TRABALHO DE CONCLUSÃO DE CURSO

ESTIMAÇÃO CEGA DA QUALIDADE DE VÍDEO UTILIZANDO INFORMAÇÕES DE FLUXO ÓPTICO

Raffael Luna Cardoso

Brasília, Dezembro de 2018

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia

TRABALHO DE CONCLUSÃO DE CURSO

ESTIMAÇÃO CEGA DA QUALIDADE DE VÍDEO UTILIZANDO INFORMAÇÕES DE FLUXO ÓPTICO

Raffael Luna Cardoso

Trabalho de Conclusão de Curso submetido ao Departamento de Engenharia Elétrica como requisito parcial para obtenção do grau de Engenheiro Eletricista

Banca Examinadora

Prof. Mylene C. Q. de Farias, Ph.D., ENE/UNB _____
Orientadora

Vinícius de Oliveira Silva, Ms.C., ENE/UNB _____
Co-orientador

Prof. Alexandre Ricardo Soares Romariz, Ph.D, ENE/UNB _____
Examinador interno

Prof. Alexandre Zaghetto, Ph.D, CIC/UNB _____
Examinador externo

FICHA CATALOGRÁFICA

CARDOSO, RAFFAEL LUNA
ESTIMAÇÃO CEGA DA QUALIDADE DE VÍDEO UTILIZANDO INFORMAÇÕES DE FLUXO ÓPTICO [Distrito Federal] 2018.
xvi, 42 p., 210 x 297 mm (ENE/FT/UnB, Engenheiro, Engenharia Elétrica, 2018).
Trabalho de Conclusão de Curso - Universidade de Brasília, Faculdade de Tecnologia.
Departamento de Engenharia Elétrica

1. Qualidade de Vídeo	2. Aprendizado de Máquina
3. Fluxo Óptico	4. Rede Neural Convolucional
I. ENE/FT/UnB	II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

LUNA CARDOSO, R. (2018). *ESTIMAÇÃO CEGA DA QUALIDADE DE VÍDEO UTILIZANDO INFORMAÇÕES DE FLUXO ÓPTICO*. Trabalho de Conclusão de Curso, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 42 p.

CESSÃO DE DIREITOS

AUTOR: Raffael Luna Cardoso
TÍTULO: ESTIMAÇÃO CEGA DA QUALIDADE DE VÍDEO UTILIZANDO INFORMAÇÕES DE FLUXO ÓPTICO.
GRAU: Engenheiro Eletricista ANO: 2018

Raffael Luna Cardoso
Depto. de Engenharia Elétrica (ENE) - FT
Universidade de Brasília (UnB)
Campus Darcy Ribeiro
CEP 70919-970 - Brasília - DF - Brasil

Dedicatória

Ao meu avô Walter, cujas memórias continuam a inspirar.

Raffael Luna Cardoso

Agradecimentos

Primeiramente, gostaria de agradecer à minha mãe, Iria Luna, e ao meu pai, Elton Cardoso, por todo o carinho, conselho, suporte, amor e dedicação, desde o meu primeiro momento de vida. À minha avó Elisa, que é mãe duas vezes, participando ativamente da minha vida, dando suporte e sempre distribuindo amor e atenção. Ao meu avô Walter (in memorian) eternamente presente em minha vida. Ao meu Vovô Caco, que também sempre esteve ao meu lado nesta jornada, e à minha Denda, Virna, sempre solícita e presente. Agradeço também minha irmã Marianna pelos ótimos momentos juntos, e, finalmente, a todos da minha família que de certa forma contribuíram para a construção da minha vida pessoal e acadêmica.

Gostaria de agradecer ao Vinícius, cuja ajuda fundamental tornou possível a elaboração deste trabalho, sempre estando disposto a ajudar e a aconselhar, e a Professora Mylène Farias, que mesmo estando longe esteve sempre à disposição. Agradeço também a Sana e o Muhammad, do GPDS, que foram muito solícitos quando necessitei de ajuda.

Não posso deixar de mencionar também os meus amigos da 402 sul, que sempre estiveram comigo, nos melhores e piores momentos da minha vida. Menciono também meus amigos de graduação, o Tea Bag's Group, que sempre compartilharam as dificuldades do curso de Engenharia Elétrica, bem como os momentos felizes fora dele. Além disso, gostaria de agradecer a todos os meus amigos pelos bons momentos compartilhados.

Meus sinceros agradecimentos à minha namorada, Maria Luiza, pelo carinho, apoio, compreensão e paciência.

Raffael Luna Cardoso

RESUMO

A quantidade de vídeos digitais presente na vida das pessoas é crescente devido ao surgimento e barateamento de novas tecnologias que permitem e facilitam a disseminação dos vídeos. A partir disso, a quantidade de dados produzidos por esta massiva disseminação é tão grande que os métodos tradicionais de avaliação de qualidade de vídeo podem não ser mais eficientes, quando se tratam de aplicações em tempo real. Dessa forma, a utilização de Inteligência Artificial, mais especificamente Aprendizado de Máquina, pode ser uma estratégia a ser utilizada. É nesse contexto que surge a abordagem proposta neste trabalho: a implementação de um método de avaliação de qualidade de vídeo objetiva, utilizando uma Rede Neural Convolucional, cujas entradas são pilhas de fluxo ópticos, que visa predizer notas de vídeo. Os resultados obtidos neste trabalho mostraram que o modelo proposto tem potencial para realizar o objetivo mencionado, porém é necessário realizar ajustes e adicionar complementos, a ser realizados em trabalhos futuros, para ser comparado ao estado da arte.

Palavras-Chaves: Qualidade de Vídeo, Aprendizado de Máquinas, Fluxo Óptico, Rede Neural Convolucional

ABSTRACT

The amount of digital videos present in people's lives is growing due to the emerging and cheapening of new technologies that allow and make it easier to disseminate. Thereby, the amount of data produced by this massive dissemination it's so big that the traditional Video Quality Assessment methods may not be so efficient for real time purposes. So, Artificial Intelligence applications, specifically Machine Learning, could be a new strategy to solve this problem. In this context arrises the approach that will be used in this work: the implementation of a objective Video Quality Assessment method based on a Convolutional Neural Network, whose inputs are optical flow stacks, aiming to predict video scores. The results obtained in this work shows that the proposed model has potential to perform the suggested task, but it is necessary to make some adjusts and complements, to be accomplished in future works, to be compared to the state of art.

Keywords: Video Quality, Machine Learning, Optical Flow, Convolutional Neural Network

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 INTRODUÇÃO

A sociedade globalizada está cada vez mais imersa em dados digitais, que podem carregar informações relevantes de forma explícita ou não. Estes dados podem estar organizados de forma a produzir estímulos visuais, formando imagens, audíveis, como sons, e, combinando os dois, audiovisuais. Os vídeos digitais, sequências de imagens digitais amostradas no tempo, estão mais presentes nas vidas das pessoas graças aos avanços das tecnologias que tornam dispositivos tanto de captura quanto de reprodução mais baratos economicamente, além do crescente aparecimento de serviços de reprodução de vídeos *on-line* por *streaming*, como *Netflix*, *YouTube*, *Facebook*, *Instagram*, entre outros.

Dada a crescente popularização dos vídeos digitais fazem-se necessárias análises a respeito da qualidade dos mesmos e, se possível, em tempo real, para aplicações na *internet*, que também está cada vez mais popular. As análises de qualidade de vídeo podem ser realizadas de forma subjetiva, isto é, quando é realizado um experimento psico-físico com pessoas reais, as quais devem julgar uma nota para o vídeo em questão. Entretanto, essa forma de avaliação de qualidade de vídeo não é viável economicamente e não possui aplicação em tempo real, sendo necessário pensar em formas objetivas de avaliação. As formas objetivas de avaliação de qualidade de vídeo podem ser discriminadas em três principais categorias: referências cega, quando não há informações do vídeo original; referência reduzida, quando são passados apenas alguns parâmetros do vídeo original; e referência completa, quando o vídeo original está disponível.

Como a quantidade de dados é muito grande, as técnicas convencionais de avaliação objetiva de qualidade de vídeo podem não ser eficazes, fazendo com que seja necessário buscar novas abordagens para esta tarefa. Uma possibilidade é a utilização de Redes Profundas para fazer a captura de características dos vídeos e baseado nelas estimar uma nota, buscando se aproximar de julgamentos humanos.

Uma possível forma pela qual as informações de movimento do vídeo podem ser passadas para a rede é baseada em fluxos ópticos, que são imagens do campo de vetores resultantes do deslocamento de *pixels* entre dois quadros consecutivos de um vídeo. Existem algumas formas de calcular o fluxo óptico, uma delas é através do algorítimo proposto por Farnebäck [1], conhecido como Fluxo Óptico Denso.

1.1 OBJETIVOS

Este trabalho foi inspirado no artigo [2], em que foi construída uma Rede Convolucional para qualidade de imagens. Entretanto, aqui a proposta é a implementação de um método objetivo de referência cega para avaliação de qualidade de vídeo. Para isso, será construída uma Rede

de Aprendizado Profundo, neste caso uma Rede Neural Convolucional para a estimação de nota. Essa rede funcionará extraindo atributos relevantes de uma pilha de fluxos ópticos, que são imagens concatenadas de fluxos ópticos densos de um vídeo, que passará por uma rede que fará uma regressão desses atributos, estimando uma nota.

É de se esperar deste projeto uma estimação de notas de forma que seja possível estudar como uma Rede Neural Convolucional se comporta quando sua tarefa é predizer notas de qualidade de vídeo, utilizando como entradas pilhas de fluxos ópticos.

1.2 ORGANIZAÇÃO DO TRABALHO

Este projeto está dividido em duas grandes partes: uma de introdução e conceituação geral, necessárias para compreender este trabalho, e a parte experimental, onde será explicada a metodologia, os resultados obtidos, suas análises e conclusões. O capítulo dois versa sobre uma definição geral a respeito de vídeos digitais, fazendo uma breve apresentação de conceitos básicos, tipos de compressões de vídeo e seus consequentes artefatos, além de uma referenciação com respeito a conceitos de qualidade de vídeo.

Após apresentado os conceitos básicos sobre vídeos digitais, parte-se para uma análise de mais baixo nível de um vídeo: os quadros que o compõe. O capítulo 3 trata de uma estratégia para representar o movimento dos *pixels* entre dois quadros consecutivos, chamado de Fluxo Óptico, e os métodos propostos por Farnebäck [1] e Horn-Schunck [3]. Em seguida, o capítulo 4 apresenta um dos pilares de Inteligência Artificial, o aprendizado de máquinas. Nele, é tratado sobre os conceitos gerais de Aprendizado de Máquina, e depois parte-se para Redes Neurais Artificiais, desde o *perceptron* de MCCulloch e Pitts [4] passando pelo *perceptron* multicamadas até, finalmente, as Redes Neurais Convolucionais, modelo a ser utilizado neste trabalho.

A segunda grande parte do trabalho incia-se no capítulo 5, onde será abordado a metodologia do trabalho. Lá estará registrado como serão construídas as entradas da rede, a arquitetura geral do modelo proposto, o método de predições de notas e como elas serão analisadas. O capítulo 6 traz os experimentos realizados para a obtenção dos resultados que serão também analisados neste capítulo. O trabalho termina no capítulo 7, onde será feito conclusões a respeito dos resultados obtidos seguindo a metodologia proposta.

2 VÍDEOS DIGITAIS

Este capítulo trará noções básicas a respeito de vídeos digitais que serão necessárias para a compreensão deste trabalho. Neste sentido, serão apresentados alguns conceitos iniciais, bem como noções de compressões de vídeos e informações relativas à qualidade de vídeo.

2.1 CONCEITOS INICIAIS

Segundo Murat [5], um vídeo é definido como um padrão de intensidade espacial que varia no tempo e, além disso, um vídeo pode ser encarado como uma sequência temporal de imagens estáticas (quadros). Esta noção de quadros será importante no capítulo seguinte, no qual será apresentado o conceito de fluxo óptico, que é base do presente trabalho.

Vídeos Digitais

Nos dias de hoje, é comum estar cercado de informações digitalizadas. Com os vídeos não são diferentes. A digitalização de vídeos é só mais um meio de permitir a transmissão desses tipos de sinais com mais eficiência e qualidade. Dessa forma, as sequências de imagens não são mais representadas de maneira contínua e passam a ser amostradas em intervalos regulares [6].

Os vídeos capturados de maneira analógica passam por uma amostragem através da qual suas linhas serão substituídas por *pixels* [6]. Após todo o processo de digitalização, o vídeo digital será uma sequência discreta no tempo de matrizes de *pixels*.

Dimensão Espacial

A dimensão espacial de um vídeo diz respeito a relação entre a quantidade de *pixels* horizontais e verticais, além de trazer a informação sobre a definição do vídeo. Os vídeos podem ser classificados em definição padrão (720x576 ou 720x488), alta (1920x1080) ou ultra alta (3840x2160) [5], mas podem, também, não estar incluídos nesses padrões de classificações e podem possuir dimensões arbitrárias.

Resolução Temporal

Uma imagem discretizada no tempo consiste em três canais de cores representados na forma de matrizes com valores inteiros e correspondem aos quadros de um vídeo. A quantidade desses quadros amostrados em um segundo caracterizam a taxa de quadros em *frames per second* (FPS). Os valores típicos de taxa de quadros são 60/50 FPS [5], mas podem ser valores maiores ou

menores, como 24, 25, 30 FPS.

2.2 ARTEFATOS EM VÍDEOS

Os efeitos visuais, perceptíveis e indesejáveis, causados pelas compressões, transmissões, capturas, entre outros, são conhecidos como artefatos [7]. Os artefatos de vídeos mais comuns são:

- *Perda de Pacotes:* Ocorre quando há perda de partes do vídeo durante a transmissão, resultando em partes faltantes em vários quadros, como na imagem 2.5(a).
- *Borrado:* Ocorre quando é suprimida os coeficiente de alta frequência durante o processo de quantização, na compressão, ocasionando perda de detalhamento e menos percepção das bordas. É observado o borrado no quadro mostrado na figura 2.2(a) .
- *Blocagem:* É caracterizado como padrões de blocos aparente na imagem. É resultado de quantizações grosseiras durante a compressão, gerando descontinuidades de blocos próximos. É observada na figura 2.3(a).
- *Anelamento:* É também conhecido como serrilhado e é resultado da quantização, que leva a irregularidades na reconstrução. É percebido como "serras"nos contornos das imagens.
- *Jitter:* É o efeito causado pelo corte de quadros para reduzir a quantidade de informação do vídeo, ocasionando movimentos não suaves, como se fossem fotografias.

Ainda segundo [7], a performance dos vídeos são melhores quando os artefatos que estão presentes são conhecidos, porque podem ser utilizados algorítimos que os reduzam.

2.3 COMPRESSÃO DE VÍDEO

No cenário atual, com a constante evolução das tecnologias de captura de vídeos, a quantidade de informação a ser processada é muito grande e tende a aumentar cada vez mais. Sabendo disso, surge a necessidade de transmitir estes dados de maneira rápida e sem perder informações significativas, visto que a transmissões de dados cru, isto é, sem compressão, exige altas taxas de transmissões e são inviáveis comercialmente.

Nesse contexto, foram desenvolvidos os padrões de compressão de vídeos, que viabilizam a tecnologia de vídeos digitais e são padronizados para garantir a compatibilidade de *hardware* de diferentes fornecedores [5].

Compressão é o procedimento utilizado para reduzir a quantidade de dados necessários para representar uma dada quantidade de informação [8]. Dentre os mais diversos tipos de compressão,

as que aqui serão mencionadas são aquelas que causam perdas de dados e introduzem algum tipo de distorção no vídeo.

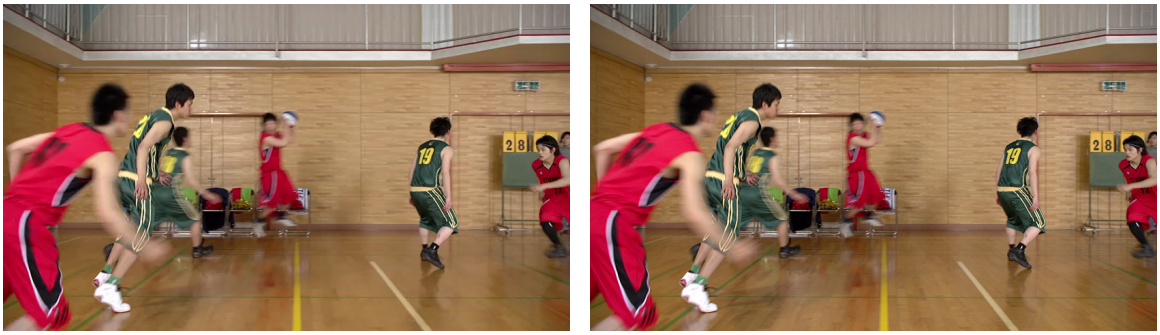
Segundo [7], a compressão é feita removendo redundâncias dos vídeos, que, essencialmente, são de quatro tipos:

- **Redundância Perceptiva:** São aquelas informações que o sistema visual humano não é capaz de captar com clareza;
- **Redundância Temporal:** Como os *pixels* tem grande similaridade em quadros consecutivos, o movimento de blocos de *pixels* tende a não alterar seus valores, nem sua correlação;
- **Redundância Espacial:** Indica a forte correlação entre um *pixel* e sua vizinhança;
- **Redundância Estatística:** São as redundância que estão relacionadas à estatística dos dados de informação, como os bytes que formam um vídeo digital.

Para cada tipo de remoção de redundâncias, existem variados métodos de compressão de vídeos, que serão brevemente descritos abaixo. Dessa forma, cada tipo de compressão introduz uma determinada distorção ao vídeo, juntamente com ruídos e perda de dados ocasionado pela transmissão dos mesmos, causando diferentes efeitos visuais. Alguns exemplos de compressões que causam perdas de dados são a H.264, HEVC, MJPEG, compressões do tipo *wavelet*, entre outros.

Compressão segundo o padrão H.264

A compressão H.264/AVC é uma técnica de compressão de vídeos baseada em blocos com compensação de movimento e consiste na divisão do quadro em macro-blocos que pode variar de 4×4 até 16×16, nos quais serão aplicadas a transformada discreta do cosseno (DCT), e, em seguida, é utilizado um estimador de movimento para realizar predições entre os quadros [9]. A distorção introduzida pela compressão H.264 pode ser observada na figura 2.1 (a).



(a) (b)

Figura 2.1: **Compressão H.264:** A sub-figura (a) mostra as distorções causadas pela compressão H.264, enquanto a sub-figura (b) é a imagem de referência.

Compressão segundo o padrão H.265 (HEVC)

A compressão HEVC é o padrão que sucede o padrão H.264. Neste padrão, os macro-blocos podem variar até 64×64 e se dividem em blocos menores de codificação, predição e transformação, os quais são mais eficientes e inteligentes. Este padrão de compressão tornou possível maiores velocidades de transmissão de vídeos de alta resolução e qualidade, se destacando em transmissões *on-line*, além de poder ser utilizado em arquiteturas de processamento paralelo [10] de maneira mais eficiente. Mostra-se a distorção causada pela compressão HEVC na figura 2.2.

(a) (b)

Figura 2.2: **Compressão H.265 (HEVC):** A sub-figura (a) mostra as distorções causadas pela compressão HEVC, enquanto a sub-figura (b) é a imagem de referência.

Compressão segundo o padrão *Motion* JPEG (MJPEG)

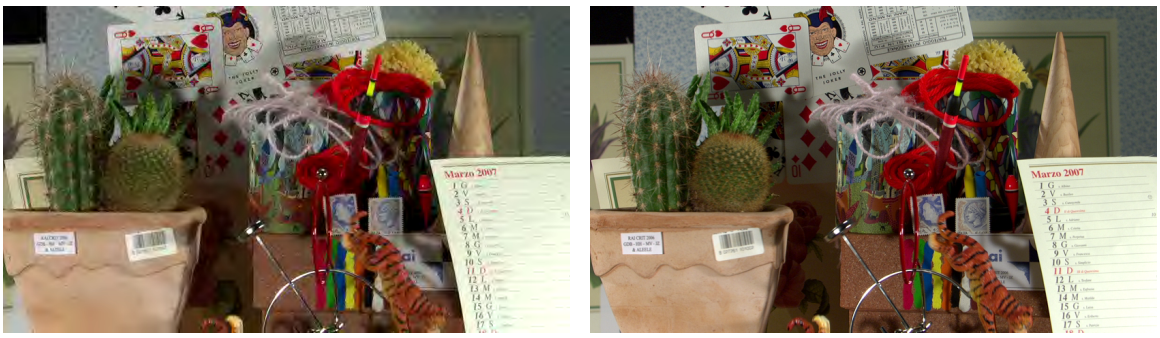
Este tipo de compressão consiste em simplesmente aplicar a compressão de imagens JPEG a cada quadro do vídeo. O algorítimo JPEG consiste em dividir o quadro em blocos de 8x8 *pixels* e neles aplicar a transformada discreta de cosseno (DCT), resultando em coeficientes para cada bloco. O resultado da compressão MJPEG é vista na figura 2.3.

(a) (b)

Figura 2.3: **Compressão *Motion* JPEG (MJPEG):** A sub-figura (a) mostra as distorções causadas pela compressão MJPEG, enquanto a sub-figura (b) é a imagem de referência.

Compressão do tipo *wavelet* usando o *codec Snow*

A transformada *wavelet* é uma transformação que utiliza a série *wavelet*. Quando utilizada para a compressão de imagens, a compressão *wavelet* possui um aspecto superior, tendo em vista sua alta correspondência com o sistema de visão humana [11]. A figura 2.4 mostra uma imagem comprimida utilizando o *codec Snow*, baseado na compressão *wavelet*.



(a) (b)

Figura 2.4: **Compressão *wavelet* usando o *codec Snow*:** A sub-figura (a) mostra as distorções causadas pela compressão do tipo *wavelet*, enquanto a sub-figura (b) é a imagem de referência.

Compressão H.264 com a perdas de pacote devido a transmissão *wireless*

Um sinal digital quando precisa ser enviado é dividido em pacotes de informação menores que são transmitidos a algum tipo de canal de comunicação. Em todo canal de comunicação real ocorrem perdas. Essas perdas foram simuladas para um canal de transmissão *wireless* utilizando a compressão H.264. A Figura 2.5 mostra as distorções causadas tanto pela compressão H.264 quanto por pacotes de informações perdidos durante a comunicação.

(a) (b)

Figura 2.5: **Compressão H.264 com a perdas de pacote devido a transmissão *wireless*:** A sub-figura (a) mostra as distorções causadas pela compressão H.264 junta com a perda de informação oriunda da transmissão *wireless*, enquanto a sub-figura (b) é a imagem de referência.

Ruído Branco Aditivo (AWGN)

O ruído branco aditivo é o ruído oriundo do canal de comunicação ou de processamento. Este ruído é uma componente aleatória adicionada ao sinal que ocupa todo o espectro de frequências. A Figura 2.6(a) mostra o ruído na imagem.

(a) (b)

Figura 2.6: **Ruído Branco Aditivo:** A sub-figura (a) mostra o ruído branco aditivo incorporado a uma imagem de referência (b).

2.4 QUALIDADE DE VÍDEO

Avaliar a qualidade de vídeo significa estabelecer métricas para julgar determinadas características de um vídeo sob dada circunstância. Essencialmente, existem duas formas de avaliar um vídeo: de maneira **subjetiva**, que está relacionada à percepção humana, ou de maneira **objetiva**, através de análises matemáticas dos *pixels* do vídeo.

Análise Subjetiva

Quando uma análise é dita subjetiva, refere-se à experimentos psico-físicos aplicados a sujeitos humanos que são submetidos à experiencia de assistir uma série de vídeos e julgar uma nota. Estes experimentos subjetivos representam a forma mais acurada de se medir a qualidade de um vídeo [7], mas são caros e demandam muito tempo.

Existe uma série de métodos para dimensionar um experimento subjetivo para avaliação de qualidade de vídeo propostos pela União Internacional de Telecomunicações (ITU), que estabelecem uma espécie de "padronização"dos experimentos subjetivos, bem como a condição nas quais o espectadores devem estar submetidos, métodos de análises de dados, tipos de equipamentos, formas pontuação e procedimentos de avaliação [12, 13].

Existem, ainda, duas formas de fazer experimentos para medições de qualidade de vídeo subjetiva: estímulo simples e estímulo duplo. Na primeira forma apenas o vídeo de teste é apresentado para análise, enquanto na segunda forma é apresentado o vídeo teste juntamente com seu

vídeo de referência.

Análise Objetiva

Análises objetivas pressupõem avaliações automatizadas, baseados em algorítimos computacionais, chamados de métricas, que tentam estimar a qualidade de vídeo baseada em informações retiradas de seus *pixels*. Existem três categorias de métricas de avaliação de qualidade de vídeo objetiva: Referência Cega, Referência Reduzida e Referência Completa.

- **Referência Cega:** É fornecido ao algorítimo apenas o vídeo de teste;
- **Referência Reduzida:** São fornecidos um vídeo de teste e descrições e/ou parâmetros do vídeo original;
- **Referência Completa:** São fornecidos o vídeo de teste juntamente com o vídeo de referência.

As principais métricas de referência completa para estimação de qualidade de vídeo objetiva é PSNR (*Peak Signal-to-Noise Ratio*) e o erro médio quadrático (MSE: *Mean Squared Error*), calculados fazendo:

$$MSE = \frac{1}{N}\sum_{p=1}^{N}\left(O_p - D_p\right)^2, \tag{2.1}$$

e

$$PSNR = 10\log_{10}\frac{255^2}{MSE}. \tag{2.2}$$

Finalmente, o objetivo deste trabalho é propor um meio de análise objetiva de referência cega de qualidade de vídeos, isto é, sem passar o vídeo de referência, alimentando uma Rede Neural Convolucional com Pilhas de Fluxos Ópticos, que serão vistas nos capítulos seguintes.

3 FLUXO ÓPTICO

Este capítulo abordará uma noção básica a respeito do fluxo óptico. Fluxo óptico é o campo de vetores de velocidade resultantes de deslocamento de *pixels* semelhantes entre duas imagens consecutivas. Como visto no capítulo anterior, um vídeo é uma sequência de imagens estáticas, também chamadas de quadros. A partir desses quadros, pode-se, então, calcular o fluxo óptico. Existem alguns métodos de se calcular o fluxo óptico entre duas imagens estáticas, dentre elas será detalhado o método proposto por Horn e Schunck [3] e o método de fluxo óptico denso adotado por Farnebäck [1], técnica que será utilizada neste trabalho.

Alguns métodos de cálculo, como os propostos por Horn e Schunck, são baseados na ideia do fluxo óptico diferencial, que é calculado em pequenas janelas Ω de quadros consecutivos de um vídeo. Dentro da janela Ω, assume-se que existem regiões em que a intensidade de brilho não varia em dois quadros consecutivos. Sendo $I(x, y, t)$ a intensidade do brilho de um quadro nos pontos (x, y) em um instante t e $I(x+\Delta x, y+\Delta y, t+\Delta t)$ a intensidade do brilho em um quadro consecutivo, tem-se

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t). \tag{3.1}$$

Fazendo a expansão por série de Taylor da equação 3.1 e fazendo $\Delta x = udt$, $\Delta y = vdt$ e $\Delta t = dt$, onde u e v são as componentes horizontais e verticais, respectivamente, do fluxo ópticos, obtém-se

$$I(x, y, t) = I(x+udt, y+vdt, t+dt) = I(x, y, t)+\frac{\partial I}{\partial x}udt+\frac{\partial I}{\partial y}vdt+\frac{\partial I}{\partial t}dt+OS(x, y, t). \tag{3.2}$$

Subtraindo $I(x, y, t)$ da Equação 3.2, desconsiderando os termos de ordem superior $OS(x, y, t)$ da expansão por série de Taylor e fazendo $E_x = \frac{\partial I}{\partial x}$, $E_y = \frac{\partial I}{\partial y}$ e $E_t = \frac{\partial I}{\partial t}$, chega-se a

$$E_x u + E_y v + E_t = 0, \tag{3.3}$$

que é a conhecido como restrição do fluxo óptico, introduzido por [3]. O algorítimo de Horn-Schunck busca soluções para a Equação 3.3, utilizando alguns métodos numéricos, como será visto na seção seguinte.

3.1 CÁLCULO ATRAVÉS DO ALGORÍTIMO DE HORN E SCHUNCK

Para o algorítimo elaborado por Horn e Schunck [3], devem ser consideradas algumas restrições a fim de que se obtenha um sistema de equações com solução única. Assim, considera-se que o brilho em um ponto (x, y) da imagem é proporcional a refletância, relação entre o fluxo de luz incidente em uma superfície e o fluxo refletido, da superfície do ponto correspondente. Assume-se, a fim de garantir que o brilho é diferenciável, que a refletância varia de forma suave e não possui descontinuidades. Dessa forma, é necessário que, na imagem, um objeto não oclusa outro, garantido a continuidade do fluxo.

Dada as restrições acima mencionadas, o problema passa a ser de minimização, cujo o objetivo é minimizar a soma dos erros da equação da taxa de variação do brilho da imagem $\mathscr{E}_b$ (ver Equação 3.3), visto que não é possível garantir que ela é igual a zero, devido a erros na quantização, e a variação espacial do fluxo óptico $\mathscr{E}_c$, mostradas respectivamente nas equações abaixo:

$$\mathscr{E}_b^2 = E_x u + E_y v + E_t \tag{3.4}$$

e

$$\mathscr{E}_c^2 = \left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2, \tag{3.5}$$

no qual u e v são as componentes de fluxo óptico nas direções horizontais e verticais, respectivamente. Então o erro total é dado por

$$\mathscr{E}_T^2 = \iint (\alpha^2 \mathscr{E}_c^2 + \mathscr{E}_b^2) dx dy, \tag{3.6}$$

em que α é o fator de peso, cujo objetivo é prevenir ajustes aleatórios do fluxo óptico estimado ocasionados por ruídos. A minimização da Equação 3.6 é feita solucionando as seguintes equações:

$$E_x^2 u + E_x E_y v = \alpha^2 \nabla^2 u - E_x E_t, \tag{3.7}$$

$$E_x E_y u + E_y^2 v = \alpha^2 \nabla^2 v - E_y E_t. \tag{3.8}$$

Usando a aproximação laplaciana mencionada no trabalho [3], obtém-se

$$(\alpha^2 + E_x^2) u + E_x E_y v = \alpha^2 \bar{u} - E_x E_t, \tag{3.9}$$

$$E_x E_y u + (\alpha^2 + E_y^2)v = \alpha^2 \bar{v} - E_y E_t, \tag{3.10}$$

em que $\bar{u}$ e $\bar{v}$ são médias locais para as velocidade u e v em sua vizinhança. As Equações 3.9 e 3.10 podem sem reescritas como

$$(\alpha^2 + E_x^2 + E_y^2)(u - \bar{u}) = -E_x(E_x\bar{u} + E_y\bar{v} + E_t), \tag{3.11}$$

$$(\alpha^2 + E_x^2 + E_y^2)(v - \bar{v}) = -E_y(E_x\bar{u} + E_y\bar{v} + E_t), \tag{3.12}$$

e, de forma iterativa, sendo n a n-ésima iteração,

$$u^{n+1} = \bar{u}^n - \frac{E_x(E_x\bar{u}^n + E_y\bar{v}^n + E_t)}{(\alpha^2 + E_x^2 + E_y^2)}, \tag{3.13}$$

$$v^{n+1} = \bar{v}^n - \frac{E_y(E_x\bar{u}^n + E_y\bar{v}^n + E_t)}{(\alpha^2 + E_x^2 + E_y^2)}. \tag{3.14}$$

3.2 CÁLCULO DO FLUXO ÓPTICO DENSO

Para a execução deste trabalho será utilizado o algorítimo de cálculo de fluxo óptico desenvolvido por Farnebäck [1]. A ideia central deste algorítimo é a expansão polinomial, cujo objetivo é aproximar a vizinhança de cada *pixel* através de um polinômio quadrático da forma

$$f(\mathbf{x}) \sim \mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{b}^T\mathbf{x} + c, \tag{3.15}$$

em que **A** é uma matriz simétrica, **b** um vetor e c um escalar. Estes coeficientes são obtidos através do método de mínimos quadrados ponderados, que visam encaixar na vizinhança do *pixel* em questão.

Para o cálculo do deslocamento **d** de *pixels* entre dois quadros consecutivos de um vídeo, faz-se necessário estimar a vizinhança destes *pixels*, aplicando a equação 3.15 para cada quadro. Segue-se, então

$$f_1(\mathbf{x}) = \mathbf{x}^T\mathbf{A}_1\mathbf{x} + \mathbf{b}_1^T\mathbf{x} + c_1, \tag{3.16}$$

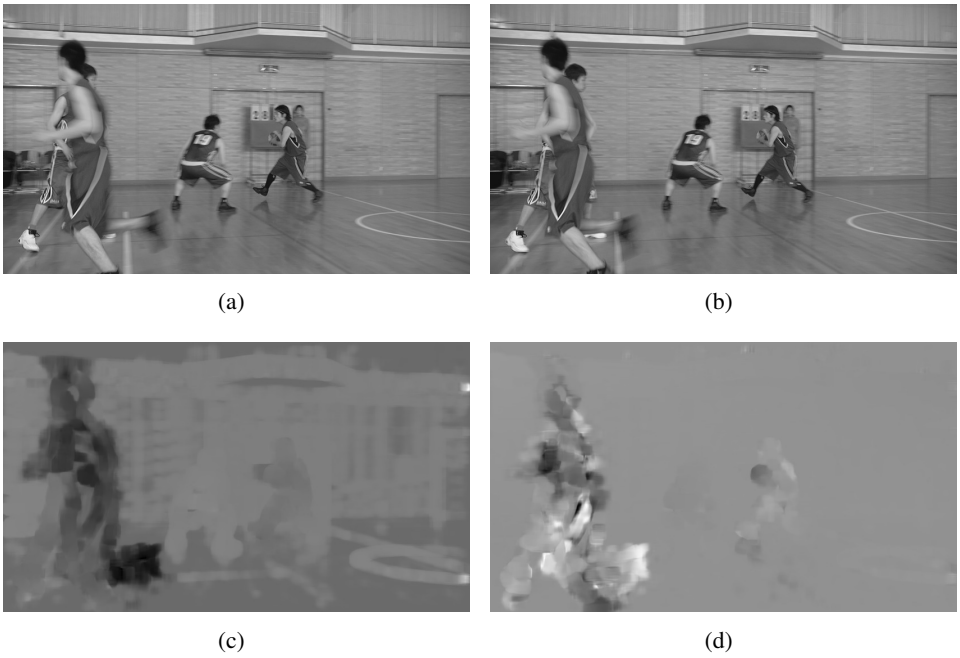
para o primeiro quadro e

$$
\begin{aligned}
f_2(\mathbf{x}) = f_1(\mathbf{x}-\mathbf{d}) &= (\mathbf{x}-\mathbf{d})^T\mathbf{A}_1\mathbf{x} + \mathbf{b}_1^T(\mathbf{x}-\mathbf{d}) + c_1 \\
&= \mathbf{x}^T\mathbf{A}_1\mathbf{x} + (\mathbf{b}_1 - 2\mathbf{A}_1\mathbf{d})^T\mathbf{x} + \mathbf{d}^T\mathbf{A}_1\mathbf{d} - \mathbf{b}_1^T\mathbf{d} + c_1 \\
&= \mathbf{x}^T\mathbf{A}_2\mathbf{x} + \mathbf{b}_2^T\mathbf{x} + c_2,
\end{aligned} \tag{3.17}
$$

para o segundo quadro. Então, é possível determinar o deslocamento **d**, fazendo

$$
\mathbf{b}_2 = \mathbf{b}_1 - 2\mathbf{A}_1\mathbf{d} \Rightarrow \mathbf{d} = -\frac{1}{2}\mathbf{A}_1^{-1}(\mathbf{b}_2 - \mathbf{b}_1). \tag{3.18}
$$

A Figura 3.1 mostra a aplicação do algorítimo descrito acima em dois quadros consecutivos (a) e (b) resultando na componentes horizontais (c) e verticais (d) do fluxo óptico calculado.



(a) (b)

(c) (d)

Figura 3.1: **Fluxo Óptico:** As sub-imagens (c) e (d) mostram as componentes horizontais e verticais, respectivamente, do fluxo óptico calculado a partir de dois quadros consecutivos (a) e (b) de um mesmo vídeo.

4 APRENDIZADO DE MÁQUINA

Assim como a robótica, a visão computacional e a linguagem natural, o aprendizado de máquina é um dos pilares fundamentais da inteligência artificial. Alpaydin [14] conceitua aprendizado de máquinas como o ato de programar computadores para otimizar a performance de um critério, baseado em um conjunto de dados ou experiências passadas.

Outra forma de pensar aprendizado de máquinas é segundo Tom Mitchell [15], que define como o ato de programar um computador para aprender de uma experiência **E** a respeito de alguma tarefa **T** e performance de medida **P**, se essa performance com relação a tarefa **T**, medida por **P**, se otimiza com a experiência **E**.

Nesse sentido, desenvolveram-se e continua-se desenvolvendo algorítimos que visam, a partir de um conjuntos de dados, a aprender sobre uma determinada tarefa. Conforme [16], algumas aplicações de aprendizado de máquinas são: sistema de *ranking* nos resultados de sistemas de buscas *on-line*, sistemas de recomendações, tradução automática de documentos, classificação, reconhecimento de fala, entre outras.

4.1 CONCEITOS BÁSICOS

Como já mencionado, existem vários algorítimos de aprendizado de máquinas. Eles podem ser separados conforme sua tarefa, classificação ou regressão, e conforme o processo de aprendizado: supervisionado, não supervisionado ou por reforço.

Um aprendizado é dito **supervisionado** quando um algorítimo objetiva mapear suas entradas a uma saída, cujos valores corretos, chamados rótulos, são fornecidos por um supervisor [14]. Exemplos de aprendizagem supervisionada são as tarefas de classificação e regressão. O processo de aprendizagem **não supervisionada** é quando os rótulos não são fornecidos, isto é, o algorítimo recebe apenas um conjunto de entradas, e este então busca regularidades nas entradas [14]. Exemplo de aprendizagem não supervisionada são os algorítimos de *clustering*, que podem ser aplicados em compressão de imagens e segmentação de dados. Existe, ainda, uma outra forma de aprendizado que é a por **reforço**. Nesta forma de aprendizagem, o algorítimo dá como resposta um conjunto de ações que deve fazer sentido ao contexto no qual é inserido e deve aprender com as consequências destas ações. Como, por exemplo, as decisões tomadas por um robô para interagir com o ambiente a sua volta.

Neste trabalho, a forma de aprendizagem do algorítimo a ser desenvolvido é supervisionada. Nesta forma de aprendizagem, as duas principais aplicações são tarefas de classificação e regressão, que é a tarefa escolhida para este projeto. A tarefa de **classificação** consiste em mapear as entradas em saídas com rótulos discretos, isto é, categorizar as amostras de entrada em classes

discretas. Já a tarefa de **regressão** se dá por meio de algorítimos que retornam um valor contínuo, como quando se tenta prever a nota de qualidade de um vídeo, baseado no seus atributos.

Dentre os mais populares algorítimos de aprendizado de máquinas, estão as árvores de decisão, máquinas de vetor suporte, estimador de k vizinhos mais próximos, redes neurais artificiais (RNA), entre outros. A árvore de decisão é um método baseado na ideia de dividir e conquistar [14], no qual as amostras de entradas se convertem em informações conforme um conjunto de regras. Já o algorítimo de estimador de k vizinhos mais próximos, segundo [15], consiste em assumir que todas as instâncias correspondem a pontos em um plano e seus vizinhos mais próximos, que são determinados pela distância euclidiana padrão, dizem informações a respeito da amostra. As máquinas de vetor suporte são estimadores binários, cujo a superfície de decisão é um hiperplano de tal forma que a margem de separação entre exemplos positivos e negativos são maximizadas, segundo Haykin [17]. As redes neurais artificiais ganharão uma abordagem mais detalhada na seção seguinte, visto que elas são os principais elementos que compõem o algorítimo utilizado neste trabalho.

Por mais sofisticado que um algorítimo seja, deve-se tomar cuidado com os dados de entrada, fazendo-se necessário analisar a necessidade de aumentar a diversidade do conjunto de treinamento para evitar fenômenos indesejados como o *overfitting*, que é o processo que ocorre quando o modelo é treinado acima do necessário, fazendo com que sejam aprendidos características que só são encontradas nas amostras de treinamento, muitas vezes ruídos. Uma forma de evitar o *overfitting* é reamostrar o modelo, através de validação cruzada, por exemplo. Outro problema comum nas técnicas de aprendizado de máquinas é o *underfitting*, que ocorre quando o algorítimo não é capaz de modelar os dados de entrada, nem ajustar novos dados, fazendo com que seja necessário aumentar a complexidade do modelo, por exemplo.

4.2 REDES NEURAIS ARTIFICIAIS

As redes neurais artificiais (RNA) merecem uma explicação mais detalhada, visto que elas são base do algoritmo de treinamento utilizado neste trabalho, as redes neurais convolucionais, que serão detalhadas na seção seguinte. As RNAs são um artifício computacional baseado no cérebro humano, compostas por neurônios, que são as unidades de processamento dos dados.

A Figura 4.1 mostra um modelo simplificado do neurônio humano. Essa estrutura é composta por dendritos, que fazem a recepção de sinais de outros neurônios; axônios, as linhas de transmissão que se conectam a outras células nervosas [17]; e o corpo que faz o processamento da informação.

Essa estrutura é a base do modelo matemático do neurônio, primeiramente proposta por McCulloch e Pitts [4] em 1943. A subseção seguinte abordará com mais detalhes sobre o modelo matemático do neurônio.

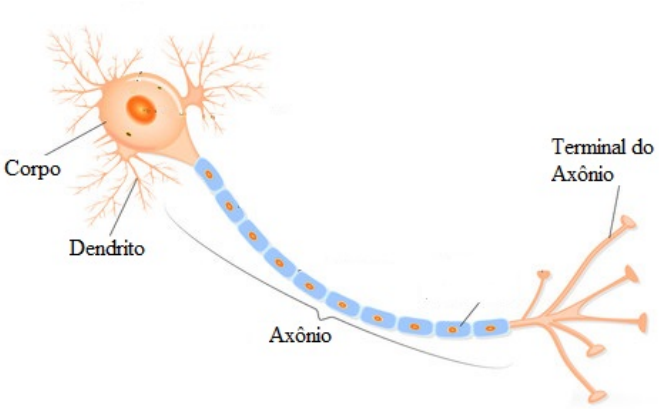


Figura 4.1: Estrutura simplificada de um neurônio humano.

4.2.1 Modelo Matemático do Neurônio

Uma das primeiras utilizações do modelo matemático do neurônio proposto por McCulloch e Pitts [4] é também conhecido como *perceptron* e foi elaborado por F. Rosenblatt [18], como é mostrado na Figura 4.2.

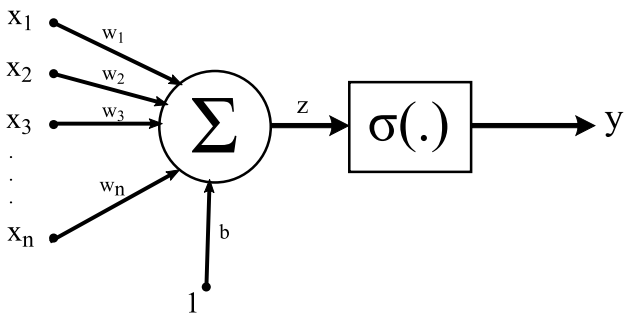


Figura 4.2: Modelo do *perceptron*.

O neurônio recebe um conjunto de dados de entrada $x_1, x_2, \ldots, x_n$ e cada um dos elementos deste conjunto é multiplicado pelo seu respectivo peso sináptico $w_1, w_2, \ldots, w_n$ e, em seguida, somados, resultando em z, o campo local induzido. O campo local induzido z é entrada de uma função $\sigma(\cdot)$, que será detalhada em breve. O parâmetro b é o *bias* e serve como um "limitador"da saída do *perceptron*.

$$y = \sigma\left(\sum_{i=1}^{n} x_i w_i + b\right). \tag{4.1}$$

Para fins de treinamento, a saída y, Equação 4.1, do *perceptron* deve ser comparada ao valor de referência através de uma função de custo. A função de custo é computada fazendo, para todas as m amostras de treinamento, a diferença entre a saída do *perceptron*, calculado através ds n atributos de uma amostra, e a saída esperada ao quadrado. Neste trabalho, será utilizado o método dos mínimos quadrados como função de custo, que é da forma

$$\mathcal{E} = \frac{1}{2}\sum_{j=1}^{m}\left(\sigma\left(\sum_{i=1}^{n} x_i w_i + b\right) - y_{ref,m}\right)^2 . \tag{4.2}$$

A partir da função de custo, mede-se o quanto que os pesos devem variar para, na próxima iteração, mais se aproximar do valor esperado y_{ref}, isto é, minimizar a função de custo.

4.2.2 Funções de Ativação

A função $\sigma(\cdot)$, vista na Equação 4.1, recebe o nome de função de ativação, porque é através dela que um neurônio será "ativado"ou não, isto é, a forma como a saída do neurônio contribuirá na aprendizagem de uma tarefa.

Para o *perceptron*, a sua função de ativação é linear $\sigma(z) = z$ e a função de ativação mais básica é a função degrau $step(z)$, na qual $step(z)$ é zero se $z \leq 0$ e $step(z)$ é 1, caso contrário. Entretanto, faz-se necessário que a função de ativação seja diferenciável para os algoritmos que serão detalhados nas etapas seguintes. surgindo outras opções de função de ativação:

- **Logística:**

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

- **Tangente Hiperbólico:**

$$tanh(z) = 2logistic(2z) - 1$$

- ***Rectified Linear Unit** (ReLU):*

$$ReLU(z) = \begin{cases} 0 & \text{se} \quad z \leq 0 \\ z & \text{se} \quad z > 0 \end{cases}$$

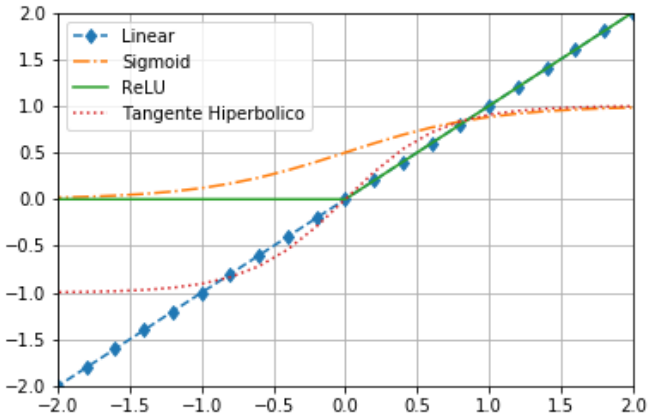


Figura 4.3: Funções de Ativação.

4.2.3 O *perceptron* multicamadas

O *perceptron*, anteriormente descrito, é capaz de gerar discriminantes lineares. Entretanto, quando são conectadas uma ou mais camadas de *perceptrons*, de forma não recorrente, chamadas camadas ocultas, pode-se gerar discriminantes não lineares, para o caso de classificação, e aproximar funções não lineares, para o caso da regressão [14].

As redes de *perceptron* multicamadas (MLP), ou redes neurais artificiais, são modelos de propagação direta e não possuem realimentação da saída em neurônios anteriores. Dessa forma, antes de cada camada de neurônios existem entradas, que podem ser saídas das camadas de neurônios anteriores ou o vetor de entrada, e seus respectivos pesos. A Figura 4.4 mostra uma RNA que recebe um vetor de entrada n características relativas a uma amostra, L camadas ocultas, em que cada uma possui $m^{(L)}$ neurônios, e uma camada de saída, com k neurônios.

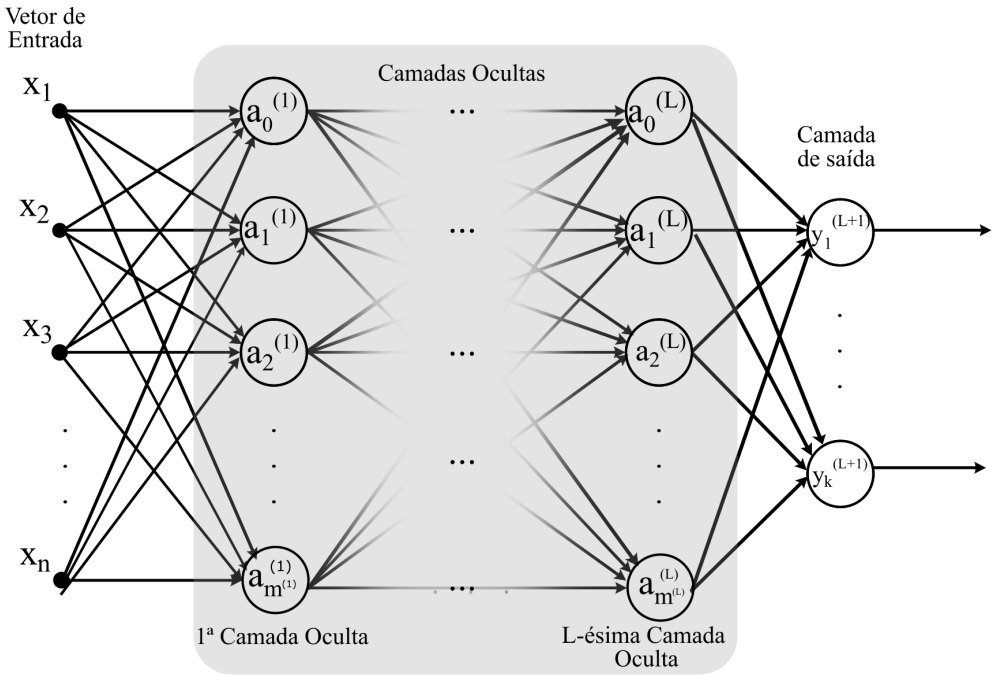


Figura 4.4: *Perceptron* multicamadas.

Dessa forma, um neurônio j, em uma camada l, tem como saída

$$a_j^{(l)} = \sigma(z_j^{(l)}), \text{ onde } z_j^{(l)} = \sum_{k=1}^{m^{(l-1)}} w_{j,k}^{(l)} a_k^{(l-1)} + b_{j,0}^{(l)}. \tag{4.3}$$

Para o treinamento de uma RNA, os pesos de cada camada devem ser inicializados aleatoriamente, para que a atualização dos pesos não seja igual, de modo que a rede não aprenda uma única característica do conjunto de treinamento. Assim, enquanto as saídas da camada de saída não estiverem suficientemente próximas dos valores desejado, os pesos são reajustados até convergirem.

Uma possível forma com que os pesos são atualizados em cada iteração i depende de como a

função de custo $\mathcal{E}$ varia com relação aos pesos, dos atributos **x** da amostra selecionada m naquela iteração, e da taxa de aprendizagem η, que define o quão rápido o modelo irá aprender. Assim, tem-se que

$$\Delta \mathbf{w}(i) = -\eta \frac{\partial \mathcal{E}_m}{\partial \mathbf{w}}, \text{ onde } \frac{\partial \mathcal{E}_m}{\partial \mathbf{w}} = \mathbf{x}_m(i) \sum_m (y_m - y_{ref,m}), \tag{4.4}$$

é a atualização dos pesos, e

$$\mathbf{w}(i+1) = \mathbf{w}(i) + \Delta \mathbf{w}(i), \tag{4.5}$$

o vetor dos pesos para a iteração seguinte. É por este motivo, como mencionado na Sub-seção 4.2.2, que as funções de ativação precisam ser diferenciáveis. As Equações 4.4 e 4.5 descrevem um método de atualização de pesos chamado **descida do gradiente**. A partir desta ideia, pode-se introduzir o algorítimo de retropropagação, um algorítimo para redes de propagação direta capaz de lidar com grandes números de dados.

O **algoritmo de retropropagação** inicia-se propagando os dados de entrada **x** diretamente pela rede e, ao final dela, calcula-se o erro. Após isso, as informações de erro da rede propagam-se de trás para frente, fazendo, para um neurônio j em uma camada oculta arbitrária l, cuja saída $a_j^{(l)}$ é dada por

$$a_j^{(l)} = \sigma \left(\sum_{k=1}^{m^{(l-1)}} w_{jk}^{(l)} a_k^{(l-1)} + b_{j,0}^{(l)} \right), \tag{4.6}$$

em que, se $l = L + 1$ é a camada de saída, então $a_j = y_j$ e e_j é a diferença de um neurônio dessa camada para o seu respectivo valor desejado $y_{ref,j}$,

$$\delta_j^{L+1} = \frac{\partial \mathcal{E}_m}{\partial y_j^{(L+1)}} \cdot \sigma'(z_j^{(L+1)}) = e_j^{(L+1)} \sigma'(z_j^{(L+1)}), \tag{4.7}$$

para a camada de saída, em que $\sigma'(\cdot)$ significa a derivada de σ, e para uma camada arbitrária l, diferente da camada de saída

$$\delta_j^{(l)} = \sigma'(z_j^{(l)}) \sum_{k=1}^{m^{(l+1)}} w_{jk}^{(l+1)} \delta_k^{(l+1)}, \tag{4.8}$$

em que

$$z_j^{(l)} = \sum_{k=1}^{m^{(l-1)}} w_{jk}^{(l)} a_k^{(l-1)} + b_{j,0}^{(l)}. \tag{4.9}$$

E, portanto, para cada camada, a atualização dos pesos fica

$$w_{jk}^{(l)}(i+1) = w_{jk}^{(l)}(i) + \eta \delta_j^{(l)}(i) a_k^{(l-1)}(i). \tag{4.10}$$

Em resumo, o algorítimo de retropropagação é:

1. Propagação direta dos dados de entrada pela rede;
2. Cálculo de retropropagação para a camada de saída, segundo a equação 4.7;
3. Cálculo da retropropagação para as camadas ocultas, segundo a equação 4.8;
4. Atualização de pesos, vide equação 4.10.

4.3 REDES NEURAIS CONVOLUCIONAIS

Quando as entradas passam a ser imagens, existe a necessidade de haver algum meio de se extrair e quantificar os atributos das entradas para gerar um modelo que seja capaz de se encaixar e prever informações de novos dados de entrada. É nesse contexto que surgem as Redes Neurais Convolucionais como extratoras de atributos, como proposto por LeCun [19]. As redes neurais convolucionais são compostas de diferentes tipos de camadas e conforme a complexidade da arquitetura desejada diversos tipos de camadas podem ser empilhadas. As redes neurais convolucionais podem ser compostas de camadas convolucionais, com não-linearidade ou não, camadas de subamostragem, camada de retificação e camada totalmente conectada, que serão vistas nas sub-seções subsequentes.

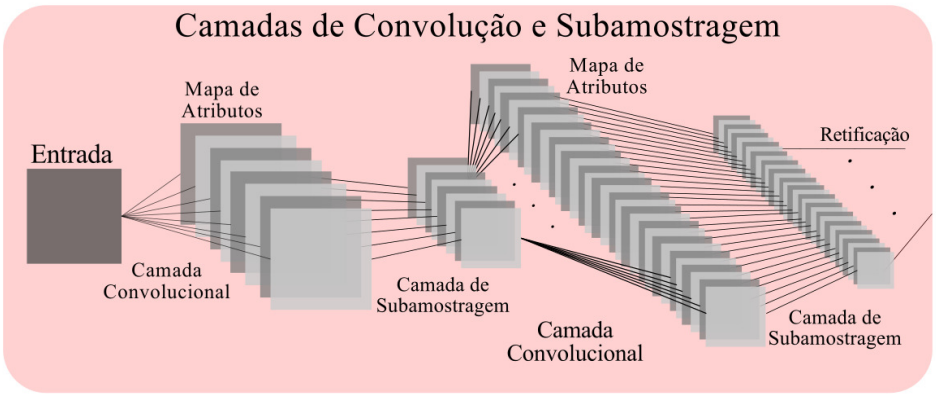


Figura 4.5: Arquitetura básica de uma rede convolucional.

A Figura 4.5 mostra uma arquitetura simples de uma rede que faz a extração de características de uma entrada. A rede como é mostrada na figura é constituída de duas camadas convolucionais, duas camadas de subamostragem e uma camada de retificação, que seguirá numa próxima etapa para uma camada totalmente conectada, isto é, uma RNA.

4.3.1 Camada Convolucional

A convolução é uma operação que tem como característica o reconhecimento de padrões dado uma entrada. A camada convolucional é baseada nesta ideia, visto que *pixels* próximos estão diretamente relacionados. Portanto, a camada convolucional é responsável por identificar padrões em uma imagem extraindo suas características.

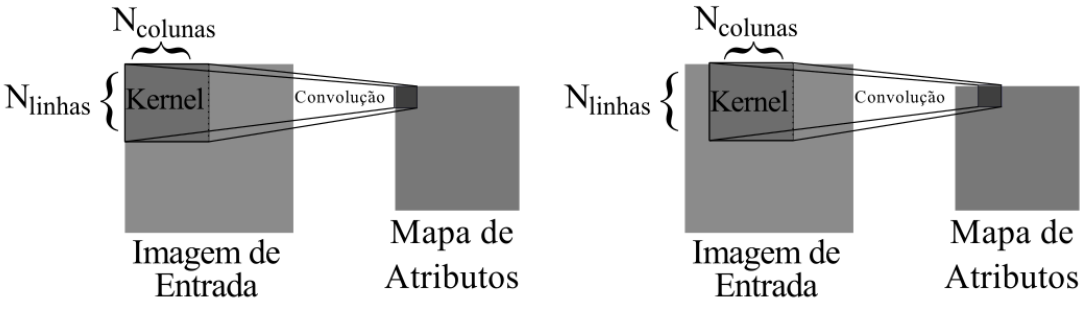
A convolução nesta camada é realizada entre uma pequena região da imagem **x** e um filtro f, gerando como resultado um mapa de atributos, fazendo

$$\mathbf{x} * f[x] = \sum_{i=-\infty}^{\infty} x[i] f[x-i]. \tag{4.11}$$

O filtro é uma unidade que abrange uma pequena área da imagem chamado *kernel* de convolução, que é uma matriz de pesos, na qual o seu produto com os *pixels* da região da imagens passa pela função de ativação do subconjunto de neurônios da camada subsequente, chamados **campos receptivos**. Assim, um filtro com *kernel* de convolução (m, n) de tamanho $N_{linhas} \times N_{colunas}$, aplicado em uma área (x, y) de uma imagem, produz seus mapas de atributos fazendo uma convolução bidimensional discreta,

$$\mathbf{x} * f[x, y] = \sum_{m=1}^{N_{linhas}} \sum_{n=1}^{N_{colunas}} x[m, n] \cdot f[x-m, y-n]. \tag{4.12}$$

em que **x** são os *pixels* contidos na área de convolução. Para o caso da imagem, $x[m, n] \cdot f[x - m, y - n]$ é a saída do neurônio de uma camada l arbitrária, como na Equação 4.3. A Figura 4.7 mostra como é o funcionamento da convolução bidimensional, em que o *kernel* de convolução se desloca entre os pixel.



(a) (b)

Figura 4.6: **Convolução:** Convolução bidimensional discreta.

É comum no deslocamento do *kernel* a imagem não ser completamente abrangida, ocasionado pelo efeito de borda da imagem, fazendo com que seja necessário um preenchimento, de forma que são adicionados *pixels* com valor zero ao redor da imagem.

4.3.2 Camada de Subamostragem

Esta camada é responsável por fazer o agrupamento de um conjunto de *pixels* de uma imagem de acordo com o estabelecimento de alguma regra. Em geral, as arquiteturas usam camadas de subamostragem do tipo *Max Pooling* e *Average Pooling*, porém existem outras formas menos convencionais, como, por exemplo, a *Min-max Pooling*.

A camada de subamostragem é capaz de diminuir o custo computacional do modelo, visto que a quantidade de pesos é reduzida, além de mapear o comportamento das entradas com relação a distorções. As principais formas com que as subamostras são calculadas estão listadas a seguir.

Max Pooling e *Average Pooling*

Quando é feito um *Max Pooling*, são aplicados filtros, assim como nas camadas convolucionais, com tamanho geralmente 3×3 ou 2×2, nos mapas de atributos resultantes das camadas convolucionais. Estes filtros quando aplicados, preservam a quantidade de mapas de atributos e reduzem suas dimensões. Assim, a camada de *Max Pooling* aplica um filtro em uma região (x, y) de um mapa de atributos e capta o maior valor de *pixel* desta região para construir uma nova matriz com este valor, e o filtro se movimenta para a o próxima área, repetindo o processo. Já o *Average Pooling* é aplicada uma ideia semelhante ao *Max Pooling*, porém ao invés de captar o maior valor de uma região, é feita uma média dos valores de *pixel* contida na região.

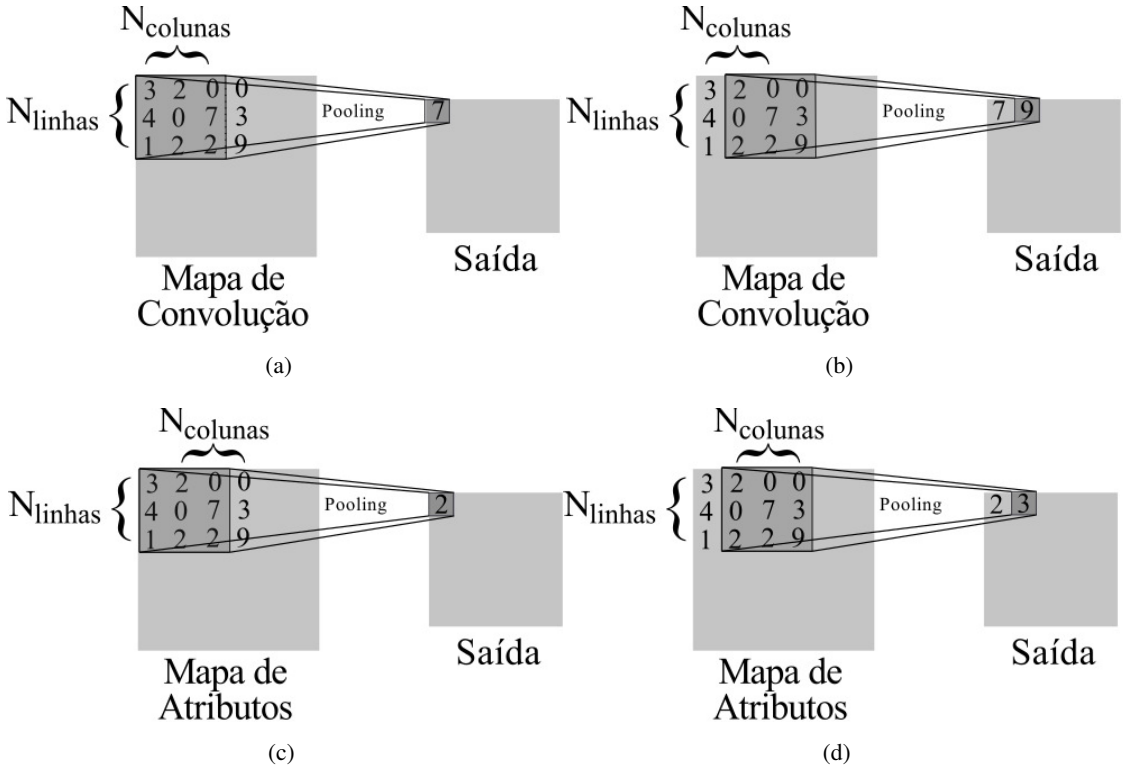


Figura 4.7: **Subamostragem**: (a) e (b) são do tipo *Max Pooling*, e, (c) e (d) são do tipo *Average Pooling*.

Min-max Pooling

Neste trabalho será utilizada uma camada de subamostragem personalizada, chamada *Min-Max Pooling*. Essa camada consiste em duas camadas de *Pooling* que são aplicadas aos mapas de atributos resultantes da camada convolucional anterior. O primeiro *Pooling* é o *Max Pooling*, em que, neste caso pega-se apenas o maior elemento do mapa de atributos, e o segundo é o *Min Pooling*, que pega apenas o menor elemento do mapa de atributos. O maior e o menor valor extraído do mapa de atributos são guardados nos seus respectivos vetor e são concatenados ao fim do processo.

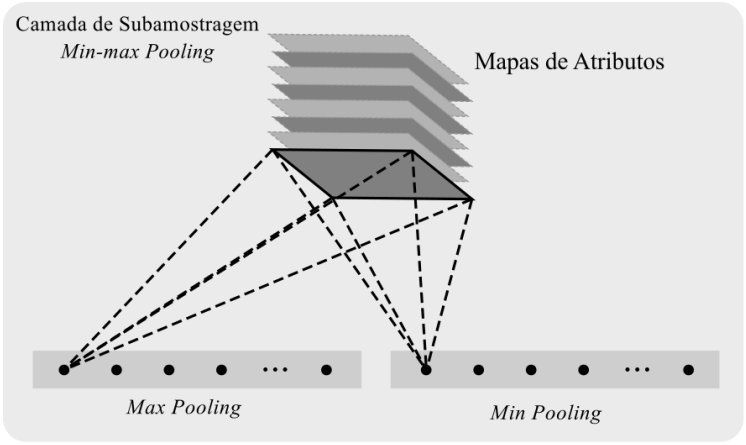


Figura 4.8: Camada de *Min-max Pooling*.

4.3.3 Camada Totalmente Conectada

Ainda, faz-se necessário planificar os dados resultantes da camada de subamostragem para fazê-los servir de entrada para a camada totalmente conectada, que é um rede neural artificial, que por sua será usada como um regressor. As arquiteturas das redes convolucionais são definidas escolhendo a quantidade e disposição de cada camada, bem como seus hiper-parâmetros, como número de filtro, tamanho do *kernel*, o tamanho do deslocamento dos filtros, se haverá preenchimento ou não, entre outros. A Figura 4.9 mostra um modelo básico de uma estrutura convolucional.

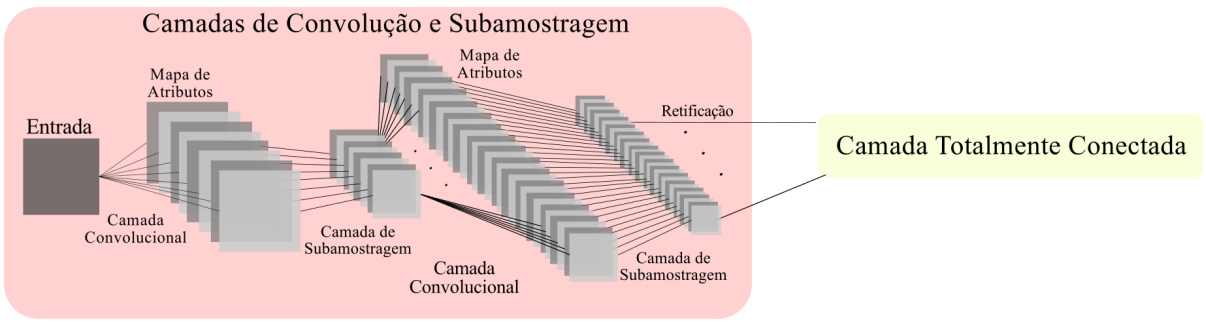


Figura 4.9: **Rede Convolucional genérica:** Arquitetura de camadas de convolução e subamostragem.

5 MÉTODOS ADOTADOS

Este trabalho tem por objetivo estimar a qualidade de vídeos, através de redes neurais convolucionais, utilizando informações de fluxo óptico. Dessa forma, as etapas iniciais de execução deste projeto, isto é, a montagem dos dados de entrada da rede, consistiram no cálculo do fluxo óptico denso, visto na Subseção 3.2. Depois, faz-se necessário empilhar os fluxos ópticos resultantes e normalizar as pilhas geradas, para passar noção de movimento à rede neural convolucional que fará a extração de atributos de cada pilha de fluxo óptico para, assim, realizar a regressão que estimará uma nota para um vídeo de entrada.

O modelo da rede será treinado utilizando vídeos com duas dimensões espaciais, e, além disso, as pilhas de fluxos ópticos serão montadas utilizando suas componentes verticais e horizontais, bem como o módulo dessas componentes. Após o treinamento da rede, novos vídeos serão testados e correlacionados com suas notas originais, através de coeficientes de correlação para qualidade de vídeo, com o objetivo de avaliar a performance do modelo proposto. As seções seguintes mostrarão como são montadas as pilhas de fluxo óptico e a arquitetura da rede escolhida.

5.1 PILHAS DE FLUXO ÓPTICO

Para passar a ideia de movimento para a rede, faz-se necessário utilizar pilhas de fluxos ópticos, segundo o algoritmo mencionado em [20], que são imagens de fluxos ópticos densos concatenados.

A primeira abordagem a respeito da pilha de fluxo óptico é chamado de pilha de fluxos ópticos das componentes, e é realizado fazendo o cálculo do fluxo óptico denso para quadros consecutivos. Como o fluxo óptico é um campo de vetores, para cada fluxo óptico gerado, separa-se cada vetor em componentes verticais e horizontais. Após isso, concatena-se D_x fluxos ópticos de componentes horizontais, resultando em uma pilha horizontal, e concatena-se D_y fluxos ópticos de componentes verticais. Finalmente, as pilhas horizontais e verticais são concatenadas, formando a pilha final, conforme a Figura 5.1(b).

A segunda abordagem é chamada de pilha de fluxos ópticos de módulo. Neste método, calcula-se, também, o fluxo óptico denso de quadros consecutivos. Novamente, do campo de vetores resultantes, separa-se as suas componentes horizontais e verticais, e, a partir delas, calcula-se o seu módulo, fazendo

$$D_m = \sqrt{D_x^2 + D_y^2}, \tag{5.1}$$

em que D_m é o fluxo óptico do módulo, D_x é o fluxo óptico da horizontal e D_y é o fluxo óptico da vertical, calculados como na primeira abordagem. Então para dois fluxos ópticos, um horizontal e um vertical, é gerado o fluxo óptico de módulo, que será concatenado com outros fluxos ópticos de módulo, formando a pilha final, conforme a Figura 5.1(a).

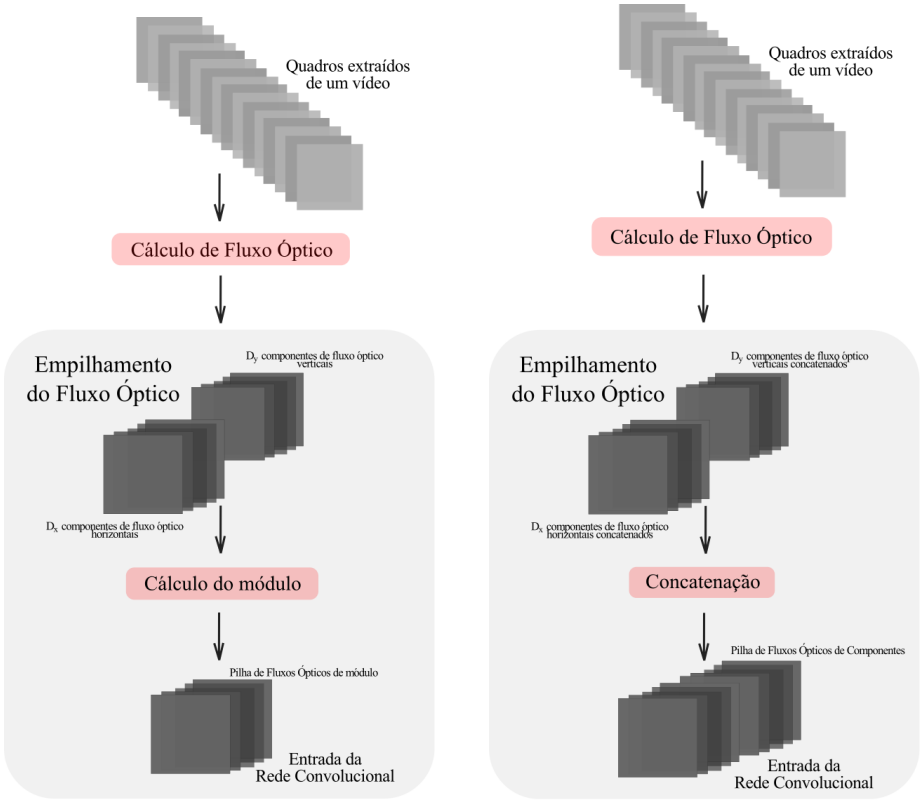


Figura 5.1: **Empilhamento de Fluxos Ópticos:** A sub-figura (a) mostra o empilhamento de fluxos ópticos de módulo, enquanto o empilhamento de fluxos ópticos de componentes é mostrado na sub-figura(b).

5.2 NORMALIZAÇÃO DAS PILHAS

Após a montagem das pilhas é de boa prática fazer a normalização gaussiana dos dados de entrada, com o objetivo de prevenir grandes variações dos valores dos *pixels* das imagens de fluxo ópticos que compõe a pilha. Dessa forma, é necessário, para cada imagem, subtrair a média dos seus *pixels* e dividir pelo seu desvio padrão.

Seja I_k a k-ésima imagem de uma pilha de fluxo óptico, com n_i imagens de fluxo óptico, com dimensão espacial $(n_l \times n_c)$. Então a média $\bar{I}_k$ dos valores de *pixel* de uma imagem é dada por

$$\bar{I}_k = \frac{1}{n_l n_c} \sum_{i=1}^{n_l} \sum_{j=1}^{n_c} I_k(i,j) \tag{5.2}$$

e o seu desvio padrão std_k,

$$std_k = \sqrt{\frac{1}{n_l n_c} \sum_{i=1}^{n_l} \sum_{j=1}^{n_c} \left(I_k(i,j) - \bar{I}_k\right)^2}. \tag{5.3}$$

A partir das Equações 5.2 e 5.3, obtém-se a imagem normalizada $I_{k,norm}$ fazendo, para cada imagem da pilha,

$$I_{k,norm} = \frac{I_k - \bar{I}_k}{std_k}. \tag{5.4}$$

5.3 ARQUITETURA DA REDE E TREINAMENTO

O próximo passo após a manipulação das pilhas de fluxo ópticos, que serão as entradas da rede, é definir a arquitetura da rede neural convolucional, bem como seus hiper-parâmetros. As pilhas de fluxo ópticos passarão por uma arquitetura de rede com uma camada convolucional, para fins experimentais, semelhante a [2]. A camada de convolução bidimensional com possuirá 50 filtros de tamanho de *kernel* 7×7, e será seguida de uma camada de subamostragem, que é do tipo *Min-max Pooling*.

Após isso, a saída da camada de subamostragem será retificada e seguirá para uma camada totalmente conectada atuando como uma rede regressora. Essa camada totalmente conectada é uma rede neural artificial tradicional com duas camadas com 800 neurônios cada, com *Dropout* na segunda camada que faz com que cada neurônio dessa camada tenham probabilidade de 50% de ser desativado, com fins de prevenção de *overfitting*, e com função de ativação ReLU, seguindo para uma camada final com um único neurônio, com ativação logística, que dará a nota final do vídeo. A Figura 5.2 ilustra a topologia da rede escolhida, na qual serão realizados experimentos.

Os treinamentos foram realizados conforme a arquitetura de rede proposta. O conjunto de dados foi reduzido a metade, por questões computacionais, dos quais 50% dos vídeos são do conjunto de treinamento, 33% dos vídeos para o conjunto de teste e 17% para o conjunto de validação. Devido a limitações computacionais, para cada pilha de fluxo óptico de um vídeo de treinamento é necessário fragmentá-la em unidades menores chamadas *batches*. Foi escolhido experimentalmente um *batch* de tamanho de 32 pilhas de fluxos ópticos.

Para cada época, após todos os vídeos de treinamento, foi calculado o erro de predição para o conjunto de validação com o objetivo de fazer uma análise da quantidade de épocas necessárias e a respeito do comportamento do modelo em geral. O otimizador escolhido foi o SGD, descida do gradiente estocástico, cuja a atualização de pesos é feita de forma semelhante a [2], com *momentum* $m = 0,9$, taxa de aprendizagem inicial $\eta_0 = 0,001$, com deixaimento $d = 0,0004$.

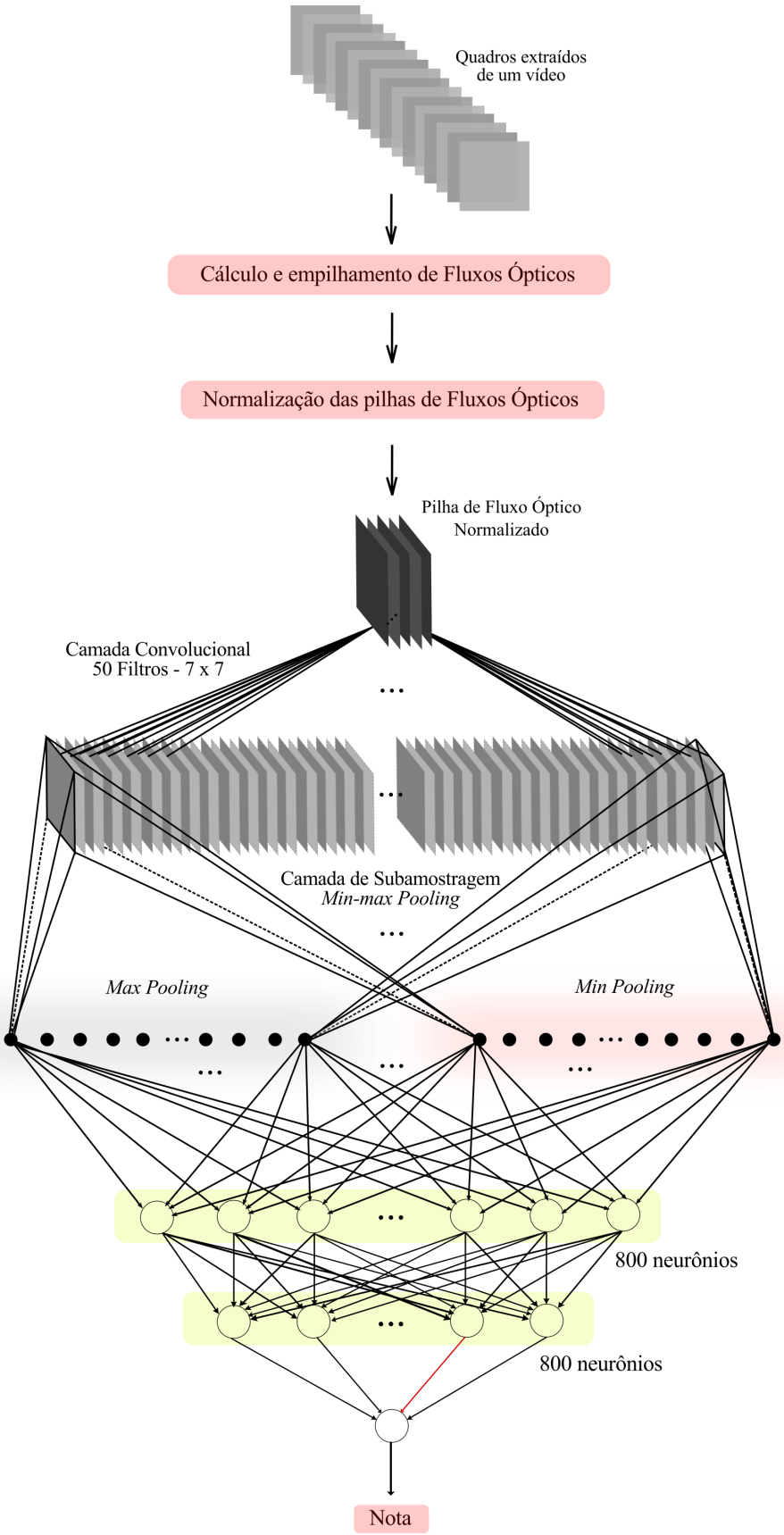


Figura 5.2: Arquitetura proposta.

5.4 COEFICIENTES DE CORRELAÇÃO

Com o modelo treinado, serão feitas as predições de notas em um conjunto de teste de vídeos que nunca foram vistos pela rede. Estas predições deverão ser correlacionadas com suas respectivas notas verdadeiras, por meio do Coeficiente de Correlação de Postos de *Spearman* (SROCC) e pelo Coeficiente de Correlação Linear (LCC).

Para compreender o Coeficiente de Correlação de Postos de *Spearman* é necessário entender que funções monótonas são funções que, ou nunca crescem ou nunca decrescem, quando sua variável cresce. Dessa forma, o SROCC mede o quão monótona é a relação entre um par de dados. A medida SROCC R_s é dada por

$$R_s = 1 - \frac{\sum d_i^2}{n(n^2-1)}, \tag{5.5}$$

em que d_i é a diferença entre postos distintos de uma medida i e n é a quantidade total de postos diferentes. A Equação 5.5 indica que $-1 < R_s < 1$, sendo $R_s = -1$ uma relação monotônica perfeitamente decrescente e $R_s = 1$ uma relação monotônica perfeitamente crescente.

O Coeficiente de Correlação Linear ρ indica quanto dois conjuntos de dados estão linearmente associados. Novamente, o indicador está restrito a $-1 < \rho < 1$ e $\rho = -1$ mostra uma forte relação linear decrescente entre os dados e $\rho = 1$, uma forte relação linear crescente. No caso deste trabalho, calcula-se o LCC das predições das notas y com suas respectivas notas verdadeiras y_{ref}, fazendo

$$\rho = \frac{\sum_i (y_i - \bar{y})(y_{i,ref} - \bar{y}_{ref})}{\sqrt{\sum_i (y_i - \bar{y})^2}\sqrt{\sum_i (y_{i,ref} - \bar{y}_{ref})^2}}. \tag{5.6}$$

6 RESULTADOS

Este capítulo versará sobre os resultados obtidos seguindo o método de trabalho mencionado no Capítulo 5. Estará detalhado qual o conjunto de dados que foi utilizado para servir como entrada da topologia de Rede Neural Convolucional e para a estimação das notas dos vídeo. Ainda, será mostrado os parâmetros e condições de cada experimento, produzindo os resultados deste trabalho.

6.1 O CONJUNTO DE DADOS

O conjunto de dados escolhido foi o *CSIQ Video Quality Database* [21], desenvolvido pela Universidade do Estado de Olkahoma. Este conjunto de dados é composto de 228 vídeos, sendo 12 vídeos de referência e 216 vídeos com algum tipo de distorção. Para cada vídeo de referência, existem 18 vídeos distorcidos, com 6 diferentes tipos de distorções, cada uma com 3 níveis de intensidade.

Todos os vídeos desta base de dados estão no formato YUV420, com dimensão espacial de 832×480 *pixels*, duração de 10 segundos, e resolução temporal de 24, 25, 30, 50 e 60 quadros por segundos. Os tipos de distorções que os vídeos apresentam são causadas pelas compressões H.264, HEVC, do tipo *wavelet* utilizando o *codec Snow*, e MJPEG, e por transmissão de sinal, como a introdução de ruído aditivo gaussiano e a simulação de perda de pacotes por transmissão *wireless*.

Figura 6.1: ***CSIQ Video Database***: quadros exemplos.

6.2 EXPERIMENTOS

Para todos os experimentos realizados, foram utilizados vídeos com todas as resoluções temporais disponíveis (24, 25, 30, 50 e 60 quadros por segundo). Foram testadas as duas dimensões espaciais 832×480 e 640×360, afim de se julgar a influência da dimensão espacial na capacidade de predição de notas da rede. O tipo de fluxo óptico que foi utilizado na entrada da rede neural convolucional foi o de módulo. Dado o tamanho da base de dados escolhida, a etapa de treinamento demandava bastante tempo, ocasionando em um número reduzido de experimentos. Os experimentos não realizados serão feitos em trabalhos futuros como será discutido adiante.

Todos os experimentos realizados foram rodados nas placas de vídeos *Nvidia GeForce GTX 1080* e *Nvidia Quadro P6000* e os fluxos ópticos calculados utilizando o pacote *OpenCV* em *Python*. Além disso, para a Rede Neural Convolucional foi utilizado o otimizador *SGD* e as medições de erro MSE, *Mean Squared Error* e MAE, *Mean Absolute Error*. As sub-seções seguintes trazem os experimentos realizados.

6.2.1 Experimento 1

O primeiro experimento foi realizado para a dimensão espacial de 640×360, com pilhas de fluxo óptico do tipo módulo e arquitetura como proposta no capítulo 5. Para o treinamento, utilizou-se *batch* de 32 amostras e foram realizadas 30 épocas, em que ao final de cada época foi realizada validação dos dados e ao final de todas as épocas o conjunto de teste foi testado pela rede. A Tabela 6.1 abaixo mostra o resumo da rede escolhida.

Tipo de Camada	**Ativação**	**Dimensão de Saída**	**Número de Parâmetros**
Entrada	-	(32, 5, 360, 640)	-
Convolução 2D	ReLU	(32, 50, 354, 634)	12300
Min-max Pooling	Linear	(32, 100, 1, 1)	-
Retificação	-	(32, 100)	-
Totalmente Conectada 1	ReLU	(32, 800)	80800
Totalmente Conectada 2	ReLU	(32, 800)	640800
Dropout	-	-	-
Saída	Sigmoid	(32, 1)	801
Total:			734701

Tabela 6.1: Resumo da rede a ser treinada.

Durante o treinamento, foram registrados os erros quadráticos médios de treinamento e de validação para cada época, com a finalidade de analisar o aprendizado da rede. Assim, pode-se dizer se a rede comportou-se da forma esperada ou ocorreram fenômenos indesejados, como *overfitting* ou *underfitting*. Aqui, analisa-se, também, se a quantidade de épocas utilizadas para o treinamento foi suficiente para o que foi proposto. A Figura 6.2 mostra as curvas de treinamento e validação obtidas, são a partir delas que são definidas as estratégias para novas configurações da rede.

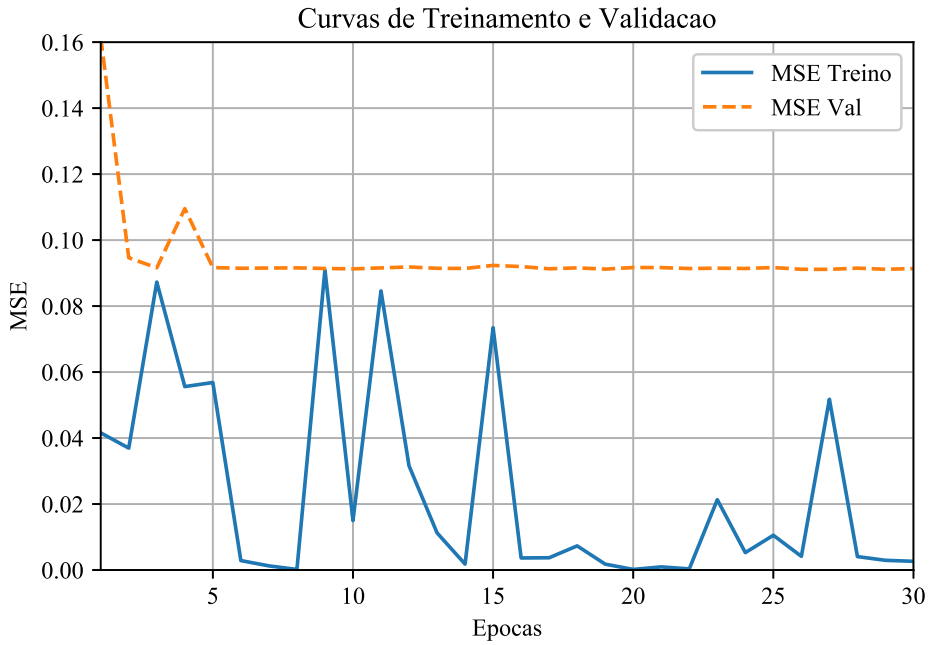


Figura 6.2: **Experimento 1**: Curvas do erro médio quadrático para os conjuntos de treinamento e validação.

Após o treinamento, foram preditas notas dos vídeos do conjunto de teste. As notas preditas foram correlacionadas com suas respectivas notas originais pelo método SROCC e LCC, descritos no Capítulo 5. Os valores obtidos para as correlações podem ser confirmadas através do gráfico de dispersão mostrado na figura 6.3.

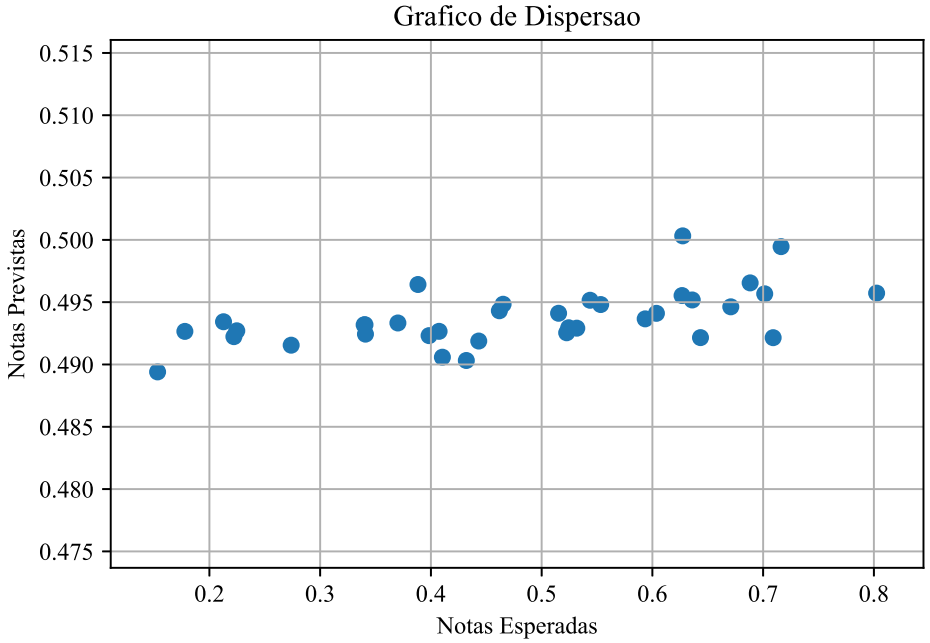


Figura 6.3: **Experimento 1**: Gráfico de dispersão das notas previstas com as originais.

Os valores obtidos para cada coeficiente de correlação está mostrado na tabela 6.2.

SROCC	LCC
0,55	0,58

Tabela 6.2: Coeficientes de Correlação do Experimento 1

Observa-se de uma análise geral do Experimento 1 que a curva de treinamento ficou relativamente afastada da curva de validação, indicando que o treinamento não realizado de forma satisfatória. Este fato pode ter sido ocasionado devido a utilização de um conjunto de dados reduzido à metade, motivado pela grande demanda de tempo que o conjunto todo necessitaria. Além disso, outro possível motivo pelo qual as curvas ficaram distantes é a ocorrência de *overfitting*, ou seja, os dados de treinamento não foram capazes de generalizar o modelo até a época proposta. Uma estratégia a ser adotada é a utilização de regularizadores, que aplicam penalidades aos parâmetros das camadas durante a o treinamento, visando reduzir a diferença entre as curvas.

Entretanto, foi observado que as notas preditas se comportam de forma semelhante às notas originais, mesmo estando relativamente distantes, isto é, as notas preditas caem conforme as distorções aumentam ou as notas aumentam com a redução das distorções, assim como nas notas originais, mostrando que a rede, mesmo com o problema mencionado acima, foi capaz de aprender sobre as distorções.

6.2.2 Experimento 2

O Experimento 2 foi realizado em circunstância semelhantes ao Experimento 1, aumentando apenas a dimensão espacial para 832×480. A arquitetura da rede foi mantida, bem como a quantidade de amostras, isto é o *batch* de tamanho 32. Também foram realizadas 30 épocas de treinamento, em que ao final de cada época foi realizada a validação dos dados e ao final de todo o treinamento foi realizada a predição de notas no conjunto de teste.

Da mesma forma que no procedimento anterior, foram montados os gráficos com as curvas do erro quadrático médio tanto para o treinamento, quanto para a validação, conforme mostra a Figura 6.4.

Após todo o treinamento, faz-se necessário avaliar a correlação entre as notas previstas e as suas respectivas notas verdadeiras afim de se determinar a validade do modelo. Foram calculados os coeficientes de *Spearman* (SROCC) e o coeficiente de correlação linear (LCC). O gráfico mostrado na Figura 6.5 traz a dispersão dos dados, reforçando os valores dos coeficientes de correlação encontrados, expostos na Tabela 6.3.

SROCC	LCC
0,53	0,52

Tabela 6.3: Coeficientes de Correlação do Experimento 2

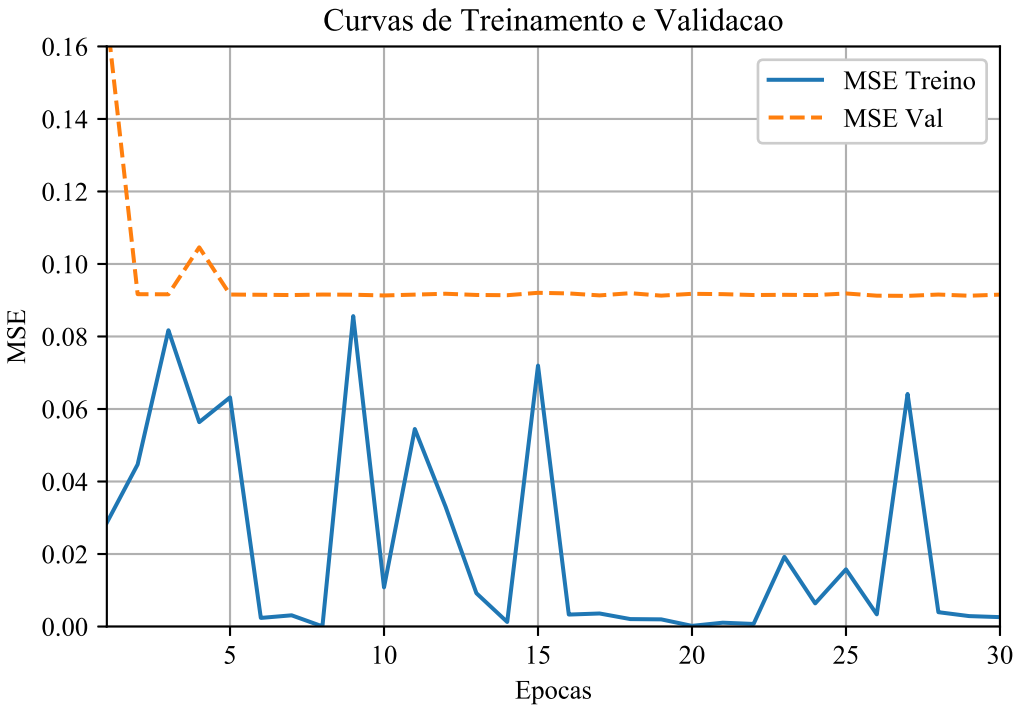


Figura 6.4: **Experimento 2**: Curvas do erro médio quadrático para os conjuntos de treinamento e validação.

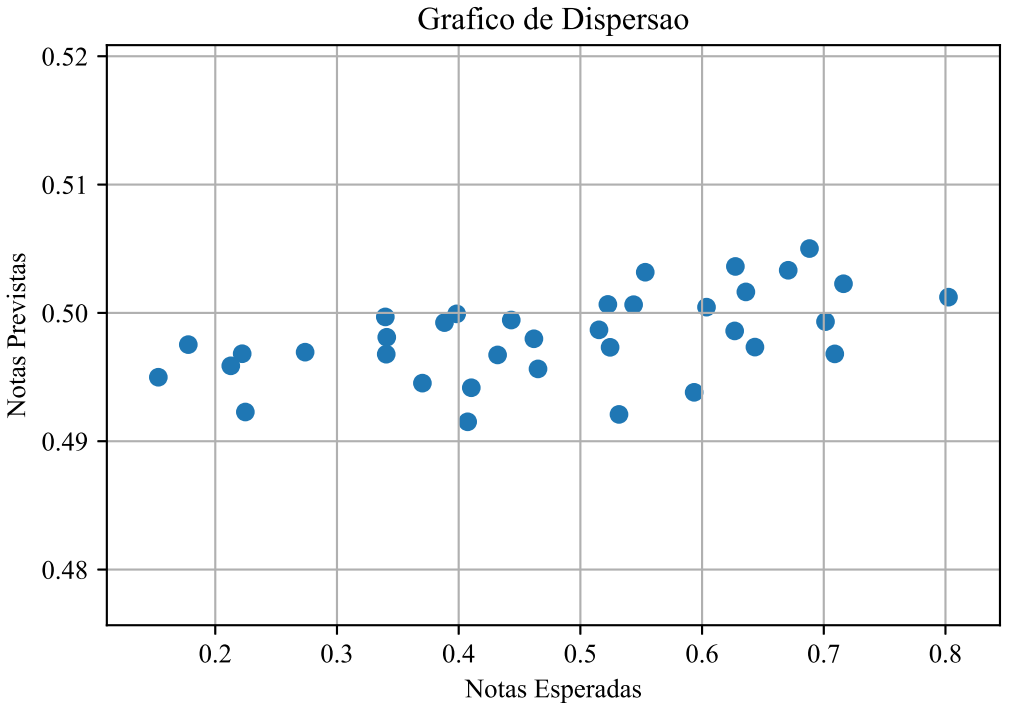


Figura 6.5: **Experimento 2**: Gráfico de dispersão das notas previstas com as originais.

As curvas de treinamento no Experimento 2 se comportaram muito semelhante ao Experimento 1, havendo mudanças pontuais em algumas épocas apenas, possivelmente por que a dife-

rença entre as dimensões não foi muito grande. Novamente, as curvas de treinamento e validação ficaram distantes entre si. Além disso, os coeficientes de correlação SROCC e LCC se comportaram conforme a dispersão dos conjuntos de notas previstas e originais.

Os coeficientes de correlação obtidos no Experimento 2 quando comparados aos do Experimento 1 estão de acordo com os resultados obtidos em [22], que mostram que os coeficientes de correlação aumentam quando a dimensões espacial dos vídeos é reduzida, ocasionado devido ao aumento em determinados tipos de distorções.

6.2.3 Experimento 3

A realização do Experimento 3 se deu dado a necessidade de adoção de estratégias para aproximar as curvas de treinamento e validação obtidas no Experimento 1. Para isso, foram realizadas 10 novas épocas de treinamento a partir dos pesos já treinados no Experimento 1, com a adição de regularizadores, aplicando penalidades nos parâmetros da rede durantes o treinamento, com o objetivo de reduzir um possível *overfitting*. As penalidades são aplicadas a função de custo, fazendo

$$\mathcal{E} = \frac{1}{2}\sum_{j=1}^{m}\left(\sigma\left(\sum_{i=1}^{n} x_i w_i + b\right) - y_{ref,m}\right)^2 + \lambda\sum_{i=1}^{n}|w_i| + \lambda\sum_{i=1}^{n} w_i^2, \tag{6.1}$$

em que λ é a proporção de penalidade que será aplicada aos pesos w e, neste caso, foi escolhido $\lambda = 0,01$.

Após o treinamento sob as novas circunstâncias, o procedimento seguiu como proposto no capítulo 5. Foram montados os gráficos das curvas de treinamento e validação para as novas épocas de treinamento, mostrado na Figura 6.6, assim como calculados os coeficientes de correlação SROCC e LCC, validados pelo gráfico de dispersão representado na Figura 6.7.

Observa-se da Figura 6.6 que a aplicação dos regularizadores foi capaz de aproximar as curvas de erro médio quadrático de treinamento e validação, durante as 10 épocas adicionais realizadas no novo treinamento, mostrando que os regularizadores foram capazes de reduzir a complexidade do modelo.

Como pode ser visto na Figura 6.7, tanto o SROCC e o LCC obtidos foram iguais a 0, indicando que o modelo perdeu sua capacidade de prever as notas. Essa abordagem sugere a ocorrência de *overfitting*, causado, principalmente, pelo excesso de treinamento do modelo, ou seja, quantidade de épocas acima do necessário, além do fato de o conjunto ainda continuar reduzido a metade, treinando o modelo com baixa variabilidade de dados de entrada. Entretanto, não é possível afirmar com convicção que houve *overfitting* sem avaliar a performance do modelo e, caso de fato ocorra, uma possível estratégia que poderia ser tomada é reduzir a quantidade total de épocas de treinamento e iniciar a aplicação dos regularizadores a partir de um número de épocas menor.

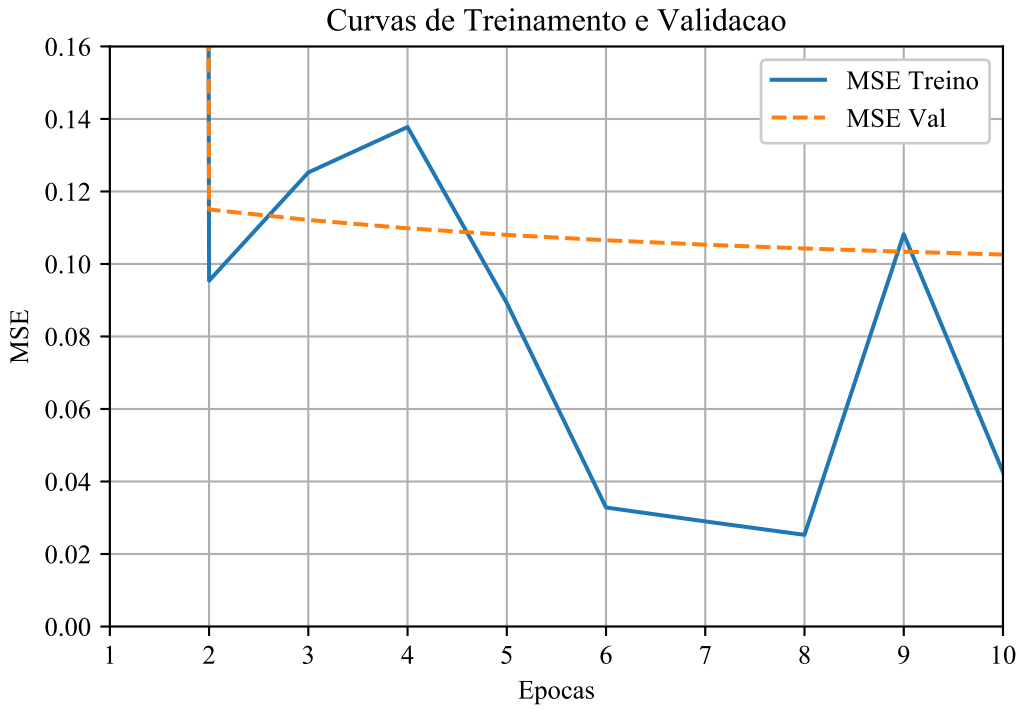


Figura 6.6: **Experimento 3**: Curvas do erro médio quadrático para os conjuntos de treinamento e validação.

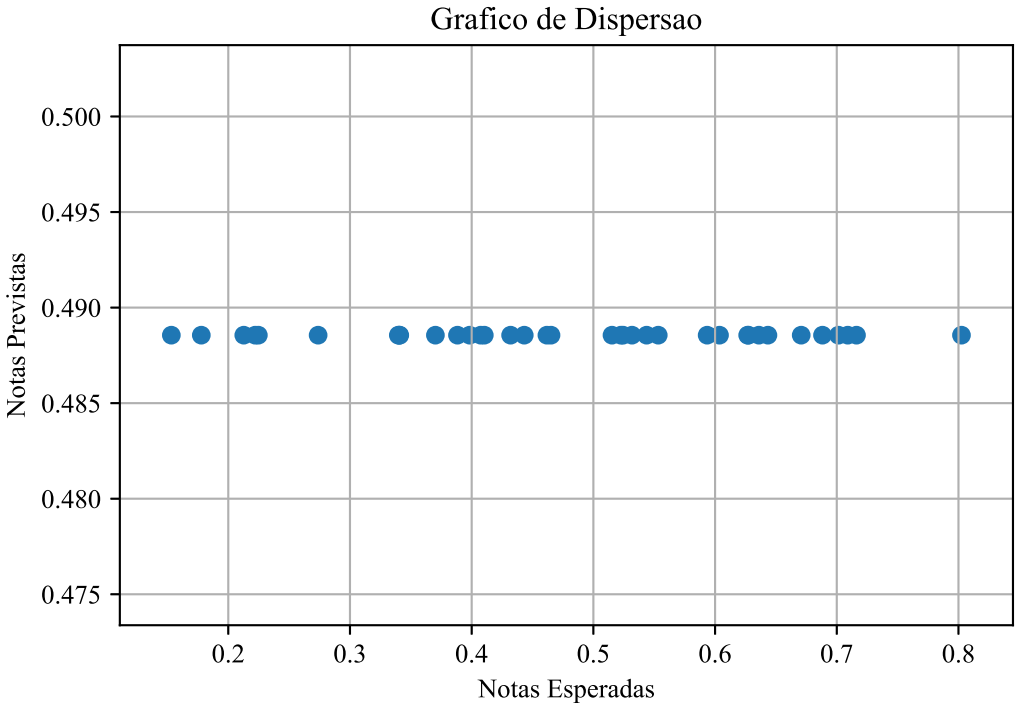


Figura 6.7: **Experimento 3**: Gráfico de dispersão das notas previstas com as originais.

6.2.4 Experimento 4

Assim como o Experimento 3, o Experimento 4 consistiu na aplicação de regularizadores de pesos em novas épocas de treinamento como continuação do Experimento 2, ou seja, para a dimensão 832×480, com o objetivo, também, de aproximar as curvas de treinamento e validação, melhorando a capacidade de generalização do modelo proposto.

Para as novas épocas treinadas foram plotadas as curvas de treinamento e validação, como mostradas na Figura 6.8. Em seguida, os coeficientes de correlação do modelo foram calculado e validados pelo gráfico de dispersão. A figura 6.9 traz o gráfico de dispersão dos notas preditas e originais, mostrando que os coeficientes de correlação foram praticamente nulos.

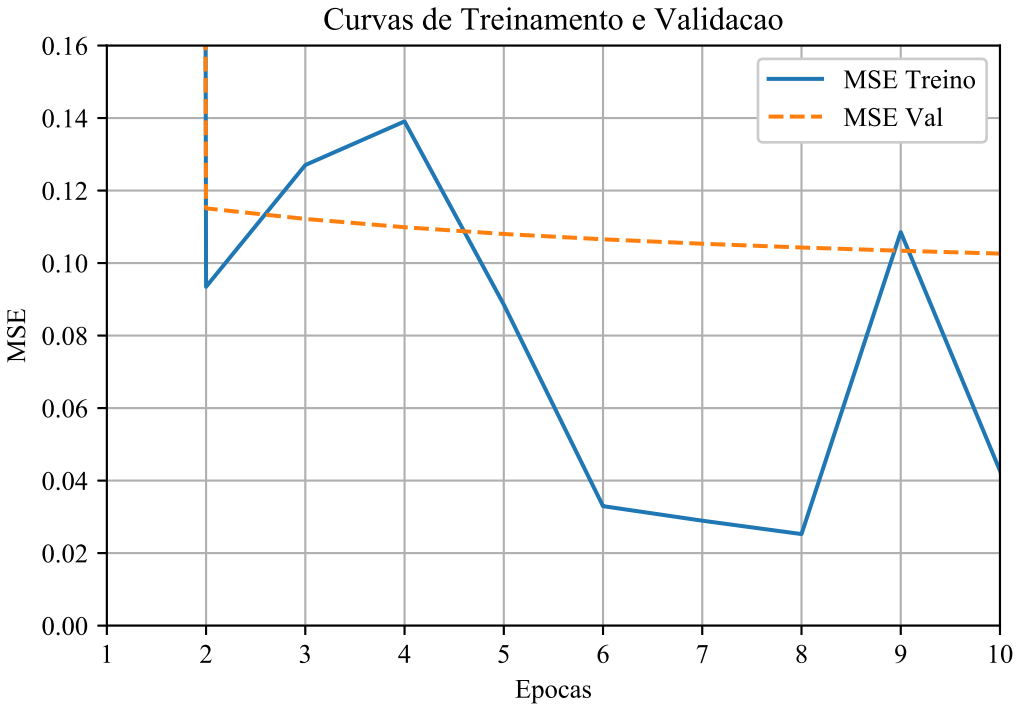


Figura 6.8: **Experimento 4**: Curvas do erro médio quadrático para os conjuntos de treinamento e validação.

Ainda, dado a semelhança com os resultados do Experimento 3, observa-se que a rede também não consegue aprender sobre as diferentes dimensões espaciais. Novamente, as estratégias que devem ser adotadas para corrigir este problema estão relacionadas com medidas para solução de *overfitting*, como aumento de dados de treinamento e/ou redução da quantidade total de épocas.

Dessa maneira, sugere-se uma nova abordagem, que será vista na sub-seção seguinte, cujo objetivo é verificar o comportamento de aprendizagem, estabelecer uma comparação entre as diferentes abordagens adotadas e, finalmente, fazer uma decisão sobre qual abordagem seria mais viável para trabalhos futuros, bem como quais novos experimentos poderiam ser propostos para esta nova situação.

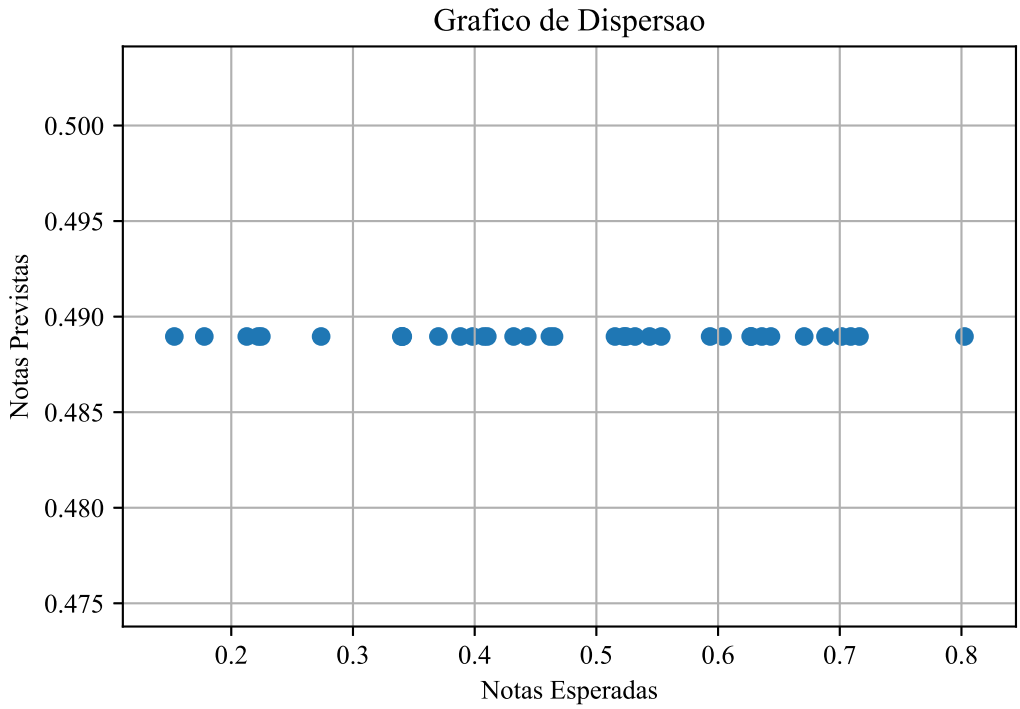


Figura 6.9: **Experimento 4**: Gráfico de dispersão das notas previstas com as originais.

6.2.5 Experimento 5

O Experimento 5 foi realizado segundo uma nova proposta de manipulação de dados de entrada, arquitetura de rede e treinamento. A avaliação dos resultados é feito da mesma forma que anteriormente, através dos coeficientes de correlação SROCC e LCC.

A arquitetura da nova abordagem consiste em duas camadas convolucionais com 32 filtros e *kernel* de tamanho 7×7. Em seguida, uma camada de *Max Pooling*, seguindo para, novamente, uma camada de convolução, agora com 64 filtros de tamanho de *kernel* 7×7, passando para a camada de subamostragem de *Min-max Pooling*. Após a etapa de extração de características, as saídas retificadas da última camada de subamostragem segue para a RNA regressora, com duas camadas ocultas com 800 neurônios, ativação ReLU, seguindo para a camada de saída com 1 neurônio sem ativação. O otimizador escolhido foi o *RMSprop* [23], com seus valores padrões.

Os dados de entrada nesta nova abordagem passam a ser as imagens de fluxo óptico, com dimensões espaciais 640×360, propriamente ditas e não mais as pilhas. Cada imagem de fluxo óptico é rotulada com as notas originais de seus respectivos vídeos e é normalizada conforme o método proposto, isto é, fazendo a subtração da média e divisão pelo desvio padrão.

A nova abordagem reduziu o tempo de treinamento, sendo possível utilizar todo o conjunto de dados, que foram divididos em 8 vídeo de treinamento, 2 de validação e 2 para teste, e treinar mais épocas para investigar o modelo. Assim, o treinamento consistiu em 50 épocas, na qual cada foi utilizada 32 amostras por iteração. A Figura 6.10 mostra as curvas de aprendizgem.

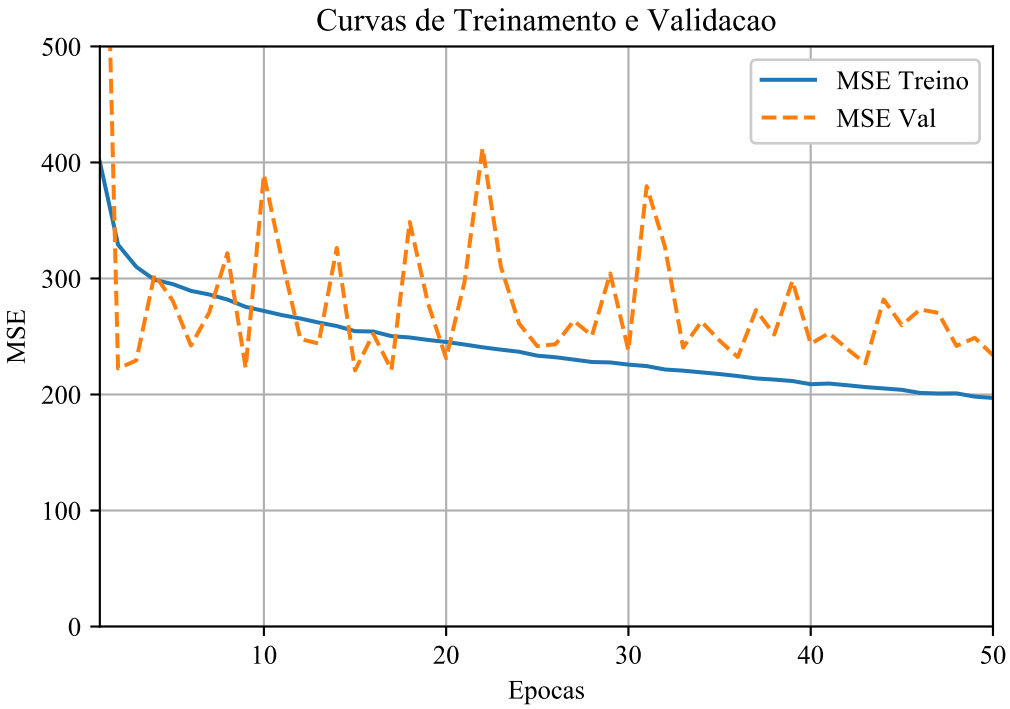


Figura 6.10: **Experimento 5**: Curvas do erro médio quadrático para os conjuntos de treinamento e validação.

Deste treinamento, observa-se que não é possível afirmar com propriedade a respeito do aprendizado do modelo, visto que a quantidade de épocas treinadas ainda foi baixa. Porém, as curvas obtidas se mostram mais condizentes com o esperado visto na literatura. Com a notas preditas ao final do treinamento, foram calculados os coeficientes de correlação SROCC e LCC, mostrados na Tabela 6.4.

SROCC	**LCC**
0,19	0,15

Tabela 6.4: Coeficientes de Correlação do Experimento 5

Estes coeficientes mostram que há uma fraca correlação entre as notas preditas e as notas originais, mas devido à quantidade de épocas treinadas este resultado é esperado, de modo que só seria possível ser mais conclusivo com mais épocas de treinamento.

7 CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho foi proposto um método objetivo de avaliação de qualidade de vídeo, utilizando Redes Neurais Convolucionais, cujas entradas eram pilhas de fluxos ópticos, isto é, informações temporais dos vídeos. As pilhas de fluxo óptico passaram por uma rede constituída de uma camada convolucional com 50 filtros de tamanho de *kernel* 7×7, seguindo por uma camada de *Min-max Pooling* e passando por uma RNA regressora, estimando, finalmente, a nota de um vídeo. Experimentos foram realizados para aferir a qualidade do modelo proposto para duas dimensões espaciais diferentes, 832×480 e 640×360, além de tentativas de otimizar o método de predição de notas, com a utilização de regularizadores.

Do experimentos 1 e 2 foi possível observar que as condições do método proposto não foram suficientes para fazer generalizar da maneira esperada, fazendo com que o modelo aprendesse sobre as distorções, mas ainda não da maneira correta. Esse fato foi observado através dos coeficientes de correlação SROCC e LCC, validados pelo gráfico de dispersão entre as notas originais e as notas preditas. Dessa forma, foram propostos novos experimentos com a aplicação de regularizadores com a finalidade de aproximar as curvas de treinamento e validação, reduzindo a complexidade do modelo.

Os experimentos 3 e 4 consistiram na realização de 10 novas épocas de treinamento com aplicação de regularizadores nas camadas totalmente conectadas, visando diminuir a complexidade do modelo e, assim, aproximar as curvas de treinamento e validação. As curvas de fato foram aproximadas, entretanto o modelo perdeu sua capacidade de estimar notas para o conjunto de teste, possivelmente pelo excesso número de épocas treinadas aliado com o fato do modelo não ser treinado com o conjunto de dados completo, ocasionando em *overfitting*. Como solução imediata, faz-se necessário a redução do número de épocas, fazendo com que o modelo treine somente o necessário.

O Experimento 5 foi realizado seguindo uma nova abordagem, com uma rede mais complexa e deixando de utilizar pilhas de fluxo óptico, tornando possível realizar mais épocas de treinamento em menos tempo. A quantidade de épocas de treinamento não foi suficiente para poder afirmar a respeito da performance do modelo, de modo que as curvas de treinamento e validação são inconclusivas, bem como os coeficientes de correlação SROCC e LCC obtidos, que indicaram baixa correlação.

Outro fato que não pode ser negligenciado neste trabalho, é que os rótulos dados para os fluxos ópticos foram definidos de acordo com o vídeo. Dessa maneira, o fluxo óptico, mesmo não contendo distorções, recebe a nota do vídeo em questão, que pode ter recebido uma nota baixa, relativa à alta distorção. Este fato dificulta o treinamento da rede, mostrando que, para trabalhos futuros, faz-se necessário um estudo dos momentos dos vídeos, para aumentar a precisão das notas dos fluxos ópticos.

Os próximos passos deste trabalho estarão relacionados a melhorias no modelo proposto. Para isso, serão realizados treinamentos em todo o conjunto de dados e em outras bases de dados, com a finalidade de aumentar os dados de treinamento do modelo, para, assim, melhorar a sua capacidade de generalização. Somado a isso, para avaliar as reais capacidades do modelo, deverá ser aplicado o método de reamostragem conhecido como Validação Cruzada, que consiste em dividir diversas vezes todo o conjunto de dados em conjuntos de treinamento e teste, mutuamente exclusivos, de modo que a cada divisão seja realizado um treinamento com os conjuntos de treinamento e teste sendo sempre diferentes.

Além disso, novos experimentos deverão ser realizados, nos quais serão testados os treinamentos utilizando como entrada da rede pilhas de fluxo ópticos do tipo componente, para ambas dimensões. Poderão ser realizadas experimentos utilizando, também, novas dimensões espaciais, para corroborar com o que foi concluído neste trabalho.

Ainda como trabalho futuro, este projeto poderá ser complementado com uma segunda rede que auxiliará na predição de notas utilizando informações espaciais, isto é, os quadros dos vídeos. Uma possível rede a ser implementada é uma Rede Geradora Adversária, que tentará produzir quadros de vídeos de melhor qualidade para ser comparados com os seus respectivos distorcidos, passando por uma rede regressora que dará a nota, e ao final se juntará com a nota dada pela Rede Convolucional Temporal proposta neste trabalho, para, assim, dar a nota final do vídeo. Outra possibilidade é a implementação de uma segunda Rede Neural Convolucional que desta vez receberá informações espaciais dos vídeos, e sua saída, juntamente com a rede temporal, produzirá a nota final do vídeo.

REFERÊNCIAS BIBLIOGRÁFICAS

1 FARNEBäCK, G. Two-frame motion estimation based on polynomial expansion. *n Scandinavian conference on Image analysis*, Springer, p. 363–370, 2003.

2 KANG, L.; YE, P.; LI, Y.; DOERMANN, D. Convolutional neural networks for no-reference image quality assessment. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, p. 1733–1740, 2014.

3 HORN, B. K.; SCHUNCK, B. G. Determining optical flow. *Artificial intelligence*, v. 17, n. 1-3, p. 185–203, 1981.

4 MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, v. 5, n. 4, p. 115–133, 1943.

5 TEKALP, A. M. *Digital Video Processing*. 2. ed. [S.l.]: Prentice Hall, 2015.

6 WATKINSON, J. *The Art of Digital Video*. 4. ed. [S.l.]: Focal Press, 2008.

7 FARIAS, M. C. Q. Video quality metrics. In: RANGO, F. D. (Ed.). *Digital Video*. Rijeka: IntechOpen, 2010. cap. 16. Disponível em: <https://doi.org/10.5772/8038>.

8 GONZALEZ, R. C.; WOODS, R. E. *Digital Image Processing*. 2. ed. [S.l.]: Prentice Hall, 2002.

9 SULLIVAN, G. J.; WIEGAND, T. Video compression - from concepts to the h.264/avc standard. *Proceedings of the IEEE*, v. 93, n. 1, p. 18–31, Jan 2005. ISSN 0018-9219.

10 SULLIVAN, J.-R. O. G. J.; WIEGAND, T. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 22, n. 12, p. 1669–1684, Dec 2012.

11 WALKER, J. S. *The Transform and Data Compression Handbook*. [S.l.]: CRC Press, 2001.

12 ITU-R. Recommendation bt.500-8: Methodology for subjective assessment of the quality of television pictures. 1998.

13 ITU-T. Recommendation p.910: Subjective video quality assessment methods for multimedia applications. 1999.

14 ALPAYDIN, E. *Introduction to Machine Learning*. [S.l.]: MIT Press, 2010.

15 MITCHELL, T. M. *Machine Learning*. [S.l.]: McGraw-Hill Science/Engineering/Math, 1997.

16 SMOLA, A.; VISHWANATHAN, S. V. N. *Introduction to Machine Learning*. [S.l.]: Cambridge University Press, 2008.

17 HAYKIN, S. *Neural networks and learning machines*. [S.l.]: Pearson Upper Saddle River, 2009.

18 ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, v. 65, n. 6, p. 386–408, 1958.

19 LECUN, Y. Generalization and network design strategies. *Technical Report CRG-TR-89-4*, p. 1–19, 1989.

20 SILVA, V. O. da. *Human Action Recognition in Image Sequences Based on a Two-stram Convolutinal Neural Network Classifier*. Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF: PPGEA-672/2017, 2017. 66 p.

21 VU, P. V.; CHANDLER, D. M. Vis3: An algorithm for video quality assessment via analysis of spatial and spatiotemporal slices. *Journal of Electronic Imaging*, v. 23, n. 1, 2014.

22 AKAMINE, W. Y. L. *On the Performance of Video Quality Assessment Methods for Spatial and Temporal Resolutions*. Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF: PPGEA-655/2017, 2017. 78 p.

23 HINTON, N. S. G.; SWERSKY, K. *Lecture 6e RMSprop: divide the gradient by a running average of its magnitude*. [S.l.]: Universidade de Toronto.