UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO

DEPARTAMENTO DE ESTATÍSTIA E INFORMÁTICA DA UFRPE

CIÊNCIA DA COMPUTAÇÃO

ALESSANDRO NAZÁRIO DA ROCHA

CLASSIFICAÇÃO DE DOCUMENTOS DE IDENTIFICAÇÃO COM REDES NEURAIS CONVOLUCIONAIS

TRABALHO DE CONCLUSÃO DE CURSO

Recife

19 de Janeiro de 2019

ALESSANDRO NAZÁRIO DA ROCHA

CLASSIFICAÇÃO DE DOCUMENTOS DE IDENTIFICAÇÃO COM REDES NEURAIS CONVOLUCIONAIS

Trabalho de Conclusão de Curso apresentado ao Curso de CIência da Computação, como parte dos requisitos necessários à obtenção do título de Bacharel em Ciência da Computação.

Orientador: Pablo Azevedo Sampaio

Recife

19 de Janeiro de 2019

.

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema Integrado de Bibliotecas da UFRPE
Biblioteca Central, Recife-PE, Brasil

R672c Rocha, Alessandro Nazário da
Classificação de documentos de identificação com redes neurais convolucionais / Alessandro Nazário da Rocha. – 2019.
45 f. : il.

Orientador: Pablo Azevedo Sampaio.
. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) – Universidade Federal Rural de Pernambuco, Departamento de Computação, Recife, BR-PE, 2019.
Inclui referências.

1. Redes neurais (Computação) 2. Imagens digitais – Classificação 3. Documentos oficiais – Identificação I. Sampaio, Pablo Azevedo, orient. II. Título

CDD 004

MINISTÉRIO DA EDUCAÇÃO E DO DESPORTO
UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO (UFRPE)
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

http://www.bcc.ufrpe.br

FICHA DE APROVAÇÃO DO TRABALHO DE CONCLUSÃO DE CURSO

Trabalho defendido por Alessandro Nazário da Rocha às 9 horas do dia 18 de janeiro de 2019, no Auditório do CEAGRI-02 – Sala 07, como requisito para conclusão do curso de Bacharelado em Ciência da Computação da Universidade Federal Rural de Pernambuco, intitulado **Classificação de Documentos de Identificação com Redes Neurais Convolucionais**, orientado por Pablo Azevedo Sampaio e aprovado pela seguinte banca examinadora:

Pablo Azevedo Sampaio

DC/UFRPE

Rafael Ferreira Mello
DC/UFRPE

Dedico este trabalho à Maria Lúcia da Silva, minha mãe, que precisou de poucas palavras que me serviu de maior motivação e transformação do meu pensamento para seguir com os estudos.

> *Filho, enquanto eu estiver com vida irei te ajudar com o que precisar para você estudar. - Silva, M. L.*

Obrigado por todo amor e compreensão nas pequenas e simples atitudes, serei eternamente grato.

E em memória ao meu pai, Lourenço Nazário da Rocha que não se encontra mais neste plano.

Agradecimentos

Agradeço à minha família e amigos, principalmente os que conheci na universidade, obrigado pela paciência, preocupação, cooperação e motivação nos momentos difícies. A todos os professores, como Jeísa e Valmir que proporcionaram conversas que ajudaram a minha vida, obrigado pela paciência e as palavras ditas, assim como a melhor secretária de curso, nossa querida Sandra, obrigado pela paciência e compreensão. Agradeço aos meus colegas de trabalho que proporcionaram motivação, conhecimentos e juntos podemos compartilhar vitórias. Agradeço também ao meu orientador Pablo, que nesses anos foi como ponte de conhecimento, não só na parte acadêmica, como na parte espiritual falando do amor do nosso Senhor Jesus Cristo, tenha certeza que tenho guardado em meu coração as suas palavras. E por fim, e não menos importante, agradeço a Deus por ter sido tão bom comigo. Sem todos os mencionados, não chegaria até aqui.

Resumo

O aprendizado profundo (*Deep Learning*) tem desempenhado um papel importante em processamento de dados, tendo com uma de suas principais técnicas as redes neurais convolucionais, uma de suas aplicações é na área de visão computacional e pode ser utilizada para aprender automaticamente características incluídas em imagens em suas camadas. No entanto, essas redes neurais artificiais profundas precisam de uma quantidade significante de imagens, para os problemas propostos, já separados em categorias para realização dos treinamentos e validações dos modelos que nem sempre são disponíveis. Nexte contexto, neste trabalho foi construído um *dataset* com imagens de documento de identificação brasileiro de Registro Nacional (RG) e Carteira Nacional de Habilitação (CNH) separando-os em algumas categorias e por se tratar de documentos com informações sensíveis, juntar uma quantidade de imagens para obter bons resultados, foi um passo que demandou tempo. Por isso, foram tomados os devidos cuidados para que essas informações sensíveis fossem preservadas. Este trabalho apresenta algumas arquiteturas de redes neurais artificias convolucionais para classificar as imagens em diferentes categorias. Experimentos foram realizados com a utilização de unidade de processamento gráfico (GPU) e com a utilização só de unidade central de processamento (CPU). Resultados de 99% de acertos foram obtidos em alguns cenários que foram testados no decorrer do trabalho para as diferentes arquiteturas propostas.

Palavras-chave: Redes Neurais Convolucionais. Classificação de Imagem. Classificação de Documentos de Identificação.

Abstract

Deep Learning has played an important role in data processing, with one of its main techniques being convolutional neural networks, which has the power to automatically learn features included in images in their layers. However, these deep artificial neural networks need a significant amount of images, for the proposed problem, already separated into categories to perform the training and validations of models that are not always available. In this context, in this work was constructed a dataset with images of Brazilian identification document of National Registry (RG) and National Qualification Card (CNH) separating them into some categories and, since they are documents with sensitive information, add a significant amount of images to obtain good results, it was a step that took time. For this reason, care was taken to preserve this sensitive information. This work presents some architectures of deep artificial neural networks to classify the images for different categories. Experiments were performed using a graphics processing unit (GPU) and using only the central processing unit (CPU). Results above 99% were obtained in some scenarios that were tested in the course of the work for the different proposed architectures.

Key-words: Convolutional Neural Networks. Image Classification. Classification of Identification Documents.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

CNH	Documento de carteira nacional de habilitação
CNN	Rede Neural Convolucional
CPU	Unidade central de processamento
FC	Camada totalmente conectada
GPU	Unidade de processamento gráfico
MLP	Perceptrons multicamadas
OCR	Reconhecimento ótico de caracteres
RG	Registro geral ou documento de identificação

Sumário

1 Introdução

> Nós somos aquilo que fazemos repetidamente. Excelência, então, não é um modo de agir, mas um hábito - Aristóteles

Neste capítulo introdutório, é apresentado o contexto que está inserido o trabalho de conclusão de curso, que é na área de aprendizado de máquina com ênfase em classificação de imagens. Em seguida uma breve visão dos assuntos tratados nos capítulos seguintes e por fim, serão apresentados os objetivos propostos.

1.1 Contexto

A cada dia, novos sistemas e aplicativos estão sendo desenvolvidos para que processos de envio de documentos de identificação passem a serem digitalizados em diferentes aplicações. Para isso, é necessário que esses documentos sejam validados previamente pela própria aplicação para constatar se a pessoa está enviando o documento correto solicitado. Em (1), é apresentado um exemplo da aplicação da classificação de documentos de identificação brasileiro para processos de avaliação e concessão de créditos, que antes eram feitos manualmente e passaram a ser automatizados. Sendo que foi constatado que 30% das propostas recebidas pelos clientes, antes da digitalização, não apresentavam todos os documentos obrigatórios, o problema da classificação de imagem vem para automatizar esta tarefa.

Existem diferentes oportunidades no mercado para aplicar aprendizado de máquina envolvendo a classificação de documentos de identificação em processos que estão deixando de ser feitos manual, passando a ser digital. O aprendizado profundo (*Deep Learning* - DL), se basea em redes neurais artificiais que são modelos de aprendizagem de máquina (Machine Learning - ML), inspirados no mecanismo das células nervosas humanas (neurônios) (2). As redes neurais artificiais são métodos computacionais que usam grupos de neurônios artificiais e são usadas em vários campos, como processamento de imagem, processamento de sinal e modelagem de fontes de energia (3). A visão computacional (*Computer Vision* - CV) é uma das áreas que se beneficia com esses métodos, e proporciona a forma como as máquinas podem ser feitas para obter uma compreensão de alto nível a partir de imagens ou vídeos e que suporta o rápido desenvolvimento da inteligência artificial (*Artificial Intelligence* - AI), e seu progresso tem feito uso prático em vários campos avançados. A CV ajuda a tomar decisões de reconhecimentos, classificações e detecções sobre objetos físicos reais e cenas baseadas em imagens (4), e também é usada para analisar imagens e produzir descrições que podem ser usadas para interagir com o ambiente.

As Redes Neurais Convolucionais (*Convolutional Neural Network* - CNN), são um tipo especializado de rede neural profunda para processamento de dados que possuem uma topologia em grande conhecida (5), este trabalho vai utilizar dados de imagens que podem ser considerados como uma grade de pixels 2D. Recentemente, as abordagens profundas de CNN foram aplicadas com sucesso a conjuntos de dados de classificação de imagens (6, 7). A popularidade atual desses tipos de redes se deu pelo aprimoramento das capacidades de processamento, as Unidades de Processamento Gráfico (Graphic Processing Unit - GPU) atuais, emparelhadas com uma implementação altamente otimizada da convolução 2D são poderosos o suficiente para facilitar o treinamento de CNN. Um outro ponto importante se da pelo aumento significativo da quantidade de dados. Ao inserir uma grande quantidade de imagens em uma *Convolutional Neural Network*, o modelo aprende as caractestícas incluídas nos dados automaticamente em suas camadas, essa estrutura e método de aprendizado é peculizar à aprendizagem profunda, o que faz com que o modelo de DL possua uma precisão extremamente alta.

Este trabalho vai utilizar CNN aplicado a um problema na área de visão computacional. Sabendo que a visão computacional possui sete áreas: pré-processamento, classificação de imagens, detecção de objetos, segmentação de instancias, controle de qualidade visual e reconhecimento ótico de caracteres (OCR), neste trabalho serão utilizados algumas técnicas de pré-processamento e CNN para atacar o problema de classificação de imagens, problema esse que recebeu grande atenção da comunidade científica. As imagens que serão utilizadas podem variar de digitalizações de alta qualidade a fotos de celulares de baixa qualidade. Primeiro será treinada uma CNN que pode ter como entrada milhares de exemplos de dados representando as classes que se quer aprender, esse tipo de treinamento é chamado de aprendizado supervisionado porque está sendo fornecido a rede neural uma imagem de uma classe e explicitamente está sendo informado que é uma imagem da classe específica. Em seguida, será realizado os testes, podendo ser entendido como classificar ou categorizar em uma das classes treinadas. Por exemplo, se uma imagem pertence à categoria Carteira de Identidade (RG) ou Carteira Nacional de Habilitação (CNH), que são documentos de identificação de pessoa física brasileiros, que é o problema que este projeto propõe atacar.

Uma CNN que pode classificar imagens de forma robusta, mesmo se essas imagens tenham diferentes rotações, tamanho, iluminação é dita ter uma propriedade chamada invariância, basicamente essa é a premissa do *Data Augmentation* (aumento de dados). Em um cenário real as imagens que vão ser classificadas podem ser capturas em diferentes situações, tais como, brilho, rotação, escala etc. O modelo a ser treinado precisa ter dados que reflitam essas realidades. Com *Data Augmentation* novos dados sinteticamente alterados são adicionados ao *dataset* de treinamento (8).

1.2 Objetivos

1.2.1 Objetivo geral

Implementar um modelo de Rede Neural Convolucional para classificar documentos de identificação de pessoa física brasileiros. O objetivo é classificar documento de identificação e não faz parte do escopo identificar se o documento é falso ou não.

1.2.2 Objetivos específicos

- Tornar este trabalho uma introdução às redes convolucionais que possa ser útil a alunos que queiram aprender e aplicar esta técnica, uma vez que este assunto não é visto na nossa graduação (do Bacharelado em Ciência da Computação da UFRPE, em Recife-PE).
- Criar um *dataset* com imagens de documentos de identificação de pessoa física brasileiros.
- Utilizar framework em python para fazer a implementação do modelo.
- Aplicar técnicas de *Data Augmentation.*

1.3 Estrutura do trabalho de conclusão de curso

Esse trabalho apresentará no **Capítulo 2** a técnica de Rede Neural Convolutional e no **Capítulo 3** alguns trabalhos encontrados sobre classificação de imagens, em seguida no **Capítulo 4**, será apresentada a metodologia utilizada para construção da base de dados, logo após no **Capítulo 5,** as arquiteturas de redes neurais artificias propostas, em seguida o **Capítulo 6** com os resultados e discursões, e por fim, o **Capítulo 7** com as conclusões e trabalhos futuros.

2 Redes Neurais Artificiais

> A imaginação é mais importante que o conhecimento. Porque o conhecimento é limitado a tudo que sabemos e entendemos, ao passo que a imaginação abrange o mundo inteiro, e tudo que será conhecido e entendido. -Einstein, Albert

Este capítulo será apresentado conceitos sobre Redes Neurais Convolucionais, apresentando as suas principais camadas e o seu funcionamento.

2.1 Redes neurais convolucionais

As redes convolucionais foram umas das primeiras redes neurais a resolver importantes aplicações comerciais, já na década de 1990, o grupo de pesquisa em rede neural da AT&T desenvolveu uma rede convolucional para leitura de cheques (9). E permanecem na vanguarda das aplicações comerciais da aprendizagem profunda (5). Hoje em dia, empresas e pesquisadores estão desenvolvendo tecnologias que utilizam redes convolucionais em diferentes cenários práticos. Segundo Goodfellow et al. (2016), redes neurais convolucionais são:

> um tipo especializado de rede neural para processamento de dados que possui uma topologia em grade conhecida. Os exemplos incluem dados de séries temporais, que podem ser considerados como uma grade 1-D levando amostras em intervalos de tempo regulares e dados de imagem, que podem ser considerados como uma grade de pixels 2D.

O nome “convolucional” indica a aplicação de uma operação matemática chamada convolução que é um tipo especializado de operação linear que será detalhado na próxima sub-seção. O trabalho (10), apresenta que um dos seus principais objetivos de uma CNN é reduzir o número de parâmetros que serão ajustados pela rede, isso faz que a parte de treinamento seja melhorada. Já (11), explica que uma CNN consiste em um ou mais camadas convolucionais, muitas vezes com uma camada de subamostragem (*pooling*), que são seguidos por uma ou mais camadas totalmente conectadas como em uma rede neural padrão. O desenho dessa rede foi motivada a partir da descoberta de um mecanismo visual, chamado de córtex visual, no cérebro.

Redes Neurais Convolucionais são muito utilizadas para tarefas de visão computacional. Essa área da visão computacional é uma subárea dentro da inteligência artificial que está preocupada com processamento de vídeo e imagem. Um dos principais exemplos é a aplicação em carros autônomos na detecção de pedestres, detecção de placas e sua classificação. A seguir será conceituado as principais camadas de uma

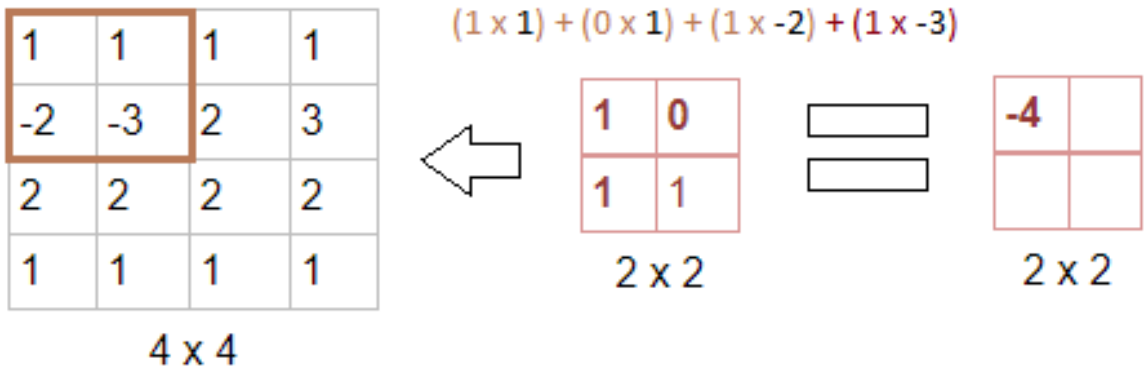
CNN: convolução, *pooling* e também será descrita a camada totalmente conectada que usualmente são colocadas após as camadas convolucionais.

2.1.1 Camada de Convolução

Segundo (10), as camadas convolucionais (*Convolutional Layer* - CL), são conjuntos de filtros que percorrem sequencialmente os dados de entrada (ou camada anterior) e então produzem matrizes chamadas de mapa de característica (*feature map*). Os filtros são detectores de características (*kernels*), inicialmente recebe valores aleatórios, na etapa de treinamento da CNN os seus valores são ajustados a fim de encontrar os melhores valores para que as *feature maps* geradas contenham padrões dos dados de entrada. Não é preciso passar como parâmetro os valores para os *kernels*, basta apenas indicar o seu tamanho m x n que a rede computa os valores para fazer a transformação da imagem.

Na Figura 1, a imagem de entrada é representada pela matriz 4 x 4 e o *kernel* representado pela matriz 2 x 2, geralmente o *kernel* é uma matriz menor com pesos, pois a convolução permite reduzir os parâmetros a cada camada. É realizado a primeira etapada da convolução, o *kernel* passa por uma parte da matriz de entrada onde cada elemento sobreposto são multiplicandos. Em seguida é feita a soma resultando no primeiro elemento da matriz de características que está sendo criada.

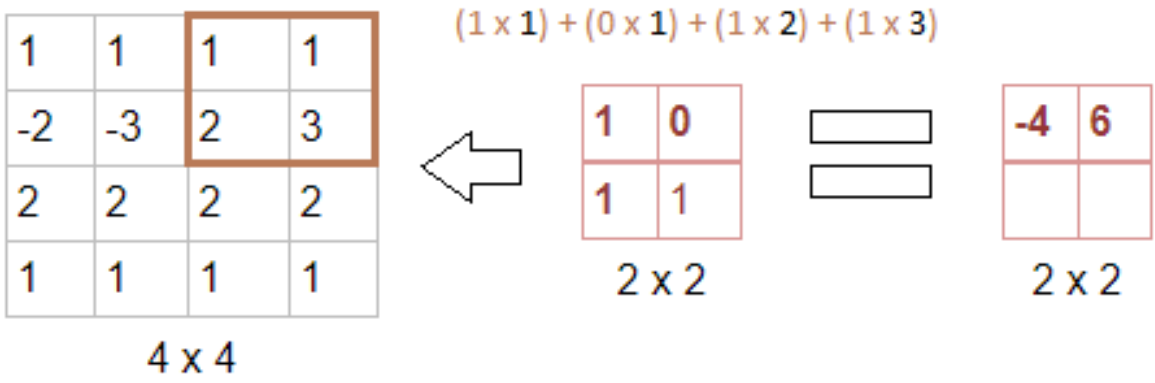
Figura 1 – Exemplo de convolução: etapa 1



Fonte: Autor

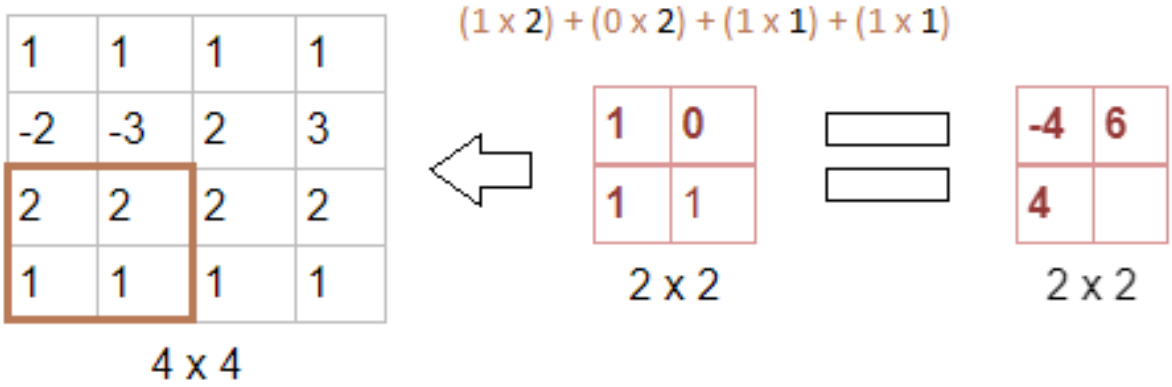
A próxima passagem do *kernel* vai depender do tamanho do passo que vai ser dado da esquerda para direita, dependendo do tamanho das matrizes esse tamanho é variável. Nesse exemplo, o tamanho do passo vai ser igual ao tamanho do *kernel,* um dos efeitos de sua escolha é o tempo de processamento da convolução que pode aumentar ou diminuir. As Figuras 2, 3 e 4 , mostram as etapas seguintes dessa convolução (todas as etapas podem ocorrer em paralelo) movendo o *kernel* pela imagem e realizando as operações formando a matriz resultante à direita.

Figura 2 – Exemplo de convolução: etapa 2



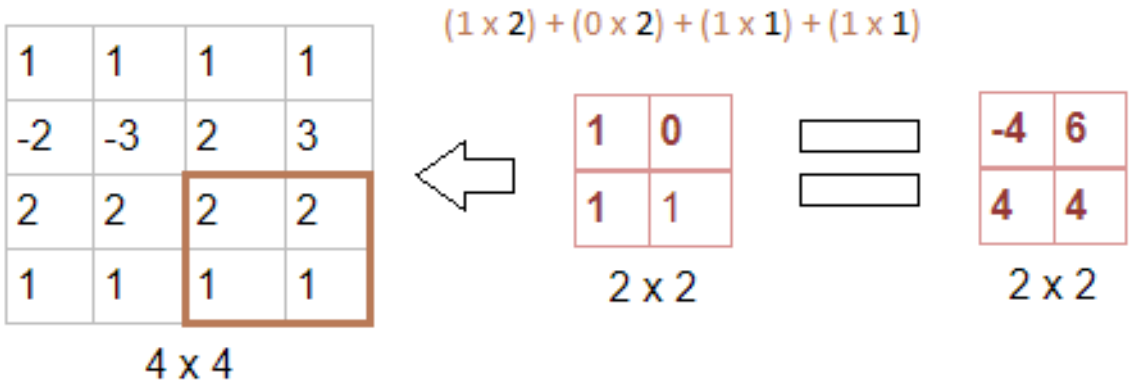
Fonte: Autor

Figura 3 – Exemplo de convolução: etapa 3



Fonte: Autor

Figura 4 – Exemplo de convolução: etapa 4



Fonte: Autor

A matriz 2 x 2 resultante dessa convolução é chamada de *feature map*. A quantidade de *feature map* geradas é especificada para a convolução, ou seja, podem ser gerados diferentes *features maps* a partir de seus respectivos *kernels*. Esse exemplo

hipotético a imagem só teria um canal, sendo uma imagem com tons de cinza. Se fosse uma imagem colorida, o *kernel* seria multiplicado pelas três matrizes que representariam cada cor do pixel formado por RGB (Red, Green e Blue), gerando três canais para o mapa de características.

A convolução destaca na imagem os padrões e características que ela possui. Ao realizar a convolução, dependendo de onde começar o passo, é possível que seja perdida as margens da imagem original, o tratamento a ser dado para isso pode ser simplesmente ignorar, zerar ou replicar os seus valores entre outros. A realização de convolução, que é uma operação linear, gerou a *feature map* resultante na Figura 4, para que a rede venha aprender para problemas que a classificação alcance não linearidade é aplicada uma função de ativação para adicionar uma não linearidade, na seção 2.2 são apresentadas algumas dessas funções. A Tabela 1 representa o resultado da sua aplicação aplicação da função de ativação ReLu, umas das vantagens é aumentar o poder de representação e aumentar a esparsidade.

Tabela 1 – *Feature map* gerada pela convolução após ser aplicada a função de ativação

0	6
4	4

Fonte: Autor

2.1.2 Camada de *Pooling*

A partir do feature *map*, vai ser aplicado a operação de *pooling* que tem como um dos seus objetivos enfatizar ainda mais as características das imagens que se quer fazer a classificação. Então, independente como essa imagem irá estar quando passar pelo algoritmo (exemplos, achatadas, rotacionadas em graus, ou/e em cima de uma mesa, mão de uma pessoa etc.), o *pooling* pode ajudar realçar suas principais características, ou não dependendo do caso.

O método mais comum utilizado para problemas de classificação é chamado de *Max Pooling* (existe também o mínimo e a média para fazer o *pooling*), pega o maior valor da sua região de dimensões m x n estabelecidas, percorrendo a matriz ao qual quer maximizar da esquerda para direita, o passo pode ser de uma em uma coluna ou até o tamanho da matriz utilizada para fazer o *pooling.*

Outros fatores importantes é que o *pooling* ajuda a evitar o sobreajuste (*overfitting*) e ruídos desnecessários. *Overfitting* é quando a rede neural se adapta demais aos dados de treinamento fazendo com que seu erro diminua nessa etapa, mas na etapa de teste o erro continua alto, ou seja, a rede passa a não generalizar para

dados diferentes, não vistos no processo de treinamento. Existe também o conceito de subajuste (*underfitting*), que aparece quando o erro da etapa de treinamento não reduz, isso significa que a rede não consegue aprender padrões.

Como exemplo, para a passagem do *max pooling* na Tabela 1, será considerado uma subamostragem de tamanho 1 x 2, resultando a Tabela 2.

Tabela 2 – Resultado do max pooling

6
4

Fonte: Autor

A vantagem de utilizar é se os pixels que representam a caracterisca encontrada nessa camada podem vir em posições diferentes na proximidade nas próximas imagens, o *pooling* tem o papel de enfatizar.

2.1.3 Camada Totalmente Conectada

A transformação do formato de matriz oriunda das convoluções para o formato de vetor é feita originando a camada totalmente conectada (*Fully connected* - FC), mas nem toda camada totalmente conectada é oriunda dessa forma, podemos dizer que uma FC é semelhante a uma camada de Perceptrons, em (12) no Capítulo 7 começa a explicação sobre os conceitos do Perceptrons e de uma camada de Perceptrons no Capítulo 8. As camadas totalmente conectadas, podem ser chamadas de redes densas. Essa camada pode ser a precedente a de saída ou sendo a própria camada de saída densa, são camadas finais com a finalidade de combinar características de alto nível, resultantes de camadas anteriores, podem existir mais de uma FC, tais redes com múltiplas camadas são chamadas de Perceptrons Multicamadas ou MPLs (Multilayer Perceptrons), conforme é explicado no Capítulo 9 em (12).

Na literatura existem variações de arquiteturas sem as camadas totalmente conectadas, originando as redes denominadas Redes Totalmente Convolucionais. As redes neurais podem ter diferentes arranjos das suas camadas, com convoluções, *pooling* e camadas totalmente conectadas, cabe a cada problema ser utilizado uma arquitetura para ser treinada e então testada para que forneça resultados promissores que venham atender as necessidades desejadas.

2.2 Função de ativação

As funções de ativações servem como propagadores de impulsos de uma camada para outra, como por exemplo, a função y = max(0 , x) que é a função de ativação linear retificada (Rectified Linear Unit - ReLu). A ReLu recebe um valor x, para valores negativos ela zera, senão continua o mesmo número, essa é a função mais comum utilizada geralmente entre camadas intermediárias em redes neurais convolucionais (13) . Em (5), é relatado que é muito utilizado a função de ativação softmax em problemas de classificação de imagens na camada de saída, que gera uma probabilidade que indica a classe identificada, os seus valores podem ser no intervalo 0 a 1, sendo que a soma dos valores de cada classe de saída será igual a 1. Existem outras funções de ativações que foram utilizadas no problema de classificação de imagens citadas em artigos da década de 90 (9), por exemplo, sigmóide e tangente hiperbólica (tanh) que atualmente não são muito utilizadas para o problema de visão computacional com redes convolucionais. Para o problema de classificação de imagem, a função de ativação tem como característica fazer uma transformação não linear ao longo do sinal de entrada (12).

3 Arquitetura de trabalhos relacionados de classificação de imagens

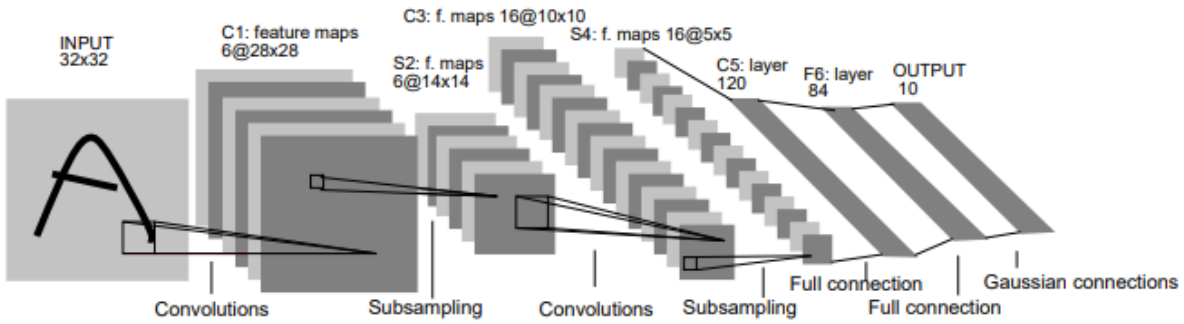
> Os computadores jamais passarão de escravos de estímulos programados, ainda que incorporem um processo de auto-aprendizagem e levem em consideração a “lógica paraconsistente”, ou seja, que admitam contradições das informações. - Cury, Augusto

A primeira arquitetura apresentada na seção 3.1, é muito referenciada na literatura e ataca o problema de classificação de imagens com a utilização de redes neurais convolucionais, mesmo sendo aplicado a um problema diferente em relação a este projeto. Por outro lado, o artigo apresentado na seção 3.2, utilizou uma arquitetura aplicada justamente ao problema de classificação de documentos. Foi encontrado apenas esse trabalho na literatura com identificação de documentos.

3.1 Arquitetura LeNet-5

O trabalho (9), propôs uma arquitetura de rede neural convolucional para reconhecimento de caracteres manuscritos que recebeu o nome de LeNet-5, a figura 5, apresenta essa arquitetura de rede neural que possui 2 camadas convolucionais e 3 camadas totalmente conectadas. A entrada (INPUT) da LeNet-5 é uma imagem em escala de cinza com tamanho de 32 pixels de altura e 32 pixels de largura (32 x 32). Cada ponto de uma imagem é chamado de pixel, essa imagem tem 1024 pixels, que é o resultado da multiplicação da sua altura pela sua largura. As camadas convolucionais são responsáveis por extrair as características (C1, C3), as camadas de *pooling* reduzem a dimensionalidade da rede (S2, S4) e e as camadas totalmente conectadas (C5, F6) ligam as saídas da camada anterior.

Figura 5 – Arquitetura da rede neural convolucional LeNet-5



Fonte: LECUN, Y. et al. (1998)

Descrição abaixo de cada camada dessa rede:

- A imagem passa pela primeira camada convolucional, é utilizado na convolução *kernel* de tamanho 5 x 5, de passo 1 e o resultado são 6 *feature maps* (C1) de tamanho 28 x 28. As dimensões da imagem mudam de 32 x 32 x 1 para 28 x 28 x 6. Apesar da arquitetura não mostrar é aplicada uma função de ativação tanh para adicionar não linearidade, esse passo nem sempre é representado na rede.
- Em seguida, é feito o *pooling* com tamanho de 2 x 2 x 6, passo 2, que está pegando 4 pixels e transformando em um só resultando uma matriz de tamanho 14 x 14 x 6 (S2). O tamanho da imagem anterior diminui a metade.
- Existe uma segunda convolução são utilizado mais *kernel* de tamanho 5 x 5, passo 1, que geram 16 *feature maps* de tamanho 10 x 10 (C3). Nessa etapa, dos 16 *feature maps* só 10 são conectados a 6 da camada anterior, sendo que para cada *feature map* os escolhidos são diferentes. A principal razão é quebrar a simetria na rede e manter o número de conexões dentro de limites razoáveis resultando em dimensões 10 x 10 x 16.
- A quarta camada é feito um *pooling* novamente de tamanho 2 x 2, passo 2, o resultado 5 x 5 x 16 (S4) é a metade da matriz anterior.
- A partir daqui as camadas passam a ser de uma rede neural artificial densa, existe um "achatamento" que o artigo chama de *flattening convolutional layer,* esse tipo de camada apenas transforma as matrizes em um vetor, gerando uma camada totalmente conectada (*full connection*) com 120 *map features* (C5), cada um com tamanho 1 x 1. Cada um dos 120 pixels são conectados a todos os 400 pixels (5 x 5 x 16) das *map features* da camada (S4). Estes 120 neurônios vão servir de "entrada" na camada oculta da rede neural densa.
- A sexta camada (F6), é mais uma camada totalmente conectada com 84 pixels.
- Por fim a camada de saída (*OUTPUT*), com uma função de ativação *sofmax* totalmente conectada com os 10 possíveis valores de saída.

3.2 Arquitetura AlexNet

No momento do desenvolvimento deste trabalho só foi encontrado um artigo que trata do problema de classificação de documentos de identificação (14) *,* tem como objetivo identificar o tipo de documento e o país de origem. O artigo utiliza pipeline de classificação de imagens com máquina de vetores de suporte (*Support Vector Machine* - SVM) e com CNN, só a parte com CNN será destacada, pois é a parte que é do interesse deste trabalho. O trabalho realizou experimentos com um conjunto de dados divididos em 9 classes de documentos franceses, chamados de:

- carteira de identidade (frente)
- carteira de identidade (verso)
- passaporte (antigo)
- passaporte (novo)
- cartão de residência (antigo frente)
- cartão de residência (antigo verso)
- cartão de residência (novo frente)
- cartão de residência (novo verso)
- Licença para dirigir

Um total de 527 imagens foram divididas em treinamento e teste variando de 26 a 136 por classe. Em seguida um conjunto maior de imagens com um total de 2399 variando de 86 a 586 por classe são usadas. O último conjunto de dados consiste de 446 imagens variando de 8 a 110 por classe, sendo o conjunto de dados divididos em 10 classes de documento belgas, chamados de:

- carteira de identidade 1 (frente)
- carteira de identidade 1 (verso)
- carteira de identididade 2 (frente)
- carteira de identidade 2 (verso)
- cartão de residência (antigo verso)
- cartão de residência (antigo front)
- cartão de residência (novo verso)
- cartão de residência (novo frente)
- passaporte (antigo)
- passaporte (novo)

Uma das arquiteturas de CNN citada, referência uma arquitetura criada por (13) chamada de CNN-F, ao ler esse outro artigo os autores dizem que se basearam na arquitetura AlexNet, que possui 5 camadas convolucionais e 3 camadas totalmente

conectadas criada por (6) para sua criação. A arquitetura tem 8 camadas, sendo 5 convolucionais e 3 totalmente conectadas sendo a imagen de entrada com tamanho 224 x 224. Essa arquitetura é apresentada na Figura 6,

Figura 6 – A arquitetura da rede neural convolucional fast (CNN-F)

Arch.	conv1	conv2	conv3	conv4	conv5	full6	full7	full8
CNN-F	64x11x11 st. 4, pad 0 LRN, x2 pool	256x5x5 st. 1, pad 2 LRN, x2 pool	256x3x3 st. 1, pad 1 -	256x3x3 st. 1, pad 1 -	256x3x3 st. 1, pad 1 x2 pool	4096 drop-out	4096 drop-out	1000 soft-max

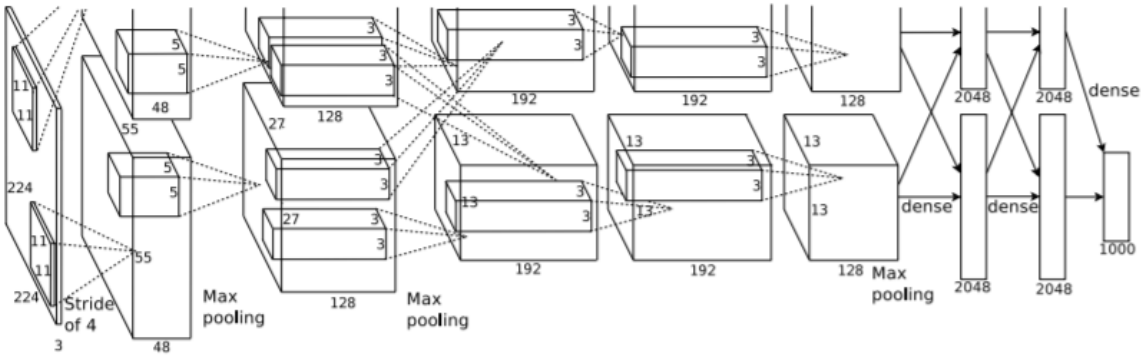
Fonte: Ken Chatfield et al. (2017)

Descrição abaixo de cada camada dessa rede:

- A primeira convolução (conv1) filtra a imagem de entrada 224 x 244 x 3 com 96 *kernels* de tamanho 11 x 11, de passo 4.
- Existe uma segunda convolução utilizando 256 kernels de tamanho 5 x 5, passo 1.
- As convoluções (conv3, conv4 e conv5) utilizam 256 *kernels* de tamanho 3 x 3 e passo 1 .
- Duas camadas (full6 e full7) totalmente conectas seguida de *dropout*
- Por fim a camada de saída com a função de ativação *sofmax.*

Um ponto a destacar dessa rede neural é a utilização de *dropout*, que consiste em configurar para zero o valor do neurônio. É possível determinar a quantidade de neurônios que terão os valores zerados, a probabilidade aplica no artigo (6) foi de 50%. Quando chega nessa camada, 50% dos neurônios não contribuem para a próxima camada, com isso, toda vez que uma entrada é apresentada, a rede neural mostra uma arquitetura diferente, mas todas essas arquiteturas compartilham pesos. Alex Krizhevsky et al. (2012), afirmam que sem a aplicação da técnica de *dropout*, a rede deles exibe *overfitting*. A Figura 7 mostra a arquitetura AlexNet que foi criada em (6).

Figura 7 – Arquitetura AlexNet



Fonte: Krizhevsky, Sutskever e Hinton (2012)

A arquitetura LeNet-5 apresentada no artigo (9), foi utilizada com base para implementação do modelo neste trabalho por ser muito referenciada na literatura. Já o artigo (14), foi importante para a melhor contextualização do problema, servindo de referência para definir o tamanho inicial das imagens (244 x 244 x 3) e mostrando que para esse problema um *dataset* não muito grande pode ser utilizado obtendo resultados promissores. Mas, por motivo de tempo, não foram feitas as implementações dos modelos para este trabalho.

4 Metodologia

Fazer a pergunta certa é mais difícil do que respondê-la. - Cantor, Georg

4.1 Levantamento das imagens para o *dataset*

As redes neurais precisam de dados para que seus modelos sejam treinados para executar determinadas tarefas em diferentes soluções e no caso de aprendizado supervisionado existe a necessidade que os dados sejam rotulados para que isso seja possível. Nesse contexto, as imagens encontradas já foram sendo colocadas em pastas de acordo com a sua categoria e por se tratar de documentos que possuem dados sensíveis, não foi encontrado nenhum *dataset* público que pudesse ser utilizado nesse projeto. Logo, foi necessarios utilizar alguns métodos para reunir um número máximo de imagens a serem utilizadas no *pipeline,* sendo as imagens recolhidas de duas formas:

- Através de buscas na internet
- Campanha com pessoas conhecidas, que expandiram para outras pessoas

O *dataset* foi criado baseado em 6 categorias representadas na Tabela 3 abaixo.

Tabela 3 – Distribuição das imagens por categoria de documento

Categoria	Quantidade
cnh	268
cnh_front	364
cnh_back	269
rg_front	185
rg_back	146
another_image	105

Categoria	Quantidade

Fonte: Autor

- cnh - documento de Carteira Nacional de Habilitação (CNH) mostrando frente e verso do documento.
- cnh_front - CNH mostrando só a parte que contém a foto do documento
- cnh_back - CNH mostrando o verso do documento
- rg_front - Registro Geral ou Documento de Identificação (RG) mostrando a frente do documento
- rg_back - RG mostrando o verso do documento
- another_image - São imagens de cartão de crédito e outros documentos não mensionados anteriormente.

4.1.1 Pré-processamento dos documentos

Inicialmente foi preciso fazer um tratamento manual de corte em todas as imagens utilizando a ferramenta *Photos do* Windows *10* a fim de separar em imagens diferentes documentos localizados na mesma foto. A seguir alguns exemplos de documentos encontrados com as informações sensíveis ocultadas:

- A Figura 8, foi uma foto tirada com a CNH em cima de uma mesa.

Figura 8 – Parte da frente da Carteira Nacional de Habilitação antes do corte

Fonte: Autor

- A Figura 9, é o resultado da extração da categoria cnh_front retirada da figura acima.

Figura 9 – Parte da frente da Carteira Nacional de Habilitação depois do corte

Fonte: Autor

- A Figura 10, possibilitou extrair 4 categorias de documentos:
  - cnh - Figura 11

- cnh_front - Figura 12
- cnh_back - Figura 13
- rg_back - Figura 14

Figura 10 – Documentos que foram escaneados de RG e CNH antes do corte

Fonte: Autor

Figura 11 – Carteira Nacional de Habilitação depois do corte

Fonte: Autor

Figura 12 – Carteira Nacional de Habilitação frente depois do corte

Fonte: Autor

Figura 13 – Carteira Nacional de Habilitação verso depois do corte

Fonte: Autor

Figura 14 – Carteira de Identidade verso depois do corte

Fonte: Autor

Mesmo com limitações de disponibilidade de imagens de documentos de identificação de pessoa física brasileiros suficientes para treino, que pode dificultar o treinamento de modelos que obtenham resultados robustos para serem colocados em aplicações reais, este trabalho conseguiu juntar um bom número de imagens (foram 1337 imagens reunidas) comparado ao *dataset* utilizado no artigo (14) que ataca esse problema de classificação de imagens de documentos.

4.1.2 *Data Augmentation*

Os tipos de imagens que são necessárias para esse projeto possuem informações sensíveis, por isso, foi um desafio conseguir o maior número de dados. Uma maneira de contornar este problema, foi utilizada a metodologia de *Data Augmentation* (aumento de dados), que consiste em expandir o conjunto de dados de treinamento com novos exemplos de dados derivados do conjunto de exemplos original. Em um cenário

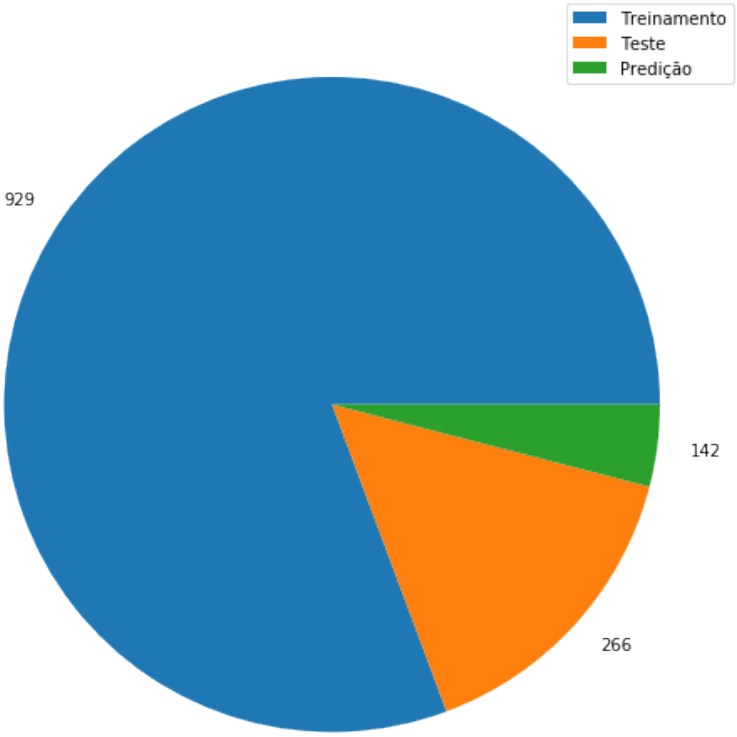
real as imagens que vão ser classificadas podem ser capturas em diferentes situações, tais como, brilho, rotação, escala etc. O modelo a ser treinado precisa ter dados que reflitam essas realialidades, com *Data Augmentation* novos dados sinteticamente alterados são adicionados ao *dataset* (8).

As imagens iniciais vinheram com diferentes dimensões, então o primeiro passo foi deixar todas as imagens com tamanho 224 x 244 pixels, em seguida o método de *data augmentation* foi aplicado. A seguir algumas das técnicas de aumento de dados que foram aplicadas em todas a imagens separadas para o treinamento. Estas modificações podem ser operações básicas de manipulação de imagens, tais como:

- *Rotation 90, 180 e 270 degrees*
  - A rede tem que está preparada para reconhecer o documento em qualquer orientação, cada imagem irá gerar 3 novas imagens, essa técnica foi escolhida com base em experiência pessoal por ter usado um serviço pago que não funcionava bem com imagens rotacionas.
- *Minor Rotation*
  - Cada imagem vai gerar 5 novas imagens com rotações de 48, 16, -16, -48 e -80 graus.
- *Lighting condition*
  - As condições de iluminação das imagens são variadas adicionando ruído gaussiano as imagens. Essa técnica é muito importante porque simula a diversidade de iluminação em diferentes ambientes que as imagens podem ser capturadas.

Por se tratar de um problema que envolve documentos de identificação que podem ser escaneados ou fotografados rotações e iluminações diferentes podem surgir. Por isso a utilização das técnicas citadas acima. O *dataset* inicial contava com 1337 imagens, no qual foram distribuidas para as etapas do modelo conforme a Figura 15, onde aproximadamente, 70% das imagens são utilizadas para o treinamento, 20% para o teste e 10% para a predição. Apenas nas imagens de treinamento foram aplicadas as técnicas de *data augmentation*, as 929 foram aumentadas para 3716 imagens.

Figura 15 – Gráfico com a distribuição das imagens de documento para as etapas de treinamento, teste e predição.



Fonte: Autor

4.1.3 TensorFlow

O TensorFlow é uma biblioteca de aprendizado de máquina de código aberto para pesquisa e produção (15), que está sendo utiliza neste projeto para criação e treinamento da rede neural para classificação de documentos de identificação. Foi desenvolvido pela empresa Google e lançado sob a licença de código aberto Apache 2.0 em 9 de novembro de 2015. O TensorFlow tem suporte aos dispositivos de computação CPU e GPU. O desenvolvimento foi feita na ferramenta de pesquisa para educação e pesquisa em aprendizado de máquina chamada de Colaboratory ou Google Colab (Google (2018)).

5 Arquiteturas de redes neurais artificias utilizadas

> A tarefa não é ver o que nunca foi visto antes, mas pensar o que nunca foi pensado antes sobre o que você vê todos os dias . - Schrödinger, Erwin

Existe uma grande variedade de arquiteturas de redes neurais convolucionais. As escolhidas neste capítulo foram baseadas nas redes utilizadas nos trabalhos relacionados mensionadas no Capítulo 3. O treinamento é feito utilizando com o passo para frente (*forwad pass*), onde a imagem é passada pela rede então gerando as previsões de saídas, em seguida o passo para trás (*backward pass*), onde os pesos (incluindo os dos *kernels*) são atualizados, o algoritmo de *backpropagation* é utilizado.

> O objetivo do backpropagation é otimizar os pesos para que a rede neural possa aprender a mapear corretamente as entradas para as saídas. (12)

Na saída da rede neural é usada uma função de custo *softmax cross entropy*, alguns resultados de pesquisas sugerem a utilização dessa função, mais detalhes podem ser encontrados no Capítulo 18 em (12). Para calcular o erro, essa função *softmax cross entropy* recebe os valores de saídas calculados pela rede neural e os valores que eram esperados para cada imagem, em seguida é utilizado um algoritmo de otimização de taxa de aprendizagem adaptativa chamado de Adam. o Adam foi escolhido através de realização de testes de treinamentos de forma empírica, tendo o treinamento gerado bons resultados (que serão apresentados no Capítulo 6.1 deste trabalho). Existem outros algoritmos otimizadores que também poderiam ter sido escolhidos, com descida de gradiente estocástico (6), mas devido ao tempo não foram testados.

5.1 Rede neural convolucional C2P2TC2

A primeira rede neural profunda proposta foi baseada na LeNet-5 (9), possui duas camadas convolucionais, duas camadas de pooling, depois é feito um “achatamento” dos mapas de características transformando as matrizes em um vetor com as características que será a entrada da camada totalmente conectada, além das camadas de entrada e saída, a partir desse momento essa arquitetura será chamada de **C2P2TC2**. A LeNet-5 utiliza entre as camadas a função de ativação tanh, mas neste trabalho, após cada convolução é aplicado a função de ativação ReLu que é mais utilizada nos trabalhos mais recentes e para gerar os resultados de saída a função de ativação Softmax. Essa arquitetura está sendo representada pela tabela 4, que difere da tabela 5 apenas o tamanho da imagem de entrada.

Tabela 4 – Arquitetura C2P2TC2 com entrada 224 x 224

Camada		**Feature Map**	**Tamanho**	**kernel**	**Passo**	**Função de ativação**
Entrada	Imagem	1	224 x 224			
1	Convolucional	32	224 x 224	5 x 5		Relu
2	Max pooling	32	112 x 112	2 x 2	2	
3	Convolucional	64	112 x 112	5 x 5		Relu
4	Max pooling	64	56 x 56	2 x 2	2	
	“Achatamento”		200704			
6	Total Conectada		1024			Relu
Saída	Total Conectada		6			Softmax

Fonte: Autor

Tabela 5 – Arquitetura C2P2TC2 com entrada 112 x 112

Camada		**Feature Map**	**Tamanho**	**Kernel**	**Passo**	**Função de ativação**
Entrada	Imagem		112 x 112			
1	Convolucional	32	112 x 112	5 x 5		Relu
2	Max pooling	32	56 x 56	2 x 2	2	
3	Convolucional	64	56 x 56	5 x 5		Relu
4	Max pooling	64	28 x 28	2 x 2	2	
5	“Achatamento”		50176			
6	Total conectada		1024			Relu
Saída	Total conectada		6			Softmax

Camada	Feature Map	Tamanho	Kernel	Passo	Função de ativação

Fonte: Autor

5.2 Rede Neural Convolucional C1P1TC2

Essa arquitetura é parecida com a arquitetura C2P2TC2, a diferença é a remoção da segunda camada convolucional e o seu respectivo *pooling,* a camada Totalmente conectada contínua com tamanho 1024 e a de saída com tamanho 6.

Tabela 6 – Arquitetura C1P1TC2 com entrada 112 x 112

Camada		Feature Map	Tamanho	Kernel	Passo	Função de ativação
Entrada	Imagem		112 x 112			
1	Convolucional	32	112 x 112	5 x 5		
2	Max pooling	32	56 x 56	2 x 2	2	Relu
3	“Achatamento”		100352			
4	Total conectada		1024			Relu
Saída	Total conectada		6			Softmax

Fonte: Autor

5.3 Rede Neural Convolucional C1P1TC2b

Essa arquitetura teve uma alteração do tamanho da matriz para a realização do *pooling* que passou a ser 4 x 4 e passo 4.

Tabela 7 – Arquitetura C1P1TC2b com entrada 112 x 112

Camada	Feature Map	Tamanho	Kernel	Passo	Função de ativação

Camada		**Feature Map**	**Tamanho**	**Kernel**	**Passo**	**Função de ativação**
Entrada	Imagem		112 x 112			
1	Convolucional	32	112 x 112	5 x 5		
2	Max Pooling	32	28 x 28	4 x 4	4	Relu
3	“Achatamento”		25088			
4	Total conectada		1024			Relu
Saída	Total conectada		6			Softmax

Fonte: Autor

6 Resultados e Discussões

> Uma pessoa que nunca cometeu um erro, nunca tentou nada novo. - Einstein, Albert

Neste capítulo serão apresentados e analisados os dados dos experimentos realizados com as diferentes configurações das redes neurais artificiais profundas propostas.

6.1 Resultados obtidos com a arquitetura C2P2TC2

Primeiramente, foi feito um experimento sem utilização de GPU e com tamanho de 224 x 224 das imagens de entrada. No entanto, a ferramenta Google Colab travou em tempo de execução. Isso aconteceu já na parte de treinamento e mesmo com uma quantidade menor de imagens (300 e 200 imagens), os recursos disponíveis de memória da máquina utilizada não foram suficientes, pois além das imagens, ficam na GPU, a rede e as *feature maps (*no momento de treinamento*)* de cada camada. Em seguida, com a utilização de GPU, foram feitos alguns testes, mas ocorreu o mesmo problema de travamento com uma quantidade acima de 300 imagens para a etapa de treinamento da rede neural profunda mesmo com a utilização da GPU, por falta de alocação de recursos disponíveis para a conta gratuita. A continuidade da execução do pipeline se deu com a alteração do tamanho da imagem de entrada da rede para 112 x 112, isso foi feito para diminuir a quantidade de parametros para ser suportado pelos recursos de hardware disponíveis.

A tabela 8 abaixo, apresenta os resultados obtidos para o testes e predições para 10, 30, 50 e 100 épocas. O número de épocas foram escolhidos de forma empírica, o erro nas etapas de treinamentos e testes foram diminuindo de forma progressiva sem apresentar *overfitting*. Para 100 épocas resultados de teste foi de 98,4962% de acerto, em seguida foi feita uma predição obtendo 97,1830% de acertos onde sua matriz de confusão é exibido na tabela 9. A categoria cnh_front apresentou um falso negativo (o modelo classificou como another_image, quando na verdade era para ser um cnh_front), por outro lado another_image apresentou um falso positivo (foi classificado como another_image, quando era para ser cnh_front), enquanto que a categoria rg_front teve um falso positivo (o modelo classificou como rg_front, sendo que era para ser rg_back), já rg_back teve um falso negativo e dois falsos positivos (primeiro o modelo classificou com rg_front, quando se esperava rg_back, em seguida o modelo classificou como rg_back, mas era para ser another_image).

Tabela 8 – Resultados para arquitetura C2P2TC2

Nº épocas no treinamento	10	30	50	100
Acurácia - Teste	80,4511%	90,9774%	98,4962%	99,2481%
Acurácia - Predição	76,0563%	82,3943%	96,4788%	97,1830%

Fonte: Autor

Tabela 9 – Matriz de confusão para 100 épocas - C2P2TC2

	cnh	**cnh_front**	**cnh_back**	**rg_front**	**rg_back**	**another_image**
cnh	**27**	0	0	0	0	0
cnh_front	0	**35**	0	0	0	1
cnh_back	0	0	**27**	0	0	0
rg_front	0	0	0	**20**	0	0
rg_back	0	0	0	1	**16**	0
another_image	0	0	0	0	2	**13**

Fonte: Autor

6.2 Resultados obtidos com a arquitetura C1P1TC2

Um novo treinamento foi feito, dessa vez com uma arquitetura com apenas uma convolução. Os resultados obtidos de testes e predição são exibidos na tabela 10 abaixo. O treinamento com 30 épocas já apresenta ótimos resultados, com uma acurácia de 98,8721% na etapa de teste.

Tabela 10 – Resultados para arquitetura C1P1TC2

N° de épocas no treinamento	10	30	50	100
Acurácia - Teste	46,6165%	98,8721%	98,8721%	99,2481%
Acurácia - Predição	40,8450%	95,0704%	95,7746%	97,8873%

N° de épocas no treinamento	10	30	50	100

Fonte: Autor

6.3 Resultados obtidos com a arquitetura C1P1TC2b

Esta arquitetura C1P1TC2b com um número bem menores de pesos conseguiu obter valores de acurácia tão bom quanto das arquiteturas C2P2TC2 e C1P1TC2.

Tabela 11 – Resultados para arquitetura C1P1TC2b

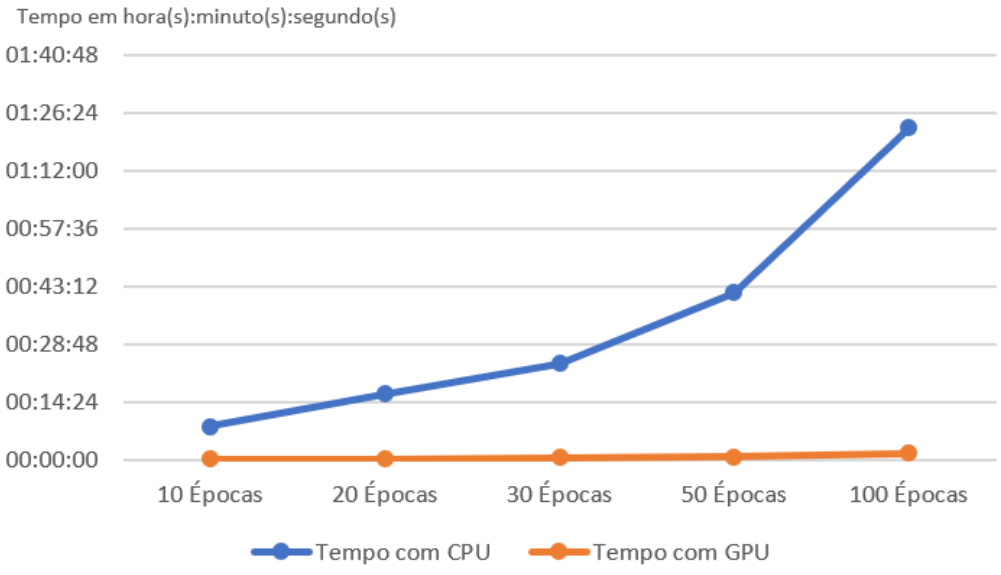
N° de épocas no treinamento	10	30	50	100
Acurácia - Teste	72,5563%	97,7443%	99,2481%	99,2481%
Acurácia - Predição	66,1971%	92,9577%	96,4788%	98,5915%

Fonte: Autor

6.4 Treinamentos realizados com CPU/GPU

Foram feitos alguns testes para apresentar a diferença de velocidade de treinamento em utilizar GPU ao invés de só CPU , conforme os tempos apresentados na Figura 16, com destaque para o número de cem épocas o tempo da etapa de treinamento passar de 1 hora e 22 minutos e 43 segundos para 1 minuto e 53 segundos.

Figura 16 – Gráfico com a distribuição da quantidade de épocas por tempo gasto para CPU e GPU



Fonte: Autor

7 Conclusão

> Tudo tem o seu tempo determinado, e há tempo para todo propósito debaixo dos céus. - Eclesiastes 3:1

Este trabalho abordou o problema de classificação de documentos de identificação, sendo utilizadas três diferentes arquiteturas de redes neurais convolucionais, os resultados obtidos para as três chegaram aproximadamente a 97% de acertos, com destaque para arquitetura C1P1TC2b, que com um número bem menor de pesos conseguiu obter uma acurácia de 98,59% de acerto no momento de validação do modelo. Este trabalho possibilitou passar por todas as etapas de criação de um pipeline, partindo da criação do *dataset*, o modelo e a validação desse modelo. Uma arquitetura com bem menos parâmetros se comparado aos trabalhos relacionados citados na seção 3, com apenas uma camada convolucional foi suficiente para obter resultados robustos para o problema de classificação de imagens de documentos brasileiros.

7.1 Lições aprendidas

- É possível realizar o treinamento em mini-batch, ou seja, carregar as imagens aos poucos para o modelo ir treinamento. Isso resolve o problema de pouca memória disponível na máquina, que não possibilita carregar um número grande de imagens de uma só vez para o modelo. Toda vez que iniciava o treinamento do modelo o Google Colab travava. No momento de treinamento ficam na memória todas as imagens que forem carregadas para o Google Colab, além de todas as representações dos mapas de caracteristicas e pesos fazendo que a memória não suportasse. Carregando o conjunto de imagens aos poucos (mini-batch), foi uma forma encontrada para solucionar esse problema de falta de recurso hardware.
- É importante fazer a matriz de confusão e validar os casos que deram falsos negativos (FN) e falsos positivos (FP), porque ao verificar essas imagens que foram classificadas como FN ou FP constatou que algumas imagens estavam marcadas com o label errado. Era de uma categoria e estava assinalado como fosse de outra no *dataset,* logo é importante ficar atento a isso.
- É fundamental que as imagens utilizadas na etapa do treinamento representem todos os possíveis cenários que as imagens de predição podem serem recebidas. Primeiro o modelo foi treinado sem ter entradas de imagens rotacionadas e foram feitos alguns testes iniciais com imagens rotacionadas para predição e foram obtidos resultados abaixo de 40% de acerto. Logo, foi necessário aplicação de

técnicas de *Data Augmentation* para aumentar a quantidade de imagens para etapa de treinamento do modelo, fazendo que imagens rotacionadas fizessem parte do conjunto de treinamento. Após realizar novos testes, o modelo passou a generalizar para esses cenários de entradas rotacionadas.

- Utilizar GPU para o treinamento do modelo é melhor do que CPU, pois a GPU foi 44 vezes mais rápida do que a CPU na execução do treinamento.

7.2 Trabalhos futuros

- Aumentar a quantidade de categorias de documentos de identificação de pessoa física brasileiros e também de outros países.
- Após classificar o documento utilizar um reconhecimento ótico de caracteres (OCR) para extrair as informações importantes de cada documento.

Referências

1 DECISION REPORT. **Simply usa nuvem em processo de análise de crédito**. 2017. Disponível em: <http://www.decisionreport.com.br/financas/simply-usa-nuvem-em-processo-de-analise-de-credito/#.XFGziflKg2x>. Acesso em: 18/06/2018.

2 BARRETO, J. M. **Introdução às Redes Neurais Artificiais**. Florianópolis, 2002. Disponível em: <http://www.inf.ufsc.br/~j.barreto/tutoriais/Survey.pdf>.

3 ALZAHRANI, A. et al. Solar Irradiance Forecasting Using Deep Neural Networks. In: SCIENCE, P. C. (Ed.). **Complex Adaptive Systems Conference with Theme**: Engineering cyber physical systems. Chicago: ELSEVIER, 2017. v. 114, p. 304 – 313. Disponível em: <https://sciencedirect.com/science/article/pii/S1877050917318392?via%3Dihub>.

4 SHAPIRO, L.; IMBRÓGLIO. **Computer Vision**. [s.n.], 2000. Disponível em: <http://nana.lecturer.pens.ac.id/index_files/referensi/computer_vision/Computer%20Vision.pdf>.

5 GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. MIT Press, 2016. Disponível em: <http://www.deeplearningbook.org>. Acesso em: 01/07/2018.

6 KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Image-Net Classification with Deep Convolutional Neural Networks. In: . [s.n.], 2012. p. 1 – 9. Disponível em: <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>.

7 SZEGEDY, C. et al. Going deep with convolutions. In: **IEEE Computer Society Conference on Computer Vision and Pattern Recognition**. [s.n.], 2015. Disponível em: <https://arxiv.org/pdf/1409.4842.pdf>.

8 RAJ, B. **Data Augmentation | How to use Deep Learning when you have Limited Data—Part 2**. 2018. Disponível em: <https://medium.com/nanonets/how-to-use-deep-learning-when-you-have-limited-data-part-2-data-augmentation-c26971dc8ced> Acesso em: 30/01/2019.

9 LECUN, Y. et al. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278 – 2324, November 1998. Disponível em: <http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf>. Acesso em: 19/11/2018.

10 MIYAZAKI, C. K. **Redes neurais convolucionais para aprendizagem e reconhecimento de objetos 3D**. 2017. 56 p. Monografia (Engenharia Elétrica com Ênfase em Sistemas de Energia e Automação) — Universidade de São Paulo ESCOLA DE ENGENHARIA DE SÃO CARLOS. Disponível em: <http://www.tcc.sc.usp.br/tce/disponiveis/18/180500/tce-22022018-121624/publico/Miyazaki_caio_tcc.pdf>. Acesso em: 25/11/2018.

11 HIJAZI, S.; KUMAR, R.; ROWEN, C. Using Convolutional Neural Networks for Image Recognition. 2015. Disponível em: <https://ip.cadence.com/uploads/901/cnn_wp-pdf>.

12 DATA SCIENCE ACADEMY. **Deep Learning Book**. Online, 2017. Disponível em: <deeplearningbook.com.br>.

13 CHATFIELD, K. et al. Return of the Devil in the Details: Delving Deep into Convolutional Nets. In: CORNELL UNIVERSITY, 2017, -. **BMVC 2014**. -, 2017. v. 4, p. 1 – 11. Disponível em: <https://arxiv.org/pdf/1405.3531.pdf>.

14 SICRE, R.; AWAL, A.; FURON, T. Identity documents classification as an image classification problem. In: PROJECT-TEAMS LINKMEDIA, 488., 2017, Bretagne Atlantique. **Technical Report**. Bretagne Atlantique: AriadNext, 2017. p. 1 – 15. Disponível em: <https://hal.inria.fr/hal-01503541v2>.

15 GOOGLE. **TensorFlow**. 2018. Disponível em: <https://www.tensorflow.org/tutorials/>.