

UNIVERSIDADE FEDERAL DE VIÇOSA

PATRICK ÁGTON DE OLIVEIRA

DEEP LEARNING NA SEGMENTAÇÃO AUTOMÁTICA DE IMAGENS DE SATÉLITE

**VIÇOSA - MINAS GERAIS
2020**

PATRICK ÁGTON DE OLIVEIRA

DEEP LEARNING NA SEGMENTAÇÃO AUTOMÁTICA DE IMAGENS DE SATÉLITE

Relatório final, apresentado a Universidade Federal de Viçosa, como parte das exigências, para obtenção do título de Engenheiro Agrícola e Ambiental.

Orientador: Domingos Sárvio Magalhães Valente

VIÇOSA - MINAS GERAIS
2020

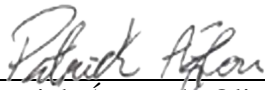
PATRICK ÁGTON DE OLIVEIRA

**DEEP LEARNING NA SEGMENTAÇÃO AUTOMÁTICA DE IMAGENS DE
SATÉLITE**


Relatório final, apresentado a Universidade Federal de Viçosa, como parte das exigências, para obtenção do título de Engenheiro Agrícola e Ambiental.

APROVADO: 01 de dezembro de 2020.

Assentimento:



Patrick Ágton de Oliveira
Autor



Domingos Sárvio Magalhães Valente
Orientador

*A meus pais, irmãos e amigos pelo incentivo
e carinho durante a minha trajetória*

DEDICO

“Cada ser humano é uma pequena sociedade” (Novalis)

RESUMO

A classificação de uso e ocupação do solo é um processo de grande importância para o manejo adequado de áreas, contudo, pode ser susceptível a erros humanos, visto que é feito de forma manual e, também, pode ser trabalhoso, principalmente se for feita em grandes áreas. Dado essa problemática, nos últimos anos a inteligência artificial vem ganhando grande papel na segmentação de imagens para diferentes fins e, aproveitando dessas ferramentas, este trabalho propôs criar um modelo de aprendizado profundo, através de arquitetura *U-Net*, para classificação de uso e ocupação de solo. Para isto, foi definido uma bacia hidrográfica e, através do satélite *Sentinel 2*, foi selecionada uma imagem com boas condições que abrange toda extensão dessa bacia. Assim, foi criada uma máscara através de um programa de sistema de informação geográfica e estes, imagem e máscara, foram introduzidos em recortes na rede para geração de parâmetros e, então, pode-se avaliar a acurácia e perda do modelo gerado. Como conclusão, foi avaliada como positiva a inserção de inteligência artificial na classificação de imagens de satélite para uso e ocupação do solo, com o modelo apresentando acurácia máxima de até 95%.

Palavras-chave: aprendizado profundo; segmentação semântica; uso e ocupação do solo.

ABSTRACT

The classification of land use and occupation is a process of paramount importance for the proper management of areas, however, it can be susceptible to human errors, since it is done manually and it can also be laborious, especially if done in large areas. Given this problem, artificial intelligence has been playing a major role in the segmentation of images for different purposes in recent years and, taking advantage of these tools, this work proposed to create a deep learning model, through U-Net architecture, for classification of land use and occupation. For this, a hydrographic basin was defined and, using the Sentinel 2 satellite, an image with good conditions was selected that covers the entire extension of this basin. Thus, a mask was created through a geographic information system program and these, image and mask, were introduced in cutouts in the network to generate parameters and, then, the accuracy and loss of the generated model can be evaluated. As conclusion, the insertion of artificial intelligence in the classification of satellite images for land use and occupation was evaluated as positive, with the model showing maximum accuracy of up to 95%.

Keywords: *deep learning*; land use and occupation; semantic segmentation.

SUMÁRIO

1- INTRODUÇÃO	08
2- MATERIAIS E MÉTODOS.....	10
2.1- Desenvolvimento do algoritmo	10
2.1.1 Rede Neural Convolutacional <i>U-Net</i>	10
2.2- Análise de desempenho do algoritmo	11
2.2.1 Dados e Pré-processamento.....	11
2.2.2 Treinamento do algoritmo.....	13
2.2.3 Predição para classificação.....	15
2.2.4 Acurácia e avaliação do modelo gerado.....	16
3- RESULTADOS E DISCUSSÃO.....	17
3.1 Dados	17
3.2 Parâmetros.....	18
3.3 Perda e acurácia do modelo.....	19
4- CONCLUSÃO.....	21
5- REFERÊNCIAS.....	22
6- ANEXO.....	23

1 INTRODUÇÃO

O mapeamento de uso e ocupação do solo é fundamental para conhecimento das características ambientais, sociais e econômicas de uma bacia hidrográfica. Ele consiste em segmentar áreas de uma região determinada de acordo com seu uso ou ocupação atual, podendo este modificar de forma frequente. Dessa forma, há necessidade de constante monitoramento das informações de uso e ocupação do solo em uma bacia hidrográfica (FORESTI & HAMBURGUER, 1995 com; BITTENCOURT et al., 2006). Atualmente, este trabalho é realizado de forma manual e visual, em imagens de satélites utilizando softwares de SIG (Sistema de informação Geográfica), tais como, ARCGIS e QGIS. Esse é um processo laborioso que poderá levar dias de trabalho e, dependendo do tamanho da área, fica inviável este mapeamento de forma frequente. Assim, o desenvolvimento de ferramentas que melhorem a precisão e otimizem o processo de classificação se tornaram imprescindíveis para maior facilidade de manejo das bacias.

Assim, sistemas de informação geográficas se estabeleceram indispensáveis para preparação e análise de dados espaciais, criando, então, um poderoso grupo de ferramentas para classificação de uso e ocupação do solo. Porém, as formas de segmentação de imagens, obtidas através de sensoriamento remoto, utilizadas atualmente, ainda necessitam de supervisão humana, no qual são feitas manualmente com auxílio de algum programa de geoprocessamento. Contudo, nos últimos anos, a aprendizagem profunda (*deep learning*) se mostrou apta para ser empregada em diversos processos de classificação de imagens, incluindo aquelas provenientes de sensoriamento remoto (ZHANG; LIU; WANG, 2018).

O uso de aprendizado profundo em trabalhos de diferentes áreas passíveis de automação tem crescido ultimamente. Essa técnica envolve redes neurais que utilizam mais de duas camadas ocultas, sendo assim denominadas de profundas (ZHU et al., 2017). Uma Rede Neural Convolutacional (CNN- *Convolutional Neural Network*) se configura por ser uma variação de redes de Percéptrons de Múltiplas Camadas, originando-se da inspiração do processo biológico de processamentos de dados visuais (OLIVEIRA, 2019). Estudos indicam alta eficácia das redes neurais convolucionais no reconhecimento de imagens em larga escala, detecção de objetos e segmentação semântica (ZHU et al., 2017).

O desenvolvimento de redes neurais começou no final da década de 80, mas só recentemente vem se popularizando pelos grandes avanços tecnológicos, e com isso, foi proposta, em 2012, a rede neural *AlexNet*, precursora do movimento de CNN's que processam grande volume de imagens. Desde então, várias arquiteturas CNN's foram desenvolvidas como a *U-Net*, *VGG16*, *GoogleNet*, etc, todas elas com objetivo similar de segmentação e classificação, porém, com estruturas, número de camadas e filtros diferentes. A arquitetura *U-net*, proposta por Ronneberger et al .(2015), se mostrou mais rápida para trabalhos de segmentação de imagens e vem sendo globalmente utilizada. Essa estrutura consegue treinar uma quantidade de dados substanciais, utilizando-se de uma janela de contração para prever os pixels com maior potencial da classificação desejada (PAN et al., 2020). Além disso, a *U-Net* também pode ser empregada em trabalhos envolvendo poucos treinamentos, sendo, então, uma ferramenta com aplicação para diferentes tipos de segmentação semântica. Sendo assim, torna-se evidente a crescente necessidade de monitoramento da distribuição espacial em grandes áreas para gestão de recursos e controle espacial para manejo de bacias hidrográficas. Diante disso, com este trabalho tem-se por objetivo desenvolver um código em aprendizagem profunda para classificação de uso e ocupação de solo através de imagens de satélite e avaliar a acurácia da metodologia.

2 MATERIAL E MÉTODOS

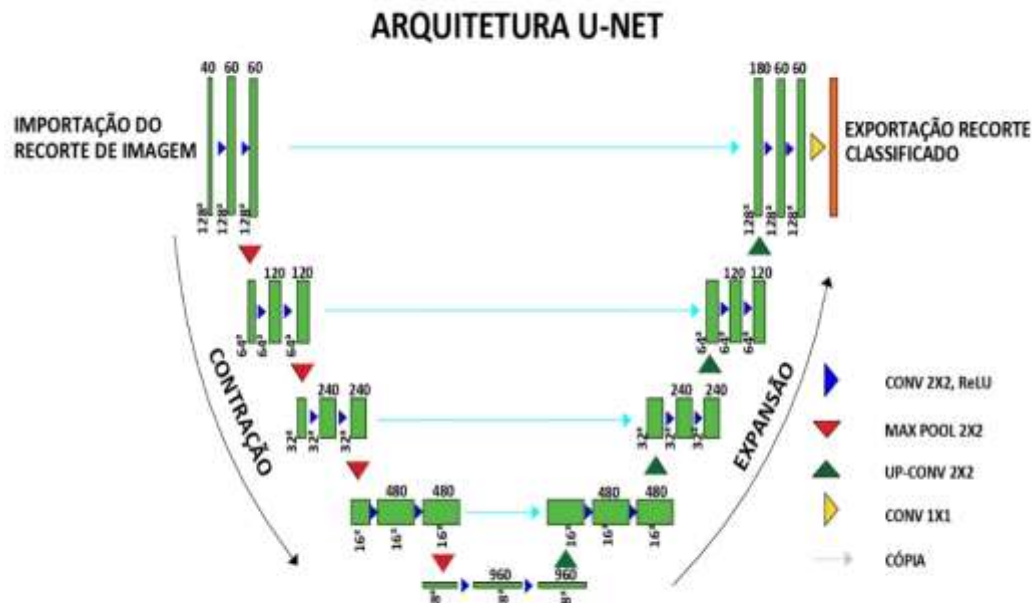
2.1 Desenvolvimento do algoritmo

A modelagem e predição serão realizadas utilizando-se de um *ACER ASPIRE, CORE I3, 4 GB, SSD 240 GB*, com sistema operacional *Windows 10*, sendo feita em ambiente do *Google Colab (Processamento em nuvem)*. Utilizou-se, também, das bibliotecas *Keras Python Deep Learning Library*, além do driver *TensorFlow* e de outras bibliotecas como *Numpy, Pandas, Scikit-learn, OpenCV, Gdal e Matplotlib*.

2.1.1 Rede Neural Convolutacional *U-Net*

No presente trabalho foi utilizada a rede neural *U-Net* que é uma estrutura composta por um filtro convolutacional que age, primeiramente, contraindo a imagem em diferentes resoluções, filtrando e detectando diferente estruturas e texturas. Cada nível foi representado por um passo correspondente ao nível de resolução daquele ponto. Após isso, houve a expansão também em níveis, fazendo interconexões entre imagens e escalas equivalentes (FLOOD; WATSON; COLLETT, 2019), como mostra a Figura 1.

Figura 1 – Arquitetura da *U-Net Neural Network*. Cada caixa representa um raster. O número acima delas representa o número de bandas e o ao lado ou abaixo o número de pixels



Fonte: Adaptado de Ronneberger et al. (2015).

2.2 Análise de Desempenho do Algoritmo

2.2.1 Dados e Pré-processamento

Para treinamento do código foram utilizadas imagens da bacia do São Bartolomeu, localizada no município de Viçosa. As imagens foram obtidas através do satélite *Sentinel 2*, do Programa Copernicus, financiado pela Agência Espacial Europeia (ESA), com resolução espacial de 10 metros, datando de 05 de novembro de 2019. Foram utilizadas 4 bandas do espectro, sendo elas R (vermelho), G (verde), B (azul) e IR (infravermelho).

Ao se obter as imagens, elas foram trabalhadas no software QGIS Desktop 3.10.8 (QGIS, 2020) por ser uma ferramenta aberta e muito importante para tratamento de dados de informação geográfica por ser fácil e intuitivo de usar (CORREIA et al., 2018). Assim, para as imagens de treinamento foi feita uma adequação ao formato da bacia e, após isso, a união das 4 camadas do espectro. Já para as máscaras, foram criadas 7 classificações de uso e ocupação do solo para serem segmentadas de forma manual e foram utilizadas como base, além da imagem do satélite Sentinel

2, datada de 05 de novembro de 2019, uma composição de bandas NDVI (*Normalized Difference Vegetation Index*) e, também, auxílio de imagens do *Google Earth* de mesma data. Após essa classificação o arquivo foi então, transformado em vetor para raster destacando a rotulação das 7 classificações em uma paleta de cores com variação de acordo com número de identificação (ID) que possui cada pixel. Na imagem de satélite do *Sentinel 2* utilizada, cada pixel possui 10 por 10 metros, tendo a imagem cerca de 576 km². Contudo, essa foi limitada à bacia estudada que tem uma área 313,75 km² e sua borda, visto que a bacia não é uniforme, não sendo de interesse os pontos fora da bacia. A Figura 2 mostra o passo a passo da preparação das imagens em ambiente do software de geoprocessamento.

Figura 2 – Passos para criação da máscara para treinamento do código



Fonte: O autor.

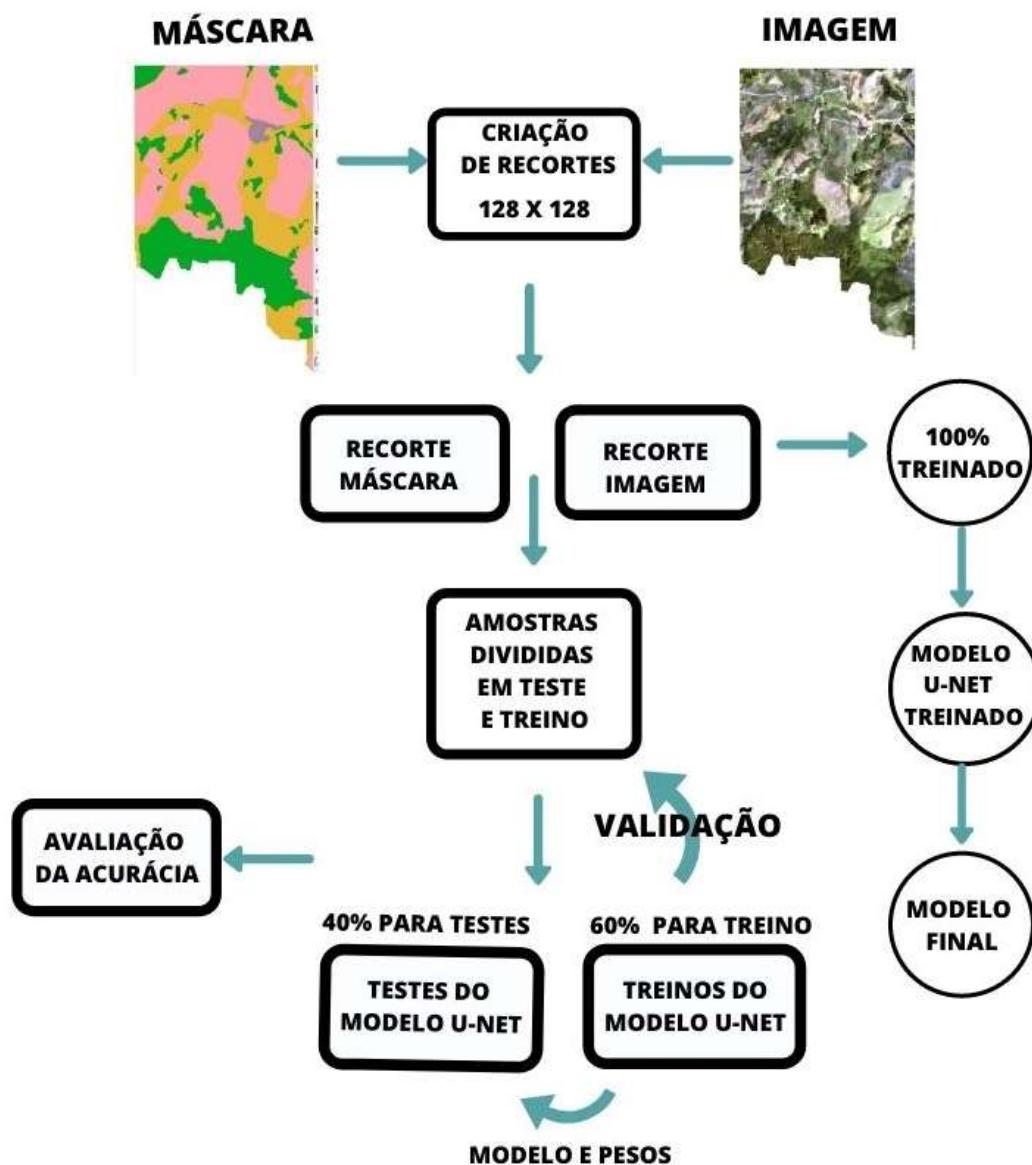
2.2.2 Treinamento do algoritmo

Para o treino das imagens, deve-se reduzi-las, pois, a arquitetura *U-Net* foi gerada para receber imagens com resolução de 128 x 128 pixels. Portanto, após essa redução, foram gerados 223 recortes de imagens com 4 bandas, sendo elas vermelho, verde, azul e infravermelho. Também foram gerados 223 recortes de máscaras, ou seja, recortes rotulados com 7 classificações, sendo elas floresta, pastagem, áreas urbanizadas, solo exposto, corpos d'água, silvicultura e áreas agrícolas.

Todas as imagens e máscaras passaram pela estrutura *U-Net* mostrada na Figura 1, em um processo iterativo longo, contendo contração e expansão. Sendo que quanto maior o número de imagens e máscaras dispostas ao treinamento do modelo, maior é a precisão da classificação. Assim, foram gerados 2000 recortes de imagens e máscaras aleatórias, visando, então, aumentar a acurácia do modelo. Ainda antes da estrutura *U-Net*, eles passaram por um filtro que avalia se o recorte está dentro ou fora da bacia, que é o local de estudo de uso e ocupação do solo.

Já dentro da estrutura os recortes passaram por reduções em sequência que puderam selecionar o objeto em questão através dos pixels com maior valor, sendo estes direcionados para um recorte cada vez menor, para, assim, aumentar a precisão de classificação, sendo essa fase denominada de contração. Cada recorte acompanha as 4 bandas da imagem de satélite e também sua máscara correspondente com as 7 classes determinadas. Após isso, no processo de expansão, esses recortes vão retornando à resolução inicial, sempre intercomunicando com seus pares de contração. A Figura 3 mostra como foi o processo de geração do modelo.

Figura 3 – Fluxograma dos processos de modelagem



Fonte: Adaptado de FLOOD et al. (2019a).

Os parâmetros utilizados para inserção na rede neural estão elencados no Quadro 1. A resolução foi determinada de acordo com o mais usual para estrutura *U-Net*. Já o número de bandas foi definido por aquelas que geralmente estão associadas ao trabalho de uso e classificação de solo, o mesmo critério sendo empregado ao número de classes. O tamanho do lote de recorte de imagens

foi definido para gerar uma maior acurácia ao processo e o número de épocas definido de forma arbitrária.

Quadro 1 – Características da estrutura *U-Net* utilizada (R-vermelho, G-verde, B-azul e IR-infravermelho)

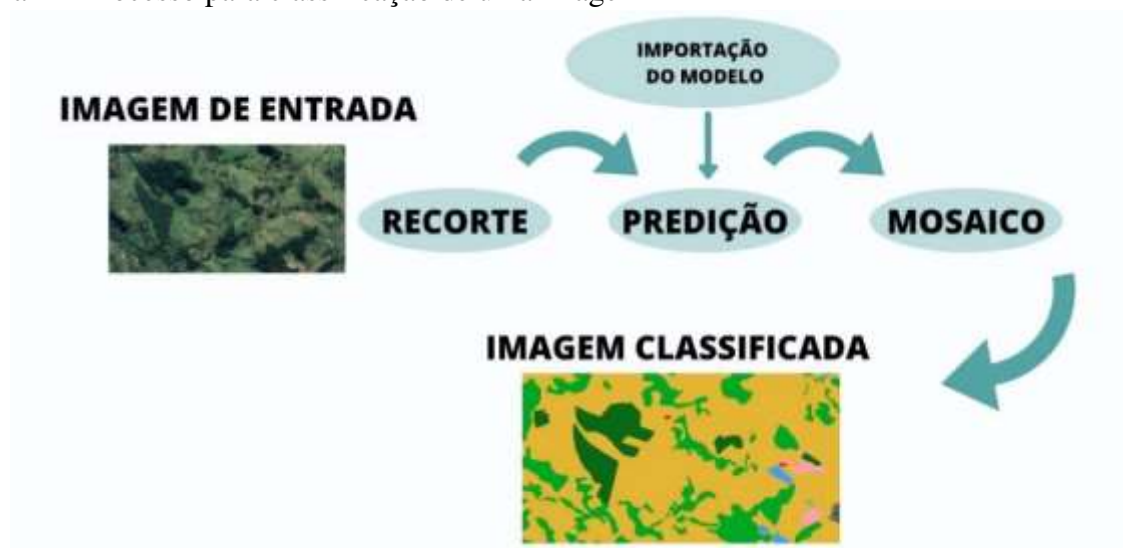
Resolução de imagens	128	128 x 128
Número de Bandas	4	R, G, B e IR
Número de Classes	7	Definido pelo autor
Tamanho do lote de imagens	10	Definido pelo autor
Número de épocas usadas	50	Definido pelo autor

Fonte: O autor.

2.2.3 Predição para classificação

O modelo desenvolvido, após passar por treinamento foi salvo, sendo importado ao ser chamado pela função de predição, a fim de aplicar o modelo para novas imagens, que retorna com as classes previstas para as imagens de recorte. As fases da classificação de novas imagens podem ser vistas na Figura 4; há importação da imagem; ela é recortada na resolução de 128 x 128; juntamente com o modelo ocorre a predição de segmentação das imagens; sendo o mosaico a união dos vários recortes; e como resultado dos passos anteriores é gerada a imagem classificada.

Figura 4 – Processo para classificação de uma imagem



Fonte: O autor.

2.2.4 Acurácia e avaliação do modelo gerado

A acurácia do algoritmo foi determinada após o treinamento do modelo da rede neural no qual estimou sua calibração frente as imagens disponibilizadas para testes, contando, assim, qual a porcentagem de classes corretas, baseadas no modelo treinado, foram preditas. A estrutura *U-Net* fornece classificação por pixel, sendo assim, a acurácia também é mensurada levando-se em conta os pixels individualmente (FLOOD; WATSON; COLLETT, 2019).

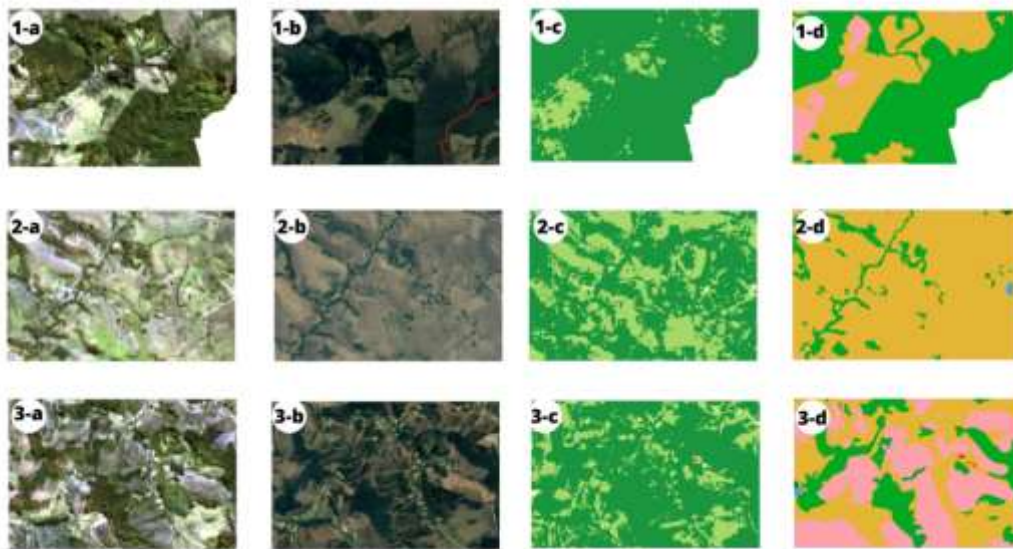
Foram utilizadas as funções de perda *Cross Entropy*, que tem por objetivo estimar as perdas ocorridas na saída da rede neural durante o treinamento. Já a função de otimização tem por objetivo aumentar a eficiência da rede neural, diminuindo as perdas ocorridas, sendo aqui utilizada a função *Adam*. Já para medir o desempenho da rede foi utilizado o módulo *Sklearn Metrics* que implementa várias funções para mensurar a acurácia.

3 RESULTADOS E DISCUSSÃO

3.1 Dados

A imagem obtida do *Sentinel 2* para o dia 05 de novembro de 2019 se encontra com boa visualização, visto a baixa incidência de nuvens, menos de 1%. Isso ajudou na criação da máscara, feita, também, com auxílio das imagens do *Google Earth* do mesmo dia. Além disso, pôde-se criar, como apoio, o NDVI. A Figura 5 mostra a comparação das imagens obtidas com a máscara criada no software QGIS Desktop 3.10.8 (QGIS, 2020) para geração do modelo.

Figura 5 – Imagens 1-a, 2-a e 3-a representam recortes de diferentes pontos da imagem de satélite utilizada para treinar o código. Imagens 1-b, 2-b e 3-b são imagens do *Google Earth* usadas como auxiliares para classificação da máscara, assim como as imagens NDVI 1-c, 2-c e 3-c. Enquanto as imagens 1-d, 2-d e 3-d são recortes da máscara criada também para treinamento do algoritmo. Todas as imagens são do dia 05/11/2019



Fonte: O autor.

A boa qualidade das imagens e a criação da máscara com precisão são fundamentais para o treinamento correto do código. Uma imagem com maior incidência de nuvens ou possíveis erros humanos na hora da geração da imagem rotulada, poderia levar o código a erros, que seriam, depois, repassados para a classificação das imagens preditas.

3.2 Parâmetros

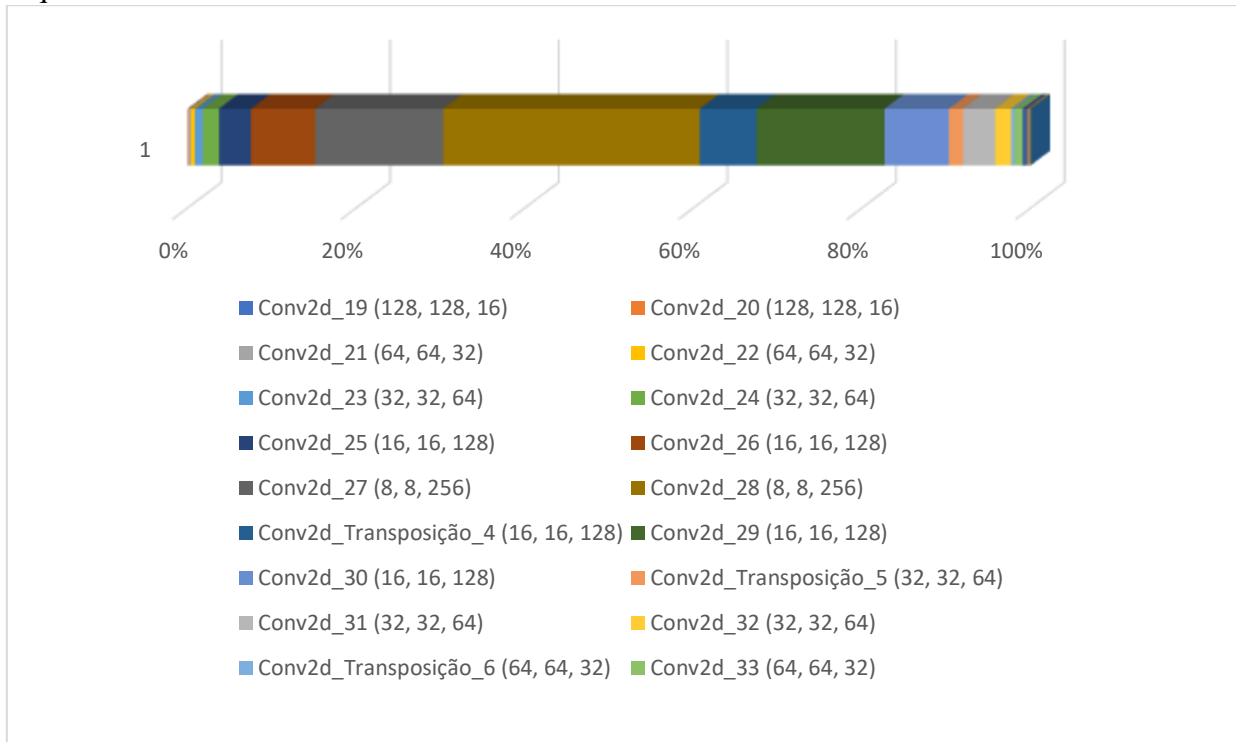
Ao passar por cada etapa convolucional, os recortes de imagens são multiplicados por filtros, que atribuem pesos e vieses em cada pixel do recorte e, ao acionar o *Max Pooling*, há a contração do recorte, sendo selecionado os pixels com maior valor. Quanto mais se avança na etapa de contração, maior o número de filtros e, consequentemente, de parâmetros gerados. A Tabela 1 mostra a quantidade de parâmetros gerados em cada etapa de convolução da arquitetura *U-Net* e o Gráfico 1 expõe a influência de cada uma dessas etapas no processo.

Tabela 1 – Quantidade de parâmetros gerados a cada etapa convolucional da arquitetura *U-Net*, criando, assim, uma rede neural com expressiva gama de dados para treinamento do modelo (Conv2D-Convolução 2D)

Convoluções	Resolução	Número de parâmetros
Conv2d_19	(128, 128, 16)	592
Conv2d_20	(128, 128, 16)	2320
Conv2d_21	(64, 64, 32)	4640
Conv2d_22	(64, 64, 32)	9248
Conv2d_23	(32, 32, 64)	18496
Conv2d_24	(32, 32, 64)	36928
Conv2d_25	(16, 16, 128)	73856
Conv2d_26	(16, 16, 128)	147584
Conv2d_27	(8, 8, 256)	295168
Conv2d_28	(8, 8, 256)	590080
Conv2d_Transposição_4	(16, 16, 128)	131200
Conv2d_29	(16, 16, 128)	295040
Conv2d_30	(16, 16, 128)	147584
Conv2d_Transposição_5	(32, 32, 64)	32832
Conv2d_31	(32, 32, 64)	73792
Conv2d_32	(32, 32, 64)	36928
Conv2d_Transposição_6	(64, 64, 32)	8224
Conv2d_33	(64, 64, 32)	18464
Conv2d_34	(64, 64, 32)	9248
Conv2d_Transposição_6	(32, 32, 16)	2064
Conv2d_35	(32, 32, 16)	4624
Conv2d_36	(32, 32, 16)	2320
Conv2d_37	(128, 128, 16)	119
Total		1.941.351

Fonte: O autor.

Gráfico 1- Comparação em porcentagem dos parâmetros gerados por etapa convolucional da arquitetura *U-Net*



Fonte: O autor.

Assim, diante da Tabela 1 e do Gráfico 1 percebe-se como funciona a arquitetura *U-Net*, comprimindo os recortes de imagens e selecionando os pixels com maior número de parâmetros semelhantes à máscara introduzida. E, como mostra o Gráfico 1, quanto maior a compressão, maior a quantidade de dados e a geração de parâmetros para treinamento. Neste trabalho, em específico, o número de quase 2 milhões de parâmetros mostra a quantidade de dados que a estrutura de segmentação de imagens utilizada consegue extrair, considerando as características de entrada do Quadro 1, sendo que outros trabalhos, como de Caron (2020), mostram essa quantidade expressiva de parâmetros conseguidos pela estrutura *U-Net*.

3.3 Perda e acurácia do modelo

O modelo gerado, ao ser submetido a um treinamento com 2000 recortes de imagens de 128 x 128 pixels, separados para treinamento em 50 épocas, mostrou uma acurácia de, aproximadamente, 95%, o que é um valor considerado suficiente para uma boa predição da classificação de novas imagens, valor este encontrado também em outros estudos, como por Reis

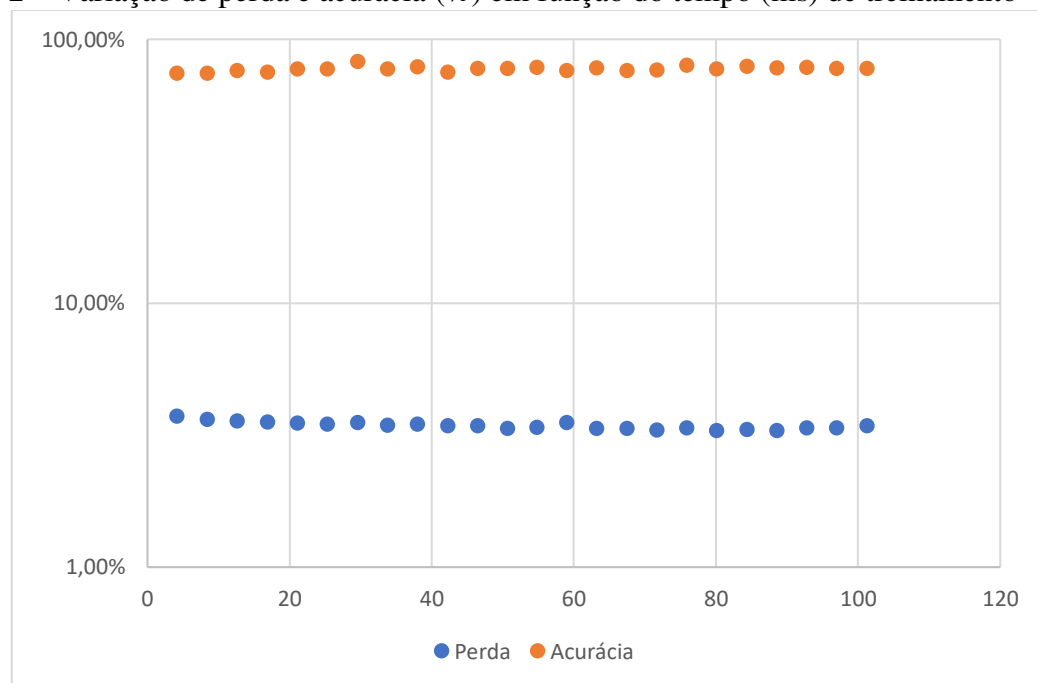
et al (2020). Já em relação as perdas, estas foram minimizadas, ficando por volta de 3%, o que, também, é considerado satisfatório, ficando um pouco acima do que o encontrado por Nunes (2018). O Quadro 2 mostra os resultados finais de acurácia e perda, já o Gráfico 2 mostra a variação destas em relação ao tempo de treinamento do modelo em uma parte do processo de validação.

Quadro 2 – Dados finais aproximados de perda e acurácia do treinamento da rede neural

Rede	Etapas	Épocas	Acurácia (máxima)	Perda (mínima)
<i>U-Net</i>	94	50	95%	3%

Fonte: O autor.

Gráfico 2 – Variação de perda e acurácia (%) em função do tempo (ms) de treinamento



Fonte: O autor.

4 CONCLUSÃO

O modelo desenvolvido mostra aplicabilidade em imagens de satélite visando classificação de uso e ocupação do solo. Ainda mais, os valores aproximados encontrados, de 95% de acurácia máxima e 3% perda mínima, demonstram, em particular, boa empregabilidade para a grande quantidade de recortes de imagens necessárias para treinamento do código para classificação de uma bacia hidrográfica utilizando a arquitetura *U-Net*. Contudo, neste trabalho, mais ações devem ser implementadas para consolidação da metodologia, como a predição das imagens para certificação do modelo.

Assim, o trabalho aqui feito pode ser considerado como um ponto de partida para diversos outras atividades que permitam explorar essas ferramentas de inteligência artificial em imagens de satélite para classificação, não só para uso e ocupação do solo, mas para outras finalidades também. Então, como trabalhos futuros se propõe realizar a predição deste modelo gerado; comparar visualmente a classificação feita com a imagem rotulada; aplicar o modelo em outras bacias; aplicar outras arquiteturas de redes neurais para segmentação de imagens; abranger a segmentação de imagens de satélite para outras finalidades como monitoramento de desmatamento e queimadas; propor um treinamento com diferentes quantidades de dados e comparar a acurácia do treinamento nessas situações.

5 REFERÊNCIAS

BITTENCOURT, L.F.F.; G.T. BATISTA; C.S. CATELANI. 2006. Sensoriamento remoto aplicado ao estudo de ocupação de solo de mata ciliar do rio Paraíba do Sul no município de Caçapava. Artigo apresentado no PRIMEIRO SEMINÁRIO DE SENSORIAMENTO REMOTO E GEOPROCESSAMENTO PARA ESTUDOS AMBIENTAIS NO VALE DOPARAÍBA - GEOVAP 2006, 07 de dezembro, Universidade de Taubaté, Taubaté, São Paulo, Brasil. Disponível em <http://www.agro.unitau.br/soac/viewabstract.php?id=30&cf=1> - Página 89-99.

CARON, E. Segmentação do disco óptico e escavação com o uso glaucoma segmentação do disco óptico e escavação com o uso. 2020.

CORREIA, R. et al. **Processing image to geographical information systems (PI2GIS)—A learning tool for QGIS Education Sciences**, 2018.

FLOOD, N.; WATSON, F.; COLLETT, L. Using a U-net convolutional neural network to map woody vegetation extent from high resolution satellite imagery across Queensland, Australia. **International Journal of Applied Earth Observation and Geoinformation**, v. 82, 1 out. 2019.

NUNES, F. S. Um Modelo De Identificação De Sal Em Imagens Sísmicas De Camadas Subterrâneas Com a Aplicação De Redes Neurais 2018.

OLIVEIRA, P. **USO DE APRENDIZAGEM DE MÁQUINA E REDES NEURAIAS CONVOLUCIONAIS**. [s.l.: s.n.].

PAN, Z. et al. Deep learning segmentation and classification for urban village using a worldview satellite image based on U-net. **Remote Sensing**, v. 12, n. 10, 1 maio 2020.

REIS, N. C. et al. Diagnóstico Automático de Glaucoma em Imagens de Retinografia baseado no CDR usando U-Net. 2020.

RONNEBERGER, O.; FISCHER, P.; BROX, T. **U-net: Convolutional networks for biomedical image segmentation**. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). **Anais...**Springer Verlag, 2015

VAEZA, R. F. et al. Uso e ocupação do solo em bacia hidrográfica urbana a partir de imagens orbitais de alta resolução. **Floresta e Ambiente**, v. 17, n. 1, p. 23–29, 2010.

ZHANG, Z.; LIU, Q.; WANG, Y. Road Extraction by Deep Residual U-Net. **IEEE Geoscience and Remote Sensing Letters**, v. 15, n. 5, p. 749–753, 2018.

ZHU, X. X. et al. Deep learning in remote sensing: a review. n. 41501462, p. 1–60, 2017.

6 ANEXO

Codificação da estrutura *U-Net* e modelo gerado:

U-Net network - FASE 1

```
inputs = tf.keras.layers.Input((128, 128, 4))
```

#redução

```
c1 = tf.keras.layers.Conv2D(16, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(inputs)
```

```
c1 = tf.keras.layers.Dropout(0.1)(c1)
```

```
c1 = tf.keras.layers.Conv2D(16, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c1)
```

```
p1 = tf.keras.layers.MaxPooling2D((2, 2))(c1)
```

```
c2 = tf.keras.layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(p1)
```

```
c2 = tf.keras.layers.Dropout(0.1)(c2)
```

```
c2 = tf.keras.layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c2)
```

```
p2 = tf.keras.layers.MaxPooling2D((2, 2))(c2)
```

```
c3 = tf.keras.layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(p2)
```

```
c3 = tf.keras.layers.Dropout(0.2)(c3)
```

```
c3 = tf.keras.layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c3)
```

```
p3 = tf.keras.layers.MaxPooling2D((2, 2))(c3)
```

```
c4 = tf.keras.layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(p3)
```

```
c4 = tf.keras.layers.Dropout(0.2)(c4)
```

```
c4 = tf.keras.layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c4)
```

```
p4 = tf.keras.layers.MaxPooling2D(pool_size=(2, 2))(c4)
```

#Centro

```
c5 = tf.keras.layers.Conv2D(256, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(p4)
```

```
c5 = tf.keras.layers.Dropout(0.3)(c5)
```

```
c5 = tf.keras.layers.Conv2D(256, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c5)
```

#expansão


```

u6 = tf.keras.layers.Conv2DTranspose(128, (2, 2), strides=(2, 2), padding='same')(c5)
u6 = tf.keras.layers.concatenate([u6, c4])
c6 = tf.keras.layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(u6)
c6 = tf.keras.layers.Dropout(0.2)(c6)
c6 = tf.keras.layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c6)

u7 = tf.keras.layers.Conv2DTranspose(64, (2, 2), strides=(2, 2), padding='same')(c6)
u7 = tf.keras.layers.concatenate([u7, c3])
c7 = tf.keras.layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(u7)
c7 = tf.keras.layers.Dropout(0.2)(c7)
c7 = tf.keras.layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c7)

u8 = tf.keras.layers.Conv2DTranspose(32, (2, 2), strides=(2, 2), padding='same')(c7)
u8 = tf.keras.layers.concatenate([u8, c2])
c8 = tf.keras.layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(u8)
c8 = tf.keras.layers.Dropout(0.1)(c8)
c8 = tf.keras.layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c8)

u9 = tf.keras.layers.Conv2DTranspose(16, (2, 2), strides=(2, 2), padding='same')(c8)
u9 = tf.keras.layers.concatenate([u9, c1], axis=3)
c9 = tf.keras.layers.Conv2D(16, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(u9)
c9 = tf.keras.layers.Dropout(0.1)(c9)
c9 = tf.keras.layers.Conv2D(16, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c9)

outputs = tf.keras.layers.Conv2D(7, (1, 1), activation='softmax')(c9)

model = tf.keras.Model(inputs=[inputs], outputs=[outputs])
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()

```