens para Tarefas de Visão Computacional*. arXiv preprint arXiv:2103.09950, 2021.

[2] K. He, X. Zhang, S. Ren, e J. Sun, *Aprendizado residual profundo para reconhecimento de imagens*. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778, 2016.

[3] C. Ledig et al., *Super-resolução de imagem única fotorrealística usando uma rede adversária generativa*. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4681-4690, 2017.

[4] I. Goodfellow, Y. Bengio, e A. Courville, *Aprendizado Profundo*. MIT Press, 2016.

Obrigado!