# Análise de Dados para Roteamento e Eliminação de Espécies Invasivas Utilizando Veículos Não Tripulados

Equipe: **Predict This!**

Autores: Guilherme Castro Silva, Humberto Monteiro Fialho, Matheus Castro Silva, Raíssa Bergamini.

RESUMO

O relatório a seguir trata da aplicação das duas principais partes do projeto de análise da localização de espécies invasivas. Um é o tratamento das imagens para identificar aquilo que é pedido (as espécies invasivas) e o outro a determinação do caminho mínimo para passar em todos os pontos determinados.

***Palavras Chaves*** **- Processamento de imagem, machine learning, roteamento, otimização, espécies invasivas, veículos não tripulados, CNN.**

## I. CONTEXTUALIZAÇÃO

Hydrangea é um género de plantas fanerogâmicas pertencente à família Hydrangeaceae, que inclui cerca de 80 espécies, na sua maioria nativas do leste e sul da Ásia (desde o Japão e China, aos Himalaias e Indonésia) mas com algumas espécies nas Américas. A maioria das espécies são arbustos (com 1 a 3 m de altura) outras são lianas que alcançam os 30 m de comprimento trepando pelas árvores. Esse grupo Inclui a hortênsia (Hydrangea macrophylla), utilizada comumente como planta ornamental.

Podemos definir espécies invasoras como sendo espécies exóticas com alta capacidade de crescimento, proliferação e dispersão, capazes de modificar a composição, estrutura ou função do ecossistema. Nessa definição, não se considera as espécies nativas que por algum desequilíbrio ecológico, passam a crescer e se multiplicar descontroladamente, comportando-se como invasoras. [2] Já para os agricultores, essas espécies são as "pragas" ou "ervas-daninhas", que são tidas numa abordagem ecológica, como "colonizadoras" ou "pioneiras".

Levando em consideração a necessidade de remover plantas que se encaixem nessas definições de invasoras, tanto considerando objetivos financeiros - plantações, remoção de pragas e espécies indesejadas - quanto ambientais, o produto proposto tem como objetivo identificar as espécies invasoras com auxílio de técnicas de visão computacional. Utilizando imagens tiradas por meio de satélites, buscaremos identificar espécies consideradas invasoras, sendo que o teste inicial para o desempenho da ferramenta será utilizando uma base de dados com 2295 figuras de uma floresta nacional brasileira. Em algumas dessas imagens temos a Hydrangea que se apresentam em diferentes quantidades, níveis de luminosidade e ângulos de inclinação. Cada uma dessas imagens está associada a uma coordenada geográfica.

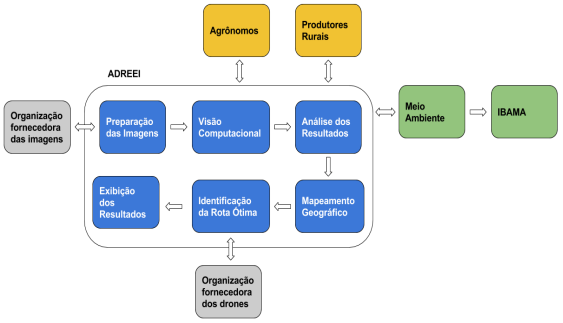


Figura 1: Representação do sistema utilizando diagrama de blocos.

Após o processo de identificação, será realizado o procedimento de reconhecimento da rota, que consiste no mapeamento das áreas mais afetadas pelas espécies invasoras. Por meio de técnicas de otimização avançadas é possível determinar a rota ótima que visa reduzir a distância e consequentemente o tempo total decorrido. Outros pontos a serem considerados para a escolha da melhor rota será: o caminho mais curto e as regiões mais afetadas, isto é maior prioridade, com o intuito de manter o equilíbrio ecológico local. Será também avaliado nível de dificuldade de locomoção para o drone, considerando o quanto a mata é fechada ou esparsa em determinados pontos.

A remoção das espécies invasivas será realizada com a utilização de veículos não tripulados. Os fatores que influenciaram na escolha de drones (veículos não tripulados) para executar a remoção destas espécies foram: simplicidade de acesso a lugares restritos, mobilidade e fácil acesso aos usuários, considerando a grande quantidade de modelos no mercado.

Figura 2: Exemplo de fotos do conjunto de dados.

Este projeto busca utilizar novas interfaces computacionais para auxiliar nessa identificação. Para isso, o projeto foi dividido em duas partes principais: reconhecimento de imagens e caminho mínimo. A identificação de uma determinada espécie através das figuras foi feita a partir de uma técnica de aprendizagem: *Convolutional Neural Network* (CNN) aplicada por meio de bibliotecas da linguagem de programação *Python*.

Já a parte de caminho mínimo teve a aplicação de métodos de otimização linear, muito utilizados na área de Pesquisa Operacional. A plataforma *open source* "lpsolve" utiliza os conceitos de grafo para calcular todos as possíveis ligações entre os pontos determinados e definir qual será o menor valor de peso total de todo o trajeto que deve ser realizado.

## II. MOTIVAÇÃO DO PROJETO

Emaranhados de Pueraria lobata, também conhecido como kudzu, dominam árvores na Geórgia, enquanto sapos de cana ameaçam habitats em mais de uma dúzia de países em todo o mundo. Estas são apenas duas espécies invasivas de muitas que podem ter efeitos prejudiciais sobre o meio ambiente, a economia e até mesmo a saúde humana. Apesar do impacto generalizado, os esforços para rastrear a localização e disseminação de espécies invasoras são tão onerosos que são difíceis de realizar em escala.

Atualmente, o monitoramento da distribuição de ecossistemas e plantas depende do conhecimento especializado. Cientistas treinados visitam áreas designadas e tomam nota das espécies que as habitam. Usar uma força de trabalho altamente qualificada é caro, tempo ineficiente e insuficiente, uma vez que os seres humanos não podem cobrir grandes áreas durante a amostragem.

O trabalho busca desenvolver um algoritmo para identificar com acurácia se as imagens apresentadas contém ou não uma espécie de hydrangea invasiva, utilizando técnicas de visão computacional e deep learning. Mais especificamente, foi utilizada uma rede neural convolucional com arquitetura otimizada para a classificação desejada. A partir dessa identificação e com o conhecimento do local correspondente a cada imagem, busca-se estabelecer a melhor rota para um veículo não tripulado eliminar a ameaça detectada, levando em consideração as restrições de hardware do drone. Desta forma, é possível remover ameaças do ecossistema com menores custos, além de uma maior eficiência, velocidade e precisão.

Subtemas envolvidos:

- Machine Learning;
- Reconhecimento de Imagem;
- Otimização Linear;
- Visão Computacional;
- Uso de heurísticas para criação de rota ótima;
- Equilíbrio ecológico;
- Sustentabilidade.

## III. DESAFIOS

Realizar a adaptação das imagens para o contexto do nosso problema, equilibrando resolução e tempo hábil de processamento referente ao conjunto de imagens a serem utilizadas como treinamento para o modelo proposto.

Apesar deste projeto ser aplicado a uma área bem específica (agrônoma), um dos desafios é conseguir desenvolver um produto que possa se generalizar em vários outros projetos.

Na última etapa do trabalho, os maiores desafios foram em função do tempo necessário para a otimização das rotas, pois para um número grande de pontos se tornava lenta.

A integração das diferentes partes do sistema entre si e com a interface também foi um desafio, tanto pela adaptação do formato de saída dos subsistemas e pelas tecnologias novas utilizadas. Todo o desenvolvimento da interface de software foi feita em python usando o *framework dash*, ferramenta com a qual nenhum dos membros do grupo tinha experiência anteriormente.

## IV. DESCRIÇÃO DA APLICAÇÃO DO PROJETO

O projeto se baseia na criação de um produto que associa visão computacional e um processo de otimização, sendo que a aplicação desenvolvida pelo grupo visa a classificação de espécies como invasoras

ou não e posterior obtenção de uma rota ótima para acesso a todas as espécies classificadas como invasoras.

As principais aplicações adequadas de acordo com este esboço do projeto são:

- Rastreamento e identificação de espécies invasoras em ambientes naturais para evitar a extinção de outras espécies;
- Evitar a disseminação de pragas em área de plantações de grande porte;
- Identificação de padrões a partir de um conjunto de imagens;
- Determinação de clusters entre os pontos de interesse;
- Cálculo de menor rota aérea a partir de qualquer localização geográfica, dentro das limitações do meio de transporte utilizado (drone);

## V. METODOLOGIA DE DESENVOLVIMENTO

Para realizar o projeto dentro do prazo determinado, o grupo precisou ser dividido em duas equipes. A metodologia utilizada em ambas as equipes foi: O desenvolvimento iterativo e incremental.

O Desenvolvimento Incremental é uma estratégia de planejamento estagiado em que várias partes do sistema são desenvolvidas em paralelo, e integradas quando completas. A alternativa ao desenvolvimento incremental é desenvolver todo o sistema com uma integração única.

O Desenvolvimento iterativo é uma estratégia de planejamento de retrabalho em que o tempo de revisão e melhorias de partes do sistema é pré-definido. Isto não pressupõe desenvolvimento incremental, mas funciona muito bem com ele. Uma diferença típica é que a saída de um incremento não é necessariamente assunto de um refinamento futuro, e seu teste ou retorno do usuário não é utilizado como entrada para planos de revisão ou especificações para incrementos sucessivos. Ao contrário, a saída de uma iteração é examinada para modificação, e especialmente para revisão dos objetivos das iterações sucessivas.

## VI. DIAGRAMAS SYSML

Com o objetivo de descrever o sistema e as interações existentes dentro dele mesmo e com o ambiente externo, são utilizados diagramas, que servem como uma ferramenta modelagem do sistema. Esses diagramas têm também a função de deixar claro o escopo do projeto, quais serão as funcionalidades que o desenvolvedor se compromete a entregar ao cliente, bem como a interação entre o sistema, os usuários, os stakeholders e o ambiente de utilização do mesmo. São apresentados, portanto, cinco diagramas que se relacionam entre si para dar uma visão geral do sistema: o diagrama de requisitos, de atividades, de estados, de pacotes e de casos de uso.

### A. Diagramas de Requisitos

O diagrama de requisitos é a base e o princípio do desenvolvimento do sistema, pois é a partir dele que o sistema passa a ser estruturado. Através dos requisitos são explicitadas, dentre outras coisas, as necessidades do cliente e as restrições do sistema, sejam físicas, de orçamento ou de outra natureza. Outros diagramas da SysML são extremamente dependentes desse diagrama. Por exemplo, os casos de uso de um sistema mostram quais partes do sistema e quais ações devem ser realizadas para uma determinada situação de operação do mesmo, o que está intimamente ligado às restrições do sistema e às funcionalidades dele. Da mesma forma, todos os outros diagramas necessitam desse diagrama como ponto de partida. E ao mesmo tempo, os outros diagramas servem como feedback, para refinar, melhorar e incrementar o próprio diagrama de requisitos, pois novos requisitos surgem a partir da perspectiva encontrada em outros diagramas. Isso aconteceu durante o desenvolvimento do projeto, por exemplo, com o diagrama de caso de uso, em que foi necessário alterar alguns dos requisitos originais e acrescentar outros que não estavam presentes. Foram feitos os seguintes diagramas:

- Funcionais

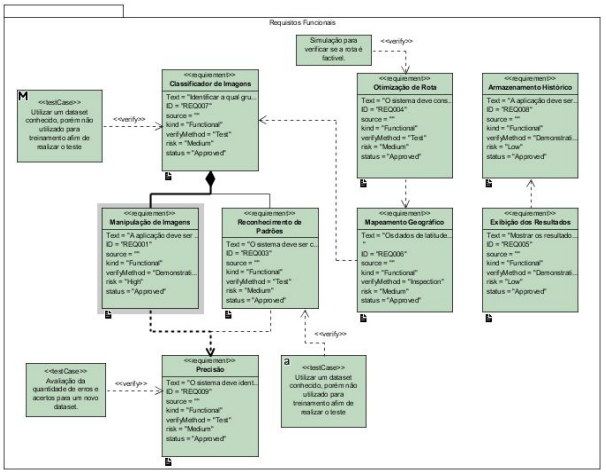


Figura 1: Diagrama de requisitos funcionais.

- Não Funcionais

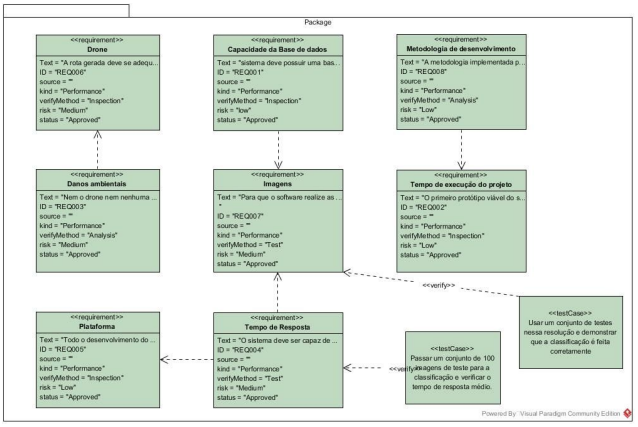


Figura 2: Diagrama de requisitos não funcionais.

- Interface

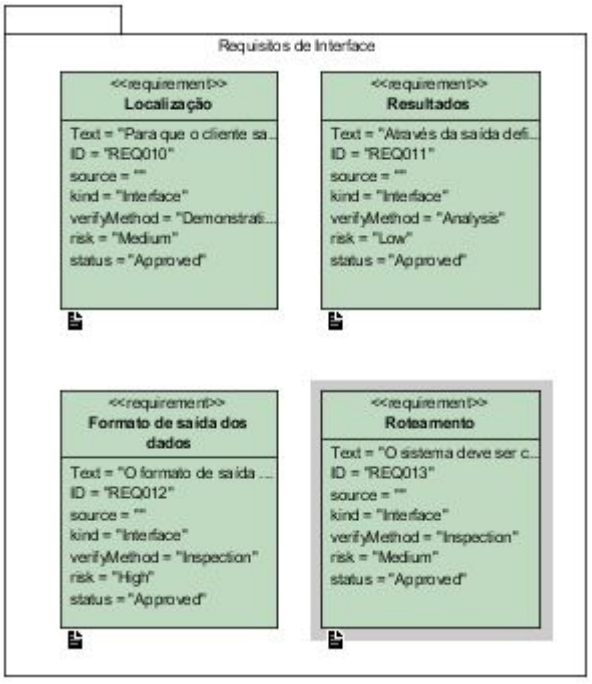


Figura 3: Diagrama de requisitos de interface.

B. Diagramas de Atividades

O diagrama de atividades tem como objetivo principal mostrar o comportamento do sistema, do ponto de vista funcional, isto é, das suas funcionalidades. A partir dele é possível entender como será a execução das funcionalidades e a respectiva atuação do sistema.

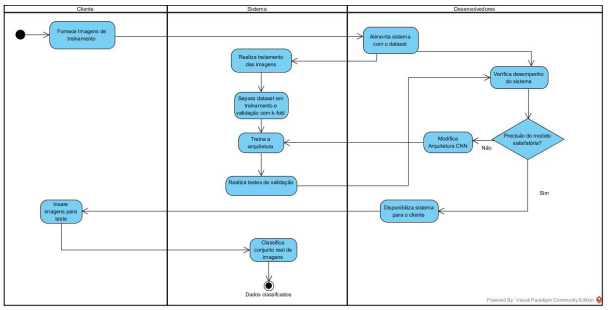


Figura 4: Diagrama de atividades entre cliente, sistema e desenvolvedores.

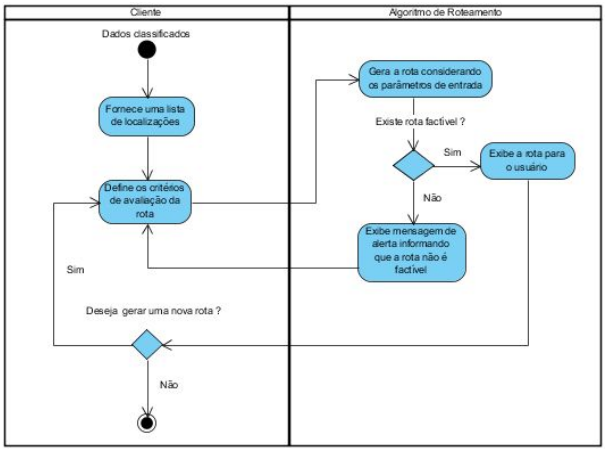


Figura 5: Diagrama de atividades entre cliente e algoritmo de roteamento.

C. Diagramas de Casos de Uso

O diagrama de caso de uso tem como objetivo auxiliar a comunicação entre os desenvolvedores do sistema e o cliente, descrevendo um cenário que mostra as funcionalidades do sistema do ponto de vista do usuário. Geralmente o diagrama de caso de uso é composto por atores, ações e os relacionamento entre atores e ações. Este diagrama está ligado diretamente com o diagrama de requisitos, principalmente nos requisitos funcionais do projeto, que por sua vez ilustra quais elementos externos estão presentes e como eles interagem com as funcionalidades do sistema.

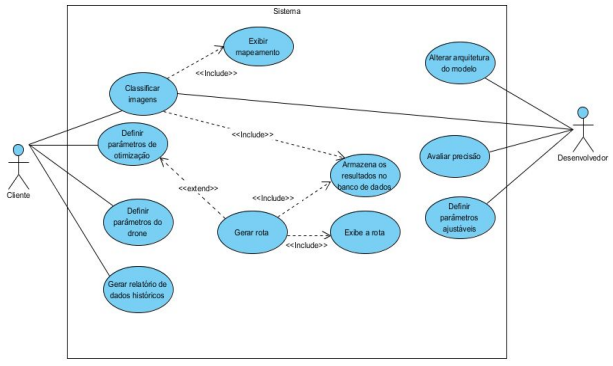


Figura 6: Diagrama caso de uso.

D. Diagramas de Pacotes

O diagrama de pacotes é uma forma conveniente de representar qualquer parte do processo/modelagem, sendo este diagrama composto por pacotes e relacionamentos entre eles. Para se definir os pacotes é necessário encontrar um grau de similaridade entre os elementos e agrupá-los, pois estes tendem a serem modificados em conjunto num mesmo pacote.

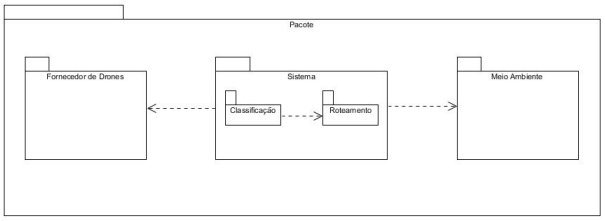


Figura 7: Diagrama de pacotes.

E. Diagramas de Máquina de Estados

O diagrama de máquina de estados tem como propósito mostrar o comportamento de cada parte do sistema como uma sequência de estados em que um componente pode estar dependendo de eventos externos ou interação com o mesmo. Neste diagrama fica definido o que o componente pode fazer em alguns cenários e a partir disso pode-se definir o limite do escopo do componente, ou seja, o que ele não é capaz de fazer.

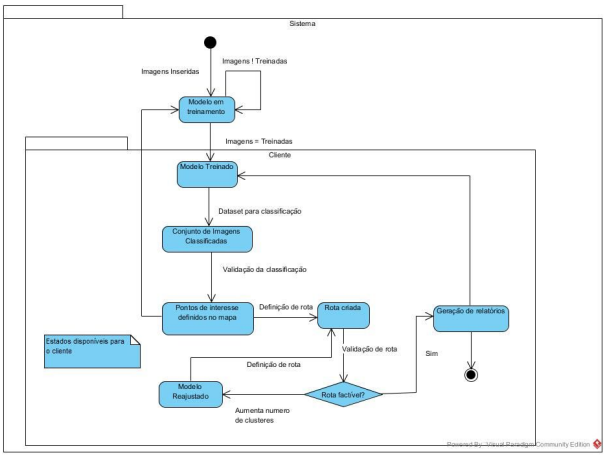


Figura 8: Diagrama máquina de estados.

## VII. MODELAGEM

O modelo para identificação das espécies invasoras foi construído utilizando a linguagem Python, com o auxílio da biblioteca de desenvolvimento Keras que possibilita o desenvolvimento de uma *Convolutional Neural Network (CNN)*.

O modelo de otimização também foi desenvolvido usando Python, mas em conjunto com o solver linear Lpsolve e a biblioteca *sklearn* para a clusterização dos dados.

A. Arquitetura CNN

A base de dados de treinamento foi dividida em dois conjuntos, um de treinamento e outro de validação, sendo 70% para treinamento e 30% para validação do modelo. A figura abaixo apresenta a distribuição os pontos de teste a serem classificados em um mapa.

Figura 9: Representação das coordenadas dos pontos de teste.

Uma CNN é composta por uma sequência de camadas. Além da camada de entrada, que normalmente é composta por uma imagem com largura, altura e profundidade (RGB), existem três camadas principais: camada convolucional, camada de pooling e camada totalmente conectada. Além disso, após uma camada de convolução é comum uma camada de ativação (normalmente uma função ReLu). Essas camadas, quando colocadas em sequência (ou empilhadas), formam uma arquitetura de uma CNN, como ilustrada na figura abaixo.

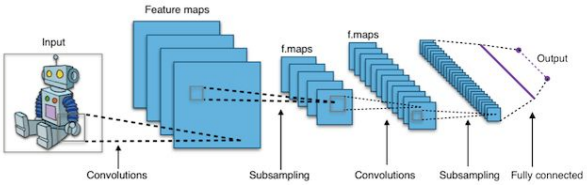


Figura 10: Arquitetura típica de uma *Convolutional Neural Network (CNN)*.

O passo de treinamento do modelo. Nesta fase, serão ajustados os pesos das conexões. É importante considerar, nesta fase, alguns aspectos tais como o modo de treinamento e o tempo de treinamento.

Quanto ao modo de treinamento, na prática é mais utilizado o modo padrão devido ao menor armazenamento de dados, além de ser menos suscetível

ao problema de mínimos locais, devido à pesquisa de natureza estocástica que realiza. Por outro lado, no modo batch se tem uma melhor estimativa do vetor gradiente, o que torna o treinamento mais estável. A eficiência relativa dos dois modos de treinamento depende do problema que está sendo tratado.

Quanto ao tempo de treinamento, vários fatores podem influenciar a sua duração, porém sempre será necessário utilizar algum critério de parada. Mas, devem ser considerados a taxa de erro médio por ciclo, e a capacidade de generalização do modelo. Pode ocorrer que em um determinado instante do treinamento a generalização comece a degenerar, causando o problema de over-training, ou seja o modelo se especializa no conjunto de dados do treinamento e perde a capacidade de generalização.

O treinamento deve ser interrompido quando o modelo apresentar uma boa capacidade de generalização e quando a taxa de erro for suficientemente pequena, ou seja menor que um erro admissível. Assim, deve-se encontrar um ponto ótimo de parada com erro mínimo e capacidade de generalização máxima.

As funções de ativação utilizadas no treinamento da CNN foram; Unidade Linear Retificada (ReLU) e Sigmóide.

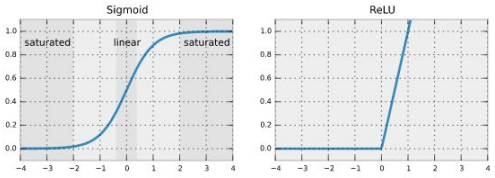


Figura 11: Funções de ativação utilizadas.

O último passo é o teste da CNN. Durante esta fase o conjunto de teste é utilizado para determinar a performance do modelo com dados que não foram previamente utilizados. A performance da arquitetura, medida nesta fase, é uma boa indicação de sua performance real.
testes.

Devem ser considerados ainda outros testes como análise do comportamento do modelo utilizando entradas especiais e análise dos pesos atuais do modelo, pois se existirem valores muito pequenos, as conexões associadas podem ser consideradas insignificantes e assim serem eliminadas (prunning). De modo inverso, valores substantivamente maiores que os outros poderiam indicar que houve over-training do modelo.

Figura 12: Saída do classificador: coordenadas dos pontos afetados

B. Otimização de rotas

A otimização de rotas foi dividida em duas etapas: clusterização, para determinar de forma aproximada a melhor distribuição de subgrupos dentro da totalidade de pontos existentes para o roteamento e posterior uso de um modelo de otimização linear para determinação da melhor sequência a ser percorrida levando em consideração as demais restrições e a minimização de tempo e custo.

Para tornar a ferramenta personalizável as necessidades de cada cliente definimos uma série de parâmetros de otimização que podem ser ajustáveis, levando em consideração caracteristicas específicas dos drones utilizados. Segue abaixo a lista de parâmetros personalizáveis:

- Quantidade de drones;
- Capacidade do drone
    - Velocidade,
    - Distância máxima,
    - Tempo máximo de vôo,
    - Carga máxima suportada,
    - Tempo de recarga e manutenção;
- Carga de inseticida necessária para a viagem;
- Quantidade e localização de pontos acessíveis;
- Margem de segurança (por padrão 0.75);
- Número máximo de clusters.

A abordagem utilizada para encontrar a solução do problema foi estruturada como um problema de minimização na forma padrão, para que ele pudesse ser resolvido com o *lpsolve*. Essa ferramenta foi escolhida por ser gratuita e por ser eficiente levando em consideração a complexidade do problema a ser resolvido. A função objetivo consiste em em minimizar a distância total que será percorrida pelo drone para remover as ameaças. Como parâmetros de avaliação do

modelo o número de clusters e o tempo total gasto para percorrer as rotas. Durante a implementação buscamos minimizar ambos os critérios sendo que apenas a minimização de tempo foi usada como função objetivo do otimizador linear.

**Clusterização**

A clusterização, particionamento de dados em grupos de forma que as observações existentes dentro de cada um deles sejam similares, é uma tarefa comum para análise estatística de dados usada em muitas áreas, como por exemplo mineração de dados, reconhecimento de padrões, processamento de imagens, aprendizado de máquina e bioinformática. Diversos métodos são usados para a realização deste processo, sendo um deles o k-means, onde cada uma das presentes observações pertence ao grupo de média mais próxima a ele. O problema computacional *k-means* é NP-difícil, sendo comumente tratado por heurísticas e tem como função objetivo a minimização da variância dentro de um mesmo cluster (ou maximização da diferença entre conjuntos). Isso quer dizer que em um dado grupo queremos que todos os elementos sejam tão próximos quanto possível da média dos demais indivíduos desse conjunto.

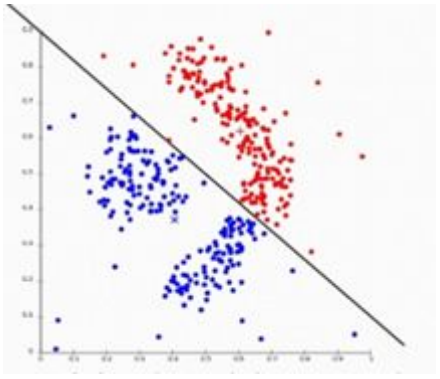

Figura 13: Separação de classes no K-Means: cada elemento pertence exatamente a um grupo

Para implementação deve-se escolher um número K de grupos em que os dados serão agrupados. São escolhidos K centros aleatoriamente e enquanto o centro continua sendo alterado calcula-se a distância entre todos os pontos e cada um dos centros os pontos são atribuídos ao centro em que o elemento possui um maior grau de proximidade, após isso calcula-se novamente o centro do grupo. Apesar de ser eficiente, o algoritmo *k-means* frequentemente converge para um mínimo local e o resultado depende fortemente do número de clusters escolhidos.

A Minimização do número de clusters foi feita a partir da verificação da viabilidade de uma distribuição de grupos levando em consideração a distância máxima entre dois pontos e pelo número de pontos existentes em cada cluster. Após encontrarmos um número de clusters aparentemente viáveis, a busca pela rota ótima era realizada. Caso se verificasse que este número de clusters era factível, o algoritmo avalia com um número inferior de grupos. Se o número encontrado ainda não era realmente factível aumentava se esse número até encontrar um solução viável.

$$pontos <= maxCargaDrone/pesoInseticida$$

**Otimização Linear**

O tempo para um único cluster pode ser dado pelo somatório da distância entre os pontos multiplicado pela velocidade média do drone para aquela carga, além de um minuto extra por ponto percorrido, referente a descida até o ponto onde a espécie invasora se encontra e liberação do inseticida.

$$min\ distanciaTotal * velocidadeDrone$$

Para todos os clusters entretanto, é necessário levar em consideração também o número de drones sendo utilizados e o tempo que cada um precisa para manutenção e recarga entre voos, sendo o número de drones e o tempo de recarga informado pelo usuário. Tais parâmetros são específicos de cada drone e devem ser informados ao modelo.

$$min \sum_{i}^{N} distanciaTotal_i * velocidadeDrone + N * Tempo\ de\ recarga\ /Num\_drones$$

O problema foi modelado com um grafo de conexões entre pontos, representado por uma matriz de adjacências binária *np* por *np*, onde *np* corresponde ao número de pontos de um determinado cluster. O um nesta matriz representa a existência de conexão entre dois pontos. Por exemplo, se temos o valor um na quinta linha quarta coluna, sabemos que ao sair do ponto denominado 1 o drone se dirigira ao ponto 4.

**Restrições**

A modelagem do roteamento e análise das restrições necessárias foi alterada nesta etapa visando corrigir pro

A matriz de distância possui valor infinito na diagonal principal para indicar que aquele nó não pode ser ligado a ele mesmo.

$$distanciaTotal >= \sum_{i}^{np}\sum_{j}^{np} matrizDistancia_{ij} * matrizConexoes_{ij}$$

$$\sum_{j}^{np} matrizConexoes >= 1\ \forall\ linha\ i\ em\ np$$

$$-\sum_{j}^{np} matrizConexoes <= -1\ \forall\ linha\ i\ em\ np$$

$$\sum_{i}^{np} matrizConexoes <= |S| - 1\ \forall\ coluna\ j\ em\ np$$

A variável matrizConexoes assume o valor 1 se o arco (i,j) $\varepsilon A$ for escolhido para integrar a solução e 0, caso contrário, S é o subgrafo de G, em que |S| representa o número de vértices deste subgrafo. Nesta formulação está implícito que não existe a diagonal principal e tem-se n(n-1) variáveis inteiras 0-1.

**Saída do modelo**

A saída do modelo é dado por um mapa representando as ligações entre os pontos de um mesmo cluster e os dados referentes ao modelo como tempo total e número de viagens realizadas por cada drone. Esses gráficos foram feitos com o auxílio da biblioteca plotly, mas ainda não foram unificadas com o restante das informações em um único relatório. É realizada também uma priorização dos clusters a serem atacados levando em consideração a densidade de espécies invasoras.

**Possíveis Melhorias**

Uma possível melhoria a ser feita no modelo na próxima iteração seria a implementação da minimização dos custos e não somente do tempo, o que poderia ser usado talvez para auxiliar na escolha do drone a ser comprado para atender aos requisitos mínimos do cliente.

## VIII. INTERFACE

Como dito anteriormente, a interface do sistema foi construída utilizando o *framework dash* em associação com gráficos do plotly. O Dash é uma ferramenta recente de código aberto criado pela equipe plotly que possibilita criar aplicativos personalizados de visualização de dados. Ele pode ser associado com javascript, CSS e associado com outras tecnologias externas o que traz muitas possibilidades para seu uso. Neste projeto, usamos o framework para a criação da interface gráfica e definição do layout do relatório

**Possíveis Melhorias**

Uma possível melhoria a ser feita é o bloqueio de funcionalidades dependendo da etapa em no modelo na próxima iteração seria a implementação da minimização dos custos e não somente do tempo, o que poderia ser usado talvez para auxiliar na escolha do drone a ser comprado para atender aos requisitos mínimos do cliente.

## IX. SIMULAÇÃO

Inicialmente, foram criados 10 modelos para realizar a predição das imagens, considerando apenas 20 iterações para o treinamento. Os resultados referente ao desempenho do treinamento de cada modelo será exibido nas imagens abaixo:

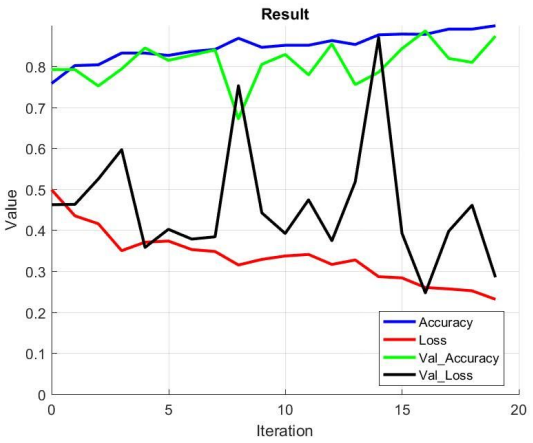


Figura 14: Gráfico do desempenho do modelo 1.

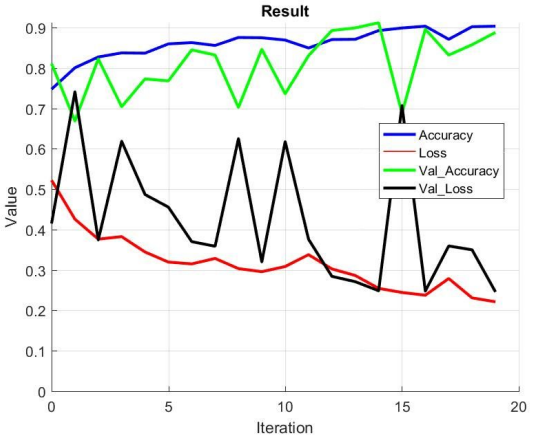


Figura 15: Gráfico do desempenho do modelo 2.

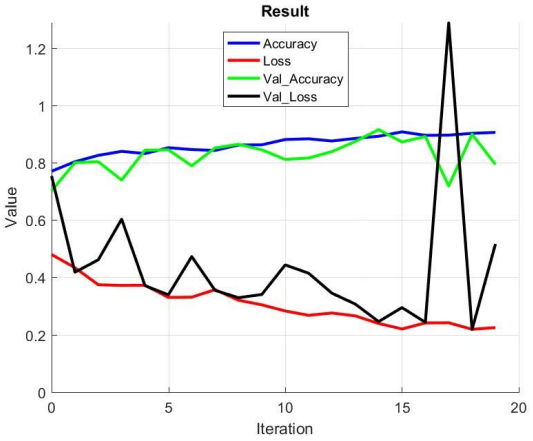


Figura 16: Gráfico do desempenho do modelo 3.

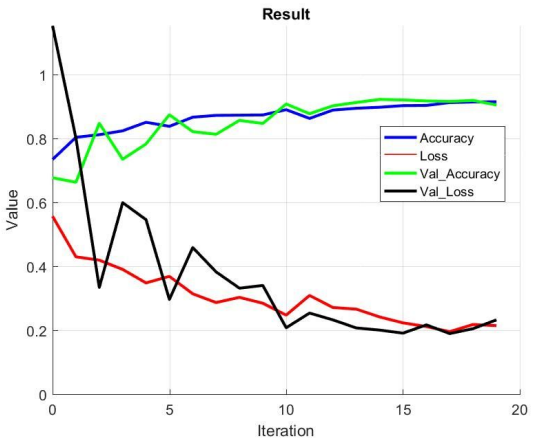


Figura 17: Gráfico do desempenho do modelo 4.

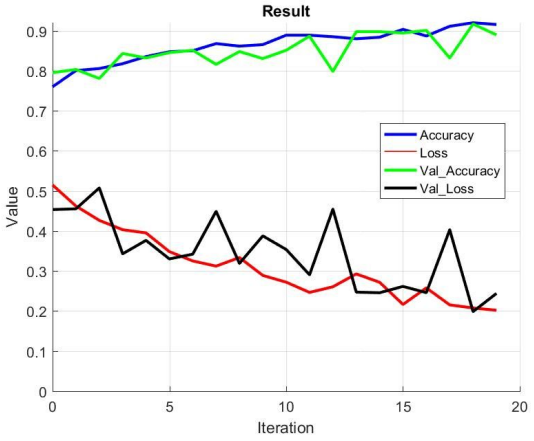


Figura 18: Gráfico do desempenho do modelo 5.

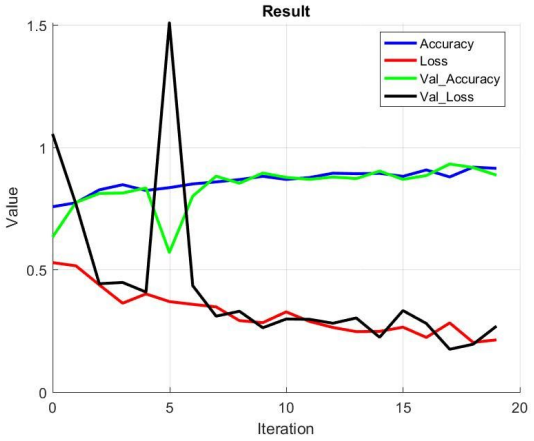


Figura 19: Gráfico do desempenho do modelo 6.

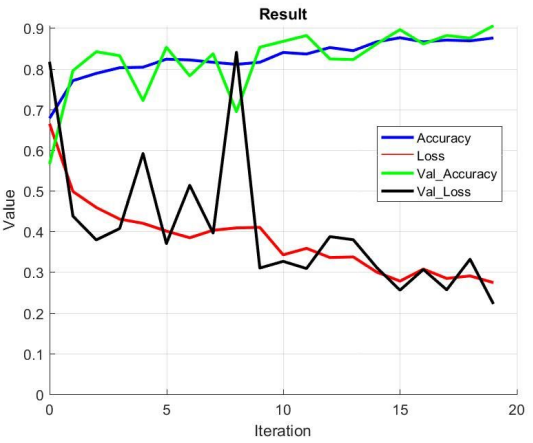


Figura 20: Gráfico do desempenho do modelo 7.

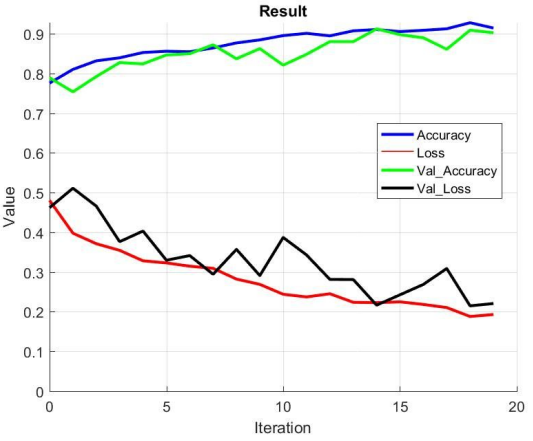


Figura 21: Gráfico do desempenho do modelo 8.

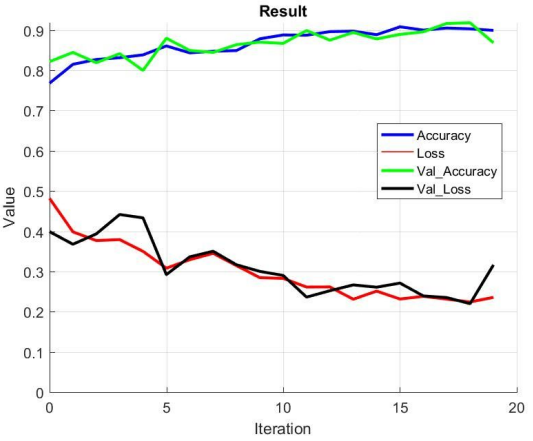


Figura 22: Gráfico do desempenho do modelo 9.

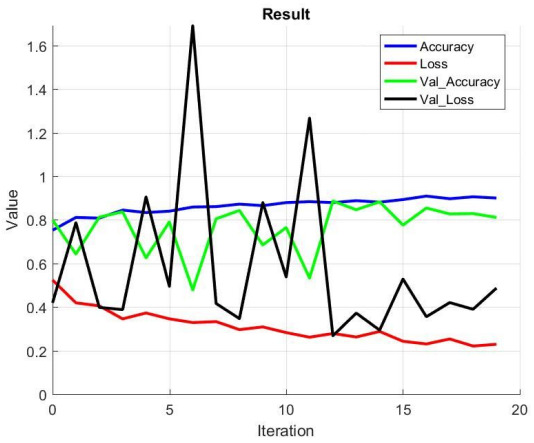


Figura 23: Gráfico do desempenho do modelo 10.

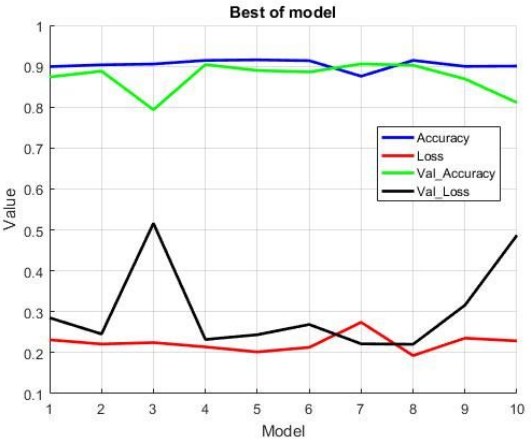


Figura 24: Gráfico do resultado final de cada modelo

Os valores do resultado final de cada modelo está presente na tabela a seguir:

Tabela 1: Resultados obtidos para cada modelo.

| **Modelo** | **Treinamento** | | **Validação** | |
|---|---|---|---|---|
| | Acurácia | Loss | Acurácia | Loss |
| **1** | 0.8990 | 0.2314 | 0.8736 | 0.2852 |
| **2** | 0.9034 | 0.2211 | 0.8880 | 0.2457 |
| **3** | 0.9053 | 0.2246 | 0.7936 | 0.5162 |
| **4** | 0.9142 | 0.2141 | 0.9040 | 0.2322 |
| **5** | 0.9155 | 0.2018 | 0.8896 | 0.2439 |
| **6** | 0.9136 | 0.2130 | 0.8859 | 0.2689 |
| **7** | 0.8755 | 0.2741 | 0.9056 | 0.2216 |
| **8** | 0.9142 | 0.1929 | 0.9024 | 0.2205 |
| **9** | 0.8996 | 0.2355 | 0.8688 | 0.3161 |
| **10** | 0.9003 | 0.2288 | 0.8112 | 0.4868 |

Conclusões retiradas a partir dos modelos testados:

- Modelos 1/2/3: O aumento da densidade pode causar maior erro;
- Modelos 4/5/6: Alterar a densidade não influencia muito no resultado quando as camadas de convolução tem valores altos;
- Modelos 1/7/8: Diminuir o dropout diminui o erro, mas pode gerar overfitting;
- Modelos 1/9/10: Variar o parâmetro de normalização para valores “extremos” (entre 0 e 1) aumenta o erro do modelo;

Com base nas conclusões retiradas referente ao resultado de treinamento e validação de cada arquitetura, considerando os parâmetros de acurácia e loss. O modelo 8 foi o que apresentou o melhor comportamento analisando estes dois parâmetros. Pois as curvas de treinamento e validação estão bem próximas umas das outras e também por apresentar uma baixa variação ao longo das iterações em relação às outras arquiteturas testadas.

## X. TESTE DO PRODUTO

Para realizar os testes de desempenho e eficiência do produto, foi criada uma máquina virtual Windows 10 64 bits, 14 GB de memória RAM, Intel Xeon E5-2660 @2.2 Ghz.

Os testes do produto foram executados utilizando a interface criada, conforme já descrita. No primeiro momento é aberto uma página WEB, onde é mostrada um mapa contendo um conjunto de pontos de teste (em azul).

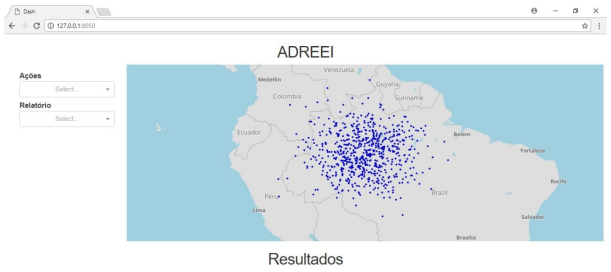


Figura 25: Tela inicial da interface do ADREEI.

Em seguida o botão *select* fica disponível para clicar em *treinar*. Ao selecionar esta opção, o conjunto de imagens são passadas para o modelo de classificação para identificar as possíveis espécies invasivas contidas neste dataset. Ao término da classificação, o mapa é atualizado e cada amostra é exibida no mapa com as cores vermelha e verde, sendo vermelho as imagens consideradas como contendo espécies invasoras e verde as não invasoras.

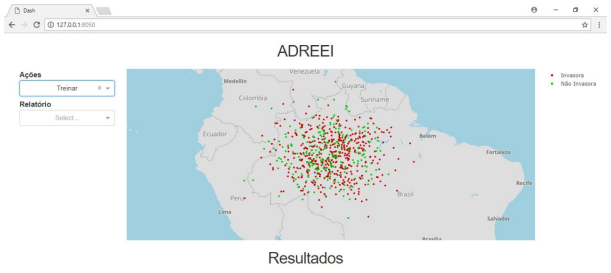


Figura 26: Resultado da classificação de um conjunto de dados de teste.

Em seguida o botão *select* fica disponível para clicar em *gerar rotas*. Ao selecionar esta opção, o conjunto de imagens detectadas como invasivas são passadas para o algoritmo de geração de rotas para encontrar a quantidade de clusters presentes nos dados e em seguida gerar a rota de caminho mínimo para cada cluster conforme mostrado na Figura 27.


Figura 27: Resultado da geração de rotas para eliminação do conjunto de espécies invasivas.

Logo após gerar a rota de caminho mínimo para cada cluster, é informado algumas informações relevantes sobre o modelo, tais como: Acurácia do modelo, número de espécies classificadas como invasoras e o número de clusters gerado.

## XI. IMPACTOS

Como o projeto envolve diretamente ambientes naturais, um dos impactos socioambientais é evitar a dispersão de espécies inadequadas em uma determinada área, ajudando em manter a ordem natural. Além disso, ajuda a biólogos a entender o que poderia estar causando o deslocamento de algumas espécies para fora do seu habitat natural.

Impede a perda em produção de alimentos em larga escala pois os veículos não tripulados podem aplicar fertilizantes, pesticidas e herbicidas, auxiliando economicamente aos agricultores em garantir a produtividade de sua área. Outro impacto é a inserção de novas tecnologias para aprimorar o sistema de produção, influenciando fortemente no setor primário em nível mundial.

## XII. ANÁLISE DE RESULTADOS

Após testes e análises dos resultados obtidos podemos perceber que os resultados encontrados pelo algoritmo são reais e factíveis, além de terem qualidade similar à esperada no início do projeto.

Em relação ao rede CNN para a classificação de imagens, obtivemos uma acurácia média de 92%, um pouco menor que a desejada inicialmente mas aceitável para o projeto atual e com margem para melhorias em relação ao tratamento prévio das imagens. O tempo para a classificação obtido foi menor que o almejado, sendo possível classificar 600 imagens em pouco mais de um minuto em média.

O modelo de otimização do roteamento apresentou maiores dificuldades do que o previsto pelo grupo, tanto pelo surgimento de problemas inesperados relacionados a versão utilizada inicialmente do LPSolve, que não atendia bem às restrições de binaridade das variáveis, quanto pela necessidade de melhorias nas restrições propostas para que ele atendesse realmente a necessidade do projeto. Na segunda etapa de implementação, onde teríamos principalmente a integração entre as partes do sistema, criação da interface e do relatório para o cliente, uma boa parte do tempo foi gasta resolvendo questões referentes a melhoria da modelagem. Em relação a velocidade de execução, conseguimos realizar o roteamento para 45 clusters de 15 pontos em aproximadamente dois minutos em média, sem a paralelização do código. A versão paralelizada da geração de rotas começou a ser implementada, mas o grupo não teve tempo hábil para validar a sua execução.

Levando em consideração à gestão de projetos como um todo, acreditamos que o projeto teve um bom resultado. Com exceção da modelagem, todas as etapas

do projeto atenderam o cronograma proposto. Todos os membros do grupo apoiaram em todas as partes referentes ao desenvolvimento, apesar de manterem o foco em atividades distintas, o que permitiu que todos se capacitarem nas tecnologias abordadas neste trabalho.

## XIII. CONCLUSÕES

O projeto alcançou boa parte das expectativas criadas no contexto inicial, dentro das limitações de tempo e ferramentas disponíveis para a execução do projeto.

Além disso, o sistema criado é capaz de mostrar os resultados de maneira compreensível ao cliente final, apesar dele não ter consciência dos métodos utilizados e da complexidade do projeto.

Já com relação à escalabilidade, apesar do produto ter sido aplicado a somente um banco de dados de uma única espécie, o produto é capaz de ser utilizado a partir de outras fontes de informação desde que atenda às características essenciais das restrições utilizadas, tais como: qualidade de imagem, quantidade de amostras, capacidade do drone, localização referente a cada imagem, dentre outras.

## XIV. ANEXOS

Segue anexo ao trabalho o documento "*Levantamento de Requisitos - ADREEI.pdf*" contendo a elicitação e especificação de requisitos realizados na etapa anterior, pois este documento foi utilizado como referência para a criação de todos os diagramas SYSML. Para melhor leitura e interpretação dos diagramas, gerou-se o arquivo "*Diagramas SYSML - ADREEI.pdf*" com todos os diagramas em uma maior escala.

## XV. REFERÊNCIAS


[1] Kaggle.com. (2017). Invasive Species Monitoring | Kaggle. [online] Disponivel em: https://www.kaggle.com/c/invasive-species-monitoring #description [ 7 Agosto de 2017].

[2] Cienc. Cult. vol.61 no.1 São Paulo 2009 O IMPACTO DAS PLANTAS INVASORAS NOS RECURSOS NATURAIS DE AMBIENTES TERRESTRES - ALGUNS CASOS BRASILEIROS. Dalva M. Silva Matos e Vânia R. Pivello.

[3] VARGAS, A. C. G. , VASCONCELOS A. P. VASCONCELOS C. N. Um estudo sobre Redes Neurais Convolucionais e sua aplicação em detecção de pedestres.

[4] DLUGOSZ, F. L.; ROSOT, N. C.; ROSOT, M. A. D.; OLIVEIRA, Y. M. M.; ARRASTAZU, M. C. Uso do levantamento aéreo expedito convencional e digital para o monitoramento da cobertura florestal no Paraná: estado da arte e potencialidades. Pesquisa Florestal Brasileira, Colombo, PR, v. 30, n. 63, p. 245-252, ago./out. 2010.

[5] FERREIRA, S. ALEXANDRE. Redes Neurais Convolucionais Profundas na Detecção de Plantas Daninhas em Lavoura de Soja. Março 2017.

[6] LAKHANI, Paras. Deep Convolutional Neural Networks for Endotracheal Tube Position and X-ray Image Classification: Challenges and Opportunities. 9 de junho de 2017

[7] SUN, X. HUINAN, Q. Chinese Herbal Medicine Image Recognition and Retrieval by Convolutional Neural Network. 03 de Julho de 2016.

[8] FERREIRA, A. S. Redes Neurais Convolucionais Profundas na Detecção de Plantas Daninhas em Lavoura de Soja, 2017.

[9] ARAÚJO Tereza, ARESTA Guilherme, CASTRO Eduardo, ROUCO José, AGUIAR Paulo, Eloy Catarina, POLONIA António, CAMPILHO Aurélio. Classification of breast cancer histology images using Convolutional Neural Networks. 01 de julho de 2017.