UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
CAMPUS DE GUARATINGUETÁ

GIOVANA RAMON

Aplicação de Machine Learning para Classificação de Imagens Astronômicas

Guaratinguetá
2023

Giovana Ramon

Aplicação de Machine Learning para Classificação de Imagens Astronômicas

Trabalho de Graduação apresentado ao Conselho de Curso de Graduação em Física Bacharelado da Faculdade de Engenharia do Campus de Guaratinguetá, Universidade Estatual Paulista, como parte dos requisitos para obtenção do diploma de Graduação em Física Bacharelado .

Orientador: Prof. Dr. Rafael Sfair
Coorientador: Prof. Dr. Altair Gomes

Guaratinguetá
2023

Luiz Neto, Giovana Ramon
L953a Aplicação de Machine Learning para classificação de imagens astronômicas / Giovana Ramon Luiz Neto – Guaratinguetá, 2023.
37 f : il.
Bibliografia: f. 36-37

Trabalho de Graduação em Física (Bacharelado) – Universidade Estadual Paulista, Faculdade de Engenharia e Ciências de Guaratinguetá, 2023.
Orientador: Prof. Dr. Rafael Sfair de Oliveira
Coorientador: Prof. Dr. Altair Gomes

1. Astronomia. 2. Redes neurais (Computação). 3. Aprendizado de máquinas. I. Título.

CDU 52

Luciana Máximo
Bibliotecária/CRB-8 3595

UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
CAMPUS DE GUARATINGUETÁ

GIOVANA RAMON

ESTE TRABALHO DE GRADUAÇÃO FOI JULGADO ADEQUADO COMO PARTE DO REQUISITO PARA A OBTENÇÃO DO DIPLOMA DE **"GRADUANDO EM FÍSICA "**

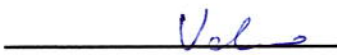
APROVADO EM SUA FORMA FINAL PELO CONSELHO DE CURSO DE GRADUAÇÃO EM FÍSICA

Prof. Dr. MARCO AURÉLIO ALVARENGA MONTEIRO
Coordenador

BANCA EXAMINADORA:

Prof. Dr. Rafael Sfair
Orientador/UNESP-FEG

Profa. Dra. Daniela Cardozo Mourão
UNESP-FEG



Prof. Dr. Valerio Carruba
UNESP-FEG

Fevereiro , 2023

DADOS CURRICULARES

GIOVANA RAMON

NASCIMENTO	17/05/2001 - Taubaté / SP
FILIAÇÃO	Antonio Luiz Neto Cristiane Ramon de Carvalho Luiz Neto
2019 / 2022	Graduação em bacharelado em Física Faculdade de Engenharia e Ciências de Guaratinguetá

AGRADECIMENTOS

Agradeço à minha família que desde cedo fomenta meu interesse pela ciência e apoia minha decisão ao escolher física como profissão. Por estar ao meu lado em todos os momentos difíceis e me ajudar a seguir em frente.

Sou grata a todos os professores que transpuseram meu caminho até aqui, pelos conhecimentos passados e a motivação de nunca desistir do curso ou de uma matéria. Em especial agradeço meu orientador Prof. Dr. Rafael Sfair, que me acompanhou durante a iniciação científica e a escrita do trabalho de conclusão, sou grata por toda atenção, incentivo e aprendizado.

À Rep. Tcheca que foi minha casa nesses quatro anos e permitiu que meus dias fossem mais leves e alegres. Especialmente a Iris, Thaís e Giovanna que se tornaram minhas verdadeiras irmãs, estando sempre ao meu lado nesse ciclo e a quem tenho certeza que levarei a amizade para vida.

Este trabalho contou com o apoio da(s) seguinte(s) entidade(s):
CAPES - Coordenação de Aperfeiçoamento de Pessoa de Nível Superior
CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico
FAPESP - Fundação de Amparo à Pesquisa do Estado de São Paulo - Proc: 2016/24561-0

"Tudo o que temos de decidir é o que fazer com o tempo que nos é dado."
(J. R. R. Tolkien)

RESUMO

Atualmente na Astronomia é crescente a quantidade de imagens coletadas. O objetivo principal desse projeto foi implementar e analisar métodos de *Machine Learning* para classificação de imagens geradas em problemas de dinâmica orbital. O ML é uma área que busca a automatização de processos a partir de métodos computacionais, utilizando a experiência para melhorar o desempenho ou para fazer previsões precisas. Dentre os métodos de ML foi escolhido a CNN (*Convolutional Neural Network*), que tem como objetivo a análise e classificação de imagens.
Para verificar a exequibilidade da proposta foi realizada uma análise preliminar utilizando dados orbitais da evolução dinâmica de NEAs (*Near Earth Asteroids*), cujo modelo de ML foi construído utilizando a biblioteca *keras* de redes neurais em Python.
Nesse caso, o intuito foi classificar gráficos que representavam a excentricidade em relação ao semi-eixo maior da órbita dos objetos. Partimos de um conjunto com 429 gráficos e realizamos testes agrupando as imagens em duas ou três categorias. Para os casos com duas categorias foi possível obter uma acurácia superior a 0,9 usando 30% das imagens como etapa de treino, podendo chegar a 0,98 caso o conjunto de treino corresponda a 50% da amostra; no caso com três classes a acurácia foi inferior (menor que 0,86). Este último resultado está relacionado a similaridade entre duas das classes, dificultando a separação entre elas.
Também realizamos a classificação de gráficos da evolução do ângulo ressonante no tempo das partículas do arco do anel G de Saturno. Nesse caso o conjunto foi amplo, contendo 18 mil imagens. Foram desenvolvidos dois tipos de cenário, com duas classes (libração e circulação) e três classes (libração, alternado e circulação). Alcançamos uma acurácia de até 0,99.
O desempenho variou conforme a quantia de épocas (número de vezes em que as camadas são aplicadas), quanto maior a época maior o desempenho, havendo um limite de parada para o aprendizado.

PALAVRAS-CHAVE: redes neurais; astronomia; aprendizado de máquina.

ABSTRACT

Nowadays in astronomy, it is increasing the number of produced images. The main objective of this project was to implement and analyze Machine Learning methods to classify images generated in orbit dynamic problems. ML is an area that seeks to automate processes through computational methods, using the experience to improve performance or to make accurate predictions. Among the methods of ML, we chose the CNN (Convolutional Neural Network) which has the objective of analyzing and classifying images.

In order to verify the proposed feasibility, we made a preliminary analysis using dynamic evolution orbital data of NEAS (Near Earth Asteroids), whose model of ML was built using the Keras library of neural networks in Python.

In this case, our intention was to classify graphics that represented the eccentricity versus the semi-major axis of the object's orbit. We started from a set with 429 graphics and performed tests grouping images into two or three categories. For the cases with two categories, it was possible to obtain an accuracy higher than 0.9 using 30% of the images as a training set, where the model could reach 0.98 accuracy if the training set corresponded to 50% of the sample: in the case with three classes the accuracy was inferior (less than 0.86). This last result is related to similarities between the two classes, making them difficult to separate.

We also perform the classification of graphics of the time evolution of the resonant angle of Saturn's G ring particles. In this case, the set was large, containing 18 thousand images. Two scenarios were developed, with two classes (libration and circulation) and with three classes (libration, alternating, and circulation). We reach an accuracy of up to 0.99.

The performance varied according to the number of epochs (number of times that the layers are applied), the higher the epoch, the higher the performance, having a stop limit for learning.

KEYWORDS: neural network; astronomy; machine learning.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Nos últimos anos o volume de dados trabalhados pela ciência tem crescido cada vez mais. Em particular, a Astronomia lida com uma quantidade imensa de imagens tanto no registro de objetos e fenômenos, quanto na geração de gráficos e simulações com as informações coletadas. Como exemplo podemos citar o *Panoramic Survey Telescope and Rapid Response System* (PAN-STARRS) que produz aproximadamente 500 exposições por noite, correspondendo a ∼750 bilhões de pixeís de dados de imagem (MAGNIER et al., 2020) e o *Large Synoptic Survey Telescope* (LSST) que promete uma carga enorme de dados, gerando a cada noite de funcionamento aproximadamente 15 TB.

Analisar manualmente esse material é uma tarefa dispendiosa e muitas vezes impraticável. Isso levou a um amplo desenvolvimento de tecnologias responsáveis por automatizar esse tipo de ação. Nesse cenário a Ciência de Dados ganhou reconhecimento, com destaque para a utilização do Aprendizado de Máquina (*Machine Learning*).

Este trabalho visa discutir brevemente os conceitos de *Machine learning* e redes neurais, introduzindo a arquitetura da rede neural convolucional e o funcionamento de sua aplicação básica. Assim, será estudado a atuação de um modelo de CNN (*Convolutional Neural Network*) sobre gráficos gerados com dados de NEAS e ângulos ressonantes. Apresentamos os resultados obtidos no Capítulo 3 e, por conseguinte, a conclusão que alcançamos com eles.

1.1 MACHINE LEARNING

"O *Machine Learning* é uma área de estudo da computação que busca a automatização de processos, sendo caracterizada a partir de métodos computacionais que usam a experiência para melhorar o desempenho ou para fazer previsões precisas"(MOHRI et al., 2012). Existem três grandes áreas dentro do *Machine Learning*, sendo elas aprendizado supervisionado, aprendizado não-supervisionado e aprendizado por reforço. Além disso, podem ser encontrados métodos tais como árvores de decisão (*decision trees*), máquina de vetores de suporte (*support vector machines*) e redes neurais (*neural networks*).

Apesar do amplo desenvolvimento de técnicas de *Machine Learning* em áreas da astronomia como astronomia solar, estrelas variáveis, galáxias, entre outros, em dinâmica orbital ainda trata-se de uma aplicação emergente. Como é discutido no estudo de Carruba et al. (2022b) nos últimos três anos apenas 12 artigos abordaram a ação desses métodos em dinâmica de asteroides.

Usualmente análises de resultados gráficos gerados em problemas de ângulos ressonantes e classificação orbital são feitas por inspeção visual, entretanto com a produção da ordem de milhares de imagens busca-se opções que automatizem o processo. Dentre as redes neurais artificiais, as CNNs (*Convolutional Neural Network*) destacam-se na análise de imagens e vídeos e serão utilizadas neste projeto. Tratando-se de um método supervisionado, aprende a partir de resultados pré-definidos, ou seja, ele utiliza valores da variável de entrada para aprender quais

devem ser seus resultados de saída. Estes mesmos valores servem como “supervisão” destas previsões, permitindo o ajuste com base nos erros.

As CNNs são compostas por três tipos de camadas: as camadas de entrada, as camadas "ocultas"e as camadas de saída. Elas buscam e aprendem padrões das imagens de entrada por meio do processo de convolução, sendo as camadas ocultas responsáveis por essa função. Esse processo é finalizado na camada de saída onde o resultado é concluído e apresentado.

1.2 REDES NEURAIS ARTIFICIAIS

As redes neurais artificias fazem uma mímica do funcionamento dos neurônios biológicos. Segundo Haykin (2007), as redes se assemelham ao cérebro humano pelo processo de aprendizado adquirido de acordo com a experiência e pela força de conexão entre os neurônios, que são células computacionais nesse caso, responsáveis por armazenar a informação processada.

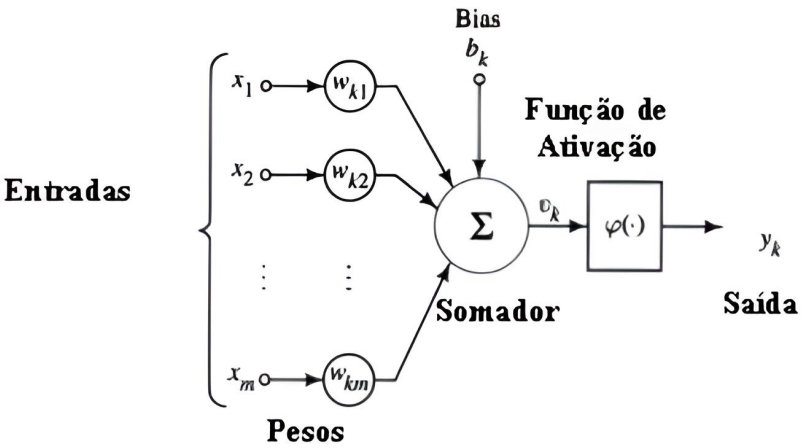
As Redes Neurais Artificiais (RNA) são capazes de entender padrões e extrair características dos dados automaticamente a partir de um treinamento. Elas são compostas por neurônios artificiais denominados *perceptrons*, que são passíveis de entender padrões e extrair características de dados sozinhos, ao passo que simulam o funcionamento do neurônio biológico.

Os dados são introduzidos como sinais $x = [x_1, x_2, x_3, ..., x_N]$ que sofrerão conexões sinápticas e passarão por uma combinação linear caracterizada pela multiplicação por pesos sinápticos $[w = w_1, w_2, w_3, ..., w_N]$, gerando assim a saída y:

$$y = \sum_{j=1}^{d} w_j x_j + b \tag{1}$$

O treinamento realizado determinará valores para w e b, sendo b o viés responsável por adicionar um grau de liberdade a mais. Nesse processo, os pesos e vieses serão ajustados e atualizados para melhor extrair os padrões dos dados analisados (Figura 1). O processo de atualização dos pesos é conhecido como época e representa uma passagem nova dos dados de entrada pela rede neural.

Figura 1 – Representação dos elementos de um neurônio artificial.



Fonte: GSIGMA (2023).

Com a eq.1 é gerado um potencial de ativação Z que passa por uma função de ativação não linear. Da mesma forma que um potencial de ação em um neurônio biológico, que decide a taxa de disparo de neurônios, a função de ativação em um neurônio artificial restringe a saída do neurônio a valores normalizáveis. A função mais utilizada é a *Softmax* (para problemas de múltiplas classes) ou *Sigmoid* (problemas binários).

Nesse projeto utilizamos as funções *Rectified Linear Unit* (ReLU) e *Softmax*, a função ReLU é não linear e retorna a saída 0 para todas as entradas negativas e $f(x)$ para entradas positivas,

$$f(x) = max(0, x) \tag{2}$$

já a função *Softmax*, utilizada em redes neurais de classificação, força a saída a representar a probabilidade dos dados de compor uma das classes definidas:

$$\sigma(y)_j = \frac{e^{y_j}}{\sum_{k=1}^{K} e^{y_k}}; \quad \text{para j = 1,...,K;} \tag{3}$$

normalmente é à última camada que resultará na classificação.

Com a resposta obtida pela rede, uma perda é calculada conforme a diferença entre o valor predito e o real esperado. Diferenças significativas próximas a 1 recebem uma pontuação grande, enquanto diferenças próximas a 0 recebem uma pequena pontuação. A função de perda de entropia cruzada (*cross entropy loss*) é definida como:

$$L_{CE} = -\sum_{i=1}^{n} t_i log(p_i) \tag{4}$$

sendo t_i o rótulo real e p_i a probabilidade da i-ésima classe.

1.3 REDES NEURAIS CONVOLUCIONAIS

A principal etapa das CNNs é refletida na convolução, responsável por extrair características das imagens. Esse processo é realizado com a aplicação de máscaras nos dados de entrada, o que dá origem a filtros de convolução que serão empregados para obter os mapas de características.

Em uma CNN, o produto escalar na eq.1 é substituído por um operador de convolução. Portanto, w representará um vetor, que pode ser chamado de kernel ou filtro. Segundo Arbib (2003), isso facilita a operação sobre dados brutos, como imagens ou dados de séries temporais, ao contrário do que ocorre com vetores em redes neurais normais.

Os filtros possuem formato de matrizes compostas por pesos de conexão entre os neurônios, assim percorrem todos os píxeis da imagem. Quando há uma área de sobreposição entre o padrão presente no filtro e aquele constatado na imagem, a convolução retorna um número, registrado no mapa de características que indicará as regiões de coincidência.

1.3.1 Treinamento

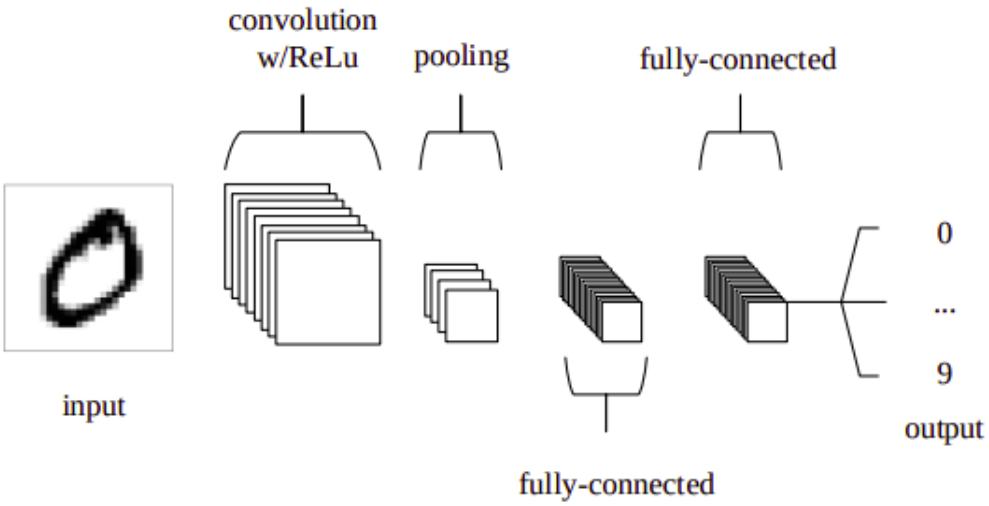
Na prática, um modelo de CNN é caracterizado por três etapas: treino, teste e validação. No treinamento utiliza-se um conjunto de imagens que são pré-classificadas pelo usuário e através delas o modelo aprende como associar cada padrão de píxel a determinado rótulo (categoria). Assim, é com base no treino que o programa capta informações para classificar o restante dos dados durante o teste.

Já as imagens de teste são aquelas que inicialmente não estão classificadas, ou seja, não são associadas diretamente à classe correta pela CNN como ocorre durante o treino. Dessa forma, são realizadas predições sobre as imagens que irão retornar a porcentagem de acerto do programa, isto é, quanto do conjunto de teste ele rotulou corretamente, representando a validação do desempenho do modelo construído.

1.3.2 Arquitetura

As Redes Neurais Convolucionais são compostas principalmente por três tipos de camadas (Figura 2), a camada convolucional, a camada de *pooling* e a camada completamente conectada.

Figura 2 – Arquitetura simples de uma CNN com apenas 5 camadas.



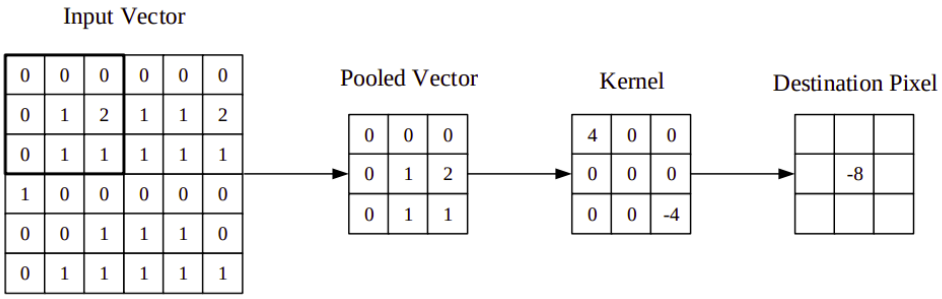
Fonte: O'Shea e Nash (2015).

O funcionamento básico da CNN é caracterizado pela contenção dos valores (píxeis das imagens) na camada de entrada, enquanto a convolução determinará a saída dos neurônios que serão conectados a regiões da entrada, por meio do produto escalar dado pela eq.1. O *pooling* irá reduzir a dimensionalidade espacial da entrada, diminuindo o número de parâmetros a serem trabalhados. Ao aplicar uma camada totalmente conectada, todas as entradas de uma camada serão conectadas a cada unidade de ativação da próxima camada.

1.3.2.1 Convolução

Como o nome indica, é a parte principal do processo, concentrando-se no uso de kernels que serão aprendidos. Quando os dados encontram a camada de convolução são percorridos por filtros que irão resultar em um mapa de ativação 2D (Figura 3). Assim, todos os valores de entrada são transpostos pelo produto escalar, fazendo com que a rede aprenda kernels que disparam na presença de um recurso específico em dada posição espacial. Esses kernels são conhecidos como ativações.

Figura 3 – Representação visual de uma convolução.



Fonte: O'Shea e Nash (2015).

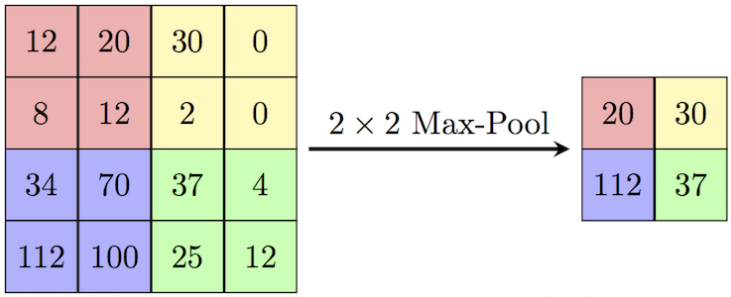
1.3.2.2 Pooling

A camada de *pooling* visa diminuir gradualmente a dimensionalidade e, assim, reduzir ainda mais o número de parâmetros e o custo computacional. Ela opera sobre cada mapa de ativação na entrada, percorrendo sua dimensão. O *pooling* pode ser utilizado de diversas formas:

1. *Max Pooling*: seleciona o maior valor dentro de um quadrante.
2. *Average Pooling*: responsável por calcular a média dos valores de um quadrante.
3. *Global Average Pooling*: é capaz de substituir as camadas Flatten. Ele gera um mapa de recursos para cada categoria correspondente da tarefa de classificação na última camada de convolução. Obtendo a média de cada mapa de recursos, o vetor resultante é alimentado diretamente na camada *softmax*.

Na maioria das CNNs, o método mais utilizado são as camadas de *max-poooling* com kernels de dimensão 2×2 aplicados com um passo 2. Isso dimensiona o mapa de ativação até 25% do tamanho original. Esse processo pode ser visualizado como exemplo sobre uma matriz de dimensão 4×4, ela será dividida em quatro partes e dentro de cada "quadrante" será selecionado o maior número, resultando em uma matriz 2×2 com esses valores (Figura 4).

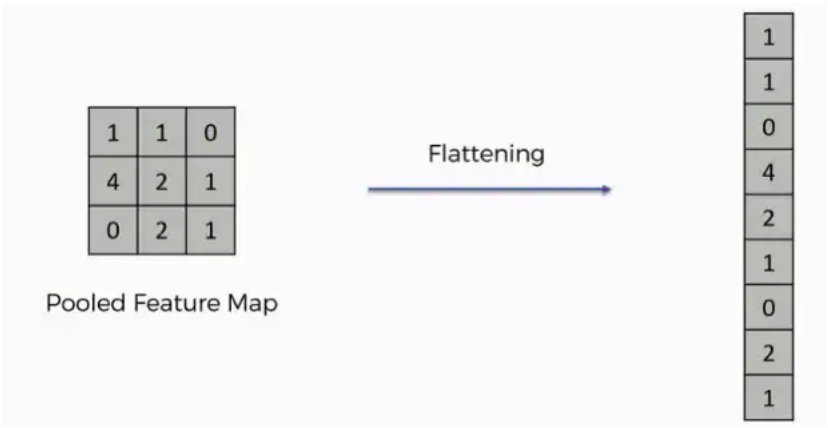
Figura 4 – Exemplo da aplicação de MaxPooling.



Fonte: MaxPooling (2023).

Na passagem entre as camadas de convolução e a camada totalmente conectada pode ser operada a função *Flatten*, que transforma o formato da matriz da imagem em um *array*. Ela não tem parâmetros para aprender, somente reformata os dados. Por exemplo, uma imagem de dimensão 50 × 50 se tornará um *array* de 2500 posições.

Figura 5 – Operação de *Flatten*.



Fonte: SuperDataScience (2023).

1.3.2.3 Camada totalmente conectada

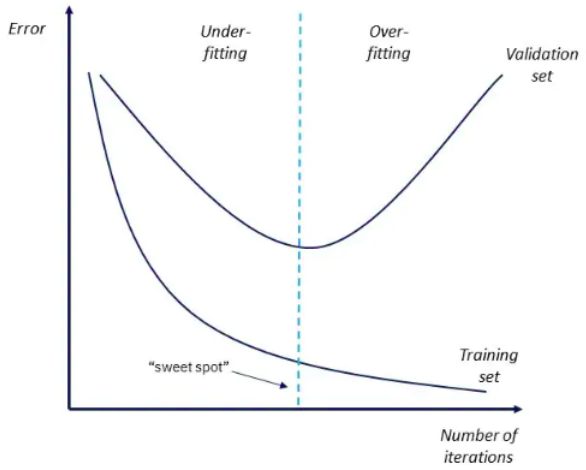
A camada totalmente conectada contém neurônios que estão diretamente conectados aos neurônios nas duas camadas adjacentes, sem estar conectado a nenhuma camada no meio deles. Ela é importante na extração de características dos dados ao passo em que precisam ser classificados em várias classes.

Dos tipos de camadas com essa função, a *Dense* é a mais utilizada. Ela visa calcular uma função de ativação em conjunto com os dados de entrada e pesos. No cenário de análise de imagens a configuração comum é composta por uma camada *dense* com a função ReLU, seguida de outra camada *dense* com mesma dimensão e a quantidade de classes a serem classificadas, nesse caso com a função de ativação *softmax*. A camada *softmax* retorna um *array* com as probabilidades da imagem pertencer a cada uma das classes definidas.

1.3.3 Overfitting

Um dos problemas que podem surgir em modelos supervisionados de *Machine Learning* é o *overfitting*, que ocorre quando o desempenho é perfeito sobre as imagens de treino enquanto não se encaixa bem sobre as imagens de teste (Figura 6). De acordo com O'Shea e Nash (2015), modelos superajustados tendem a ter dificuldade em classificar padrões no conjunto de teste que divirjam daquele memorizado no treino, além de que podem aprender ruídos de fundo no treinamento, incapazes de generalizar bem para novos dados. Na Figura 6 observamos uma ilustração do que ocorre no *overffiting*, onde a curva de erro para o conjunto de validação aumenta e se afasta da curva do conjunto de treino conforme o número de iterações crescem. O "ponto ideal" denota o espaço onde seria ideal parar o treinamento, marcando a separação do *under-fitting*, quando o treino não é suficiente para se estabelecer uma boa relação entre os padrões presentes nas imagens.

Figura 6 – Demonstração gráfica do Overfitting.



Fonte: IBM (2023).

Para contornar esse problema algumas soluções se refletem na pausa do treinamento no "ponto ideal", aumentar o conjunto de treino ou utilizar entre as camadas funções de regularização como a *Dropout*, a *Batch normalization* e a *Data Argumentation*. A função *Dropout* funciona eliminando temporária e aleatoriamente alguns dos neurônios ocultos na rede durante o treinamento em cada iteração. Assim, o procedimento é como calcular a média de inúmeras redes diferentes que serão superajustadas de maneiras distintas, então o efeito líquido do *dropout* será reduzir o superajuste.

No caso da função *Batch normalization* ocorre uma melhora da velocidade, do desempenho e da estabilidade de redes neurais artificiais. A ideia é normalizar as entradas de cada camada de forma que elas tenham uma saída de ativação média zero e um desvio padrão unitário.

Já para o uso de *Data Argumentation*, segundo Carruba et al. (2022a) há a criação de cópias de parcelas dos dados do treinamento com pequenas alterações aleatórias. Isso tem um efeito de regularização, fazendo com que o modelo aprenda os mesmos recursos gerais, mas de uma forma mais genérica enquanto também expande o conjunto de dados de treinamento.

Na construção do modelo aplicado foram utilizadas as funções *Dropout* e *Batch Normalization* comentadas, assim como a ampliação do conjunto de dados quando possível.

1.3.4 Tipos de modelos de CNNs

No histórico de modelos com tentativas de classificação de imagens com cada vez mais acurácia encontra-se como pioneiro o trabalho: "*Gradient-based learning applied to document recognition*"(LECUN et al., 1998). Utilizando a rede LeNet-5 de 7 camadas para reconhecimento de dígitos, foi aplicada para identificar números em manuscritos digitalizados como imagens de dimensão 32 × 32.

Durante um intervalo de anos o progresso foi lento devido à barreira tecnológica de recursos computacionais disponíveis, já que tais modelos exigiam um banco de dados e consequentemente uma capacidade de armazenamento e processamento amplos. Entretanto, com a introdução de *Graphics Processing Unit* (GPU) em aplicações de redes neurais convolucionais, inicou-se o uso de processadores especializados em gráficos utilizados em placas de vídeo, inicialmente utilizada para jogos de computadores e videogames.

As redes neurais convolucionais com transferência de aprendizado trabalham com modelos de pesos e parâmetros pré-treinados que agem como extrator de características. O grupo mais utilizado de banco de dados é conhecido como ImageNet [1], usualmente ajustado para a aplicação específica. A rede de dados é composta por mais de 15 milhões de imagens com 22 mil classes. Diante do extenso espaço de dados para trabalhar e do progresso técnico, houve um estopim em 2012 com a rede AlexNet.

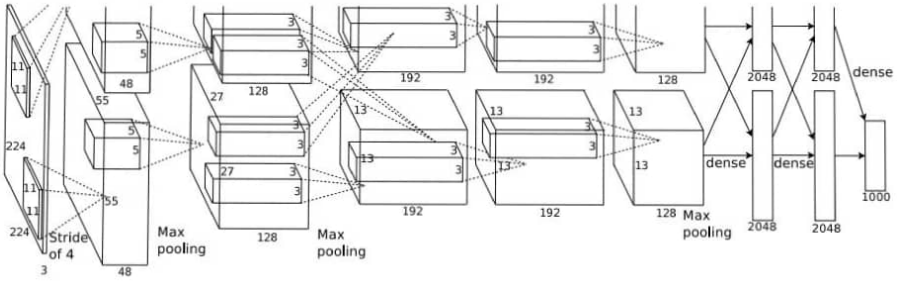
Anualmente ocorre a competição ImageNet,*ImageNet Large Scale Visual Recognition Challenge* (ILSVRC), caracterizada por analisar algoritmos de detecção de objetos e classificação de imagens em grande escala. Essa competição visa comparar o avanço na área de visão computacional.

Participando do projeto realizado pela ImageNet em 2012, o AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) reduziu o erro de 26% para 15.3%. O modelo foi treinado sobre um conjunto de 15 milhões de imagens da ImageNet, cuja dimensão de entrada era 256 × 256 píxeis, compostas por 22 mil classes. Krizhevsky, Sutskever e Hinton (2012) foram os precursores no progresso das Redes Neurais Convolucionais (CNN). Eles foram responsáveis por treinar uma das mais bem sucedidas CNNs até aquele ano e obtiveram os melhores resultados na competição.

A arquitetura utilizada na AlexNet pode ser resumida pela Figura 7 que contém oito camadas de aprendizado, cinco convolucionais e três totalmente conectadas.

[1] https://www.image-net.org/

Figura 7 – Arquitetura da rede AlexNet.



Fonte: Krizhevsky, Sutskever e Hinton (2012).

Já no ILSVRC 2013 o modelo vencedor foi a rede ZF Net (ZEILER; FERGUS, 2013) com um erro de 14,8% treinado 1,3 milhões de imagens. Em comparação ao AlexNet fez ajustes aos parâmetros, mantendo a estrutura semelhante, mas incrementada, sendo que contribuiu para a compreensão intuitiva do funcionamento da rede.

Com a necessidade de ir ainda mais fundo, foi desenvolvido por Szegedy et al. (2014) o módulo *inception* com filtros convolucionais em paralelo com camadas de *pooling*. A arquitetura utiliza filtros 1×1 para redução de dimensionalidade. Desse modo, a rede busca a melhor utilização dos recursos computacionais, aplicada ao GoogLeNet posteriormente.

A GoogLeNet alcançou, em 2014, somente 6.67% de erro (SZEGEDY et al., 2014) ao classificar as imagens do ImageNet. Outras redes de grande relevância desenvolvidas posteriormente e participantes ILSVRC foram a VGGNet com 7.3% de erro (SIMONYAN; ZISSERMAN, 2014) e a ResNet (HE et al., 2015) atingindo um erro de somente 3,57%, inferior ao erro observado no desempenho em nível humano neste conjunto de dados. A última competição foi realizada em 2017.

Figura 8 – Resumo histórico das redes neurais convolucionais. Em destaque o ano e autores dos artigos publicados, assim como o erro percentual alcançado, a colocação no projeto ILSVRC da ImageNet e o número de parâmetros utilizados.

Ano	CNN	Autores	Colocação no ILSVRC	Erro	Parâmetros
1998	LeNet	Y. LeCun et. al	-	-	60 mil
2012	AlexNet	A. Krizhevsky et. al	1°	15.3%	60 milhões
2013	ZFNet	M. D. Zeiler & R. Fergus	1°	14.8%	-
2014	VGGNet	K. Simonyan & A. Zisserman	2°	7.3%	138 milhões
2014	GoogLeNet	C. Szegedy et. al	1°	6.67%	4 milhões
2015	ResNet	K. He et. al	1°	3.6%	-

Fonte: Produção da própria autora.

2 MATERIAIS E MÉTODOS

O objetivo desse trabalho consistiu em implementar e analisar métodos de *Machine Learning* para classificação de imagens geradas em problemas de dinâmica orbital como classificação de órbitas e análise de ângulos ressonantes.

Portanto, para aplicarmos um modelo de *Machine Learning* e especificamente de Redes Neurais Convolucionais, preparamos o ambiente computacional, instalamos as bibliotecas utilizadas e analisamos as exigências de processamento. Em sequência, realizamos testes com redes mais simples para observar o comportamento da CNN antes de seguir para o caso estudado. Por fim escolhemos uma rede existente como base para a construção do nosso modelo, que nesse caso foi a rede AlexNet.

2.1 BIBLIOTECAS COMPUTACIONAIS

Uma forma de construir um modelo de CNN é empregando a biblioteca *Tensorflow* de código aberto para aprendizado de máquina. Com ele é possível manipular a biblioteca *Keras*[1] de redes neurais em Python (CHOLLET et al., 2018). Para executar o processo de forma otimizada, uma placa de processamento gráfico (GPU) é utilizada para rodar o programa. Estas bibliotecas foram instaladas e testadas nas estações de trabalho utilizadas no desenvolvimento do projeto.

A estruturação do programa é feita compondo o processamento das imagens e a aplicação das camadas da rede neural que serão responsáveis pela convolução das imagens. Em seguida são rodados os treinos com o modelo e os consecutivos bancos de dados. Após o treino é possível obter as predições e consecutiva classificação das imagens de teste. Assim, avalia-se o desempenho do modelo, ou seja, se está classificando corretamente as imagens. Durante a validação do desempenho, variam-se diferentes parâmetros para identificar aqueles que melhor se adéquam, tais como a dimensão das imagens e o número de épocas rodadas, ou seja, quantas vezes os dados passaram pela rede neural atualizando os pesos como discutido na seção 1.2. Além disso, foram alternadas a proporção de imagens correspondentes ao conjunto de treino e as características das operações de convolução.

2.2 MÉTRICAS

Com o intuito de analisar o desempenho do modelo são utilizadas métricas calculadas com base na matriz da confusão. A matriz da confusão é composta por quatro categorias que serão comparadas:

1. *True Positive* (TP): o número de imagens que pertencem à classe positiva tanto pela classe real quanto pela predição.

[1] https://keras.io/

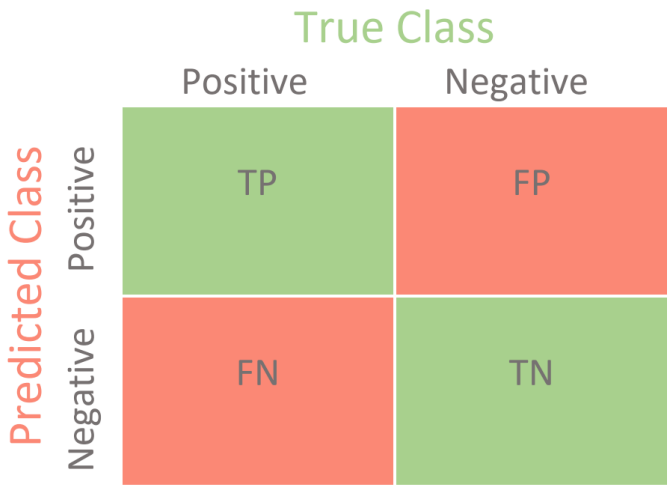
2. *True Negative* (TN): representa o número de imagens pertencentes a classe negativa tanto pela classe real quanto pela predição.

3. *False Positive* (FP): o número de imagens que somente a predição determinou como pertencente a classe positiva.

4. *False Negative* (FN): são as imagens classificadas como da classe negativa somente pela predição.

A matriz e consecutivamente as classes que a compõem estão ilustradas pela Figura 9.

Figura 9 – Exemplo gráfico de uma matriz da confusão.



Fonte: Mohajon (2023).

Assim é possível calcular as métricas *accuracy*, *precision* e *recall*. A *accuracy* (Acc) é responsável por medir a capacidade do modelo em acertar a predição da imagem em comparação com a classe real,

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

o que demonstra sua eficiência na totalidade. Já a *precision* determina a habilidade do modelo em não gerar muitos falsos positivos e assim demonstra um desempenho para a proporção de positivos corretos.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Por fim, a *recall* representa a capacidade do modelo em recuperar a atual população de uma dada classe, ou seja, dá a proporção de positivos reais identificados corretamente. Ambas podem ser identificadas pelas equações (CARRUBA et al., 2022a).

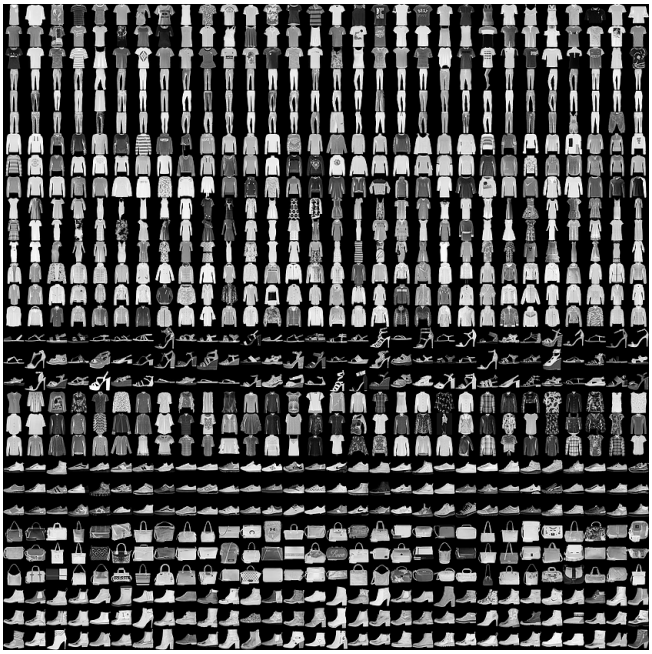
$$Recall = \frac{TP}{TP + FN} \tag{3}$$

2.3 INTRODUÇÃO A APLICAÇÃO DE UM MODELO DE CNN

As primeiras aplicações foram feitas com a motivação de observar o funcionamento de um modelo de redes neurais convolucionais na classificação de imagens. Portanto, existem tutorais disponíveis na plataforma do *TensorFlow*[2] que demonstram o uso de classificações básicas.

Nesse caso, o tutorial utiliza imagens de roupas importadas da base de dados *Fashion MNIST*[3], composta por 70 mil imagens em preto e branco (Figura 10) divididas em 10 categorias.

Figura 10 – Amostras da Fashion MNIST.



Fonte: Zalando (2023).

Foram separadas 10 mil imagens para treino e 60 mil para teste. A entrada é lida como *arrays* NumPy[4] de dimensão 28×28.

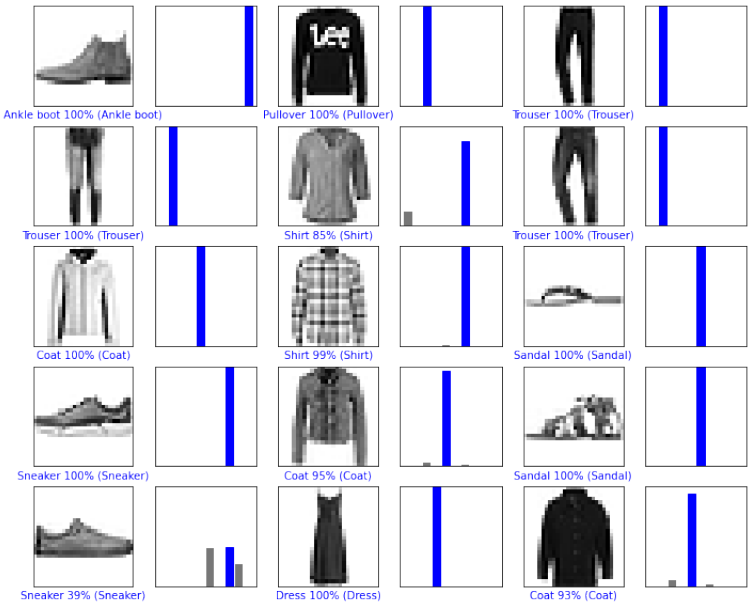
O conjunto de treinamento pelo qual o modelo irá aprender é depositado em *arrays* para as classes (*labels*) e para as imagens, assim como é feito para o conjunto de teste.

Para a construção da rede neural são encadeadas camadas, como discutido no Capítulo 1, responsáveis por extrair as características dos dados. Assim sendo, nesse caso, foi utilizada uma arquitetura básica com a sequência de uma camada *Flatten* e duas *Dense* com ativação respectivamente ReLU e *softmax*.

Nesse tutorial a acurácia alcançada após o treinamento foi de 88%, onde pode-se visualizar uma parcela de predições realizadas pelo modelo na Figura 11.

[2] https://www.tensorflow.org/
[3] https://github.com/zalandoresearch/fashion-mnist
[4] NumPy é uma biblioteca em Python com funções destinadas à computação numérica, sendo *arrays* vetores de n dimensões responsáveis por armazenar informações.

Figura 11 – Recorte de predições realizadas no caso Fashion MNIST: Labels preditas corretamente são azuis e as predições erradas são vermelhas.



Fonte: Chollet (2023).

2.4 APLICAÇÃO DO MODELO ALEXNET

Como discutido na seção 1.3.4, a rede AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) reduziu o erro para classificação da ImageNet de 26% para 15.3%, contendo oito camadas de aprendizado, cinco convolucionais e três totalmente conectadas. É aplicada sobre as camadas a função de ativação não-linear ReLU, já que segundo os autores, redes neurais convolucionais possuem um desempenho mais rápido com a utilização da mesma. É utilizada uma normalização local representada pela expressão:

$$b^i_{x,y} = \frac{a^i_{x,y}}{k + \alpha \sum_{j=max(0,i-n/2)}^{min(N-i,i+n/2)} (a^j_{x,y})^2} \tag{4}$$

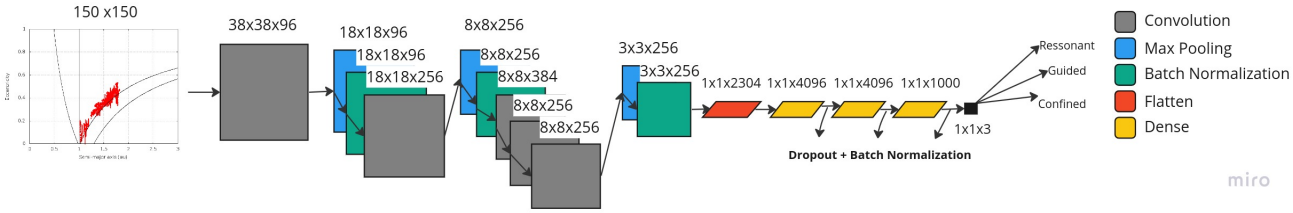
a soma percorre n mapas de *kernel* "adjacentes" na mesma posição espacial, e N é o número total de kernels na camada. As constantes k, n, α e β são hiperparâmetros determinados usando um conjunto de validação; no caso do artigo (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) foi utilizado $k = 2, n = 5, \alpha = 10^{-4}$ e $\beta = 0,75$.

Como a Figura 7 expressa, a saída da última camada totalmente conectada é alimentada para um *softmax* de 1000 vias que produz uma distribuição sobre os 1000 rótulos. Os neurônios nas camadas totalmente conectadas são associados a todos os neurônios na camada anterior. Seguem a primeira e a segunda camada convolucional camadas de normalização, que são consecutivamente encadeadas de camadas de *max-pooling*, bem como a quinta camada convolucional. A função ReLU é aplicada à saída de cada camada convolucional e totalmente conectada. O *overfitting* é combatido na rede através de *Data Argumentation* e *Dropout*, discutidos na Seção 1.3.3.

O modelo utilizado nesse projeto foi baseado na rede AlexNet, adaptado para o caso em

questão que desejamos classificar. A arquitetura caracterizada para classificação de um gráfico da excentricidade versus o semi-eixo maior de um NEAs (apresentado na Seção 3.1) é dada pela Figura 12.

Figura 12 – Arquitetura da rede utilizada.



Fonte: Produzida pela próprio autora.

A primeira camada convolucional filtra a imagem de entrada 150×150×3 com 96 kernels de tamanho $11 \times 11 \times 3$. A segunda camada convolucional recebe como entrada a saída da primeira camada convolucional e a filtra com 256 kernels. A terceira, quarta e quinta camadas convolucionais são conectadas umas às outras sem nenhuma intervenção de camadas de *pooling* ou normalização. A terceira camada convolucional tem 384 kernels conectado às saídas da segunda camada convolucional. A quarta camada convolucional tem 384 kernels, e a quinta camada convolucional tem 256 kernels. As duas primeiras camadas totalmente conectadas *dense* têm 4096 neurônios, seguidas de uma camada com 1000 neurônios e da última camada *dense* responsável pela tomada de decisão. A última camada retorna um *array* com 3 neurônios, cada um composto por um valor que representa a probabilidade da imagem pertencer a uma das classes.

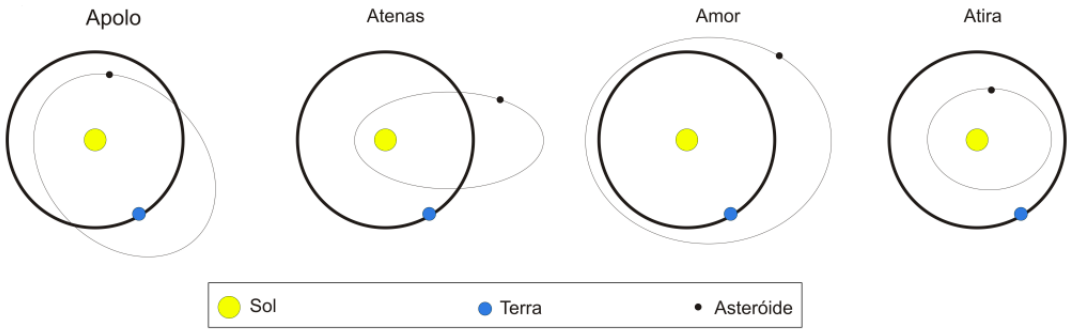
3 RESULTADOS E DISCUSSÃO

Para verificar a exequibilidade da proposta foram realizadas análises utilizando em um primeiro momento dados orbitais da evolução dinâmica de NEAs (*Near Earth Asteroids*)[1]. Além disso, realizamos a classificação de gráficos da evolução do ângulo ressonante no tempo das partículas do arco do anel G de Saturno.

3.1 APLICAÇÃO EM NEAS

NEAs são denominados aqueles asteroides cujas órbitas estão próximas ou podem cruzar a órbita da Terra (distâncias do periélio inferiores a 1,3 u.a.), existem quatro grupos principais: Amor, Apolo, Atena e Atira. Eles são separados considerando suas características orbitais [2]. Dessa forma, Apolo e Atenas cruzam a órbita da Terra e são denominados cruzadores da Terra (*Earth-crossers*). Os do tipo Amor quase cruzam a órbita da Terra e são chamados de quase cruzadores da Terra (*Nearly-Earth crossers*), porém segundo Bottke et al. (2002) em certos períodos podem evoluir em *Earth-crossers*. Os Atiras são os NEAs que têm as órbitas inteiramente na órbita da Terra, sem cruzá-las (Figura 13).

Figura 13 – Representação das órbitas de NEAs pertencentes aos grupos Apolo, Atenas, Amor e Atira.

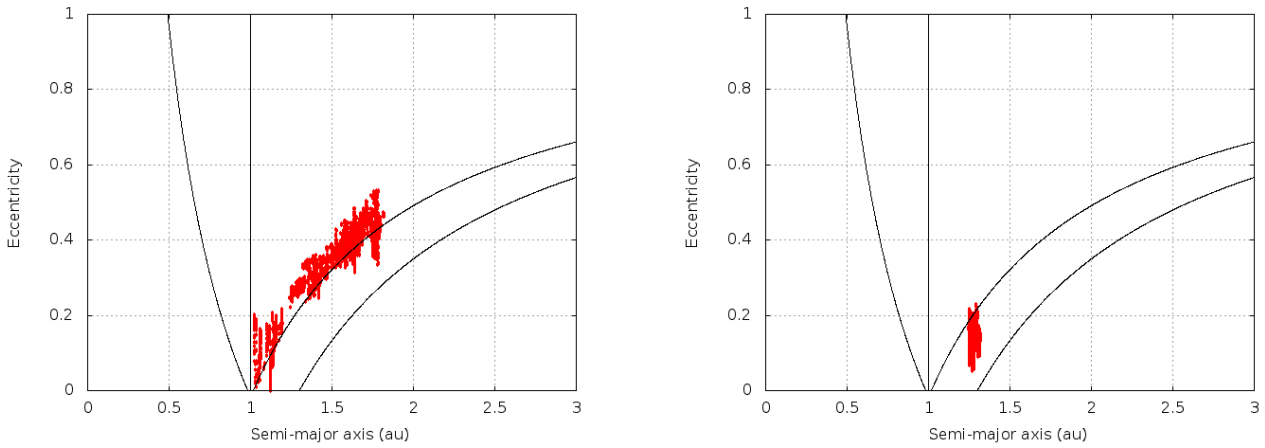


Fonte: Araujo (2011).

Foram disponibilizados para esse projeto os gráficos referentes ao conjunto Amor provenientes de um trabalho anterior, de modo que já haviam sido previamente classificadas. Portanto, a fim de caracterizar a evolução orbital desses objetos, buscamos agrupá-los de acordo com duas classificações orbitais: ressonantes ou guiado/confinado (Figura 14).

[1] As imagens foram gentilmente cedidas pela Dra. Rosana Araujo e Dr. Othon Winter.
[2] Mais informações podem ser obtidas com o estudo do trabalho de Araujo (2011)

Figura 14 – Exemplo das imagens com 2 classes.

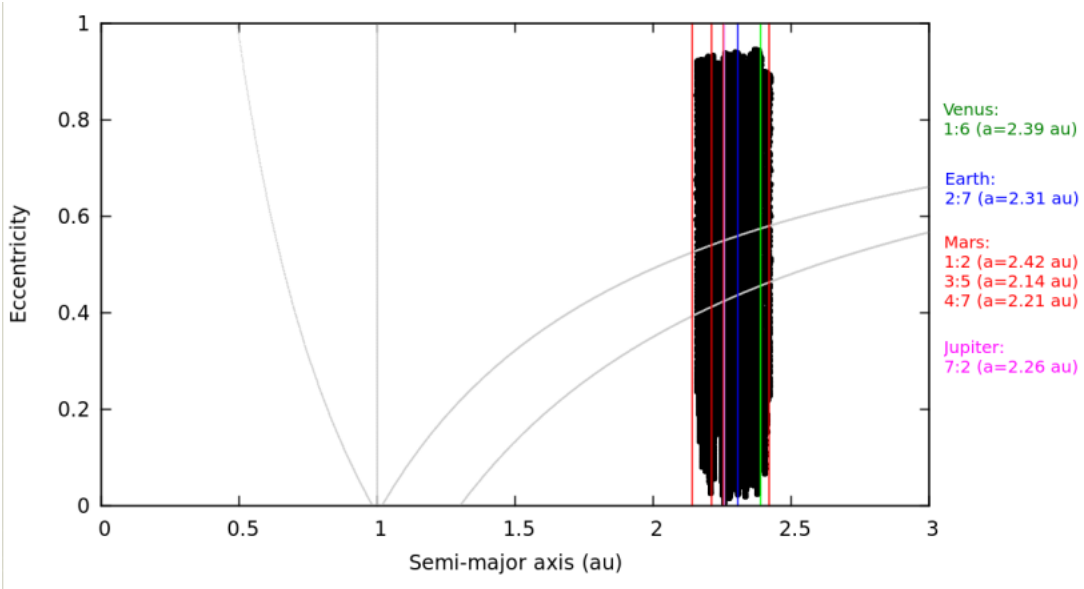


(a) Guiado/confinado (b) Ressonante

Fonte: Disponibilizados pelos pesquisadores do grupo Dra. Rosana Araujo e Dr. Othon Winter.

As órbitas guiadas são aquelas que tendem a acompanhar o apocentro da Terra e as ressonantes aquelas sobre efeito de ressonância nominal (MURRAY; DERMOTT, 1999) conforme a Figura 15. O intuito foi classificar gráficos que representavam a excentricidade versus o semi-eixo maior da órbita dos objetos e assim determinar a que categoria pertenciam.

Figura 15 – Evolução temporal no espaço (a x e) de um NEA inicialmente pertencente ao conjunto Amor.



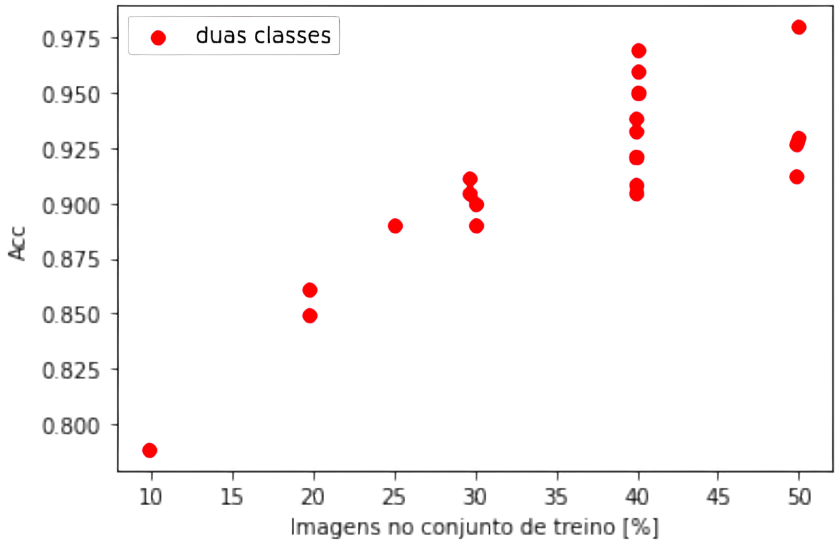
Fonte: Dra. Rosana Araujo e Dr. Othon Winter.

Com o conjunto de dados definido, uma forma de avaliar o resultado do algoritmo de classificação é registrando sua acurácia. Para esse conjunto de dados foi alcançada uma acurácia de até 98%, sendo ele composto por 273 imagens, de dimensão variada, como demonstra a Tabela 1. As imagens foram formatadas em .png e previamente classificadas pelo grupo de pesquisa para que houvesse uma comparação quando o programa fizesse a predição automática.A

Tabela 1 foi ordenada de acordo com a porcentagem do conjunto de treino (do menor ao maior), visto que um dos principais objetivos era analisar o comportamento do modelo de acordo com o número de imagens disponíveis para o treinamento e, assim, minimizar a quantidade de dados mantendo uma acurácia apreciável.

A performance do programa sobre os dados variou de acordo com a proporção entre o conjunto de teste e de treino, a quantidade de épocas, os parâmetros das camadas (níveis de convolução) e por fim o tamanho das imagens. Como observamos na Figura 16 quanto maior a porcentagem do conjunto de treino maior a acurácia, especialmente para esse caso onde o número total de imagens foi baixo. Portanto, a quantidade de imagens disponíveis para o aprendizado impacta na capacidade do modelo em aprender os padrões de cada classe. Entretanto, como o conjunto de treino exige a classificação manual, visa-se encontrar um cenário onde esse trabalho seja minimizado, mas não haja perda significativa de precisão na classificação. Para o caso em questão a porcentagem ideal se concentraria entre 30% e 40%.

Figura 16 – Relação entre a acurácia e porcentagem de imagens no conjunto de treino para o caso dos NEAs.



Fonte: Produção da própria autora.

Analisamos o comportamento do programa para o caso de utilizarmos as imagens com dimensões de 200×200, variando até o tamanho original 480×640 píxeis. O teste foi composto por duas classes e época entre 200 e 300.

Tabela 1 – Resultados para o caso de NEAS com duas classes.

Accuracy (%)	**Imagens de treino**	**Treino (%)**	**Imagens de teste**	**Teste (%)**	**Época**	**Dimensão**
89.33	68	25	205	75	200	480 x 640
89.00	82	30	191	70	200	500 x 500
90.00	82	30	191	70	200	500 x 500
96.95	109	40	164	60	300	200 x 200
95.70	109	40	164	60	300	200 x 200
94.50	109	40	164	60	300	300 x 300
96.40	109	40	164	60	300	300 x 300
98.00	136	50	137	50	200	480 x 640
93.00	136	50	137	50	200	480 x 640

Fonte: Produção da própria autora.

Obtivemos um bom acerto (maior que 90%), sendo que a maior taxa ocorreu para 50% das imagens no conjunto de treino, 200 épocas e dimensão de 480×640 píxeis (acerto: 98%). Observamos que o desempenho do modelo melhora em relação aos outros testes por ter mais parâmetros com os quais treinar, já que aumentamos a proporção da matriz imagem, entretanto como estamos lidando com fundo branco, aumenta a sobreposição de padrões em comum entre as classes.

3.2 APLICAÇÃO EM ÂNGULOS RESSONANTES

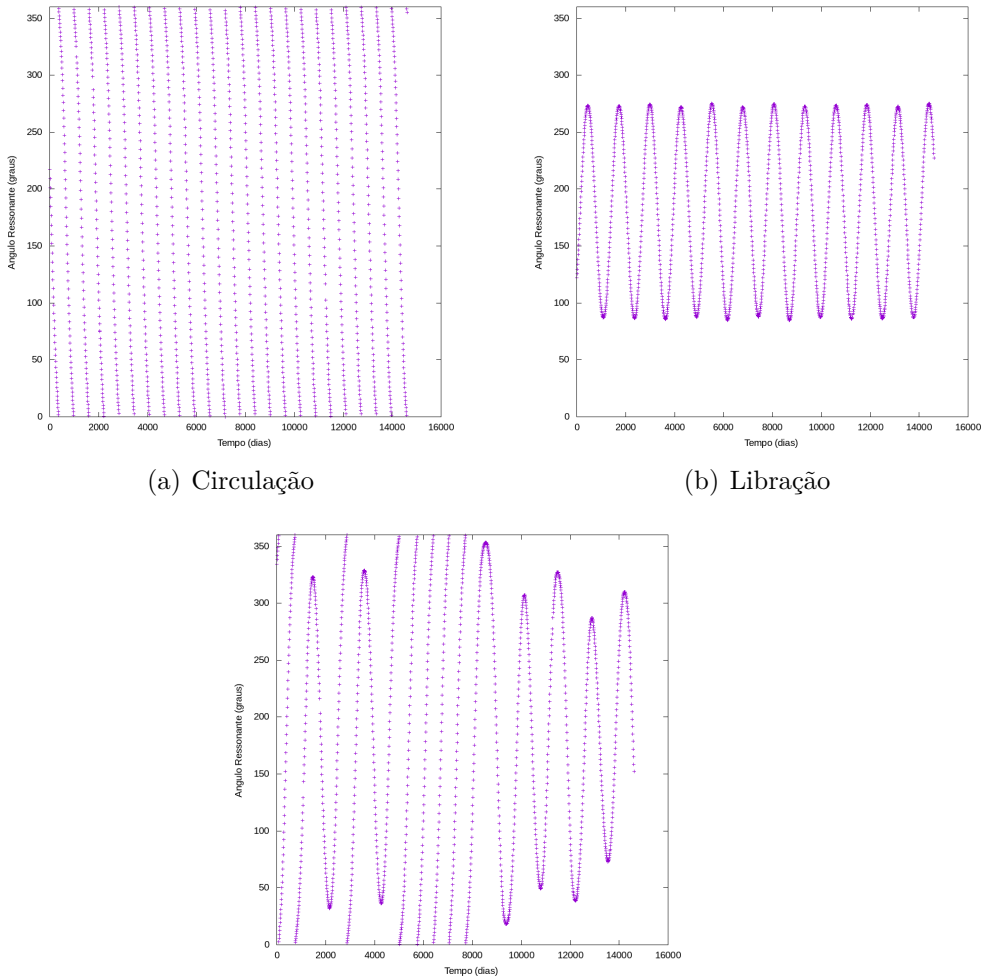
O arco do anel G se encontra confinado em uma ressonância de corrotação (consultar (MURRAY; DERMOTT, 1999)) 7:6 com o satélite Mimas. Mas no final de 2008 foi descoberto um pequeno satélite no arco, anunciado em uma circular da IAU, designado S2008/S1 e posteriormente nomeado Saturno LIII/ Aegaeon (HEDMAN et al., 2010). Aegaeon situa-se em uma ressonância 7:6 de corrotação excêntrica com Mimas. Assim, o ângulo ressonante φ_{cr} deste sistema é dado por:

$$\varphi_{cr} = 7\lambda_{Mimas} - 6\lambda_{Aegon} - \omega_{Mimas} \tag{1}$$

sendo λ a longitude média de Mimas e de Aegaeon, enquanto ω é a longitude do pericentro de Mimas.

Portanto, nessa aplicação o objetivo foi classificar os casos em que o ângulo ressonante se apresentava no estado de libração, quando a partícula oscila em um intervalo limitado, no estado de circulação, com o ângulo variando entre 0 e 360° e aquelas que alternam entre esses dois cenários (Figura 17).

Figura 17 – Exemplos da evolução do ângulo ressonante em função do tempo, definindo os casos de circulação, libração e alternado.



(a) Circulação

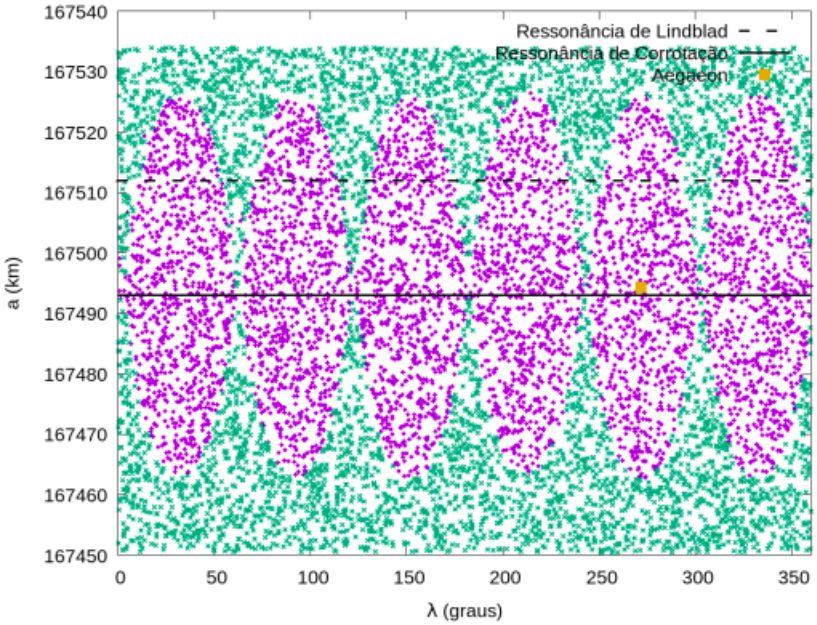
(b) Libração

(c) Alternado

Fonte: Lattari (2019).

De acordo com Lattari (2019), a composição do conjunto é distribuída de forma equilibrada entre partículas em libração e circulação. Como se observa na Figura 18, o ponto laranja corresponde a localização do satélite Aegaeon, os pontos em roxo determinam os corpos confinados na ressonância 7:6 CER com o satélite Mimas e em verde os que se encontram fora da ressonância. Já a linha contínua determina a posição teórica da ressonância 7:6 CER, enquanto a pontilhada a ressonância 7:6 CEL.

Figura 18 – Localização dos sítios ressonantes do anel G (roxo) e das ressonâncias de corrotação (linha contínua) e de Lindlbad (linha tracejada), enquanto a do satélite Aegaeon é destacada em laranja.



Fonte: Lattari (2019).

Nesse caso o conjunto foi amplo, contendo 18 mil imagens. Foram desenvolvidos dois tipos de cenário, com duas classes (libração e circulação) e três classes (libração, alternado e circulação).

Para o caso com duas classes obtivemos um bom desempenho ($acc > 0,9$) do modelo em classificar as imagens, como podemos conferir na Tabela 2:

As tabelas dessa seção foram, também, ordenadas conforme a porcentagem de imagens presentes no conjunto de treino (da menor para a maior). Entretanto, elas diferem dos resultados apresentados para o caso dos NEAs ao passo que mantemos a dimensão constante (150×150) e acrescentamos uma coluna com o comparativo do tempo de duração para treinar cada teste.

Tabela 2 – Resultado para ângulos ressonantes com duas classes.

Imagens de teste	**Teste (%)**	**Imagens de treino**	**Treino (%)**	**Accuracy (%)**	**Tempo (h)**	**Época**
16423	98	335	2	0.998	0.74	150
16256	97	502	3	0.999	1.02	150
16088	96	670	4	0.999	0.89	200
8100	95	426	5	0.869	0.49	100
8100	95	426	5	0.999	0.83	200
8100	95	426	5	0.999	0.70	200
1717	90	190	10	0.999	0.25	200
7674	90	852	10	0.999	0.82	100
7674	90	852	10	0.999	0.92	100
1526	80	381	20	0.998	0.71	200

Fonte: Produção da própria autora.

A maior acurácia alcançada foi de 0,999. As imagens utilizadas possuíam dimensão de 150 × 150 píxeis e formato .png. De acordo com a Tabela 2, quanto mais imagens presentes no conjunto de treino, mais tempo o programa tende a demorar, como vemos para o caso com 190 imagens para treino e duração de 0,25 hora, isso porque a operação de convolução percorre cada imagem. Além disso, o número de épocas também influencia na duração, conforme observado quando haviam 426 imagens para treino com 200 épocas o tempo foi quase o dobro (0,83h) comparado com 100 épocas (0,49h).

Em geral, a acurácia apresentou constância com a variação dos parâmetros, sendo mais favoráveis os casos com menos imagens de treino, como discutimos na seção anterior, e menor tempo de duração. As métricas alcançadas estão apresentadas na Tabela 3.

Tabela 3 – Média encontrada para as métricas *recall*, *precision* e *accuracy* com imagens de dimensão 150 × 150 e duração de treinamento de 0,78 hora.

Classes	Recall	Precision	Acuraccy	Distribuição do conjunto de teste (%)
Libração	1.0000	0.9993	0.9996	47.58
Circulação	0.9992	1.0000	0.9996	52.42

Fonte: Produção da própria autora.

Ao adicionarmos a terceira classe alternada ao modelo, a acurácia também foi elevada ($acc > 0,9$) como se observa na Tabela 4.

Tabela 4 – Resultado para ângulos ressonantes com três classes.

Imagens de teste	**Teste (%)**	**Imagens de treino**	**Treino (%)**	**Accuracy**	**Tempo (h)**	**Época**
17500	97	541	3	0.943	0.73	200
17139	95	902	5	0.988	0.98	150
17139	95	902	5	0.976	0.62	100
16992	95	894	5	0.961	0.45	150
16992	95	894	5	0.969	0.31	100
16456	92	1430	8	0.984	1.01	200
14309	80	3577	20	0.992	1.38	200

Fonte: Produção da própria autora.

A maior *accuracy* verificada para um conjunto de treino menor que 10% (acc = 0,988) foi com 5% dos dados de um total de 18041 imagens, 150 épocas e dimensão 150 × 150, durando 0,98 hora. Assim como para duas classes, a duração foi maior quanto mais imagens compondo o conjunto de treino.

Observa-se que o modelo confunde-se mais na predição quando haviam três classes, do que no caso composto somente pela libração e circulação. Isso porque a classe alternada apresenta padrões de ambos. Percebemos na Tabela 5 que no caso alternado a *recall* foi menor que as outras métricas, indicando que o modelo teve mais dificuldade em recuperar a classe real do conjunto, como discutido na seção 2.2.

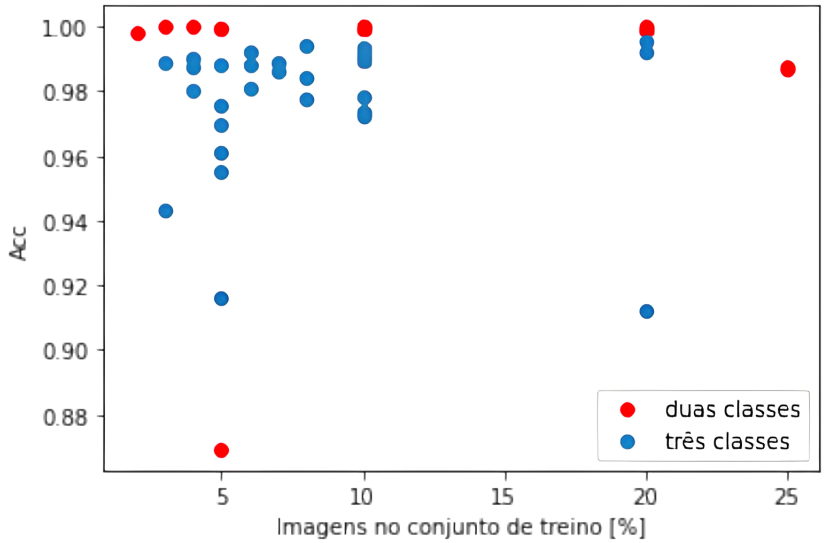
Tabela 5 – Média encontrada para as métricas *recall*, *precision* e *accuracy* com imagens de dimensão 150 × 150 e duração de treinamento de 0,73 hora.

Classes	Recall	Precision	Acuraccy	Distribuição do conjunto de teste (%)
Libração	0,9915	0,9899	0,9853	51
Alternado	0,8815	0,91969	0,9853	6.4
Circulação	0,9982	0,9989	0,9853	42.6

Fonte: Produção da própria autora.

Observa-se pela Figura 19 que a *accuracy* manteve valores altos, independente da proporção do conjunto de treino, entretanto os menores valores para o caso de duas classes foram obtidos para a menor porcentagem (5%) do conjunto de treino. Para três classes, os menores valores se concentraram também em 5%.

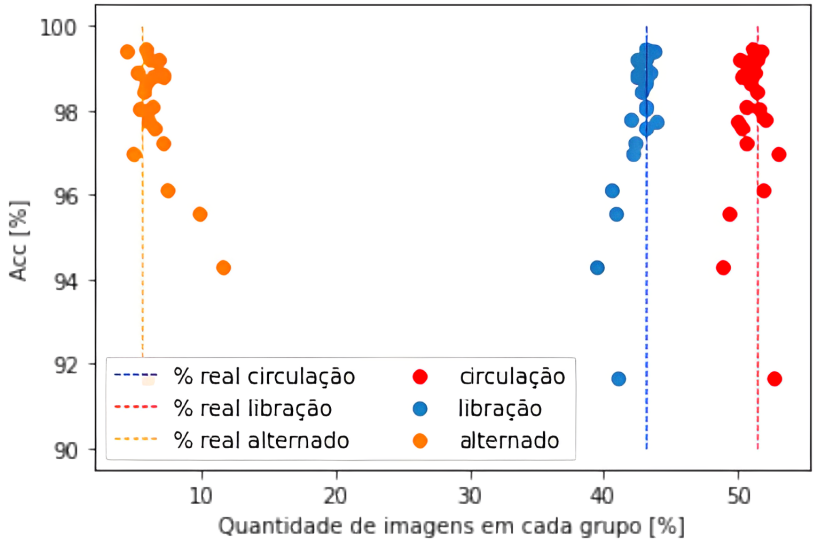
Figura 19 – Relação entre a acurácia e a porcentagem de imagens no conjunto de treino para o caso dos ângulos ressonantes.



Fonte: Produção da própria autora.

Segundo a Figura 20 percebe-se que, correspondendo ao esperado, quanto maior a acurácia, maior a correspondência de acerto da quantidade de imagens pertencentes a cada classe. A dispersão se concentrou em torno da porcentagem real obtida pela classificação do autor dos gráficos (LATTARI, 2019).

Figura 20 – Representação das porcentagens de cada classe de acordo com a classificação e a acurácia.



Fonte: Produção da própria autora.

4 CONSIDERAÇÕES FINAIS

O objetivo principal deste trabalho foi implementar um modelo de CNN para classificação de imagens geradas em problemas de dinâmica orbital, como classificação orbital e análise de ângulos ressonantes. Convém mencionar que outros métodos numéricos e de *Machine Learning* são utilizados para esse tipo de aplicação, entretanto não foram estudados nesse trabalho. Entre os modernos modelos de CNNs disponíveis, foi escolhido aquele que não levasse ao *overfitting* e fosse eficiente na previsão de grandes conjuntos de dados de imagens.

A aplicação contou com o uso da biblioteca *keras* de redes neurais em Python, assim como o uso do *Tensorflow*. Foram utilizados para teste dois conjuntos de dados disponíveis de gráficos de NEAS e de ângulos ressonantes. Portanto, a performance do modelo foi medida com uso da métrica de precisão padrão, *accuracy*, *recall* e *precision*, constatando um bom desempenho (ambas maiores que 90% nos dois cenários como demonstrado nos resultados). É importante destacar que para ambos os casos de aplicação, as classes determinadas representavam padrões muito bem delimitados, o que contribuiu para a acurácia elevada. Considerando cenários compostos por amostras com maior discrepância, haveria a possibilidade de resultados diferentes, tendo que ser essa possibilidade estudada.

O modelo agora pode ser usado para classificar imagens de ângulos ressonantes de asteroides com precisão de até 99%, assim como gráficos de NEAs com precisão de até 98%. Ademais, há o potencial do uso para outros estudos em dinâmica orbital com padrões semelhantes, ou seja, um conjunto amplo de imagens para classificação que possam ser separadas em classes definidas.

O código utilizado no trabalho será disponibilizado para consulta posteriormente no GitHub.

REFERÊNCIAS

ARAUJO, R. A. N. de. **O sistema triplo de asteróide 2001 SN263: dinâmica orbital e estabilidade**. Dissertação (Doutorado em Mecânica Espacial e Controle) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, São Paulo - Brasil, 2011.

ARBIB, M. **The handbook of brain theory and neural network**. [S.l.: s.n.], 2003. v. 26. ISBN 0262011972.

BOTTKE, W. et al. Debiased orbital and absolute magnitude distribution of the near-earth objects. **Icarus**, v. 156, p. 399–433, 04 2002.

CARRUBA, V. et al. **Optimization of artificial neural networks models applied to the identification of images of asteroids resonant arguments**. arXiv, 2022. Disponível em: <https://arxiv.org/abs/2207.14181>.

CARRUBA, V. et al. Machine learning applied to asteroid dynamics. **Celestial Mechanics and Dynamical Astronomy**, v. 134, aug 2022.

CHOLLET, F. **Treine sua primeira rede neural: classificação básica**. 2023. Disponível em: <https://www.tensorflow.org/tutorials/keras/classification>.

CHOLLET, F. et al. **Keras: the python deep learning library**. 2018. ascl:1806.022 p.

GSIGMA. **GSIGMA**. 2023. Disponível em: <https://www.gsigma.ufsc.br/>.

HAYKIN, S. **Redes neurais: princípios e prática**. Artmed, 2007. ISBN 9788577800865. Disponível em: <https://books.google.com.br/books?id=bhMwDwAAQBAJ>.

HE, K. et al. Deep residual learning for image recognition. **CoRR**, abs/1512.03385, 2015. Disponível em: <http://arxiv.org/abs/1512.03385>.

HEDMAN, M. et al. Aegaeon (saturn LIII), a g-ring object. **Icarus**, Elsevier BV, v. 207, n. 1, p. 433–447, may 2010. Disponível em: <https://doi.org/10.1016\%2Fj.icarus.2009.10.024>.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. Imagenet classification with deep convolutional neural networks. **Neural Information Processing Systems**, v. 25, 01 2012.

LATTARI, V. C. **Formação de pequenos satélites e anéis de poeira**. Dissertação (Mestrado em Física) — Universidade Estadual Paulista, Faculdade de Engenharia de Guaratinguetá, feb 2019.

LECUN, Y. et al. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278–2324, 1998.

MAGNIER, E. A. et al. Pan-starrs pixel analysis: source detection and characterization. **The Astrophysical Journal Supplement Series**, The American Astronomical Society, v. 251, n. 1, p. 5, oct 2020. Disponível em: <https://dx.doi.org/10.3847/1538-4365/abb82c>.

MAXPOOLING. **Max pooling**. 2023. Disponível em: <https://paperswithcode.com/method/max-pooling>.

MOHAJON, J. **Confusion matrix for your multi-class machine learning model**. 2023. Disponível em: <https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826>.

MOHRI et al. **Foundations of machine learning**. [S.l.]: The MIT Press, 2012. ISBN 026201825X.

MURRAY, C.; DERMOTT, S. **Solar system dynamics**. Cambridge University Press, 1999. ISBN 9780521575973. Disponível em: <https://books.google.com.br/books?id=aU6vcy5L8GAC>.

O'SHEA, K.; NASH, R. An introduction to convolutional neural networks. **CoRR**, abs/1511.08458, 2015. Disponível em: <http://arxiv.org/abs/1511.08458>.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. **arXiv preprint arXiv:1409.1556**, aug 2014.

SUPERDATASCIENCE. **Convolutional neural networks (CNN): step 3 - flattening**. 2023. Disponível em: <https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-3-flattening>.

SZEGEDY, C. et al. Going deeper with convolutions. **CoRR**, abs/1409.4842, 2014. Disponível em: <http://arxiv.org/abs/1409.4842>.

ZALANDO. **Fashion-MNIST**. 2023. Disponível em: <https://github.com/zalandoresearch/fashion-mnist>.

ZEILER, M. D.; FERGUS, R. Visualizing and understanding convolutional networks. **CoRR**, abs/1311.2901, 2013. Disponível em: <http://arxiv.org/abs/1311.2901>.