UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Amanda Costa Spolti

Classificação de vias através de imagens aéreas usando Deep Learning

Uberlândia, Brasil

2018

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Amanda Costa Spolti

Classificação de vias através de imagens aéreas usando Deep Learning

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, Minas Gerais, como requisito exigido parcial à obtenção do grau de Bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Jefferson Rodrigo de Souza

Universidade Federal de Uberlândia – UFU

Faculdade da Computação

Bacharelado em Sistemas de Informação

Uberlândia, Brasil

2018

Dedico ao meu pai, que partiu muito cedo, mas tenho certeza que ele estaria muito orgulhoso por todas as minhas conquistas. A minha mãe, que me incentivou e colaborou para que eu pudesse realizar meus sonhos mesmo que isso significasse o sacrifício dos seus. Ao meu orientador e colaboradores que não mediram esforços para me auxiliarem no desenvolvimento deste trabalho. E por último mas não menos importante, aos meus queridos amigos que acompanharam meus medos e anseios durante essa etapa e me proporcionam diariamente momentos de felicidade.

Agradecimentos

Agradeço primeiramente, à Deus, que me deu forças para superar cada dificuldade enfrentada na minha trajetória até aqui. Aos meus pais e irmão que sempre me motivaram a buscar tudo o que almejo e acredito me ensinando sempre a ser o melhor que eu posso ser.

Agradeço a todos os professores que passaram pela minha vida compartilhando seus conhecimentos e contribuindo de forma significativa para o meu crescimento acadêmico, profissional e pessoal.

Agradeço ao meu orientador Prof. Jefferson Rodrigo de Souza pela disposição, auxílio, disponibilidade e compreensão não só no desenvolvimento desse trabalho como também nas preocupações acadêmicas diárias me motivando a crescer sempre mais.

Ao Prof. Caio Mendes pelo apoio e co-orientação. Aos nossos colaboradores Vitor Guizilini e Prof. Henrique Cândido por contribuir para esse trabalho de forma enriquecedora. Agradeço também a ENGEMAP por disponibilizar as imagens aéreas utilizadas para o desenvolvimento desse trabalho.

Agradeço a minha mãe que não somente me deu à vida mas também me deu amor. Obrigada por sempre me apoiar e acreditar em mim e em meus sonhos, por mais loucos que eles pareçam ser. Ao meu pai *(in memorian)*, pelos ensinamentos que sem eles eu não teria chegado até aqui. Obrigada, saudades!

Agradeço também a todas as pessoas que fazem e fizeram parte da minha vida e contribuiram direta ou indiretamente na formação da pessoa que sou hoje.

"Que os vossos esforços desafiem as impossibilidades, lembrai-vos de que as grandes coisas do homem foram conquistadas do que parecia impossível." Charles Chaplin

Resumo

Com o avanço da tecnologia e consequentemente o aumento do poder de processamento e armazenamento de dados, técnicas que antes eram vistas como inviáveis têm ganhado espaço no âmbito de aprendizado de máquina principalmente em aplicações que envolvem o processamento de imagens. O objetivo deste trabalho é o estudo e aplicação de arquiteturas de Deep Learning para a classificação de vias a partir de uma imagem aérea visando facilitar tarefas como otimização de rotas. Para isso, foi utilizada uma arquitetura nomeada U-Net que foi comparada com outra arquitetura (AutoEnconder), onde a U-Net obteve 92.8% de precisão e 88.8% de acurácia, e o AutoEncoder 89.9% e 88% de precisão e acurácia respectivamente, nos dados de teste. Os resultados mostraram a eficiência da arquitetura usada para a extração de vias bem como a possibilidade de sua aplicação em problemas atuais.

Palavras-chave: deep learning, u-net, autoencoder, classificação de imagens

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

DL	Deep Learning
VANT	Veículo Aéreo Não Tripulado
SVM	Support Vector Machine
NiN	Network-in-Network
AR	Augmented Reality
IA	Inteligência Artificial
MLP	Multilayer Perceptron
DBN	Deep Belief Network
U-Net	Arquitetura U-Net
SR	Sensoriamento Remoto
AS	Aprendizado Supervisionado
CNN	Convolutional Neural Network
ReLU	Rectified Linear Unit

Sumário

1 Introdução

Nas últimas décadas, o uso de imagens de alta-resolução juntamente com a aplicação de técnicas de aprendizado de máquina vêm sendo amplamente utilizadas para a extração de características cartográficas. Entretanto, extrair tais características não é uma tarefa nada trivial tendo em vista a enorme quantidade de objetos distintos que interagem com as vias (MENDES; POZ, 2011). Desse modo, diversos estudos nas mais diveras áreas estão sendo desenvolvidos, como por exemplo estudos relacionados a otimização de rotas (SILVA et al., 2016), no acompanhamento de mudanças de paisagens e no auxílio para a preservação de recursos naturais presentes nessas paisagens (GALO, 2000).

1.1 Motivação

Com o constante avanço na área computacional, tecnológico, desenvolvimento de materiais mais leves e com os custos mais acessíveis, VANTs (Veículos Aéreos Não Tripulados) vem sendo utilizados nas últimas décadas para as mais diversas finalidades. Uma das principais atividades que pode ser realizada por um VANT, é a aplicação de Sensoriamento Remoto (SR). Esta função vem sendo cada vez mais empregada para a obtenção de imagens aéreas de alta resolução. Apesar da facilidade ao acesso de imagens providas de satélites, estas possuem baixa resolução temporal o que torna a extração de feições cartográficas um trabalho complexo e com resultados insatisfatórios.

Com esse avanço tecnológico, problemas de inteligência artificial que antes eram solucionados utilizando puramente códigos extensos e complexos, começaram a serem explorados pelo aprendizado de máquina. Tal técnica baseia-se no princípio de extrair padrões e características dos dados de forma a alimentar o algoritmo para que este aprenda de forma automática. A introdução de aprendizado de máquina permitiu que computadores solucionassem problemas com dados do mundo real auxiliando a tomada de decisão em diversas áreas (GOODFELLOW; BENGIO; COURVILLE, 2016). Com a sua constante evolução, classificadores simples pertencentes ao aprendizado de máquina estão sendo substituídos por métodos mais eficazes que representam melhor o funcionamento do cérebro humano, como as redes neurais e recentemente o Deep Learning (DL) (GOODFELLOW; BENGIO; COURVILLE, 2016).

Dessa forma, diversos trabalhos envolvendo redes neurais já foram e estão sendo realizados. No trabalho de (GALO, 2000), o autor aplicou redes neurais artificiais e sensoriamento remoto na caracterização ambiental do Parque Estadual Morro do Diabo. Para essa finalidade, o parque foi mapeado e caracterizado em quatro momentos distintos no tempo. Posteriormente, foi desenvolvido sistemas de classificação fundamentados

na Ecologia da Paisagem onde foi percebido a viabilidade de se utilizar redes neurais. Os resultados que foram obtidos, deixaram evidente que independente do nível de heterogeneidade pretendido pelas classes, é possível obter resultados satisfatórios para as mais diversas finalidades se os dados de entrada forem compatíveis e as redes neurais devidamente treinadas para caracterizar as classes de interesse.

No trabalho de (MENDES; POZ, 2011), eles aplicaram e avaliaram o uso de redes neurais artificiais usando dois conjuntos de dados. Primeiramente o uso de imagens aéreas R, G e B de alta resolução, e posteriormente foi adicionada uma imagem que representa os objetos elevados obtidos a partir de varredura a laser, onde ambas metodologias tiveram resultados satisfatórios embora os resultados do uso dos dois tipos de imagem combinados foram melhores.

(SILVA et al., 2016) utilizou um Sistema de Informação Geográfico em áreas florestais do estado de Minas Gerais para determinar um padrão na malha viária e definir rotas de transporte, que pode ser de extrema utilidade no transporte florestal brasileiro tendo em vista que é um dos tipos de transporte mais caros em empresas florestais.

Redes neurais e DL fornecem as melhores soluções para diversos problemas ligados ao reconhecimento de imagem, processamento de linguagem natural e reconhecimento de voz (GOODFELLOW; BENGIO; COURVILLE, 2016). Uma rede neural é uma técnica baseada na forma em que o cérebro humano processa e infere informações. Consiste basicamente em 3 camadas. A primeira é a entrada, onde os dados que vão alimentar a rede são fornecidos. A segunda, também chamada de camada escondida, é a responsável por processar os dados fazendo o uso de cálculos matemáticos que vão definir qual será a saída da rede. A terceira e última camada é a que irá analisar as saídas provenientes da camada anterior, gerando assim o resultado final. Assim sendo, DL é uma rede neural que possui uma quantidade maior de camadas escondidas que fazem uso de um conjunto de técnicas para modelar e solucionar um dado problema da melhor maneira possível. Apesar de uma arquitetura que aplica o DL possuir uma complexidade maior de treinamento quando comparada com uma rede neural superficial, esta pode, ao adicionar mais camadas e mais unidades dentro de uma camada, representar funções de crescente complexidade e por conseguinte modelar problemas de maior complexidade.

1.2 Justificativa

A geração da malha viária se dá normalmente de forma manual realizada por restituição fotogramétrica através de um operador que vetoriza pontos nas vias o que demanda muito esforço e tempo do operador, outra possibilidade é a utilização de sistemas de mapeamento móvel, onde um ou mais operadores devem percorrer todas as vias necessárias para o mapeamento do ambiente em questão.

Uma vez que seja possível a extração da malha viária utilizando Aprendizado Supervisionado (AS), é possível fazer o uso de técnicas computacionais obtendo rotas otimizadas, visando a economia de recursos, tempo e que satisfazem de forma aperfeiçoada as necessidades de um grupo de indivíduos específico. O AS é caracterizado por fornecer ao algoritmo um conjunto de dados de treinamento juntamente com as classes a que cada um desses dados pertence, sendo fundamentais para o aprendizado da rede e consequentemente para a classificação de dados de teste, onde os rótulos não são conhecidos.

Dessa forma e em virtude dos trabalhos anteriores, este trabalho utilizou imagens aéreas, como mostra a Figura 1, obtidas por avião, e futuramente pretende utilizar imagens obtidas por um VANT para classificar as malhas viárias aplicando uma rede neural de modo a solucionar o problema de classificação de vias empregando DL de forma rápida e automática. A arquitetura escolhida será comparada com uma arquitetura AutoEncoder.

Figura 1 – Imagem aérea de um sistema fotogramétrico convencional acoplado a uma plataforma Aérea Tripulada. (Fonte: ENGEMAP)

1.3 Objetivos

1.3.1 Gerais

Desenvolvimento de uma arquitetura utilizando DL, capaz de classificar vias usando imagens aéreas, distinguindo automaticamente o que é uma via ou não.

1.3.2 Específicos

- Obtenção de imagens aéreas de uma região urbana e seus respectivos rótulos;
- Implementação da arquitetura de DL U-Net;
- Treinamendo da arquitetura de DL U-Net;
- Classificação de imagens desconhecidas pela rede para obtenção de métricas que possibilite a análise visual e quantitativa da metodologia empregada.

1.4 Contribuições

A principal contribuição deste trabalho é a criação de uma solução capaz de automatizar grande parte das atividades envolvidas no processo de otimização de rotas.

1.5 Organização do Trabalho

O documento está organizado em 6 capítulos, cujos conteúdos se encontram estruturados abaixo:

- Capítulo 2 aborda a técnica de DL bem como sua definição e operações normalmente utilizadas em suas arquiteturas;
- Capítulo 3 descreve os trabalhos relacionados que utilizaram DL para a solução de problemas de classificação de imagens e extração de características;
- Capítulo 4 apresenta a metodologia usada;
- Capítulo 5 apresenta os resultados obtidos;
- Capítulo 6 apresenta a conclusão do trabalho e futuras perspectivas.

2 Fundamentação Teórica

Este capítulo apresenta o conceito de DL bem como as principais operações que são utilizadas nessa área da Inteligência Artificial (IA). Toda arquitetura de DL possui pelo menos uma das operações que serão descritas neste capítulo.

2.1 Deep Learning

O DL, também conhecido como aprendizado estruturado é um ramo da área de IA que teve seus primeiros experimentos realizados em 1950 (GOODFELLOW; BENGIO; COURVILLE, 2016). Desde então, essa abordagem vêm sendo amplamente estudada e aprimorada. Embora tenha surgido há algumas décadas atrás, sua utilização nunca esteve tão comentada no mundo acadêmico e empresarial como atualmente. Devido a falta de poder de processamento e armazenamento de dados, o DL apesar de parecer muito promissor na época de seu surgimento acabou sendo esquecido.

Contudo, com o avanço constante da tecnologia em termos de velocidade de processamento e capacidade de armazenamento de dados, o DL vêm sendo empregado nas mais diversas áreas para resolver os mais diversos problemas. Outro fator que é de extrema importância é a quantidade de dados disponíveis. A facilidade de se obter um conjunto de dados relativamente grande, o que é fundamental em arquiteturas de DL, também possibilitou a utilização desse ramo da IA.

Basicamente, o DL é uma rede neural que possui mais camadas escondidas e que possui também um maior número de operações. A sua essência consiste em representar problemas complexos em etapas mais simples. Dessa forma, cada camada é responsável por extrair determinada característica onde a junção do todo contribui para o resultado final. Uma camada escondida nada mais é do que as operações que se encontram entre a camada de entrada da rede e a camada de saída.

Arquiteturas de DL são muito flexíveis e podem ser arquitetadas da melhor maneira de acordo a sua aplicação. Por esse motivo, existem diversas arquiteturas e variações, mas todas se baseiam em alguma previamente definida e solidificada no mundo acadêmico. Uma das arquiteturas mais utilizada é a CNN. Uma CNN é basicamente uma rede neural que realiza uma operação de convolução em pelo menos uma de suas camadas (GOODFELLOW; BENGIO; COURVILLE, 2016). Geralmente, uma camada de uma CNN possui 3 etapas sendo elas convolução, função de ativação não-linear e uma camada de pooling respectivamente. Operações de convolução e pooling serão descritas em breve.

Uma função de ativação não-linear amplamente utilizada na técnica de DL é a

ReLU (Rectified Linear Unit) que é descrita da seguinte forma $f(z) = max(0, z)$, onde z é o valor de entrada e o valor propagado será z se ele for maior que 0, ou 0 caso contrário. Até o momento, não existe uma justificativa plaúsivel por a ReLU ser melhor que outras funções de ativação na literatura (GOODFELLOW; BENGIO; COURVILLE, 2016). O fato é que pesquisadores adotaram tal função e obtiveram bons resultados mesmo com uma base de dados grande, e por isso ela é a mais usada.

2.2 Convolução

No ponto de vista da matemática, a convolução é uma função linear que se baseia na multiplicação de duas matrizes. Como mostra a Figura 2, uma operação de convolução possui 3 componentes principais: a entrada, o detector de característica também conhecido como kernel de convolução e o mapa de característica que é o resultado da operação. O tamanho do detector de característica varia de acordo com o valor do parâmetro escolhido, uma escolha comum na literatura é uma janela 3x3. Outros 2 parâmetros que influenciam diretamente no mapa de carcterística resultante é o *stride* e o *padding*. O *stride* é o tamanho do passo do kernel ao percorrer a imagem e o padding define como a borda da imagem de entrada é tratada.

O kernel é uma janela deslizante que irá percorrer a imagem de entrada da esquerda para a direita e de cima para baixo de acordo com o valor do stride definido, realizando a multiplicação entre a sua matriz e o contexto atual em que a janela se encontra, somando os valores da multiplicação resultante. Geralmente, os valores do filtro são inicializados de forma randomica tendo seus valores ajustados de acordo com o treinamento da rede. De forma mais concreta, um kernel 3x3 é usado para mapear os 9 valores na matriz de entrada a 1 valor na matriz de saída. Sendo assim, uma operação de convolução forma um relacionamento de muitos para um.

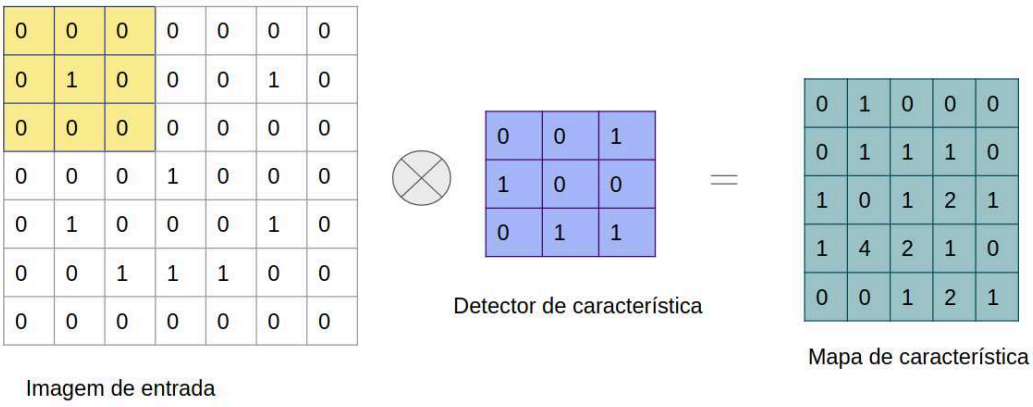

Figura 2 – Exemplo de uma operação de convolução com stride 1 e kernel 3x3. Fonte: Eremenko (2018)

O objetivo é encontrar características na imagem de entrada utilizando o detector de característica que resultará em um mapa, preservando a relação entre os pixels para manter o padrão. Na maioria das vezes, principalmente em aplicações de DL, as características que a rede irá detectar e usar para classificar imagens em certas classes não fará sentido nenhum para um humano.

Se a imagem de entrada possuir 3 canais, RGB por exemplo, como é o caso da maioria das imagens presentes em bases de dados, o detector de características também possuirá 3 dimensões. A quantidade de mapas de características resultantes também é um parâmetro da operação de convolução. Se esse valor for 4, a operação resultará em 4 mapas de características de três dimensões.

Após uma camada de convolução as características ainda são preservadas, o processamento é mais rápido devido a quantidade de parâmetros passada para as próximas camadas e ocorre perda de informação. A perda de informação é inevitável e até desejável pelo fato do objetivo do detector de característica ser justamente preservar aquelas mais discriminativas e relevantes ao contexto da aplicação e descartar o restante. Na maioria das vezes, os detectores possuem seus valores iniciados aleatoriamente o que resulta em mapas desconexos para a visão humana. Contudo, a medida que a rede vai aprendendo, os valores do detector são ajustados e a rede decide por si própria o que deve ser mantido ou alterado.

2.3 Up-Convolution (Transposição Convolucional)

Uma operação de transposição convolucional nada mais é que a operação inversa de uma convolução no âmbito de DL. Como destacado, enquanto uma operação de convolução forma um relacionamento de muitos para um, uma transposição forma um relacionamento de um para muitos. Essa operação é denominada de deconvolução e gera muita discussão sobre a forma correta de nomenclatura, pois matematicamente falando, uma deconvolução gera os valores originais de uma matriz antes da convolução. Embora uma deconvolução seja considerada a operação inversa de uma convolução, a única semelhança é que tal operação irá reconstruir a resolução espacial de antes seguida de uma convolução. Ou seja, essa etapa não reverterá o processo com relação aos valores númericos mas apenas reconstrói a entrada para que a dimensão seja a mesma.

A Figura 3 é uma representação de uma operação de deconvolução. A parte em verde localizada no centro da entrada é o mapa de característica gerado pela última operação realizada. A parte em branco geralmente é preenchida com zeros. Se houver padding, haverá um espaçamento entre os pixels do mapa de característica representado na parte verde.

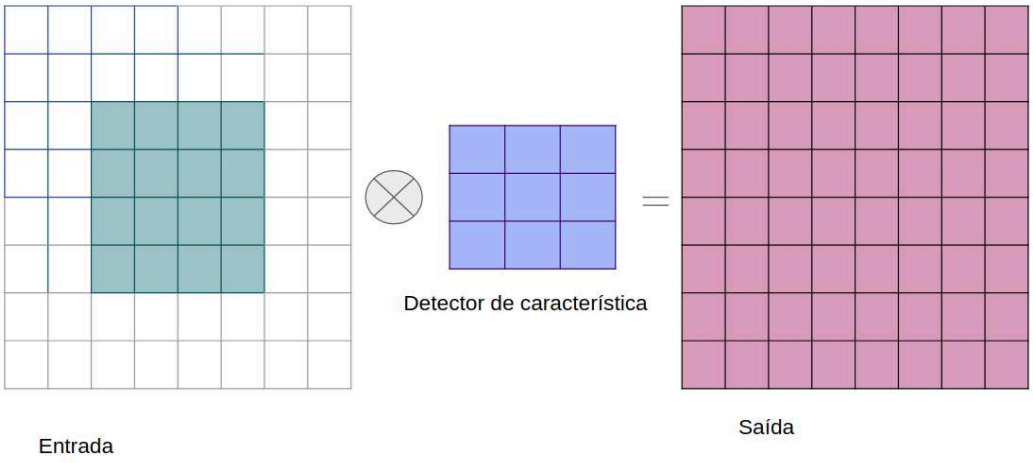

Figura 3 – Exemplo de uma operação de deconvolução com um kernel 3x3 e sem padding. Adaptada de: Pröve (2017)

2.4 Max Pooling

Operações de pooling tem um papel fundamental em arquiteturas DL. Tais operações reduzem o tamanho dos mapas de características utilizando alguma função que sumariza uma sub-região (DUMOULIN; VISIN, 2018). Existem diversas funções de pooling mas a mais aceita e utilizada na literatura atualmente é o max pooling que propaga para as próximas camadas o maior valor em uma janela NxN do mapa de característica. Como o próprio nome diz, o max pooling escolhe o maior valor em uma janela NxN do mapa de característica e ignora todo o resto.

A Figura 4 traz um exemplo de um mapa com tamanho 7x7 e o resultado de uma operação de max pooling utilizando uma janela 2x2 e stride 2 (o stride é a distância entre duas posições consecutivas da janela de pooling). O tamanho do mapa de característica, da janela de pooling e stride afetam diretamente no tamanho da saída gerada pela operação. Utilizando um mapa de característica de tamanho 256x256, janela 2x2 e stride 2 o mapa seria reduzido pela metade, resultando em um mapa de tamanho 128x128.

O pooling permite que a rede seja capaz de identificar características independente de fatores como textura, localização das características, rotação da imagem entre outros. Mesmo reduzindo o tamanho do mapa de característica, as características ainda são preservadas pois sabe-se que o maior valor é exatamente onde foi encontrado a maior similaridade com um detector de característica. Tal técnica reduz o número de parâmetros que vão para as próximas camadas da rede, prevenindo o overfitting (sobre-ajuste, termo utilizado em estatística para descrever quando um modelo se ajusta bem um conjunto de dados específico, e mostra-se ineficaz para prever novos resultados) por justamente propagar as características mais relevantes e otimizando o tempo de processamento.

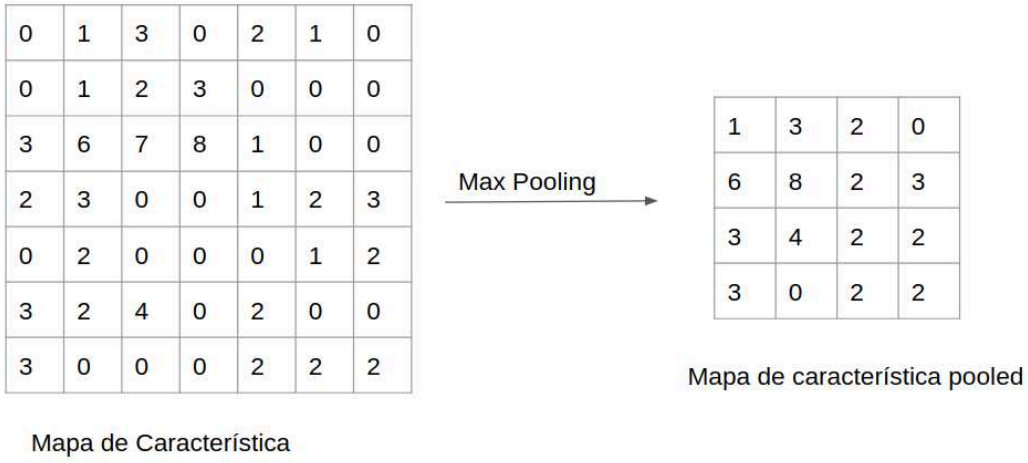

Figura 4 – Exemplo de uma operação de max pooling. Adaptada de Eremenko (2018)

3 Trabalhos Relacionados

No trabalho de Tremblay-Gosselin e Cretu (2013) foi proposto uma nova abordagem para identificação de prédios a partir da análise de imagens aéreas. A etapa de treinamento foi feita utilizando o auxílio de um usuário que selecionava pontos de interesse na imagem, prédios, e pontos de não interesse, como vegetação, ruas, etc. Uma combinação de atributos de formas com uma técnica automatizada de crescimento de região e um Support Vector Machine (SVM) foram aplicados para a classificação entre os pontos de interesse e os pontos de não interesse. Essa abordagem foi avaliada manipulando 20 imagens aéreas com resolução de 256x256 pixels que continham diferentes topologias e complexidades. Tal abordagem obteve 97,8% de precisão comprovando assim a eficácia da técnica utilizada. Dessa forma, foi concluído que os descritores codificam um conjunto apropriado de características que permitem identificar objetos de interesse corretamente na maioria das imagens, apesar das diferentes formas e tamanhos de telhados.

Desfrutando da técnica de DL (MENDES; FRÉMONT; WOLF, 2016), foi considerado o problema de detecção de rodovias, onde dada uma imagem, o objetivo era classificar cada pixel da imagem em rodovia ou não-rodovia. A arquitetura foi uma Convolutional Neural Network (CNN), visando um modelo que emprega uma grande janela contextual enquanto mantém uma inferência rápida. Para tal, usou-se uma arquitetura Network-in-Network (NiN) e convertendo o modelo em uma rede totalmente convolucional após o treinamento. Para treinamento e para avaliação da arquitetura proposta foi utilizado o conjunto de dados KITTI Vision Benchmark Suite, esse banco de dados oferece 289 imagens de treinamento juntamente com seus ground-truth e 290 imagens de teste. Foi obtido uma acurácia de 92% tanto utilizando a arquitetura NiN quanto sem, porém, com a arquitetura o tempo de inferência foi significativamente menor. Os resultados revelaram que o tempo de inferência da abordagem proposta é único nesse nível de precisão, sendo duas ordens de grandeza mais rápida do que outros métodos com desempenho semelhante.

Também aplicando DL, Badrinarayanan, Kendall e Cipolla (2015) propôs uma nova e prática arquitetura de rede neural completamente convolutiva para a segmentação semântica pixel-wise que foi denominada de SegNet. O núcleo do mecanismo de segmentação consiste em uma rede de codificador, uma rede decodificadora correspondente seguida por uma camada de segmentação por pixel. A rede do codificador consiste em 13 camadas convolutivas. Cada camada do codificador tem uma camada decodificadora correspondente, portanto a rede decodificadora também tem 13 camadas. O desempenho da arquitetura SegNet foi medido em duas situações diferentes, a primeira é a classificação de rodovias, árvores, passeios, carros, etc. A segunda é a segmentação de cena interna que é de interesse imediato para vários aplicativos de Realidade Aumentada (AR). No caso da

segunda, por ser um problema que contém diversas classes, o resultado não foi satisfatório. Porém, para a classificação de rodovias a SegNet se mostrou eficiente obtendo resultados com 90% de acurácia. SegNet foi comparada com outras arquiteturas em termos de tempo de treinamento, memória e acurácia. Algumas delas tiveram resultados melhores, porém, mais memória era necessária pela quantidade de dados que precisam armazenar. SegNet, por outro lado, é mais eficiente, pois só armazena o pool máximo dos índices dos mapas de características e os usa em sua rede decodificadora para alcançar um bom desempenho, com menos memória e ainda sim obtendo resultados satisfatórios.

Outro trabalho utilizando DL foi proposto por (KUSSUL et al., 2017) para classificar imagens de satélite com relação a cobertura dos solos e suas culturas. Para o fim de restauração de dados ausentes devido a nuvens e sombras presentes nas imagens foi necessário uma fase de pré processamento. Para fins de classificação, foram aplicadas uma rede neural supervisionada totalmente conectada Multilayer Perceptron (MLP), uma Random Forest, comparando tais técnicas com uma CNN. Os experimentos foram realizados fazendo uso de 19 imagens multitemporais da região da Ucrânia adquiridas por satélites de Landsat-8 e Sentinel-1A. Utilizando duas variações de arquitetura de uma CNN, denominadas 1-D e 2-D para explorar características espectrais e espaciais respectivamente, foi obtido uma acurácia de 93.5% e 94.6% respectivamente, enquanto Random Forest e MLP obtiveram 88.7% e 92.7% respectivamente. Consequentemente, a arquitetura proposta se mostrou mais eficaz para o problema descrito.

Uma nova abordagem baseada em DL para a extração de características no âmbito de sensoriamento remoto foi proposta por (ZOU et al., 2015), tratando o problema de extração como um problema de reconstrução de características. O método proposto seleciona as características mais reconstrutivas como sendo as discriminativas. Nos experimentos, foram utilizadas 2800 imagens de sensoriamento remoto divididas em 7 categorias (grama, fazenda, indústria, rio, floresta, residencial, estacionamento) para a avaliação do desempenho. Para tratar o problema de reconstrução de características, uma DBN (Deep Belief Network) foi utilizada. Um algoritmo iterativo para aprendizagem das características foi desenvolvido para obter pesos confiáveis de reconstrução e características com pequenos erros de reconstrução. Em média, foi obtido uma acurácia de 77%, sendo que a categoria com mais erros de classificação foi a da indústria com 65%, e a categoria menos confusa sendo a da floresta com uma acurácia de 93.5%. Tendo em vista a complexidade do tipo de classificação, os experimentos validaram a eficiência do método proposto.

4 Metodologia Proposta

4.1 U-Net

A U-Net é uma arquitetura DL que segue a estrutura de uma CNN que foi desenvolvida primordialmente para a segmentação de imagens biomédicas, que devido a complexidade e dificuldade de obter um banco de dados suficientemente grande para o treinamento da rede é um trabalho substancialmente mais complicado. Tal arquitetura foi nomeada U-Net (RONNEBERGER; P.FISCHER; BROX, 2015) por sua arquitetura representada pela Figura 5 possuir a forma de um "U".

A U-Net se resume em dois passos gerais: contração e expansão como é mostrado na Figura 5. A imagem aérea é inserida na entrada da rede e propagada para todos os caminhos, e no final tem-se o resultado da segmentação da mesma. Contração consiste em uma sucessão de camadas que são alteradas por operações de convolução e max-pooling. Na expansão a imagem é reconstruída novamente. Para localizar os recursos de alta resolução do caminho de contração, eles são combinados com a saída da expansão. Com isso aplica-se convolução novamente para montar uma saída mais precisa com base nas informações coletadas no caminho de expansão.

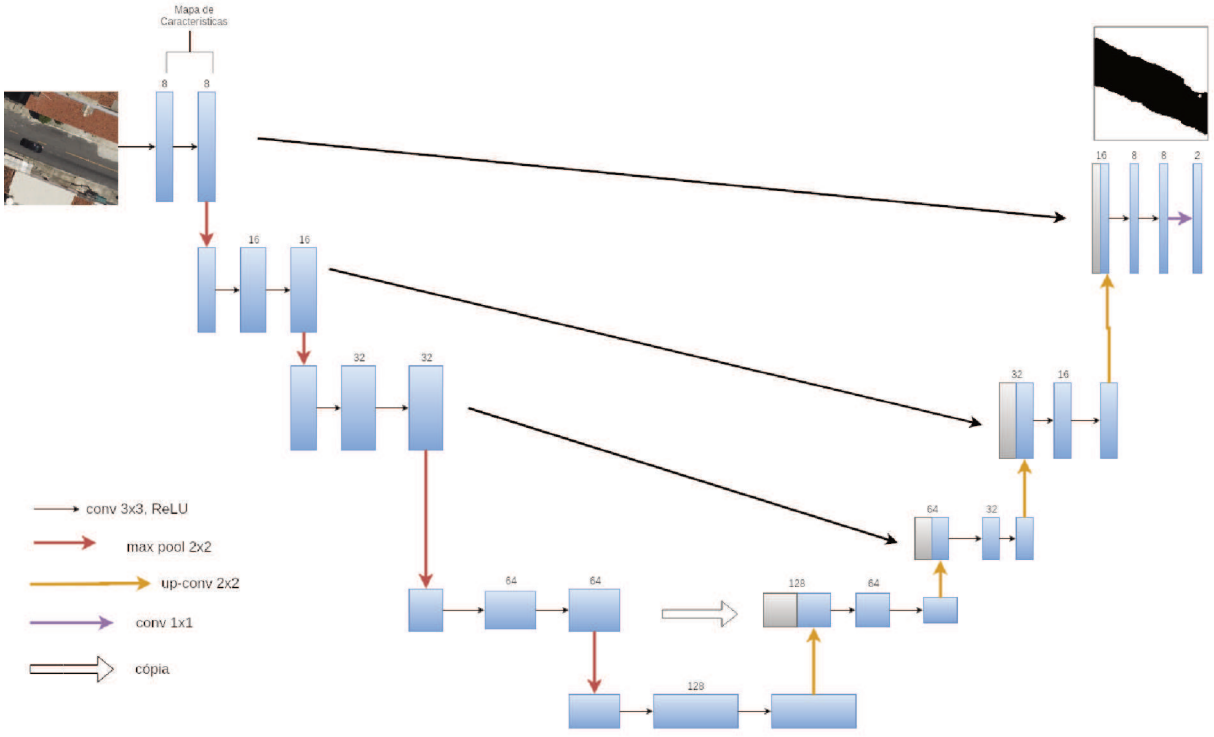

Figura 5 – Arquitetura U-Net. Adaptada de Ronneberger, P.Fischer e Brox (2015)

Como mencionado anteriormente, a U-Net possui duas fases principais: contração

e expansão. Na primeira etapa do caminho de contração, a imagem de entrada passa por 2 convoluções com kernel de tamanho 3x3, stride 1, padding que preserva as bordas e ReLU como função de ativação, gerando 8 mapas de características (usa-se 8 kernels diferentes para se produzir os mapas de características citados) seguida por uma operação de max pooling 2x2 com stride 2. Após cada operação de max pooling, a quantidade de mapa de características são aumentados por um fator de dois e o tamanho da entrada é reduzido pelo mesmo fator devido aos efeitos do max pooling. No caminho de contração, uma etapa é definida por duas convoluções e uma operação de max pooling.

Portanto, após 4 etapas, a saída resultante é passada como entrada para uma transposição convolucional (up-convolution conforme na Figura 3) com kernel 2x2, stride 2 com 64 detectores de características no primeiro passo, que é o início do caminho de expansão. Cada etapa desta fase consiste na transposição deconvolucional com os parâmetros descritos acima, concatenação com sua parte correspondente da parte de contração e convoluções como aplicadas no caminho de contração. Após cada etapa, o número de canais de recursos é reduzido por um fator de dois. Na última camada, uma convolução de 1x1 é aplicada para mapear cada 8 mapas de características para sua classe de classificação correspondente. Nessa operação de convolução usa a função de ativação sigmóide h_θ . Essa função gera resultados entre 0 e 1. Dessa forma, pode-se estabelecer regras. Por exemplo, se o resultado da função sigmóide for maior que 0.6, então a saída será 1 caso contrário produz 0. O resultado da função é exatamente a probabilidade do pixel pertencer a uma determinada classe. A função sigmóide (1) está formulada abaixo onde z é a multiplicação do pixel pelo seu respectivo peso somado ao valor bias. Neste trabalho, os pesos e os bias foram inicializados com zeros e tiveram seus valores modificados ao longo do treinamento da rede de acordo com os resultados por ela definidos.

$$\theta(z) = \frac{1}{1 + e^{-}z} \tag{1}$$

A imagem resultante é então uma imagem em preto e branco, onde pretos são os pixels classificados como vias e brancos, caso contrário.

4.1.1 AutoEncoder

Outra técnica considerada neste trabalho é baseada no trabalho de (LONG; SHELHAMER; DARRELL, 2015), que propõe o uso de redes totalmente convolucionais para segmentação semântica. Uma rede totalmente convolucional não possui nenhuma camada totalmente conectada, na qual todos os neurônios de uma camada estão conectados a todos os neurônios da camada seguinte. Esta prática era comum na maioria das topologias até este ponto, particularmente em camadas mais profundas, no entanto, há muitos benefícios na eliminação de conexões completas: 1) diminuição no número de parâmetros

treináveis; 2) preservação da correlação espacial; 3) imagens de qualquer tamanho podem ser igualmente processadas usando a mesma rede.

Priorizando a eficiência computacional, visando futuras aplicações com processamento a bordo, uma topologia relativamente simples foi utilizada neste trabalho, composta por três camadas convolucionais, com kernel de 5x5 e três camadas deconvolucionais (transposição convolucional), com o mesmo tamanho de kernel. Cada camada é seguida por uma função de ativação de ReLU (XU et al., 2015), para introduzir não linearidades, e camadas convolucionais recebem um max pooling de 2x2 para reduzir a dimensão espacial, enquanto camadas deconvolucionais aumentam as imagens de entrada pelo mesmo fator, dobrando suas dimensões espaciais. Camadas convolucionais e deconvolucionais são conectadas por uma quarta camada convolucional, com tamanho de filtro 3x3, nenhuma função de ativação e *dropout* (SRIVASTAVA et al., 2014) de 0,6 (isto é, 60 % de nós são desligados aleatoriamente durante o treinamento, para aumentar a capacidade da rede de generalizar sobre diferentes entradas). O número de neurônios em cada camada convolucional foi de 64, 128 e 256, respectivamente, e esses números foram invertidos nas camadas deconvolucionais.

A camada final produz saídas de 1 canal, com uma função de ativação sigmóide para produzir valores entre $[0, 1]$ que servem como classificação probabilística para cada pixel. A função de perda de entropia cruzada foi otimizada durante o treinamento, com base nos rótulos, usando um otimizador de Adam (KINGMA; BA, 2014) com uma taxa de aprendizado de 10^{-4}.

4.1.2 Desenvolvimento

A arquitetura U-Net foi implementada usando TensorFlow que é uma interface desenvolvido pela Google para expressar algoritmos de aprendizado de máquina (ABADI et al., 2015), e a biblioteca Keras que é uma API de redes neurais de alto nível, escrita em Python e capaz de rodar em cima do TensorFlow (CHOLLET et al., 2015). TensorFlow disponibiliza dois tipos de instalações, uma para CPU e uma para GPU. Por razões de eficiência de processamento, foi utilizada uma máquina com GPU caso contrário, o treinamento poderia levar dias para ser finalizado. O GPU utilizado foi uma *GeForce GTX TITAN X 33MHz*.

O conjunto de dados é composto de 3814 imagens de 256x256 divididas em 80% para treinamento, 10% para testes e 10% para validação. Essa divisão é importante para que o modelo resultante seja confiável. A base de treinamento é o que permite que a rede aprenda, a de validação é responsável por fornecer uma avaliação imparcial do modelo no conjunto de dados de treinamento enquanto ajusta os hiperparâmetros do modelo, e a de teste por avaliar o modelo gerado e só é utilizada quando o modelo está completamente treinado. As imagens aéreas foram obtidas por levantamento aéreo convencional que pos-

suem dimensão de 8956x6708 pixels. O conjunto de dados é extremamente importante para grande parte de algoritimos de IA, especialmente para problemas que requerem uma solução mais complexa por sua solução não ser só um grupo de operações matemáticas. Reconhecimento de voz, objetos, e outros que envolvem processamento digital de imagens requerem um banco de dados variado que representem todas as classes envolvidas na classificação. Para obter melhores resultados, o conjunto de imagens aéreas foi selecionadas cautelosamente filtrando a base de dados de forma manual, excluindo imagens com somente uma classe presente.

A obtenção dos rótulos de cada imagem foi feita manualmente através de um software de edição de imagens colorindo rodovias de preto e o restante de branco. Como a dimensão da imagem original é extremamente elevada foi necessário o desenvolvimento de um script para cortar essas imagens em pedaços menores de 256x256. A rede é treinada por 300 épocas utilizando os dados de treinamento, ou o treinamento é interrompido se a acurácia calculada com a base de validação não tiver sido melhorada por 30 épocas. Uma época é um passo completo no treinamento, ou seja, quando toda a base de dados é processada. Ao final do treinamento, o modelo gerado é salvo permitindo classificações futuras sem a necessidade de treinar a rede novamente.

Para a comparação das arquiteturas U-Net e AutoEncoder, foram aplicadas 4 métricas em cima dos dados para teste: acurácia, precisão, revocação e medida F para melhor entender a efetividade da arquitetura, obter dados quantitativos e também para futuras comparações. Para fins de comparação, a mesma base de dados foi utilizada na arquitetura de AutoEncoder descrita anteriormente.

Representada pela Equação (2), a acurácia (AC) quantifica a frequência com que a classificação foi realizada corretamente e está representada abaixo, onde VP são os verdadeiros positivos, VN são os verdadeiros negativos e N a quantidade de itens, nesse caso pixels.

$$AC = \frac{VP + VN}{N} \tag{2}$$

A precisão (P) representada pela Equação (3) calcula a porcentagem dos itens classificados como rodovias que efetivamente pertenciam a esta classe (FP são os falsos positivos).

$$P = \frac{VP}{VP + FP} \tag{3}$$

A revocação (R) é a proporção de verdadeiros positivos que foi classificada corretamente, Equação (4). Ou seja, a proporção de pixels que são rodovias e foram efetivamente classificadas como tal, e a proporção de pixels que não eram rodovias e foram identificados

corretamente (FN representa os falsos negativos).

$$R = \frac{VP}{VP + FN} \tag{4}$$

Já a medida F representada pela Equação (5) combina a precisão e revocação de modo a medir a qualidade geral do modelo criado.

$$F = \frac{2 \times P \times R}{P + R} \tag{5}$$

5 Resultados

Para analisar o desempenho da U-Net em classificar imagens com diferentes níveis de dificuldade, foram selecionadas 5 imagens do banco de dados de teste de forma manual considerando a sua complexidade de classificação. A Figura 6 mostra a imagem original, seu rótulo e a classificação obtida pela U-Net e pelo AutoEncoder. Na Imagem 1 e na Imagem 2, a classificação da U-Net foi muito precisa. No entanto, nas imagens 3, 4 e 5 a classificação foi ruim devido ao fato de essas imagens conterem objetos que podem ser facilmente misturados com rodovias considerando a similaridade de cores. A predição para a Imagem 3 possui um telhado que foi erroneamente classificado como uma via, e na Imagem 5 a classificação foi totalmente errada novamente por causa da cor que pode ser facilmente confundida até mesmo para os seres humanos.

Analisando os resultados do AutoEncoder é notável a semelhança entre os resultados das duas arquiteturas. Os mesmos detalhes de classificação analisados acima, servem também para o AutoEncoder. Porém, na imagem 4 de ambas arquiteturas a U-Net obteve um erro menor facilmente visível. Para a Imagem 3, a arquitetura AutoEncoder apresentou melhor resultado comparado a U-net, como pode ser observado. AutoEncoder conseguiu lidar melhor com o telhado parecido com a rodovia. E da mesma forma com a Imagem 5.

Considerando toda a base de dados de teste composta de 383 imagens de 256x256 pixels, a U-Net obteve o melhor desempenho como mostra a tabela de resultados (Tabela 1). Os resultados mais precisos foram obtidos pela U-Net comparado ao Auto-Encoder. As métricas das duas arquiteturas foram muito semelhantes, no entanto, a U-Net ainda obteve melhores resultados, exceto para a medida de revocação, onde U-Net obteve 90,9% e o Auto-Encoder 91,9%.

Tabela 1 – Resultados

	U-Net	Auto-Encoders
Acurácia	0.888	0.880
Precisão	0.928	0.899
Revocação	0.909	0.919
Medida F	0.919	0.907

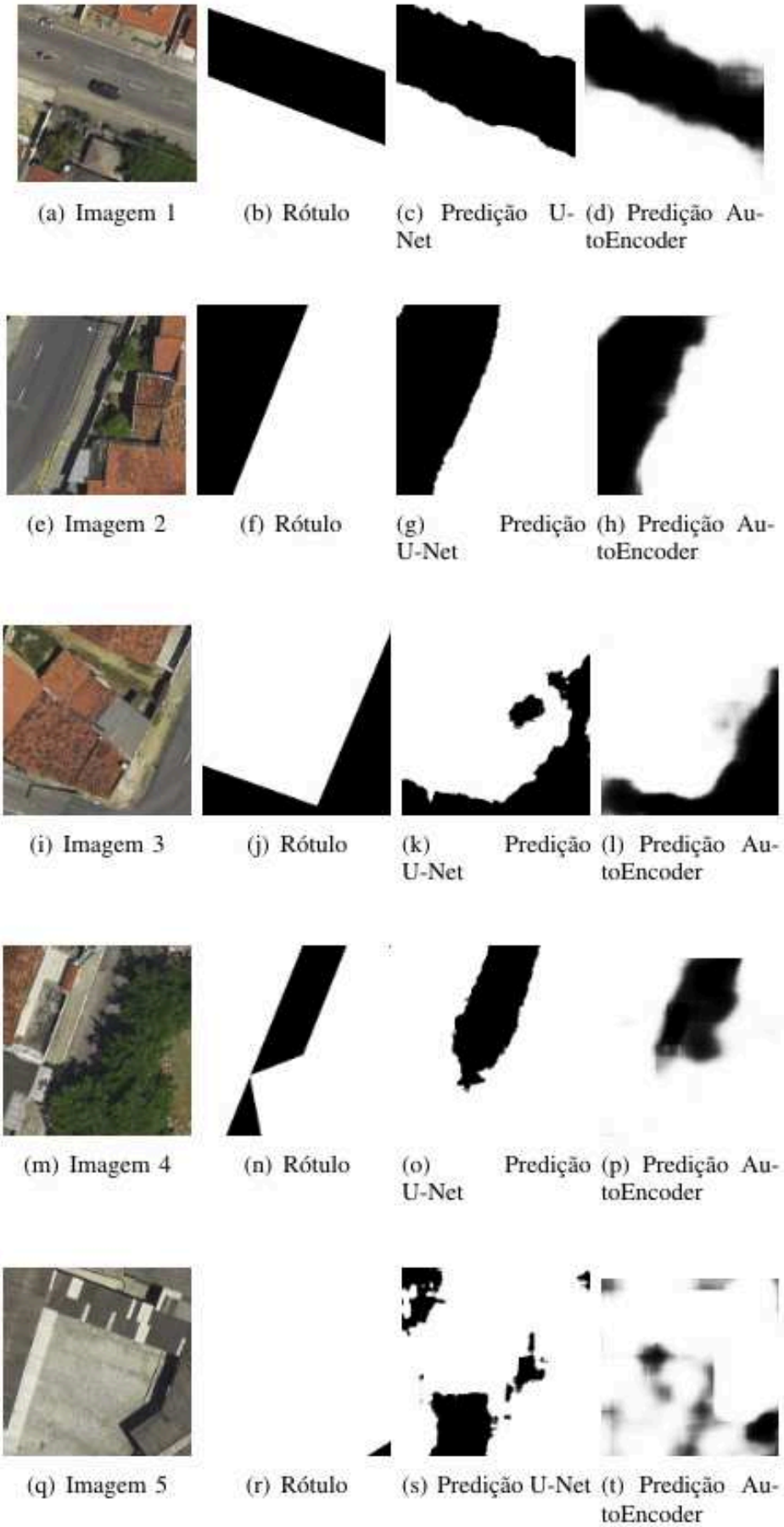

Figura 6 – Resultados das duas arquiteturas DL com diferentes níveis de dificuldade com seus respectivos rótulos e predições sob os dados de teste.

6 Conclusão

Embora existam excelentes ferramentas de mapeamento gratuitas e de fácil acesso, estas demoram meses e até anos para terem suas bases de dados atualizadas, não acompanhando o crescimento urbano e alguns municípios acabam sendo isolados. E de qualquer forma, para terem sua bases de dados atualizadas é necessário geralmente que um veículo percorra a cidade realizando tal mapeamento. A complexidade desse trabalho bem como a quantidade de tempo e recursos é extremamente elevada.

Esse trabalho apresenta a implementação e treinamento da arquitetura DL U-net e AutoEncoder para classificar rodovias a partir de uma imagem aérea, facilitando o trabalho de profissionais envolvidos com o mapeamento de uma região urbana bem como a otimização de rotas. A rede U-Net foi capaz de obter um elevado índice de acurácia para um problema relativamente complexo e com uma base de dados considerada pequena no âmbito de problemas de classificação com DL. As imagens resultantes da classificação mostraram a capacidade da arquitetura para solucionar o problema discutido nesse trabalho de forma rápida e eficiente.

O AutoEncoder também se mostrou viável para nosso problema de classificação. Embora seu resultado foi relativamente inferior ao da U-Net, tal arquitetura é mais simples e a quantidade de parâmetro utilizada é inferior ao da U-Net por utilizar o dropout de 0,6 desligando 60% dos nós durante o treinamento. Dessa forma, em trabalhos futuros é importante verificar os resultados da U-Net também utilizando dropout para diminuir o tempo de inferência e generalizar o modelo ainda mais.

Como discutido na seção de resultados é notável a incapacidade do atual modelo treinado para classificar imagens que possuem elementos com tonalidades semelhantes as de rodovias. Porém, como dito anteriormente a base de dados utilizada é relativamente pequena e tal resultado pode ser melhorado com uma base de dados maior e mais diversificada. Outro ponto é o fato das imagens terem sido obtidas por avião. Devido a impossibilidade de ter obtido imagens utilizando um VANT no atual momento em que o projeto se encontra, tem-se como objetivo futuro aumentar a base de dados com imagens coletadas por um VANT, melhorando a base e consequentemente os resultados. Outras técnicas também devem ser desenvolvidas para que a arquitetura seja capaz de analisar também o contexto em que o pixel se encontra. Dessa forma, com uma base de dados ainda mais representativa e com a capacidade da análise do contexto, os resultados podem atingir índices de sucesso ainda mais elevados.

Referências

ABADI, M. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Software available from tensorflow.org. Disponível em: <https://www.tensorflow.org/>. Citado na página 23.

BADRINARAYANAN, V.; KENDALL, A.; CIPOLLA, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015. Citado na página 19.

CHOLLET, F. et al. *Keras*. [S.l.]: GitHub, 2015. <https://github.com/fchollet/keras>. Citado na página 23.

DUMOULIN, V.; VISIN, F. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2018. Citado na página 17.

EREMENKO, K. *Deep Learning A-Z^{TM}: Hands-On Artificial Neural Networks*. [S.l.]: Udemy, 2018. <https://www.udemy.com/deeplearning/learn/v4/overview>,. Citado 3 vezes nas páginas 6, 15 e 18.

GALO, M. Aplicação de redes neurais artificiais e sensoriamento remoto na caracterização ambiental do parque estadual morro do diabo. *São Carlos*, 2000. Citado na página 10.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>. Citado 4 vezes nas páginas 10, 11, 14 e 15.

KINGMA, D.; BA, J. Adam: A method for stochastic optimization. In: . [S.l.: s.n.], 2014. Citado na página 23.

KUSSUL, N. et al. Deep learning classification of land cover and crop types using remote sensing data. *IEEE Geoscience and Remote Sensing Letters*, IEEE, v. 14, n. 5, p. 778–782, 2017. Citado na página 20.

LONG, J.; SHELHAMER, E.; DARRELL, T. Fully convolutional networks for semantic segmentation. In: IEEE. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.], 2015. p. 3431–3440. Citado na página 22.

MENDES, C. C. T.; FRÉMONT, V.; WOLF, D. F. Exploiting fully convolutional neural networks for fast road detection. In: IEEE. *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. [S.l.], 2016. p. 3174–3179. Citado na página 19.

MENDES, T. S. G.; POZ, A. P. D. Classificação de imagens aéreas de alta-resolução utilizando redes neurais artificiais e dados de varredura a laser. *Simpósio Brasileiro de Sensoriamento Remoto*, v. 15, p. 7792–7799, 2011. Citado 2 vezes nas páginas 10 e 11.

PRöVE, P. L. *An Introduction to different Types of Convolutions in Deep Learning*. [S.l.]: Towards Data Science, 2017. Citado 2 vezes nas páginas 6 e 17.

RONNEBERGER, O.; P.FISCHER; BROX, T. U-net: Convolutional networks for biomedical image segmentation. In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. [S.l.]: Springer, 2015. (LNCS, v. 9351), p. 234–241. Citado 2 vezes nas páginas 6 e 21.

SILVA, F. et al. Classification of forest roads and determination of route using geographic information system. *Revista Árvore*, SciELO Brasil, v. 40, n. 2, p. 329–335, 2016. Citado 2 vezes nas páginas 10 e 11.

SRIVASTAVA, N. et al. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, JMLR. org, v. 15, n. 1, p. 1929–1958, 2014. Citado na página 23.

TREMBLAY-GOSSELIN, J.; CRETU, A.-M. A supervised training and learning method for building identification in remotely sensed imaging. In: IEEE. *Robotic and Sensors Environments (ROSE), 2013 IEEE International Symposium on*. [S.l.], 2013. p. 73–78. Citado na página 19.

XU, B. et al. Empirical evaluation of rectified activations in convolutional network. In: *http://arxiv.org/abs/1505.00853*. [S.l.: s.n.], 2015. Citado na página 23.

ZOU, Q. et al. Deep learning based feature selection for remote sensing scene classification. *IEEE Geoscience and Remote Sensing Letters*, IEEE, v. 12, n. 11, p. 2321–2325, 2015. Citado na página 20.