RECONHECIMENTO FACIAL DE BUGIOS-RUIVO ATRAVÉS DE REDES NEURAIS CONVOLUCIONAIS

Aluno(a): Orlando Krause Junior

Orientadora: Andreza Sartori

FURB
UNIVERSIDADE DE BLUMENAU

Roteiro

- Introdução
- Objetivos
- Fundamentação teórica
- Trabalhos correlatos
- Requisitos
- Especificação
- Implementação
- Análise dos resultados
- Conclusões e sugestões

Introdução

- Preservação do bugio-ruivo
- Identificação de indivíduos
- Métodos invasivos

Objetivo geral

- Reconhecimento facial de bugios através de Redes Neurais Convolucionais.

Objetivos específicos

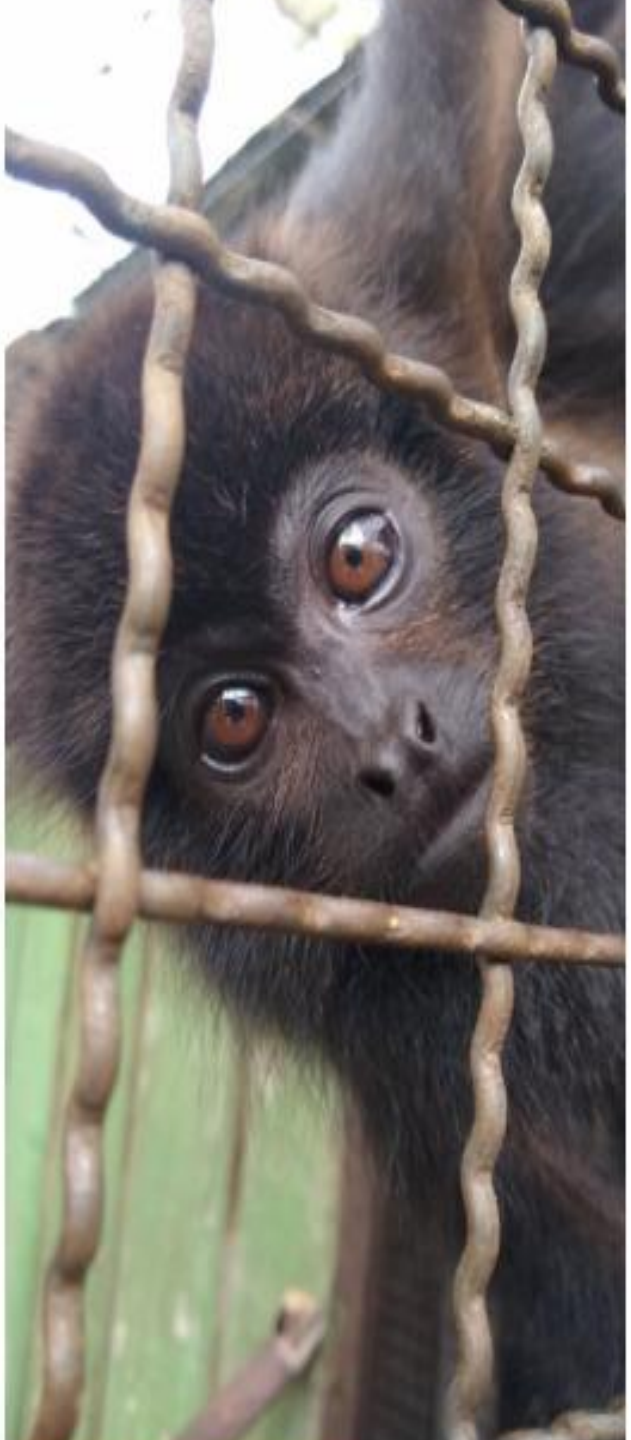
- Construir uma base de dados de imagens de bugios-ruivo em conjunto com os profissionais do Projeto Bugio-FURB;
- Extrair características relevantes utilizando os modelos de Redes Neurais Convolucionais Inception-ResNet v2, ResNet50 e Xception;
- Utilizar os classificadores KNN, rNN e SVM a fim de reconhecer os indivíduos a partir das características extraídas pela rede neural;
- Permitir One-Shot Learning, ou seja, aprender a classificar um indivíduo a partir de um único exemplo;
- Desenvolver aplicação com interface para realizar o reconhecimento facial.

Fundamentação Teórica

- Bugio-ruivo
- Projeto Bugio
- Rede Neural Artificial
- Rede Neural Convolucional
- One-Shot Learning

Bugio-ruivo

- *Alouatta Guariba Clamitans*
- Bugio-ruivo, barbado, guariba
- Comprimento entre a cabeça e o corpo de 45 a 58 centímetros
- Comprimento de cauda de 48 a 67 centímetros
- 4 a 7 kg
- Dimorfismo sexual
- Grupos
- Vivem nas árvores (10 a 20 metros)
- Se alimentam de folhas e frutos

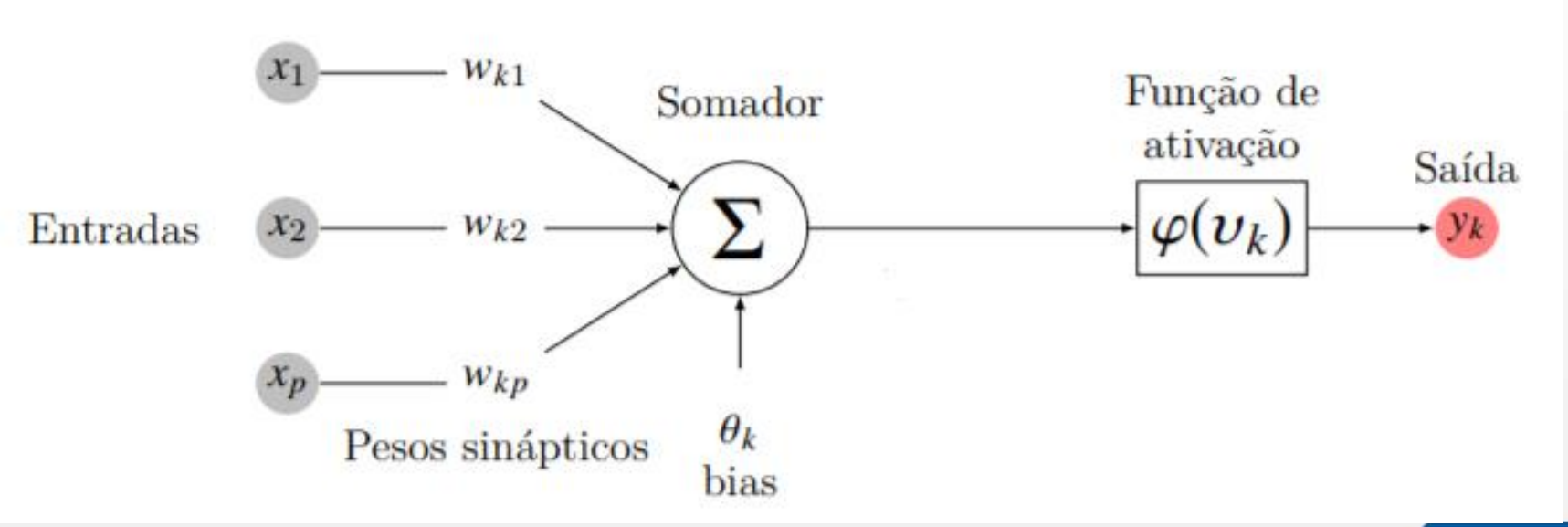


Projeto Bugio

- Criado em 1991 em parceria entre a FURB e o município de Indaial
- Preservação do bugio-ruivo
- Abrigo para os animais
- Capacitação de profissionais e estudantes
- Busca sensibilizar a comunidade

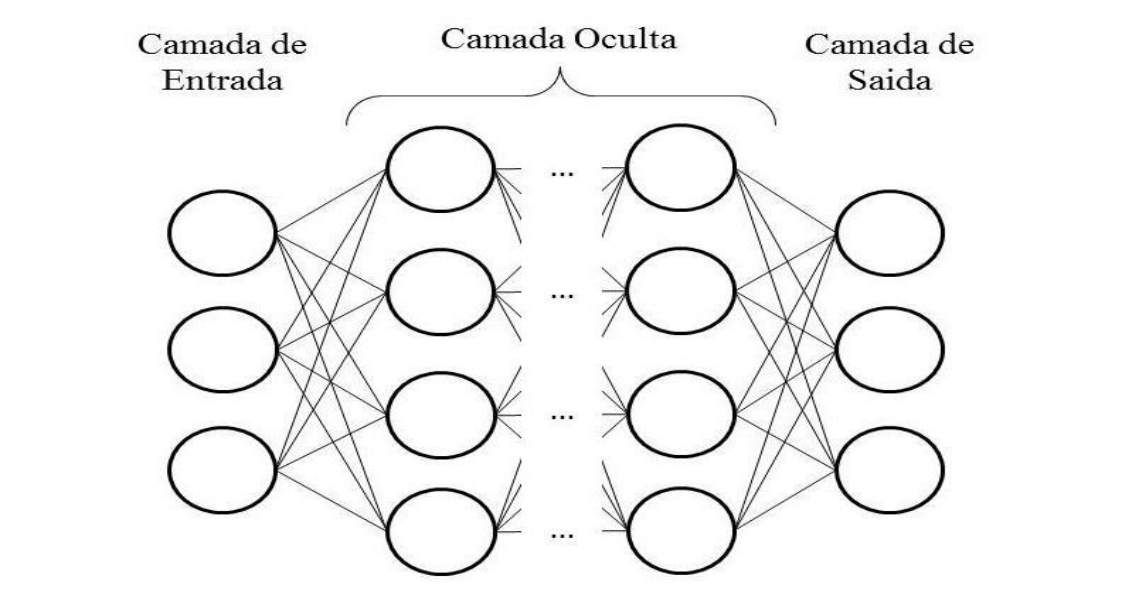
Rede Neural Artificial

- Inspirada no cérebro humano
- Neurônio artificial é a menor unidade de processamento



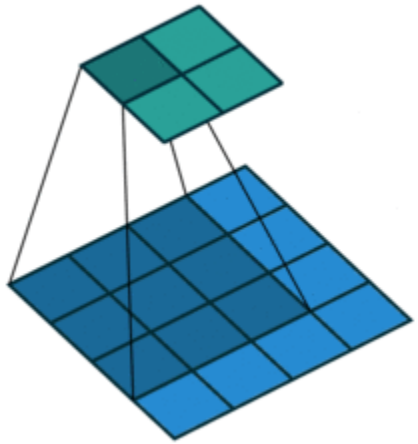
Rede Neural Artificial

- Camadas
- Feedforward ou Feedback
- Backpropagation



Rede Neural Convolucional

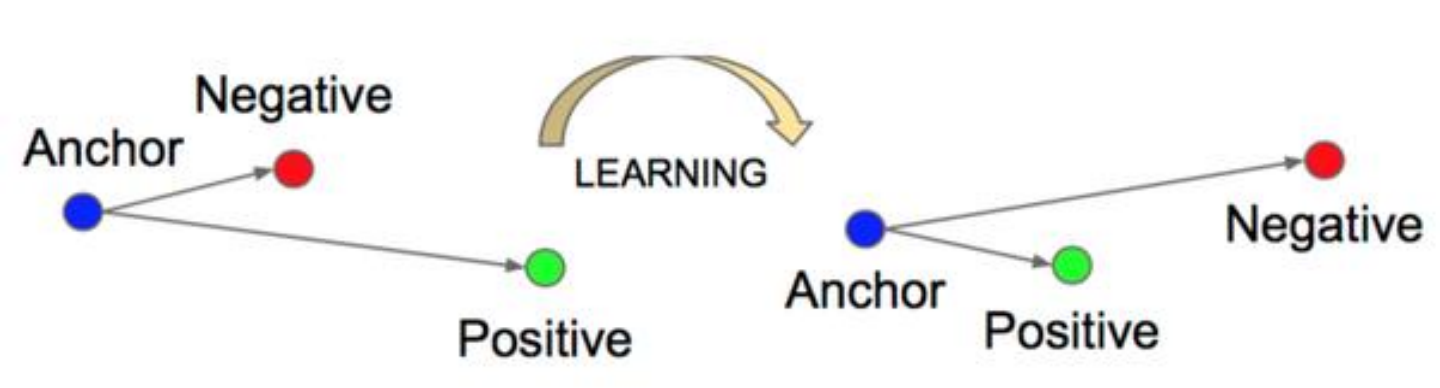
- Imagens
- Camadas com convolução
- Máscaras (filter/kernel)



- Modelos: Inception-ResNet v2, ResNet50, Xception

One-Shot Learning

- Aprendizado em um “tiro”
- Humanos aprendem de poucos exemplares
- Triplet Loss



Trabalhos Correlatos

- Crouse et al. (2017):
  - Reconhecimento facial de Lêmures; Técnicas manuais (MLBP e DoG) e distância;
  - 93,3% de precisão

Trabalhos Correlatos

- Loss e Ernst (2013):
  - Reconhecimento facial de chimpanzés;
  - Realiza localização da face
  - Técnicas manuais (Sobel, LBP, SURF e Gabor) e classificação com SRC e SVM
  - Duas bases (Zoo e Täi), 88,11% e 73,31%. Com localização de face 84,10% e 68,97%.

Trabalhos Correlatos

- Schofield et al. (2019):
  - Reconhecimento facial de chimpanzés;
  - Realiza localização e identificação de gênero;
  - Utiliza redes neurais, VGG-M para classificação (reconhecimento e gênero), VGG-16 com SSD para localização;
  - Acurácia 81% localização, 92,47% no reconhecimento e 96,16% gênero.

Trabalhos Correlatos

Características / Trabalhos	Crouse et al. (2017)	Loss, Ernst (2013)	Schofield et al. (2019)	Krause (2019)
Animal reconhecido	Lêmure	Chimpanzé	Chimpanzé	Bugio-ruivo
Detecção da face	Não	Sim (Sobel, LBP)	Sim (VGG-16 com SSD)	Não
Extração características	MLBP e DoG	Gabor e SURF	CNN (VGG-M)	CNN (Inception-ResNet v2, ResNet50 e Xception)
Classificador	Distância euclidiana	SRC e SVM	CNN (VGG-M)	kNN, rNN e SVM

Requisitos Funcionais

01 – Permitir envio de imagem pelo usuário.

02 – Extrair características relevantes da imagem informada.

03 – Realizar classificação da imagem baseada nas características extraídas.

04 – Exibir resultado da classificação para o usuário.

05 – Permitir ao usuário adicionar novo indivíduo a base de dados existente.

Requisitos Não Funcionais

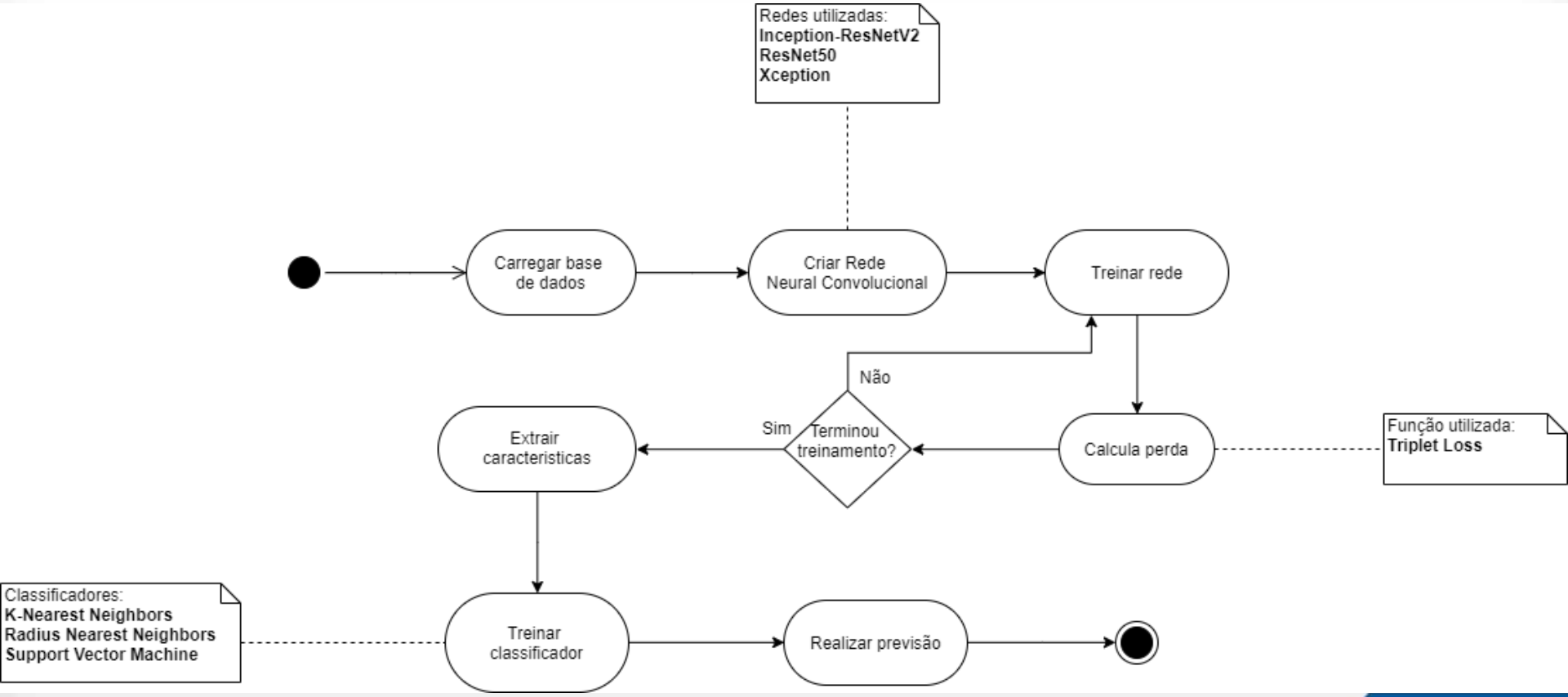
01 – A aplicação deve ser construída utilizando a linguagem Python.

02 – A aplicação deve utilizar o framework Keras para criação e treinamento do modelo.

03 – A extração de características deve ser feita utilizando Redes Neurais Convolucionais.

04 – Utilizar algoritmos k-NN, SVM e r-NN para classificação do indivíduo

Especificação

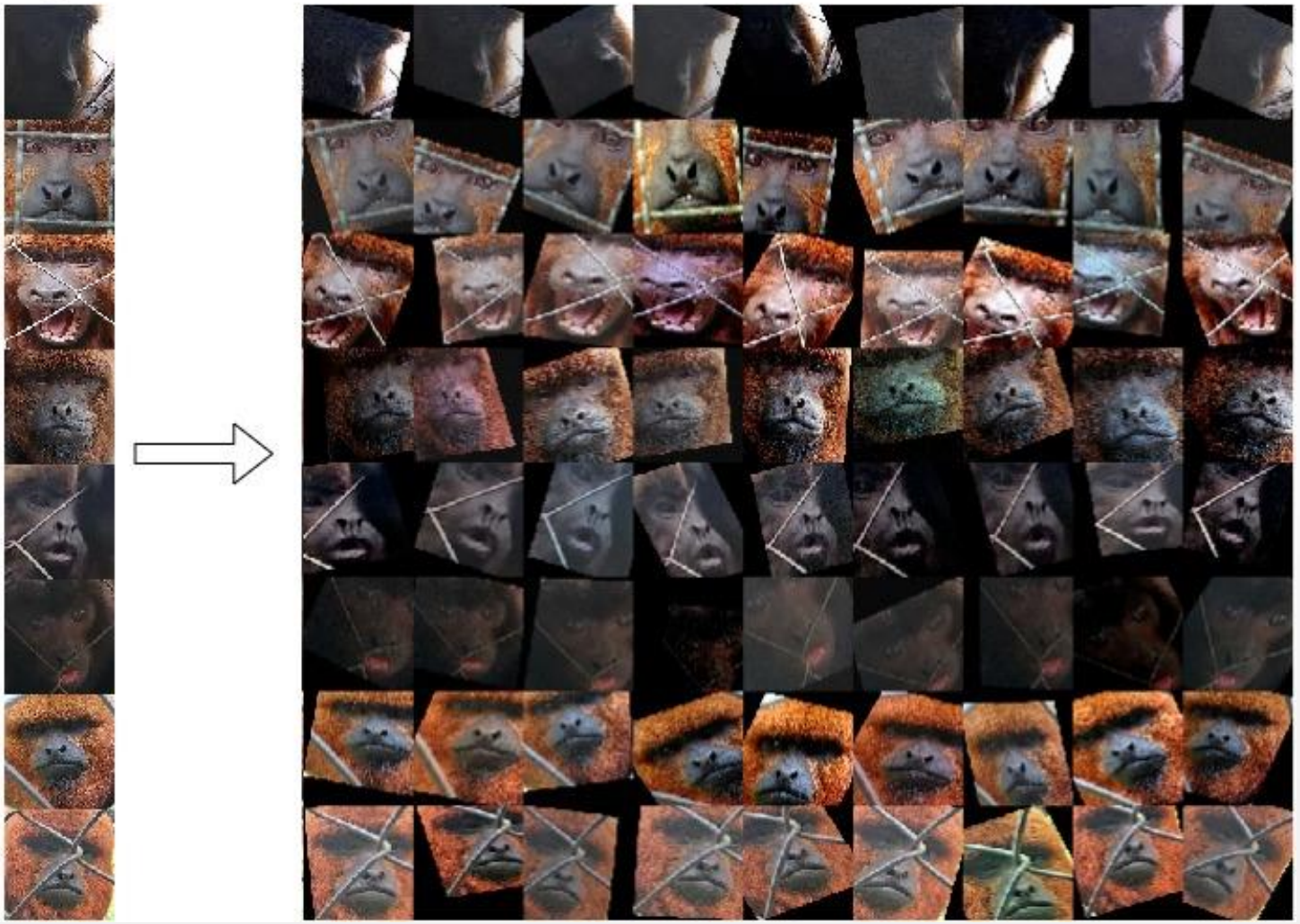


Implementação

- Base de dados:
  - Coletada junto ao Projeto Bugio;
  - 179 fotos de 20 bugios diferentes;
  - Recortadas manualmente para conter somente a face do bugio.
  - Data augmentation, mais 9 imagens cada. Total 1790 fotos.

Imagem Original

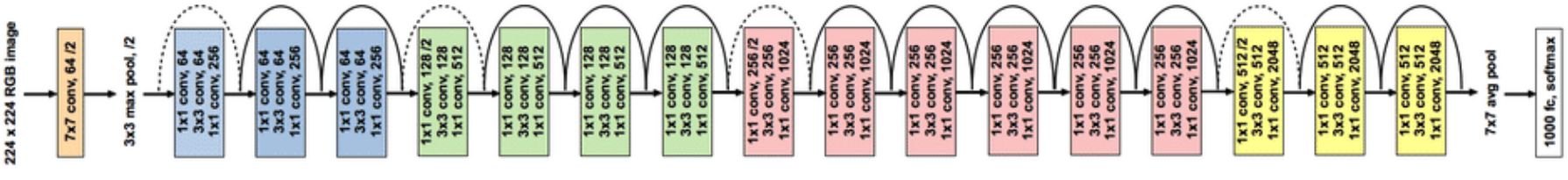
Imagens Geradas

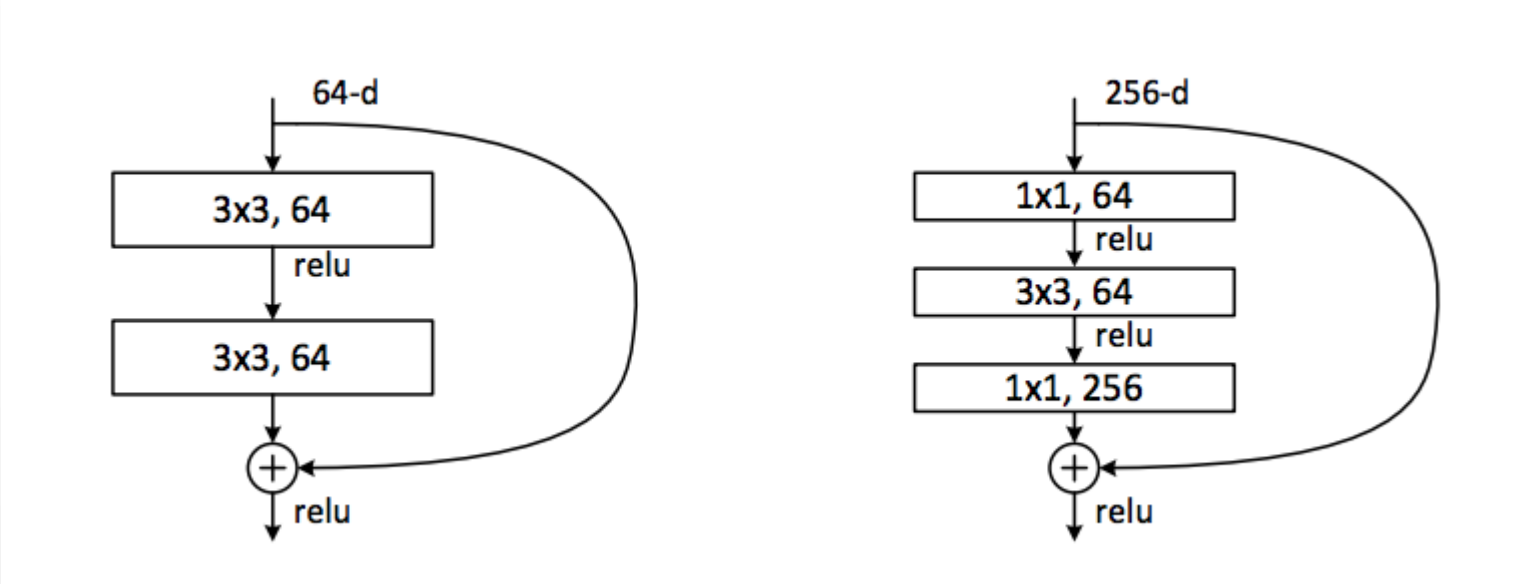


Implementação

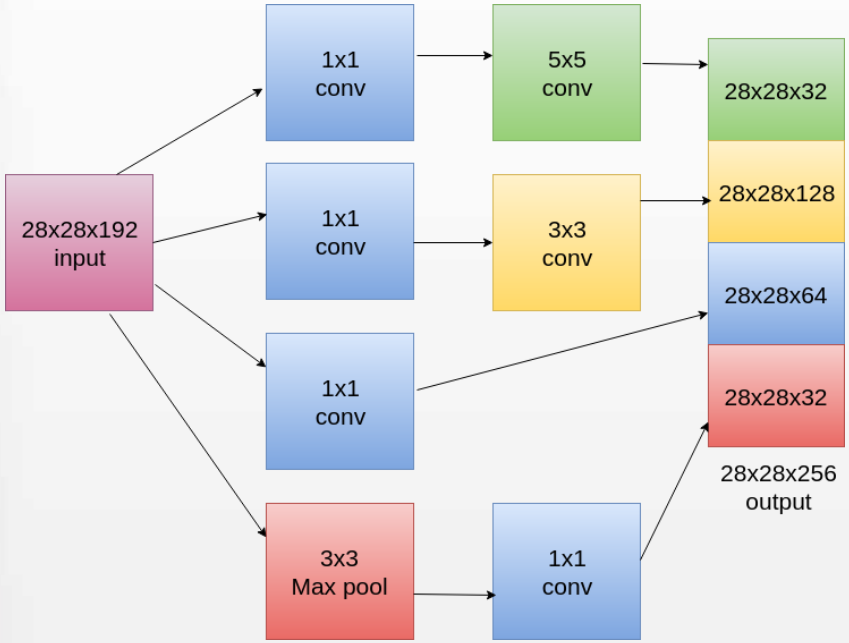
- Arquitetura:
  - Modelos: **Inception-ResNet v2** (SZEGEDY et al., 2017), **ResNet50** (HE et al., 2016) e **Xception** (CHOLETT, 2017);
  - Remoção de camadas de classificação;
  - Adicionado camada densa de 64 neurônios;
  - Função de perda Triplet Loss.

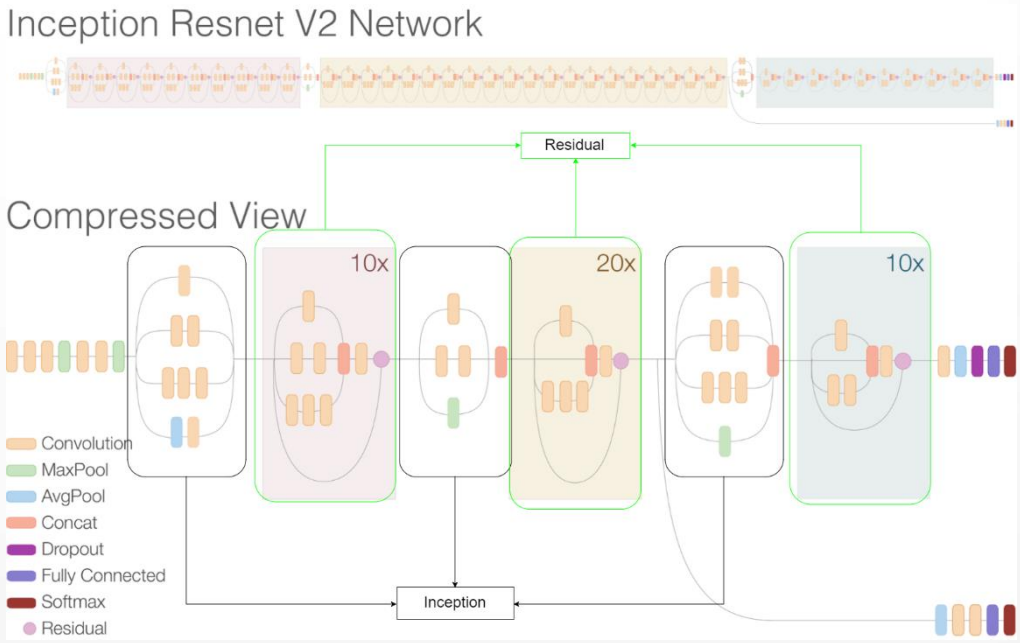
ResNet50


224 x 224 RGB image
7x7 conv, 64 /2
3x3 max pool, /2
1x1 conv, 64
3x3 conv, 64
1x1 conv, 256
1x1 conv, 128 /2
3x3 conv, 128
1x1 conv, 512
1x1 conv, 256 /2
3x3 conv, 256
1x1 conv, 1024
1x1 conv, 512 /2
3x3 conv, 512
1x1 conv, 2048
7x7 avg pool
1000 fc, softmax

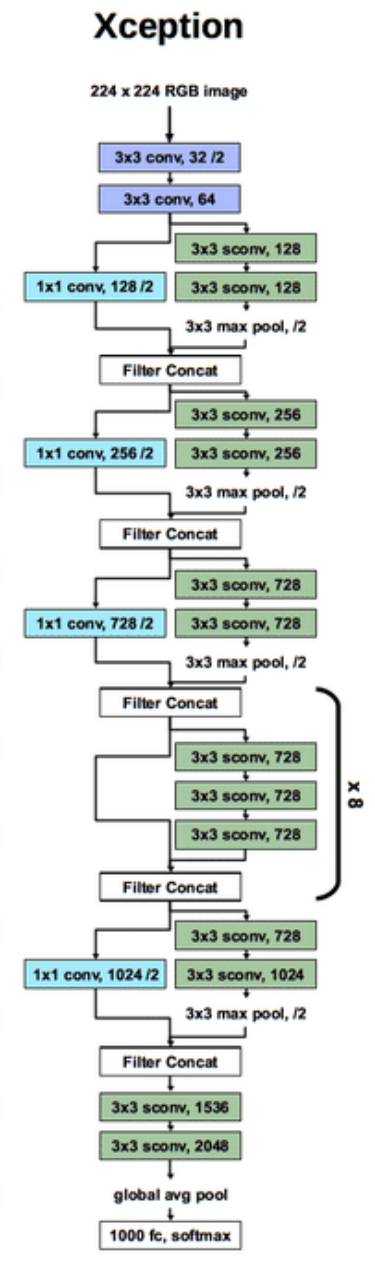

64-d
3x3, 64
relu
3x3, 64
relu
256-d
1x1, 64
relu
3x3, 64
relu
1x1, 256
relu

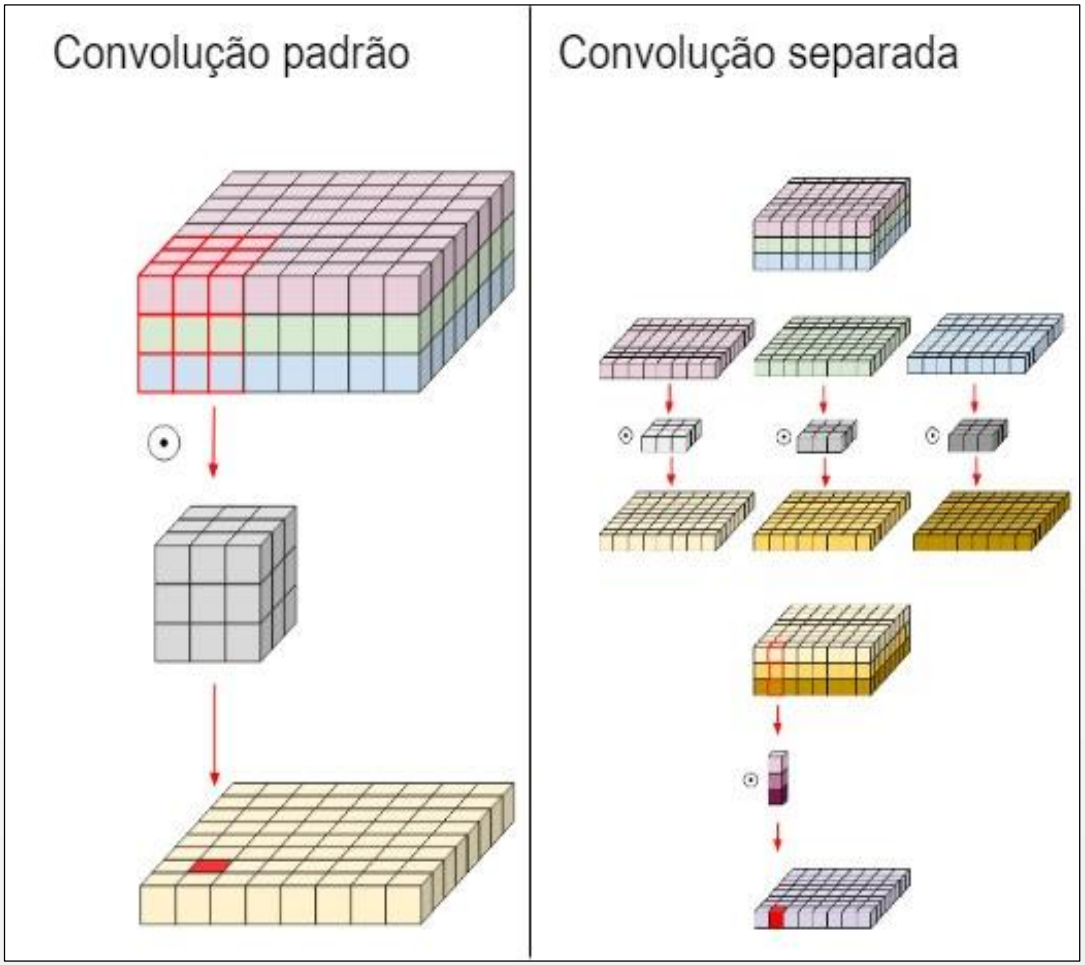
Inception-ResNet v2





Xception


Xception
224 x 224 RGB image
3x3 conv, 32 /2
3x3 conv, 64
3x3 sconv, 128
1x1 conv, 128 /2
3x3 sconv, 128
3x3 max pool, /2
Filter Concat
3x3 sconv, 256
1x1 conv, 256 /2
3x3 sconv, 256
3x3 max pool, /2
Filter Concat
3x3 sconv, 728
1x1 conv, 728 /2
3x3 sconv, 728
3x3 max pool, /2
Filter Concat
3x3 sconv, 728
3x3 sconv, 728
3x3 sconv, 728
Filter Concat
x 8
3x3 sconv, 728
1x1 conv, 1024 /2
3x3 sconv, 1024
3x3 max pool, /2
Filter Concat
3x3 sconv, 1536
3x3 sconv, 2048
global avg pool
1000 fc, softmax

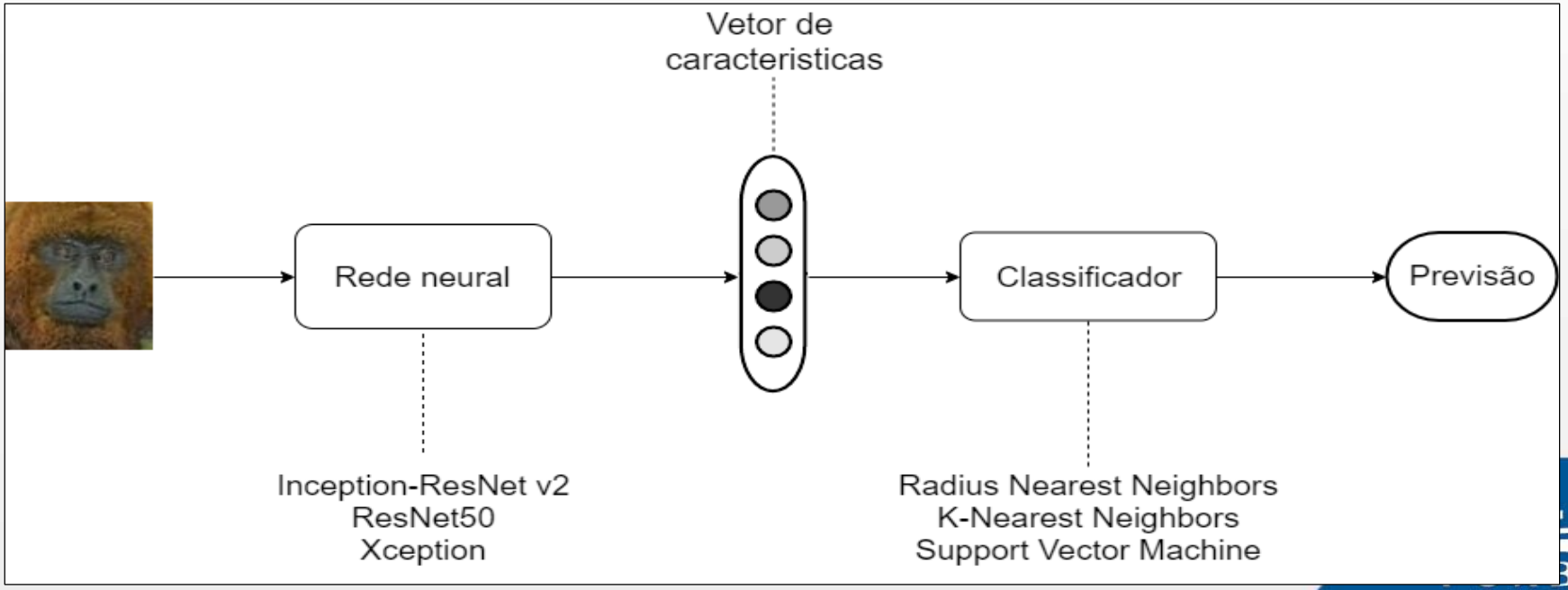

Convolução padrão
Convolução separada

Implementação

- Treinamento:
  - 20 épocas, 40 lotes por época e 64 imagens por lote;
  - Adam com lr=0,001;
  - Gerador de lotes, classes por lote e imagens por classe.

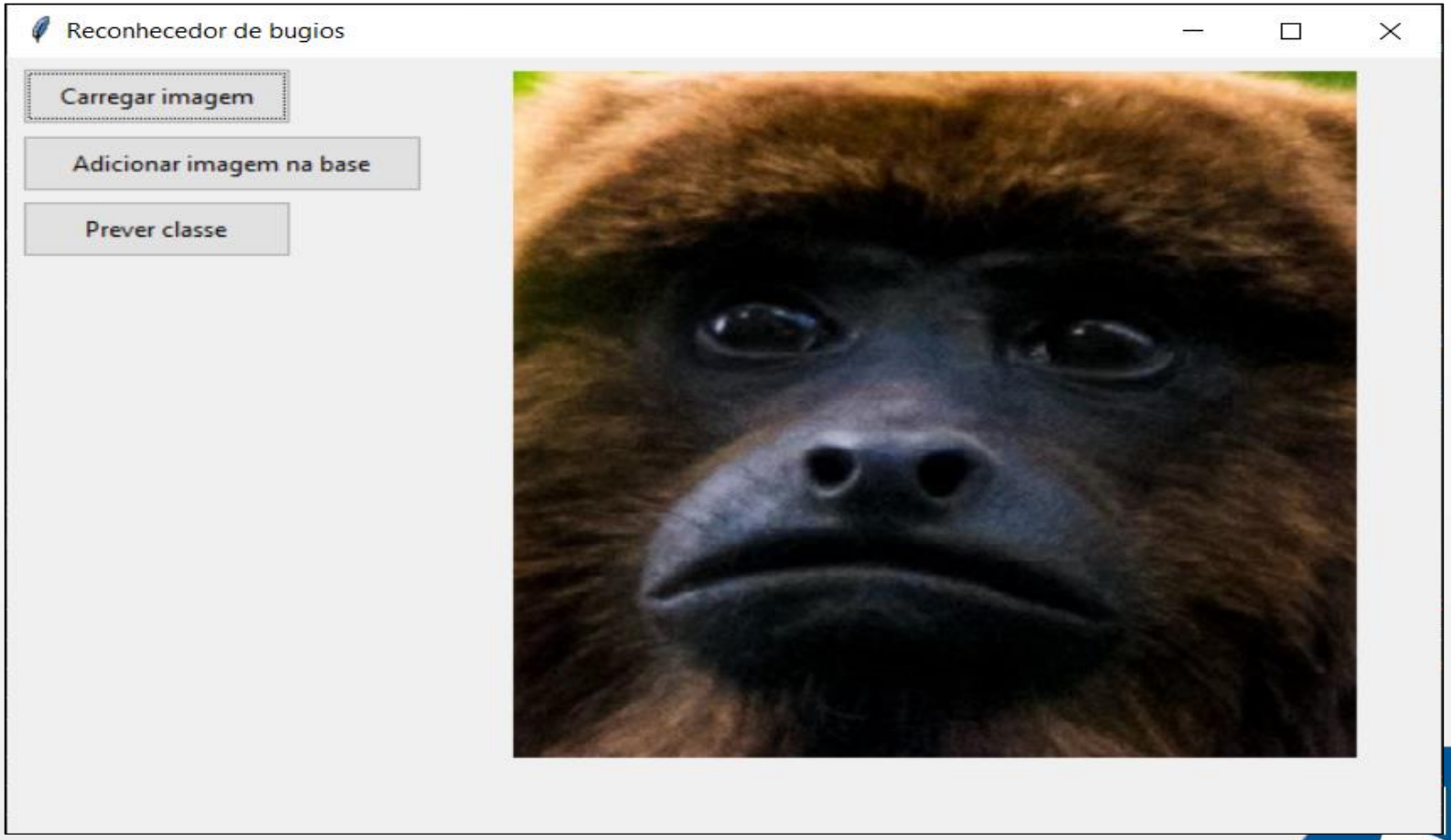
Implementação

- Inferência:
  - **k-Nearest Neighbors** (k-NN);
  - **Support Vector Machine** (SVM);
  - **Radius Nearest Neighbors** (r-NN);

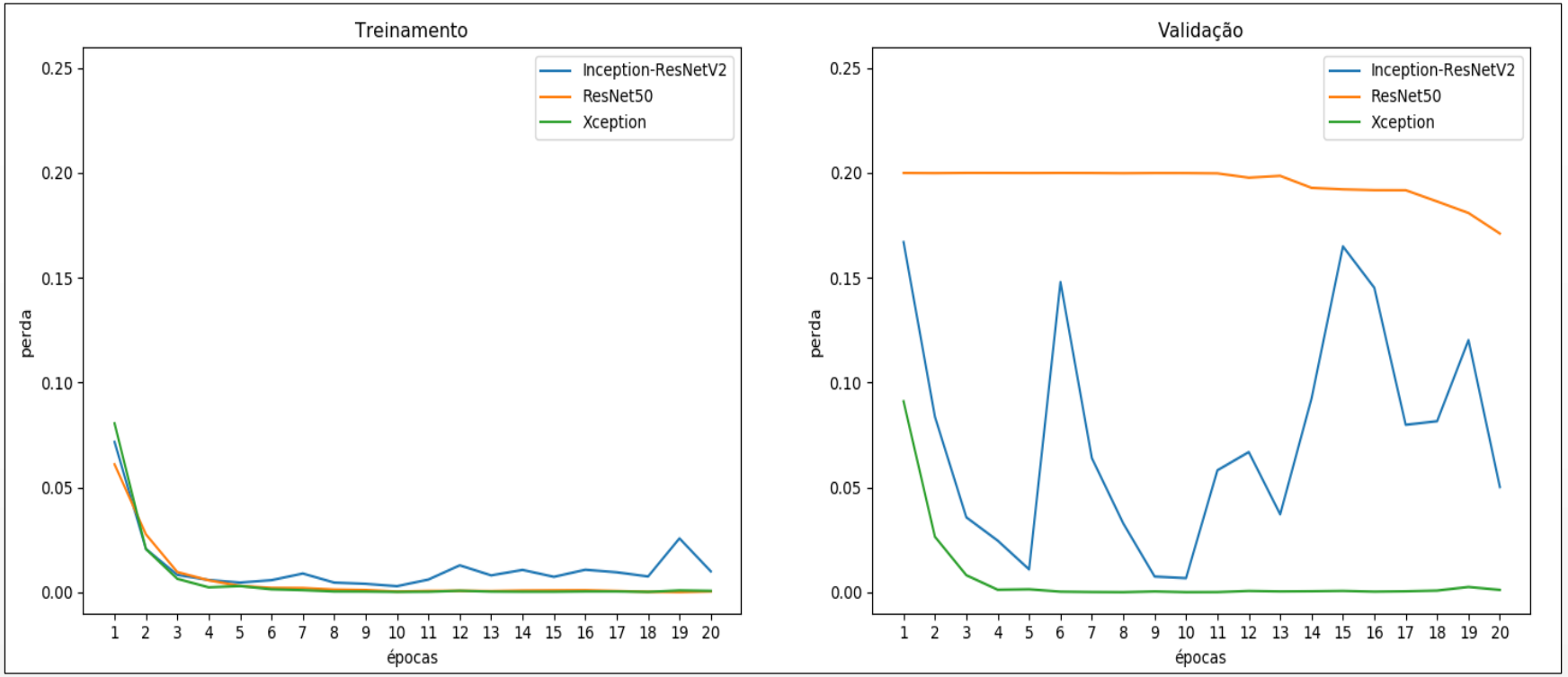


Implementação

- Aplicação desenvolvida:



Análise dos Resultados

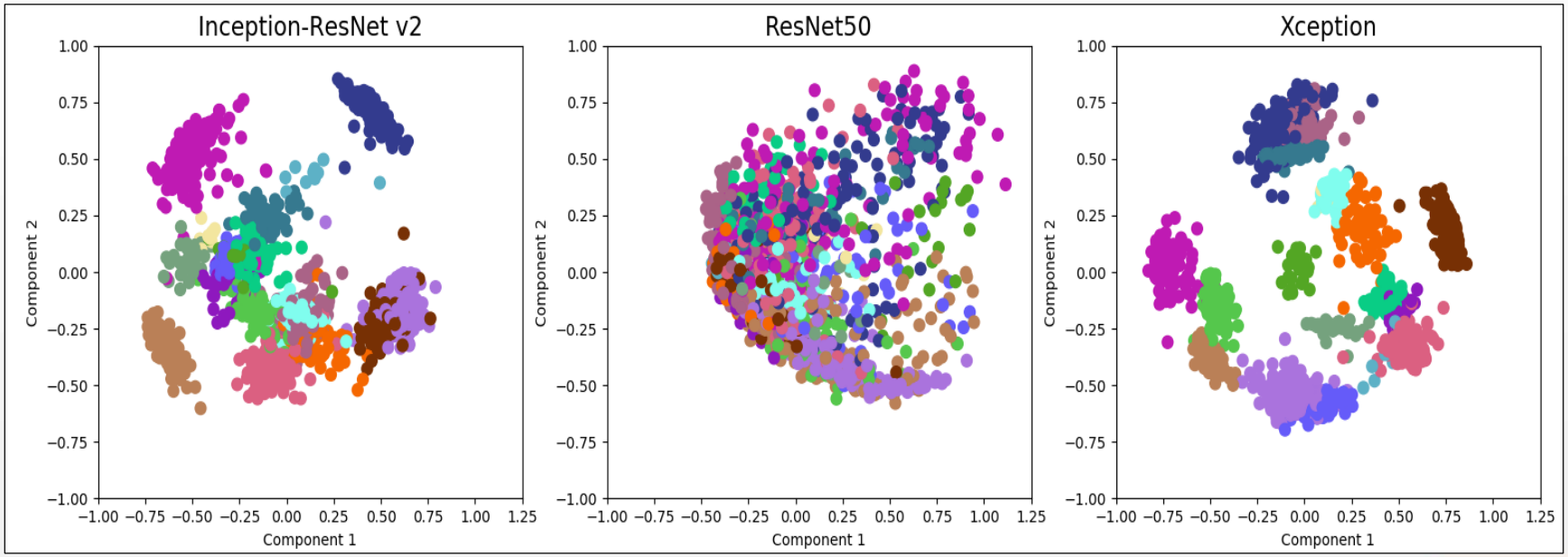


Análise de resultados

Redes	Treinamento (Menor valor de perda)	Validação (Menor valor de perda)
Inception-ResNet v2	0,002845 na época 10	0,006672 na época 10
ResNet50	**0,000022 na época 19**	0,171088 na época 20
Xception	0,000136 na época 10	**0,000003 na época 8**

Análise de resultados

- PCA nas características extraídas



Análise de resultados

- Redes x Classificadores

Redes	Pesos treináveis	r-NN, r=0.4	r-NN, r=0.5	r-NN, r=0.7	k-NN, k=5	SVM
Inception-ResNet v2	54374560	98,55%	99,39%	99,72%	**99,78%**	99,72%
ResNet50	24714304	53,46%	50,67%	46,76%	70,0%	**75,98%**
Xception	21986664	99,83%	99,89%	99,78%	**99,94%**	99,94%

Conclusões

- Atingiu o objetivo proposto.
- Melhor resultado com a rede Xception e classificador kNN, atingindo 99,94% de acurácia.
- Foi possível treinar uma nova classe a partir de um único exemplo utilizando o classificador rNN.
- A geração de lotes foi importante para a função Triplet Loss.

Sugestões

- Realizar localização da face.
- Adicionar informações do bugio.
- Aplicação mobile
- Modelar arquitetura específica.

Agradecimentos

- Professor Júlio
- Colaboradores do Projeto Bugio.