UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CIÊNCIAS DA COMPUTAÇÃO

ALEXANDRE BEHLING

RECONHECIMENTO DE EMOÇÕES EM VÍDEO UTILIZANDO REDES NEURAIS ARTIFICIAIS

Florianópolis, SC
2019

ALEXANDRE BEHLING

RECONHECIMENTO DE EMOÇÕES EM VÍDEO UTILIZANDO REDES NEURAIS ARTIFICIAIS

Trabalho de Conclusão de Curso apresentado ao curso de Ciências da Computação da Universidade Federal de Santa Catarina, como requisito parcial para a Obtenção do grau de Bacharel em Ciências da Computação.
Orientador: Prof. Dr. Mauro Roisenberg

Florianópolis, SC
2019

ALEXANDRE BEHLING

RECONHECIMENTO DE EMOÇÕES EM VÍDEO UTILIZANDO REDES NEURAIS ARTIFICIAIS

Trabalho de Conclusão de Curso apresentado ao curso de Ciências da Computação da Universidade Federal de Santa Catarina, como requisito parcial para a Obtenção do grau de Bacharel em Ciências da Computação. Orientador: Prof. Dr. Mauro Roisenberg

Florianópolis, SC, 27 de junho de 2019

BANCA EXAMINADORA

Prof. Dr. Mauro Roisenberg
Universidade Federal de Santa Catarina

Prof. Dr. Elder Rizzon Santos
Universidade Federal de Santa Catarina

Prof. Dr. Alexandre Gonçalves Silva
Universidade Federal de Santa Catarina

Dedico este trabalho ao meu avô Bruno Behling (In Memorian).

AGRADECIMENTOS

Agradeço inicialmente meus pais, Hane Louise e Beno Edson, meu irmão Guilherme, meu padrasto Santiago, que considero como um segundo pai, a minha namorada Letícia e a toda minha família, pelo carinho, paciência e apoio nos mais diversos momentos de todo o tempo na universidade.

Agradeço aos amigos de antes da faculdade e também aos grandes amigos que fiz durante essa jornada, com certeza levarei muitas histórias boas e risadas para a vida inteira.

Ao professor e orientador Mauro Roisenberg, meu muito obrigado, por aceitar me orientar e por me auxiliar durante este trabalho.

Agradeço os demais professores do curso de ciências da computação pelos ensinamentos durante a minha graduação, alguns, dos quais, levarei para a vida toda.

Agradeço a Universidade Federal de Santa Catarina por se tornar uma segunda casa para mim e proporcionar uma incrível experiência.

Muito obrigado a todos.

"Ideias, e somente ideias, podem iluminar a escuridão." - Ludwig von Mises

RESUMO

O reconhecimento de emoções em faces humanas apresenta uma série de utilizações no campo de saúde, para análise comportamental, por exemplo. É uma tarefa altamente desafiadora pois necessita ter um bom desempenho, em questão de tempo e de assertividade, mas sem necessitar de grandes clusters de computação, deste modo, podendo funcionar em pequenos dispositivos. Este trabalho apresenta uma abordagem inicial para o desenvolvimento de um classificador de emoções faciais por meio de frames de um vídeo utilizando redes neurais convolucionais e máquinas de vetores de suporte. A proposta analisa um stream de vídeo e a partir da detecção de um rosto passa este rosto para o classificador de emoções, de modo que alcance uma taxa de assertividade aceitável para as emoções conhecidas. De modo a validar o classificador de emoções faciais, foram geradas matrizes de confusão com a emoção classificada pelo classificador e a real emoção presente no vídeo. Foram obtidos resultados de 80% acurácia para a identificação de emoções.

Palavras-chave: Detecção de emoções, Análise facial, Rede Neural Convolucional, Máquina de Vetores de Suporte

ABSTRACT

The recognition of emotions in human faces presents a series of uses in the field of healthcare, for behavioral analysis, for example. It is a highly challenging task because it needs to perform well, in a matter of time and assertiveness, but without the need for large computing clusters, so that it can run on small devices. This work presents an initial approach for the development of a facial emotion classifier through the frames of a video using convolutional neural networks and support vector machines. The proposal analyzes a video stream and from the detection of a face passes this face to the emotion classifier so that it reaches an acceptable assertiveness rate for the known emotions. In order to validate the facial emotion classifier, matrices of confusion were generated with the emotion classified by the classifier and the real emotion present in the video.

Keywords: Emotion Detection, Facial Analysis, Convolutional Neural Network, Support Vector Machine

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

CNN	Convolutional Neural Network
DNN	Deep Neural Network
MLP	Multilayer Perceptron
RGB	Red Green Blue
RNA	Redes Neurais Artificiais
ROI	Region of Interest
SVM	Support Vector Machine
TCC	Trabalho de Conclusão de Curso
UFSC	Universidade Federal de Santa Catarina

SUMÁRIO

1 **INTRODUÇÃO**

Trabalhos que focam na detecção e identificação de emoções utilizando a face se propõem a fazer um computador ser hábil a identificar o estado emocional de uma pessoa. E este campo vem atraindo muita atenção nos últimos anos devido a seu imenso potencial para ser aplicado em vários campos, tal como robótica (TSAI et al., 2009), medicina (LEON, E. et al, 2005) e até mesmo aplicações investigativas (FAIRHURST, ERBILEK, LI, 2014) .

Segundo (SANCHEZ-MENDOZA et al, 2015), o comportamento dinâmico da face dispõe de uma enorme fonte de informações para a transmissão e caracterização das emoções. Assim como em outras formas de comunicação, muitas informações podem ser inferidas da mensagem original por meio das expressões faciais do seu emissor. Tais características fazem das diversas etapas do reconhecimento facial um problema desafiador e interessante para a pesquisa e a indústria de visão computacional.

Segundo (EKMAN; FRIESEN, 2003) o corpo não tem um movimento específico para o medo ou raiva, porém existem padrões faciais específicos para cada emoção. Se alguém está nervoso, seu corpo pode tentar omitir esta emoção, mas é muito difícil esconder emoções faciais.

Diversas formas e métodos de visão computacional foram abordados por diferentes pesquisadores em suas buscas por uma abordagem que fosse eficiente e permitisse a detecção das emoções faciais.

Dados estes argumentos, o presente trabalho tem como intuito o desenvolvimento de um sistema que seja capaz de reconhecer emoções por meio de redes neurais artificiais, como Redes Neurais Convolucionais e Máquinas de Vetores de Suporte, de forma rápida e eficiente, visando o seu uso em equipamentos com menor poder de processamento, sem depender de grandes clusters para processamento. Assim provendo um modelo pré-treinado para a fase de reconhecimento.

1.1 MOTIVAÇÃO

Com o avanço de poder de processamento das máquinas e dos algoritmos de aprendizado de máquinas cada vez mais complexos e eficientes são possíveis que algumas ações impossíveis, ou muito difíceis, para os seres humanos sejam realizadas por essas poderosas máquinas, por exemplo predição de ondas secundárias de terremotos (WITZE, 2018).

Em vista das grandes possibilidades abertas pela combinação de visão computacional com deep learning, a motivação vem para termos meios de identificar emoções faciais nas pessoas, tendo esta informação é possível reconhecer atitudes suspeitas em meio às massas ou até mesmo prever algumas ações.

Médicos e enfermeiros, por exemplo, devem ser capazes de identificar emoções de pacientes, que podem ser um reação a doença ou tratamento. Conforme (EKMAN; FRIESEN, 2003) pessoas tendem suprimir suas emoções na presença de outros, por vergonha ou por aspectos culturais, logo um sistema que possa identificar emoções quando pacientes estão sozinhos traria grandes benefícios ao profissionais da saúde.

1.2 OBJETIVOS

1.2.1 **OBJETIVOS GERAIS**

O objetivo principal deste trabalho é construir modelos que possam reconhecer emoções faciais por meio do reconhecimento de padrões identificados. Conseguindo, ao final do projeto, ser capaz de identificar e classificar uma face capturado por meio de uma captura de frames de vídeo.

1.2.2 **OBJETIVOS ESPECIFICOS**

Para se chegar ao objetivo geral, serão necessários objetivos específicos, listados a seguir:

- Realizar um estudo sobre os conceitos de Redes Neurais Artificiais e de Deep Learning.
- Compreender o relacionamento das Redes Neurais Artificiais com Deep Learning e realizar estudo sobre as Redes Neurais Artificiais mais utilizadas em reconhecimento de imagens.
- Criar uma aplicação em Python e com OpenCV para o reconhecimento de emoções faciais.
- Realizar testes utilizando diferentes redes neurais tendo emoções faciais como dado de entrada.
- Classificar uma emoção facial podendo essa ser capturada por uma câmera, podendo ser foto ou vídeo.
- Analisar e avaliar os resultados obtidos.

2 **METODOLOGIA DE PESQUISA**

Este trabalho foi elaborado em três etapas: uma revisão da literatura, a criação e adaptação de um banco de imagens de expressões faciais catalogadas em grupos específicos e implementação de redes neurais profundas e redes de máquina de vetores de suporte, utilizando a biblioteca de python Scikit Learn, sendo capaz de reconhecer as emoções expressas desejadas.

2.1 REVISÃO DA LITERATURA

A revisão da literatura foi realizada sobre trabalhos e artigos, acadêmicos ou não, sobre redes neurais artificiais, aprendizado profundo e visão computacional, publicados por diversas universidades e pessoas pelo mundo. A pesquisa bibliográfica também abrangeu documentações das ferramentas utilizadas neste trabalho. As pesquisas foram realizadas nas bibliotecas digitais do ramo de tecnologia e engenharia, das quais a Universidade Federal de Santa Catarina disponibiliza acesso liberado aos alunos para fins acadêmicos, como: Springer, IEEEXplore e ScienceDirect.

2.2 FERRAMENTA

Esse trabalho foi desenvolvido com um processador Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz e uma placa de vídeo dedicada de 4GB GDDR5 NVIDIA(R)GeForce(R)MX150. Para o desenvolvimento foi utilizado python e OpenCV, que juntos trazem a simplicidade da linguagem com o estado da arte em visão computacional e deep learning.

2.3 ESTRUTURA DO TRABALHO

Este trabalho está dividido em seis capítulos além da introdução, sendo eles:

- Fundamentação teórica: Explica conceitos prévios que são importantes para o entendimento deste trabalho;
- Trabalhos correlatos: Apresenta pesquisas desenvolvidas relacionadas a reconhecimento de emoções ou reconhecimento facial;

- Desenvolvimento: Baseado nos conceitos apresentados na fundamentação teórica, são demonstradas como foram montadas as arquiteturas de redes neurais para classificação e a extração de características para a detecção;
- Implementação: Mostra trechos pertinentes a elaboração deste trabalho;
- Experimentos e Resultados: Realiza-se uma análise dos resultados obtidos;
- Conclusão: Mostra-se qual foi a conclusão da pesquisa e a partir desse ponto, quais os possíveis trabalhos futuros.

3 **FUNDAMENTAÇÃO TEORICA**

3.1 REDES NEURAIS ARTIFICIAIS - RNA

Pesquisadores influenciados pelo funcionamento do cérebro humano criaram redes neurais artificiais de modo a simular as conexões realizadas na fase de aprendizagem. Com o objetivo de desenvolver sistemas inteligentes que possam ser capazes de realizar processos de classificação, identificação e reconhecimento de padrões de imagens ou outros conjuntos de dados. Para isso foram desenvolvidos modelos de neurônios artificiais, que foram conectados a outros neurônios artificiais, formando, assim, uma rede neural artificial.

Segundo (RUSSELL; NORVIG, 2016) uma rede neural é composta de um número de nodos, ou unidades, conectadas por ligações, ou arestas. Cada ligação tenho um peso numérico associado a ela. Esses pesos são os meios principais de memória de longo termo em uma rede neural e a aprendizagem acontece atualizando tais pesos. Alguns nodos têm conexões externas, podendo ser nodos de entrada ou saída da rede. Os pesos são atualizados com o objetivo de equilibrar o comportamento da entrada e saída da rede com o ambiente provendo as entradas.

3.2 MULTILAYER PERCEPTRON - MLP

Uma MLP é uma rede neural artificial profunda. É composta por mais de um perceptron ou neuronio. Elas são compostos de uma camada de entrada para receber o sinal, uma camada de saída que toma uma decisão ou previsão sobre a entrada e, entre esses dois, um número arbitrário de camadas ocultas que são o verdadeiro mecanismo computacional da MLP. MLPs com uma camada oculta são capazes de aproximar qualquer função contínua(RUSSELL; NORVIG, 2016).

Os perceptrons multicamadas são frequentemente aplicados a problemas de aprendizado supervisionados, assim, treinam em um conjunto de pares de entrada-saída e aprendem a modelar a correlação entre essas entradas e saídas. O treinamento envolve o ajuste dos parâmetros ou os pesos e vieses do modelo para minimizar o erro. A retropropagação é usada para fazer os ajustes de ponderação e

viés relativos ao erro, e o erro em si pode ser medido de várias maneiras, inclusive pelo erro quadrático médio da raiz (RMSE).

No passo para frente, o fluxo de sinal se move da camada de entrada pelas camadas ocultas para a camada de saída, e a decisão da camada de saída é medida contra o valor esperado.

No passo para trás, usando backpropagation e a regra da cadeia de cálculo, derivadas parciais da função erro, os vários pesos e vieses são propagados de volta através do MLP. Esse ato de diferenciação oferece um gradiente, ou um panorama de erro, ao longo do qual os parâmetros podem ser ajustados à medida que movem o MLP um passo mais perto do erro mínimo. Isso pode ser feito com qualquer algoritmo de otimização baseado em gradiente, como descendente de gradiente estocástico ou Adam.

3.3 REDES NEURAIS CONVOLUCIONAIS - CNN

Redes Neurais Convolucionais, conhecidas como CNN (Convolutional Neural Network), são redes artificiais inspiradas no funcionamento biológico dos nossos neurônios, com capacidade de serem treinadas e aprenderem a reconhecer representações ou padrões, independente de escala, translação, transformação e rotação (JARRETT et al., 2009). A CNN é baseada em um sistema de hierarquia que visa representar a estrutura de reconhecimento de uma imagem, com pixels individuais formam arestas, estas arestas formam padrões, que juntos formam um objeto, que podem descrever cenas.

Conforme (Abdel-Hamid et al., 2012) a CNN é formada de um ou mais pares de camadas de convolução e max-polling. A convolução é composta por um conjunto de filtros, ou operações matemáticas, que são aplicados a um pequeno pedaço da entrada e são replicados ao restante do espaço de entrada. A camada de max-polling, por sua vez, gera uma versão de menor resolução da camada de convolução aplicando um filtro de ativação máxima em diferentes posições dentro de uma janela específica. Isto gera uma invariância de conversão e tolerância a pequenas diferenças de posições de partes do objeto. Camadas mais altas usam filtros mais amplos para entradas de menor resolução para processar partes mais complexas da entrada. Ao fim, a camada totalmente conectada combina as entradas

de todas as posições para fazer a classificação da entrada original. Essa organização hierárquica gera bons resultados no processamento de imagens.

Figura 1 - Representação dos filtros das camadas

Fonte: LEE et al., 2009

3.4 MÁQUINAS DE VETORES DE SUPORTE - SVM

O SVM, em inglês, Support Vector Machine, é baseada e fundamentada na Teoria de Aprendizagem Estatística formulada inicialmente pelo pesquisador russo Vladmir Vapnik (VAPNIK, 2013). Usando o princípio de Minimização do Risco Estrutural para minimizar o erro do conjunto de treinamento, o risco empírico, junto com o erro do conjunto usado para testes.Este princípio surgiu da necessidade de desenvolvimento de limites teóricos para capacidade de generalização dos sistemas de aprendizagem. Geralmente, quanto maior a generalização na fase de treinamento, maior será os índices de acertos na fase de testes.

Uma máquina de vetores de suporte é um algoritmo de aprendizado para classificação e regressão de padrões. O princípio básico de treinamento por trás das SVMs é encontrar o hiperplano linear ótimo, de modo que o erro de classificação esperado para amostras de teste não vistas seja minimizado - isto é, um bom desempenho de generalização. De acordo com o princípio indutivo de minimização de risco estrutural (VAPNIK, 2013, uma função que classifica os dados de treinamento com precisão e que pertence a um conjunto de funções com a menor dimensão irá generalizar melhor independentemente da dimensionalidade do espaço de entrada. Com base nesse princípio, um SVM linear usa uma abordagem sistemática para encontrar uma função linear com a menor dimensão VC. Para dados linearmente não separáveis, os SVMs podem mapear (não linearmente) a entrada para um espaço de característica de alta dimensão onde um hiperplano

linear pode ser encontrado. Embora não haja garantia de que uma solução linear sempre existirá no espaço dimensional alto, na prática é possível encontrar uma solução funcional.

Dado um conjunto rotulado de amostras de treinamento M, um classificador SVM localiza o hiperplano ideal que separa corretamente, desta forma classificando, a maior fração de pontos de dados enquanto maximiza a distância entre as classes do hiperplano (a margem). Vapnik mostra que maximizar a distância da margem é equivalente a minimizar a dimensão VC na construção de um hiperplano ideal. Computar o melhor hiperplano é colocado como um problema de otimização restrito e resolvido usando técnicas de programação quadrática. O hiperplano discriminante é definido pelo conjunto de níveis de

$$f(x) = \sum (i=1 \rightarrow M)\ yi * \propto i * k(x,xi) + b$$

onde k(º,º) é uma função do kernel e o sinal de f(x) determina a associação de x. Construir um hiperplano ideal é equivalente a encontrar todos os ∝i diferente de zero. Qualquer vetor Xi que corresponda a um ∝i diferente de zero é um vetor de suporte (SV) do hiperplano ideal. Uma característica desejável das SVMs é que o número de pontos de treinamento que são retidos como vetores de suporte é geralmente muito pequeno, fornecendo assim um classificador compacto.

Já o princípio de Minimização do Risco Empírico, que é utilizado em algumas técnicas de Redes Neurais, era considerado pelos seus defensores como sendo de fácil intuição, e como apresenta bons resultados práticos, não necessitava de provas teóricas. Porém, Vapnik discordava dessa ideia e apresentava a sua própria visão sobre o tema (VAPNIK, 2013, pag.7) , dizendo que o princípio precisava ser justificado, tendo como o principal objetivo da análise teórica “encontrar um princípio indutivo com nível mais alto de habilidade de generalização e construir algoritmos que implementam esse princípio”.

O treinamento de uma SVM envolve a otimização de uma função quadrática convexa, que é um problema de Otimização Matemática. Como a SVM dispõem de poucos parâmetros livres, que precisam ser ajustados pelo usuário, e não há um dependencia, de forma explicita, na dimensão do espaço de entrada do problema, o

que leva a SVM a poder ser utilizada em problemas com grandes números de entradas.

A técnica pode ser aplicada ao Reconhecimento de Padrões, estimando funções indicadoras, Regressão, estimando funções de valores reais, e também na Extração de Características. Logo, como o problema de reconhecimento de emoções faciais é baseado na aparência, a SVM encaixa-se como um possível solução a este problema.

3.5 VISÃO COMPUTACIONAL

Um dos sentidos mais aplicados por máquinas, com o objetivo de perceber em qual ambiente estão inseridas é a visão (Russel e Norvig, 1995), como este sentido pode ser considerado uma tarefa de representação e processamento de informações, ele é um problema que pode ser computacional. Contudo, mesmo de com anos de pesquisa sobre o funcionamento da visão e o seu processamento no cérebro humano, há ainda alguns aspectos desconhecidos, deste modo, devem ser utilizadas algumas abstrações para simular esse processamento. Segundo (Marr apud. Russel e Norvig) “a visão é o processo de descobrimento através de imagens do que está presente no mundo e onde está.”. Logo, o objetivo primordial é saber a identidade dos objetos ao seu redor e as suas respectivas localizações.

Segundo PUN, GERIG e RATIBO(1994) processo de visão computacional é formado e pode ser dividido em algumas etapas, tais como: a aquisição da imagem, o pré-processamento, a segmentação, uma descrição, o reconhecimento e interpretação da imagem ou objeto.

3.5.1 **Aquisição**

É a etapa em que as informações visuais do meio são convertidas em sinais eletrônicos, por meio de dispositivos(PUN; GERIG; RATIB, 1994, p.5). A qualidade obtida na aquisição das imagens é essencial para as demais etapas.

3.5.2 **Pré-Processamento**

Nesta etapa as imagens que foram obtidas na etapa anterior são preparadas para facilitar o processamento. Caso as imagens apresentem ruídos, falta ou

excesso de luminosidade, podem ser aplicados filtros para adaptar, ou nivelar as imagens, de modo a suavizar efeitos indesejáveis ocasionados por fatores externos, além de também poder realçar alguns detalhes considerados importantes (PUN; GERIG; RATIB, 1994, p.6).

3.5.3 **Segmentação**

É a etapa em que as imagens são divididas em regiões de interesse, ou seja, regiões que contenham informações com possíveis objetos(PUN; GERIG; RATIB, 1994, p.6). A identificação de um objeto pode ser baseada na detecção de descontinuidade ou similaridades na imagem, assim, gerando uma representação abstrata do seu contorno ou região que ocupa. Na segmentação, a identificação de objetos pode ser apontada por regiões semelhantes ou por contornos. As duas estratégias proporcionam formas distintas de representação:

- Representação por Região: É a divisão da imagem em zonas que constituem um objeto. Como se fosse uma matriz com as mesmas dimensões sobre a imagem original, onde cada seção é atribuída um rótulo relativo. Através das seções, busca-se atribuir a cada objeto que possuam mesmas cores, intensidade e textura, ou seja, são similares segundo os critérios estabelecidos previamente.
- Representação de Contornos: Um dado contorno pode ser classificado como um conjunto de pixels de um objeto que o separa do resto da imagem. Uma curva de pontos conectados, dos quais pode-se reconstruir a silhueta do objeto. Essas descontinuidades podem ser obtidas por meio de métodos de detecção de bordas, transformação de Hough, operações de Threshold, entre outras.

3.5.4 **Descrição**

Após o objeto ser segmentado, este será analisado para a extração de suas características mais importantes. Esse conjunto de características é, normalmente, denominado de padrão e será a representação do objeto. As representações podem assumir diferentes formas, dependente da origem das imagens e dos objetivos do processamento(PUN; GERIG; RATIB, 1994, p.6). Se for extraído um conjunto de

algumas características que descrevem o objeto em questão, e esse conjunto passa a ser a representação do objeto, o padrão é chamado de vetor de características.

3.5.5 **Reconhecimento e Interpretação**

Na última etapa do processamento de visão computacional, o padrão encontrado é comparado às classes de padrões já conhecidas, que foram geradas a partir do treinamento, com o objetivo de determinar em qual classe este padrão pertence(PUN; GERIG; RATIB, 1994, p.6). Esta etapa pode ser realizada por meio de funções que dividam o espaço de características em classes específicas. Os classificadores são demonstrados na seção 5.

Desta forma, por meio de segmentações de regiões de interesses, tais regiões podem servir como entradas para serem gerados classificadores, como CNN ou SVM. Assim a partir de classificadores pré-treinados é possível oferecer novos segmentos com o intuíto de obter o reconhecimento da região baseado no segmentos utlizados para o treinamento.

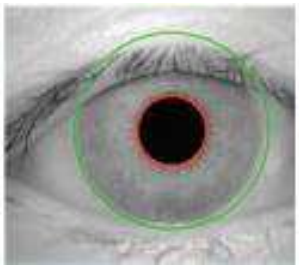
4 **TRABALHOS RELACIONADOS**

A classificação de emoções está recebendo muita atenção dos pesquisadores devido ao seu potencial em muitas áreas de pesquisa, tais como psicologia, estudos sobre o comportamento humano e interface homem-computador. Atualmente, o principal foco é em sistemas que realizam classificação baseada em informações de voz humana (NACHAMAI et al, 2015), combinadas com características extraídas de um rosto humano capturado por uma câmera, como descrito em (SU et al, 2014).

O Facial Action Coding System (FACS), desenvolvido por (EKMAN, 1997) é, atualmente, um dos sistemas mais conhecidos e mais utilizados por analisadores humanos para descrever uma dada atividade facial em termos de ações musculares visualmente observáveis, Ações de Unidades ou AUs. Com o FACS, os observadores humanos apenas dividem as expressões faciais em uma ou mais das 44 Aus que produziram a expressão em questão. Segundo (PANTIC; ROTHKRANTZ, 2003), diversos trabalhos têm aplicado várias técnicas, como Redes Neurais Artificiais, Máquinas de Vetores de Suporte e Redes Bayesianas, de modo a alcançar um sistema eficaz de classificação de emoções.

Em (BOUHABBA et al. 2011) é demonstrada uma abordagem baseada em tempo real para reconhecimento das emoções usando um buscador automático de características faciais para realizar a localização da face e extração de características. As características faciais no vídeo são então utilizadas como entrada para um classificador SVM. O método proposto é avaliado segundo sua acurácia no reconhecimento de uma variedade de cenários de interações. Em (JAGADISWARY; APPASAMI; RAJESH, 2011), é apresentado um método de segmentação de imagens que seja capaz de manusear imagens em condições piores do que as propostas. Primeiramente são utilizadas informações dos elementos dos olhos, e um novo tipo de característica que mede a proporção entre os elementos do olho em cada direção que foi avaliado. Finalmente, após o processamento dos elementos do olho, o processo é viável para aplicações em tempo real.

Figura 2 - Extração de caracteristicas do olho

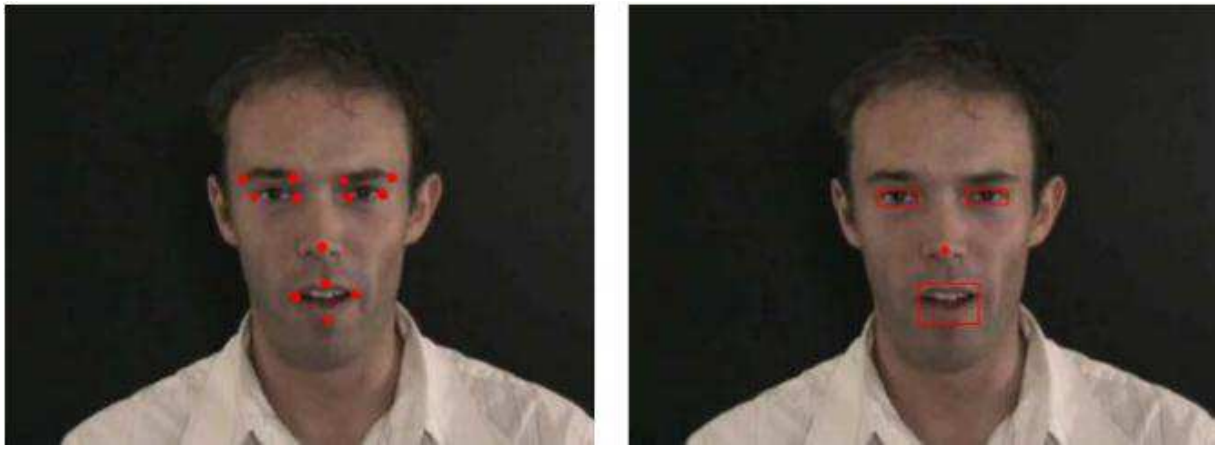


Fonte: JAGADISWARY; APPASAMI; RAJESH, 2011

Na abordagem proposta em em (FAN et al. 2014), os autores focam em detecção de emoções em falas raivosas-neutras, que são frequentes em estudos recentes de Automatic Emotion Variation Detection (AEVD). É proposto neste estudo uma nova estrutura para o AEVD, utilizar Janelas Deslizáveis Multi-escaláveis, ou Multi-scaled Sliding Window (MSW- AEVD) de modo que se possa atribuir uma classe de emoção para cada mudança de janela por decisões de fusões de todas as janelas deslizantes contidas na mudança. Usando a estratégia, eles alcançam uma eficácia de aproximadamente 92%. Porém, nenhuma informação adicional é descrita. Esta informação é crucial para explicar a alta taxa de eficiência.

Em (OH; HANG, 2014), uma região facial é detectada através da combinação do sistema de cores YCbCr com a imagem da Combinação do Gradiente Morfológico Máximo(MMGC, em inglês). A região de pesquisa para a detecção do componente facial é limitada a região da face detectada. Os componentes faciais são detectados usando o método de histograma, o método de marcação de partículas e a imagem MMGC. Neste trabalho, os autores usaram um conjunto de dados de emoção especializado (eNTERFACE'05) para avaliar a metodologia proposta e foi alcançada uma acurácia de aproximadamente 81%.

Figura 3 - Fixação de pontos de interesse do rosto



Fonte: OH; HANG, 2014

Até agora, todos os trabalhos citados vistos na literatura científica resolveram (ou propuseram uma solução definitiva) o problema de reconhecimento de emoções. No entanto, entre os trabalhos existentes, não é proposta uma abordagem inicial, viável e de simples implantação. Desta forma está sendo proposto neste trabalho o uso de uma técnica onde todas as análises são feitas no rosto capturado de um vídeo, baseando-se em uma série de pontos do rosto, tais como, extremidades dos olhos e boca, e não necessitar o uso de nenhuma outra informação, como a voz humana, para realizar o processamento do reconhecimento de emoções.

5 **DESENVOLVIMENTO**

5.1 FERRAMENTAS UTILIZADAS

5.1.1 **OpenCV**

O OpenCV (Open Source Computer Vision Library) foi originalmente, desenvolvida pela Intel, no início dos anos 2000, é uma biblioteca multiplataforma, totalmente livre ao uso acadêmico e comercial, para o desenvolvimento de aplicativos na área de visão computacional, bastando seguir o modelo de licença BSD Intel.

O OpenCV possui módulos de processamento de imagens e vídeo de entrada e saída, estrutura de dados, álgebra linear e vários outros, além de mais de 350 algoritmos de visão computacional e todo o seu processamento é em tempo real.

A biblioteca foi desenvolvida nas linguagens de programação C/C++. Porém, dá suporte às linguagens Java, Python e outras (OpenCV Documentation, 2019).

5.1.2 **Scikit Learn**

O Scikit Learn é uma biblioteca de aprendizado de máquina feita em código aberto para linguagem de programação Python(Scikit Learn Documentation, 2019). Estão incluídos diversos algoritmos de classificação, regressão e agrupamento na biblioteca. O Scikit Learn é caracterizado pela fácil integração com as bibliotecas numéricas de Python, como a NumPy e SciPy, pela API direta e simples e uma extensa documentação com exemplos no seu site.

Devido a API intuitiva o usuário pode facilmente trocar entre os diferentes algoritmos presentes, assim como todos os seus parâmetros.

5.1.3 **Cohn-Kanade Extended**

Para poder treinar um classificador de emoções utilizando redes neurais é necessário previamente ter disponível um conjunto de imagens catalogadas com qual emoção tal imagem representa. Para tal foi utilizado um conjunto estendido de imagens disponibilizado pelos pesquisadores em (LUCEY et al, 2010). Estão

disponíveis 593 sequências de fotos de 123 pessoas, em cada sequência era requisitada uma emoção e foi categorizada segundo a última foto da sequência. Das 593 sequências apenas 327 sequências têm emoções categorizadas.

Para a utilização das imagens neste trabalho foram utilizadas as últimas duas fotos ou apenas a última foto de cada sequência, de forma a isolar as características de cada emoção e não as características que fazem parte do processo de mudança emocional. Ao fim então foram separadas as imagens contendo cada tipo de emoção em um conjunto próprio, deste modo tendo as emoções a seguir e a quantidade disponível para treinamento:

- Desgosto - 59;
- Felicidade - 69;
- Medo - 25;
- Neutro - 593;
- Raiva - 45;
- Surpresa - 83;
- Tédio - 18;
- Tristeza - 28.

A classificação das emoções segue a descrição de (LUCEY et al, 2010), onde cada emoção é o resultado de várias ativações musculares, ou AUs, como visto na tabela abaixo e demonstrado no capítulo de trabalhos relacionados.

Figura 4 - Codificação das emoções pelo movimento dos musculos

AU	Name	N	AU	Name	N	AU	Name	N
1	*Inner Brow Raiser*	173	13	*Cheek Puller*	2	25	*Lips Part*	287
2	*Outer Brow Raiser*	116	14	*Dimpler*	29	26	*Jaw Drop*	48
4	*Brow Lowerer*	191	15	*Lip Corner Depressor*	89	27	*Mouth Stretch*	81
5	*Upper Lip Raiser*	102	16	*Lower Lip Depressor*	24	28	*Lip Suck*	1
6	*Cheek Raiser*	122	17	*Chin Raiser*	196	29	*Jaw Thrust*	1
7	*Lid Tightener*	119	18	*Lip Puckerer*	9	31	*Jaw Clencher*	3
9	*Nose Wrinkler*	74	20	*Lip Stretcher*	77	34	*Cheek Puff*	1
10	*Upper Lip Raiser*	21	21	*Neck Tightener*	3	38	*Nostril Dilator*	29
11	*Nasolabial Deepener*	33	23	*Lip Tightener*	59	39	*Nostril Compressor*	16
12	*Lip Corner Puller*	111	24	*Lip Pressor*	57	43	*Eyes Closed*	9

Table 1. *Frequency of the AUs coded by manual FACS coders on the CK+ database for the peak frames.*

Emotion	Criteria
Angry	AU23 and AU24 must be present in the AU combination
Disgust	Either AU9 or AU10 must be present
Fear	AU combination of AU1+2+4 must be present, unless AU5 is of intensity E then AU4 can be absent
Happy	AU12 must be present
Sadness	Either AU1+4+15 or 11 must be present. An exception is AU6+15
Surprise	Either AU1+2 or 5 must be present and the intensity of AU5 must not be stronger than B
Contempt	AU14 must be present (either unilateral or bilateral)

Fonte: LUCEY et al, 2010

Segundo (EKMAN; FRIESEN, 2003) as 7 emoções primordiais, que estão presentes no escopo do trabalho, podem ser consideradas universais. Diferentes culturas têm as mesmas expressões faciais para as mesma emoções, apenas com algumas confusões em culturas isoladas da Papua Guiné que confundem surpresa e medo, segundo os pesquisadores.

5.1.4 **dlib**

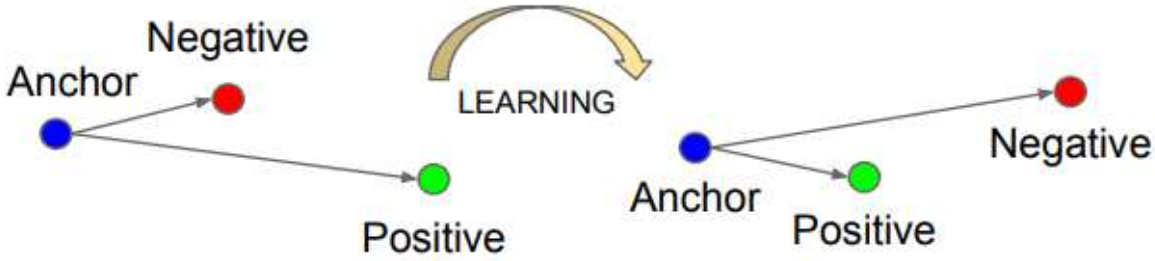
O Dlib é um kit de ferramentas C ++ moderno que contém algoritmos e ferramentas de aprendizado de máquina para criar softwares complexos em C++, com API Bindings para Python, para resolver problemas do mundo real. Ele é usado tanto na indústria quanto na academia em uma ampla gama de domínios, incluindo robótica, dispositivos incorporados, telefones celulares e grandes ambientes de computação de alto desempenho. O licenciamento de código aberto da Dlib permite que você o use em qualquer aplicativo, gratuitamente (dlib Documentation, 2019).

5.1.5 **OpenFace**

O OpenFace é uma implementação feita na linguagem de programação Python e utiliza a biblioteca de IA Torch para realizar o reconhecimento fácil com uma DNN (Deep Neural Network). Esta implementação é baseada no trabalho realizado em (SCHROFF; KALENICHENKO; PHILBIN, 2015) que são engenheiros do Google, onde puderam utilizar grandes clusters para o desenvolvimento dos modelos. Um grande benefício de se utilizar o OpenFace é o fato de ele permitir ser executado tanto em CPUs como também em CUDA.

O OpenFace fornece um vetor 128-D em espaço euclidiano que representa as características extraídas da face analisada. Para a obtenção do melhor vetor é feito o uso da função Triplet Loss, que minimiza a distância entre a âncora e a imagem positiva, ou seja, as duas imagens representam a mesma classe, e maximiza a distância entre a âncora e imagem negativa, que são imagens de classes diferentes.

Figura 5 - Função Triplet Loss do OpenFace



Fonte: SCHROFF; KALENICHENKO; PHILBIN, 2015

Para a extração de características é utilizada uma CNN com Gradiente Estocástico Descendente com backpropagation. Os autores geraram diversos modelos que foram treinados em clusters de 1000 horas até 2000 horas.

5.2 DATA AUGMENTATION

O desempenho de redes neurais de aprendizagem profunda geralmente melhora conforme a quantidade de dados disponíveis vai aumentando (RUSSELL; NORVIG, 2016). Aumento de dados é uma técnica para criar artificialmente novos dados de treinamento a partir de dados de treinamento existentes. Isso é feito

aplicando técnicas específicas de domínio a exemplos dos dados de treinamento que criam exemplos de treinamento novos e diferentes.

O aumento de dados de imagem é talvez o tipo mais conhecido de aumento de dados e envolve a criação de versões transformadas de imagens no conjunto de dados de treinamento que pertencem à mesma classe da imagem original. As transformações incluem uma variedade de operações do campo de manipulação de imagens, como mover para os lados, inversões, zooms, adições de ruídos e filtragens.

A intenção é expandir o conjunto de dados de treinamento com exemplos novos e plausíveis. Isso significa variações das imagens do conjunto de treinamento que provavelmente serão vistas pelo modelo. Por exemplo, um flip horizontal de uma foto de um rosto pode fazer sentido, porque a foto pode ter sido tirada da esquerda ou da direita. Já um flip vertical da foto de um rosto não fará sentido e não é apropriado, dado que é muito improvável que o modelo veja uma foto de um rosto de cabeça para baixo.

Como tal, é claro que a escolha das técnicas específicas de aumento de dados usadas para um conjunto de dados de treinamento deve ser escolhida cuidadosamente e dentro do contexto do conjunto de dados de treinamento e do conhecimento do domínio do problema. Além disso, pode ser útil testar os métodos de aumento de dados isoladamente e em conjunto para ver se eles resultam em uma melhoria mensurável no desempenho do modelo, talvez com um pequeno conjunto de dados, modelo e execução de treinamento de protótipo.

Algoritmos modernos de aprendizagem profunda, como a rede neural convolucional, ou CNN, podem aprender recursos que são invariantes à sua localização na imagem. No entanto, o aumento pode auxiliar ainda mais nessa abordagem de transformação invariante de aprendizado e pode auxiliar o modelo em recursos de aprendizado que também são invariantes para transformações, como ordenação da esquerda para a direita, de cima para baixo, níveis de luz em fotografias e muito mais.

Figura 6 - Visualização do aumento de dados

Fonte: O autor (2019)

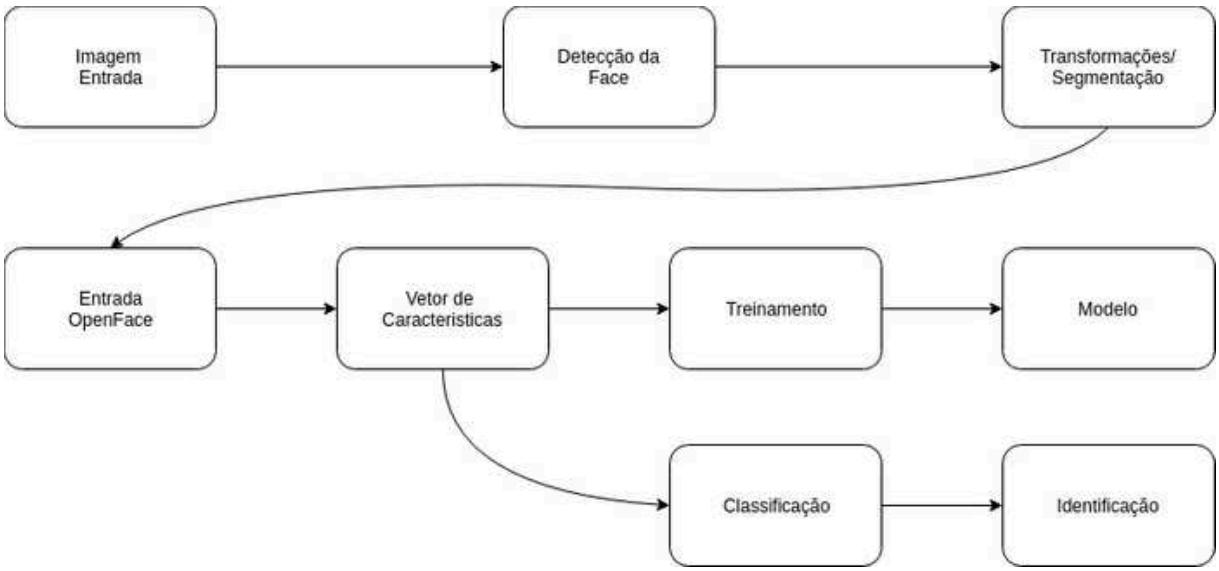
Na primeira imagem da figura anterior está representada a imagem original, a seguir é a imagem rotacionada a um nível aleatório de -25º até +25º. A terceira imagem sofreu adição de ruído e a quarta imagem foi espelhada. A quinta imagem é a combinação da rotação com o espelhamento e a última imagem é a combinação dos 3 efeitos sobre a imagem original.

5.3 FLUXO DE TRABALHO

Para alcançar o reconhecimento facial ou de alguma característica específica são necessários resolver vários problemas relacionados entre si. Primeiro, deve-se encontrar um rosto na imagem, em seguida, deve-se focar individualmente em cada rosto encontrado, mesmo se este estiver rotacionado ou com uma iluminação ruim. O terceiro passo demanda ser capaz de identificar aspectos únicos no rosto que possam diferenciá-los de outros grupos, tais como elevação das sobrancelhas, formato da boca, etc. E finalmente, comparar dados aspectos do rosto com todos os grupos pré-classificados de modo a obter o grupo com o qual este rosto mais se assemelha.

O cérebro humano é capaz de realizar todos estes passos automaticamente e quase instantaneamente. Sendo tão eficientes na identificação de rostos que acabam vendo faces em objetos comuns, segundo (EKMAN; FRIESEN, 2003). Computadores ainda não são capazes de generalizações desse nível. Logo, deve-se ensiná-los a como realizar cada passo deste processo de identificação. Para isso, é necessário um pipeline onde cada etapa será tratada individualmente, pegando o resultado do passo anterior, realizando um trabalho e passando para o passo seguinte.

Figura 7 - Fluxograma do treinamento e reconhecimento



Fonte: O autor (2019)

Os fluxos de treinamento e classificação são bem similares, basicamente seguem os mesmos passos até a obtenção do vetor de características. Após isso, na etapa de treinamento é gerado o modelo de classificação, que é utilizado na versão de classificação para identificar a emoção da face em análise.

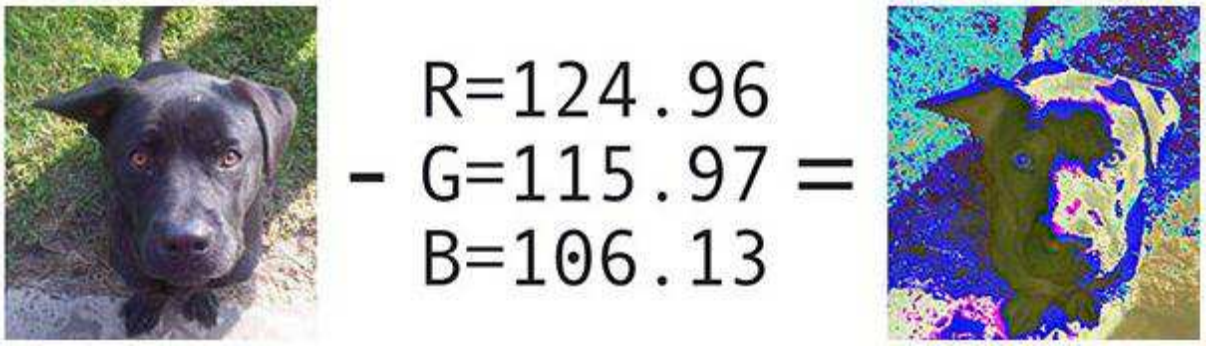
5.4 ENCONTRAR UM ROSTO NA IMAGEM

Para a primeira etapa do processo de processo de identificação do rosto é utilizada uma implementação de uma DNN(Deep Neural Network), ou Rede Neural Profunda, disponível na biblioteca OpenCV, a partir da versão 3.3. Desenvolvida principalmente pelo russo Aleksander Rybnikov, esta rede permite um método de encontrar os rostos na imagem com uma taxa maior de acerto (OpenCV Documentation, 2019)

Para a utilização são necessários dois arquivos, que são o .prototxt, que define a arquitetura do modelo e o .caffemodel, que contém os pesos das camadas do modelo utilizado. Além de ser mais eficiente que outros métodos em questão de performance, essa abordagem se mostra mais favorável, pois foi treinada com mais de 5 milhões de imagens(OpenCV Documentation, 2019), algo que não é facilmente feito.

Após o modelo ser carregado e inicializado por meio da biblioteca Caffe com os arquivos de base é carregada a imagem, ou frame, para realizar as transformações necessárias, como mudança de cores ou redimensionamento. Para isso é utilizado o método cv2.dnn.blobFromImage(), que é onde a imagem é redimensionada para um tamanho 300x300, que é o formato que a rede DNN espera, e também é gerada a versão blob da imagem. A imagem blob é uma versão em que são retirados os valores da tupla de entrada, para amenizar o efeito de mudança de iluminação. Os valores utilizados na tupla foram retirados do famoso conjunto de treinamento ImageNet, e são valores para os canais vermelho, verde e azul, ou RGB. Caso os valores cheguem a número negativos o número resultante é zero.

Figura 8 - Visualização da transformação blob



Fonte: Retirdado de PyImageSearch

A rede DNN recebe então a versão blob da entrada para realizar a detecção de faces e retorna um conjunto de contornos encontrados que possam ser faces juntos com as probabilidades estimadas para cada um. Então cada elemento tem a sua probabilidade estimada comparada com o valor de confiança limite, de modo a excluir falsas detecções, este valor é arbitrário, logo quanto menor o número, maior será a probabilidade de identificação de falso-positivos.

Por fim, apenas para demonstração é desenhado um retângulo que engloba a face encontrada, este passo, porém, não é necessário se não houver necessidade de ver a imagem.

5.5 EXTRAÇÃO DO VETOR DE CARACTERÍSTICAS

Nesta etapa serão gerados os vetores de características das faces que serão utilizados posteriormente para o treinamento e classificação dos grupos de emoções. A partir da região da face na imagem original é gerada outra imagem blob, desta vez apenas da face, não mais da imagem inteira, e a imagem mantém as mesmas cores, apenas alterando questões de tamanho e aspecto.

Logo em seguida a região da face é colocada como entrada do extrator do OpenFace. Ao fim, é gerado um vetor 128-D em espaço euclidiano que descreve a face em questão. Esse é o vetor que é utilizado para a classificação e identificação da emoção. Na etapa de treinamento, esse passo não é realizado quando se busca apenas a identificação, são adicionados o nome da classe de emoção em um vetor e o vetor de classificação em outro, para a parte do realizar o treinamento.

5.6 TREINAMENTO DOS CLASSIFICADORES

Com os vetores da última etapa, o vetor com os vetores de características extraídas das faces e o vetor com a indicação de qual categoria de emoção ela pertence, são treinadas redes com o objetivo de poder reconhecer emoções de pessoas que não participaram da etapa de treinamento.

Para o treinamento dos classificadores é utilizada a biblioteca Scikit Learn desenvolvida para Python. A biblioteca fornece uma API bem intuitiva e abstrai algumas etapas da criação das redes, possibilitando uma experimentação maior e resultados dos testes são obtidos com mais facilidade e menos tempo é gasto na criação de todos as camadas e estruturas dos classificadores, podendo, assim, focar mais nos experimentos e resultados, algo que se mostra muito proveitoso para um trabalho de conclusão de curso.

Então para realizar o treinamento das redes é necessário carregar apenas o resultado da etapa anterior. É inicializado um objeto do tipo LabelEnconder, que é o responsável por enumerar as n classes do conjunto de dados. Em seguida é iniciada a rede desejada para realizar o treinamento.

Quadro 1 - Exemplo de Inicialização dos classificadores

SVM
SVC(C=1.0, kernel="linear", probability=True)

Fonte: O autor (2019)

A API dispõe de diversos parâmetros para os classificadores, que são muito bem explicados por meio da sua documentação, que pode ser acessada pela internet, e conforme dito anteriormente, a abstração da biblioteca facilita muito a utilização dos classificadores. Após a inicialização das redes elas são enfim “treinadas” por meio do método recognizer.fit(facesVectors, labels).

Ao fim desta etapa, dois arquivos são gerados, um deles é o classificador, propriamente dito, e o seu o descritor de classes, que tem suas classes de emoções enumeradas.

5.7 RECONHECIMENTO DE EMOÇÕES

Finalmente com os modelos treinados é possível realizar a identificação das emoções em vídeo. Para isso são realizados quase os mesmos passos utilizados na etapa de treinamento dos modelos. A etapa de reconhecimento é feita sobre frames de vídeo, e não sobre imagens individuais, como no treinamento, logo são necessários outros passos para pode extrair os frames do vídeo. E isto é obtido por meio da biblioteca imutils, que disponibiliza diversas ferramentas para manipulação de dispositivos.

Na inicialização do sistema são passados como parâmetros, conforme os seguintes:

- Detector : O caminho para o detector de faces provido pelo OpenCV. Que é utilizado para encontrar onde estão as possíveis Regiões de Interesse(ROI);
- Embedding-model : O caminho para o extrator de características que gera os vetores que descrevem as faces encontradas e serão utilizadas pela classificação;
- Recognizer : O caminho para o classificador treinado nas etapas anteriores;
- Descritor de classes : caminho para o arquivo que tem as classes enumeradas, dado um certo valor do Classificador, retorna a classe que a face pertence;
- Confiança : O valor de limite mínimo de confiança para detecção das faces.

Após a camera ser inicializada começa o processo reconhecimento, passando todos os frames pelo pipeline (conforme Fluxograma 1). Os frames são transformados no aspecto 300x300, que é a entrada da rede DNN, tem a sua versão blob criada e passam pelo detector DNN para encontrar regiões de interesse. Se tais ROIs têm as características necessárias para seguir o processamento são geradas versões de tamanho 96x96 de cada região sobre a imagem original, o tamanho é o tamanho da entrada da rede OpenFace escolhida dentre as 4 disponíveis.

A região passa então pelo extrator de características e gera o seu vetor de 128-D para ser comparado os grupos de emoções. Esse vetor é então inserido como entrada no classificador previamente treinado. Na saída são dadas as probabilidades da face estar em cada um das classes presentes no classificador, logo a classe com maior probabilidade se torna a saída principal. A saída do

classificador é um número que é decodificado pelo Descritor de Classes, este por sua vez, que retorna qual é o grupo de emoções que aquela face pertence.

Para demonstração visual é gerado então um retângulo que engloba toda a face, é adicionado o valor de probabilidade da classe e o nome da classe a qual essa face pertence. Assim terminando o processo de identificação e reconhecimento da emoção

5.8 DADOS DE ENTRADA

Este trabalho visa o aprendizado supervisionado, desta maneira, tem todos os seus conjunto de dados de entrada propriamente rotulados, com o objetivo de que a saída do sistema seja de um dos conjuntos de dados. Os dados, ou no caso, as imagens, foram separadas em 7 grupos, excluindo o grupo de emoção neutra do dataset, por questões de overfitting, mas a seguir serão apresentadas figuras para demonstrar como essa categoria se comportaria no espaço vetorial.

Os grupos presentes são os seguintes, como o tamanho do conjunto antes e depois de passarem pela fase de aumento de dados, respectivamente:

- Desgosto - 59/178;
- Felicidade - 69/208;
- Medo - 25/76;
- Raiva - 45/136;
- Surpresa - 83/250;
- Tédio - 18/55;
- Tristeza - 28/85.

5.9 PARÂMETROS DAS REDES

Os parâmetros utilizados na criação das redes estão descritos a seguir. Foram testados diversos valores para os parâmetros, sendo que alguns exemplos são citados a seguir, a rede escolhida foi a que convergiu mais rapidamente.

5.9.1 **Parâmetros MLP**

- Hidden_layer_sizes : É uma tupla que representa o tamanho e o número de camadas intermediárias (Scikit Learn Documentation, 2019)

- Activation : A função de ativação para as camadas intermediárias.
- solver : Função que atualiza os pesos e os vieses da rede de forma a calibrar a rede a cada iteração, levando em conta a sua taxa de aprendizado e o seu gradiente.
- Learning_rate : O algoritmo a ser utilizado que tem como função atualizar a taxa de aprendizagem inicial (learning_rate_init). Nos testes o algoritmo que melhor se saiu foi o adaptive que divide a taxa atual por 5 a cada vez que a taxa de perda não avança por mais de duas rodadas seguidas.
- Learning_rate_init : A taxa de aprendizagem inicial é o quanto os pesos da rede podem ser atualizados a cada rodada.
- Tol : A taxa de tolerância para a otimização da rede. Quando a perda não diminui por pelo menos este valor, entra em ação o algoritmo adaptive.

5.9.2 **Parâmetros SVM**

- C : O parâmetro C informa à otimização do SVM o quanto evitar de classificar erroneamente cada exemplo de treinamento. Para valores grandes de C, a otimização escolherá um hiperplano de margem menor, mesmo se esse hiperplano fizer um trabalho melhor ao classificar todos os pontos de treinamento corretamente. Por outro lado, um valor muito pequeno de C fará com que o otimizador procure por um hiperplano com separação de margem maior, mesmo se esse hiperplano classificar incorretamente mais pontos.
- Kernel : Determina qual estilo de kernel será utilizado no algoritmo. Nos testes, o melhor kernel foi o RBF (Radial basis function), pois permite a criação de contornos entre as classes, ao invés de seguir um modelo linear, por exemplo.
- Tol : A taxa de tolerância para a otimização da rede.

6 **IMPLEMENTAÇÃO**

Neste capítulo são explicadas e demonstradas como foram feitas as implementações, assim como os parâmetros utilizados.

O primeiro algoritmo é responsável pela detecção das faces na imagem, que posteriormente serão utilizadas no treinamento ou identificação. Inicialmente é carregada cada imagem na memória, no caso do reconhecimento apenas muda esta etapa, pois é recebido o frame que já está na memória, por meio do método VideoStream.read().

É feito então um redimensionamento da imagem com largura 600 pixels para poder padronizar o tamanho das imagens e após são adquiridas as novas dimensões da imagem. A imagem passa pela transformação blob com o objetivo de realçar as diferenças e é transformada para o tamanho 300x300, que é a entrada esperada pelo identificador de faces do OpenCV. Ao final são retornados os pontos da face, as dimensões, a imagem e o nome, que será usado para treinamento apenas.

O segundo algoritmo é o responsável por validar as faces e extrair os respectivos vetores de características. No caso do treinamento sempre haverá apenas uma face por detecção, logo, quando for a etapa de reconhecimento é necessário passar por um loop do número de faces detectadas. Se a confiança obtida pelo detector de faces for maior que o threshold estabelecido o processamento continuará, o valor utilizado for 50%. São então, obtidos os pontos que englobam a face, e se as dimensões são acima do previamente estabelecidos, e gera-se uma versão 96x96 da imagem para o extrator do OpenFace. Ao fim, são adicionados o vetor de características e o respectivo nome ao vetores que serão usados no treinamento.

Algoritmo 1: Deteção da Face

```
def DetectFace(image):
    name = image.split(os.path.sep)[-2]
    image = cv2.imread(image)
    image = imutils.resize(image, width=600)
    h = image.shape[0]
    w = image.shape[1]
   image_blob = cv2.dnn.blobFromImage(cv2.resize(image, (300, 300)), 1.0,
(300, 300), (124.96, 115.97, 106.13), swapRB=False, crop=False)
    detector.setInput(image_blob)
    detection = detector.forward()
    return detection, h, w, image, name
```

Algoritmo 2: Validação da Face

```
def FaceValidation(detection, h, w, image, name):
    i = np.argmax(detection[0, 0, :, 2])
    confidence = detection[0, 0, i, 2]
    if confidence > confidenceDefault:
       x1 = int(detection[0, 0, i, 3] * w)
       y1 = int(detection[0, 0, i, 4] * h)
      x2 = int(detection[0, 0, i, 5] * w)
      y2 = int(detection[0, 0, i, 6] * h)
      face = image[y1:y2, x1:x2]
      faceH = face.shape[0]
      faceW = face.shape[1]
      if faceW < 20 or faceH < 20:
         return
     face_blob = cv2.dnn.blobFromImage(face, 1.0 / 255, (96, 96), (0, 0, 0),
swapRB=True, crop=False)
      extractor.setInput(face_blob)
      vec = extractor.forward()
```

```
    facesLabels.append(name)
    facesVectors.append(vec.flatten())
```

O terceiro algoritmo é encarregado de treinar as redes. As criações seguem os parâmetros descritos demonstrados no capítulo anterior. Graças a abstração oferecida pela biblioteca scikit learn a alteração dos parâmetros é simples, algo que ajuda muito nos testes em busca dos melhores modelos para os conjuntos de dados. O algoritmo inicializa a rede desejada e treina a rede com o vetor global que contém todos os vetores de características extraídas e com os labels, que são os números de cada uma das classes.

Algoritmo 3: Treinamento da Rede

```
def Train(network):
    labelEnc = LabelEncoder()
    labels =  labelEnc.fit_transform(facesLabels)
    recognizer = None
    // método desejado
    // provido pelo scikit

    recognizer.fit(facesVectors, labels)
    return labelEnc, recognizer
```

O quarto algoritmo é o responsável pelo reconhecimento e identificação da emoção. Para isto, é gerada a imagem 96x96, como explicado anteriormente, e passa-se o vetor de características para o classificador, que retorna a probabilidade e o número da classe a qual a emoção pertence. Logo após então é transformado o número da classe no nome da classe e o processamento de reconhecimento da emoção está terminado.

Algoritmo 4: Reconhecimento da Emoção

```
face_blob = cv2.dnn.blobFromImage(face, 1.0 / 255, (96, 96), (0, 0, 0), swapRB=True, crop=False)
embedder.setInput(face_blob)
vec = embedder.forward()
prediction = recognizer.predict_proba(vec)[0]
i = np.argmax(prediction)
probabilidade = prediction[i]
name = le.classes_[i]
```

---

7 **EXPERIMENTOS E RESULTADOS**

7.1 TREINAMENTO DAS REDES

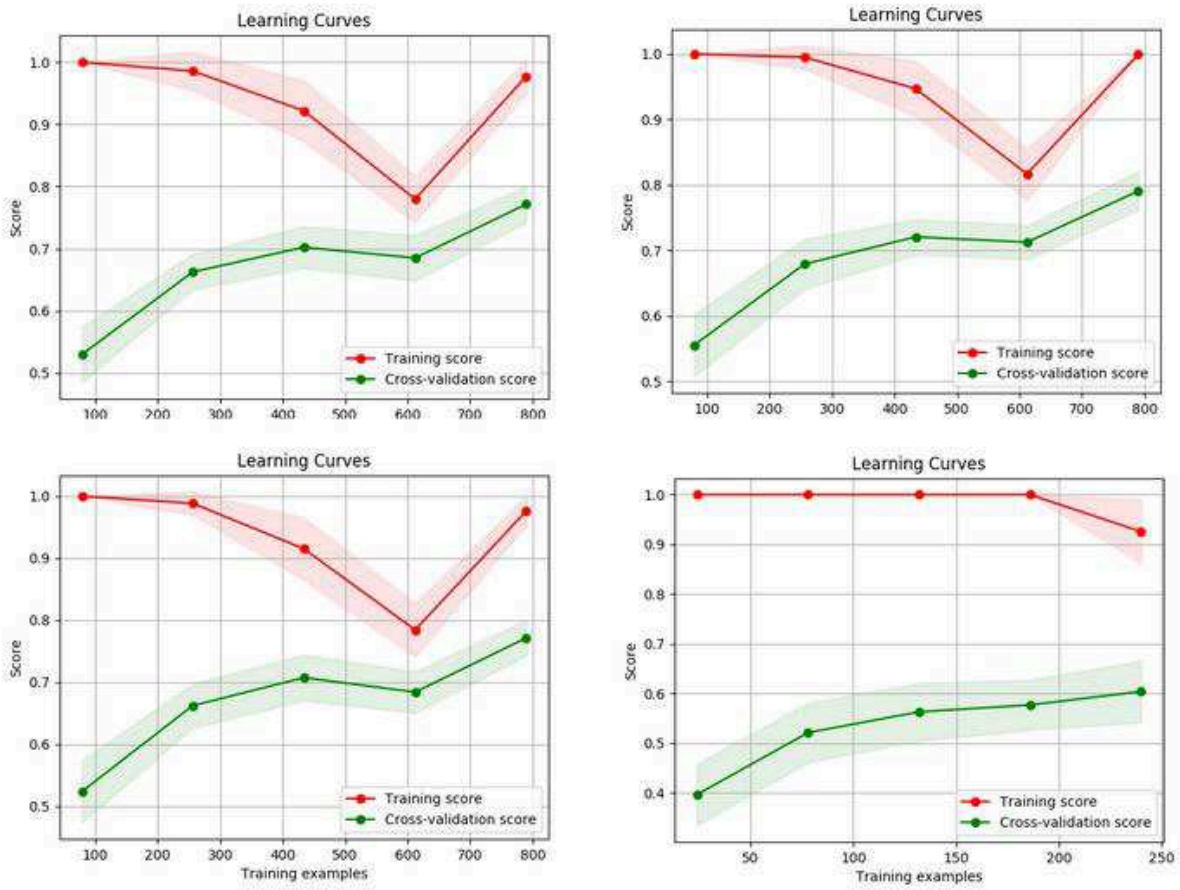
Foram testadas diversas configurações de redes com objetivo de encontrar a rede que obtém o melhor desempenho para o conjunto de dados disponível. Abaixo estão as configurações das 3 melhores redes e a seguir os seus ritmos de aprendizagem e validação, para efeito de comparação foi adicionada uma imagem do treinamento sem aumento de dados, onde pode-se ver que a rede tem menos eficiência.

Figura 9 - Algumas das configurações de testes de MLP

A	B	C
Camada de Entrada da Rede		
128	64	64
ReLu		
SGD	Adam	Adam
64	32	32
ReLu		
SGD	Adam	Adam
	16	
	ReLu	
	Adam	
Camada de Saída		
Classificação		

Fonte: O autor (2019)

Figura 10 - Desempenho MLP com validação de amostra



Fonte: O autor (2019)

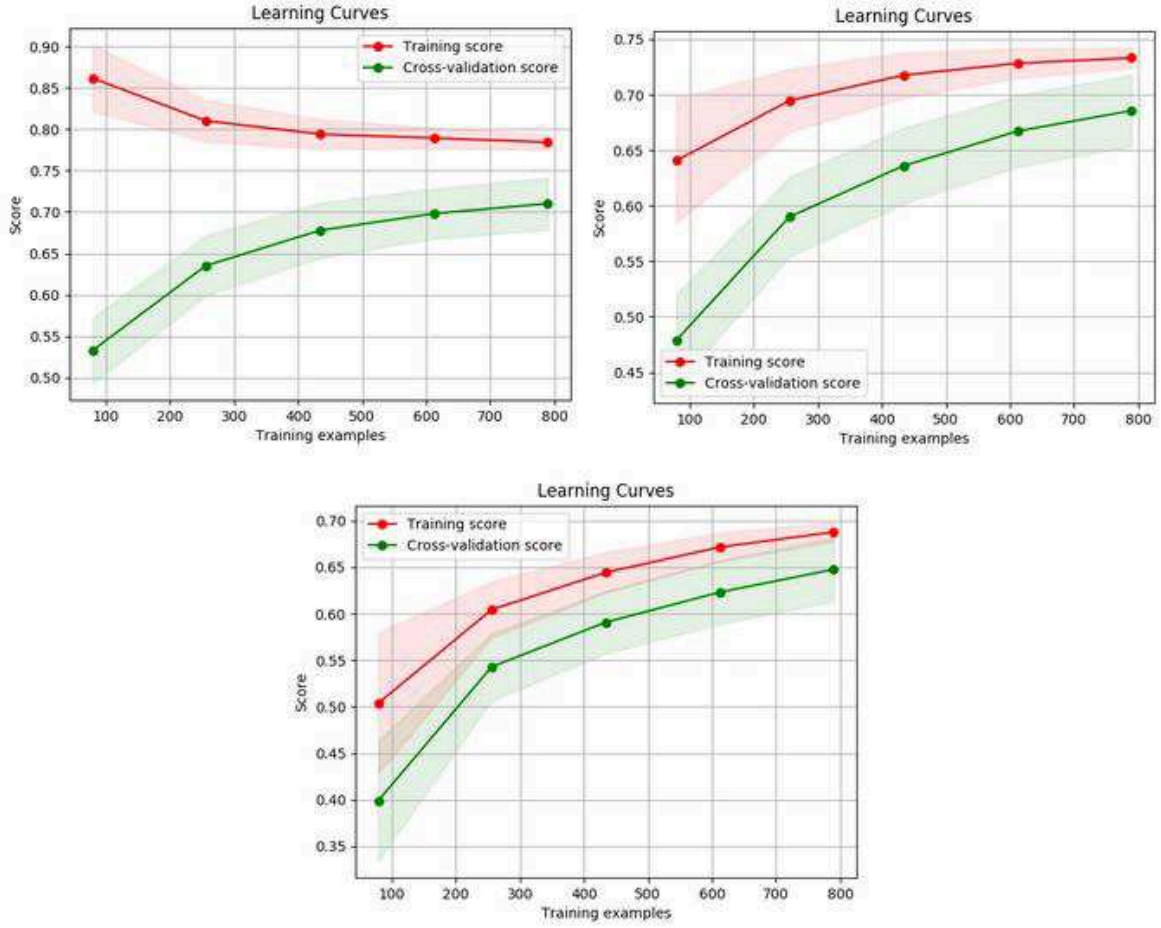
A imagem anterior representa a média e o desvio padrão dos ritmos de aprendizado das redes MLPs demonstradas no Quadro 2, isso com 30 treinamentos em cada uma. Para efeitos de comparação foi a adicionada uma rede que foi treinada em conjunto de dados sem o aumento de dados, podendo-se observar que o aumento de dados trouxe uma melhora para o treinamento. Para validação é selecionada uma amostra dos vetores para gerar um *score* de treinamento. A rede que obteve o melhor resultado foi a B, mas não muito distantes das outras.

Figura 11 - Configurações de testes de SVM

A	B	C
RBF	Linear	Poly
C = 1.0	C = 4.0	C = 1.0
Gamma = 0.1	Gamma = 0.2	Gamma = 0.2

Fonte: O autor (2019)

Figura 12 - Desempenho SVM com validação de amostra



Fonte: O autor (2019)

Assim como na imagem de desempenho das MLPs, esta última imagem demonstra a média e os desvios padrões, mas agora de 100 treinamentos de cada configuração das redes SVM. A rede com o melhor desempenho foi a primeira, a rede A.

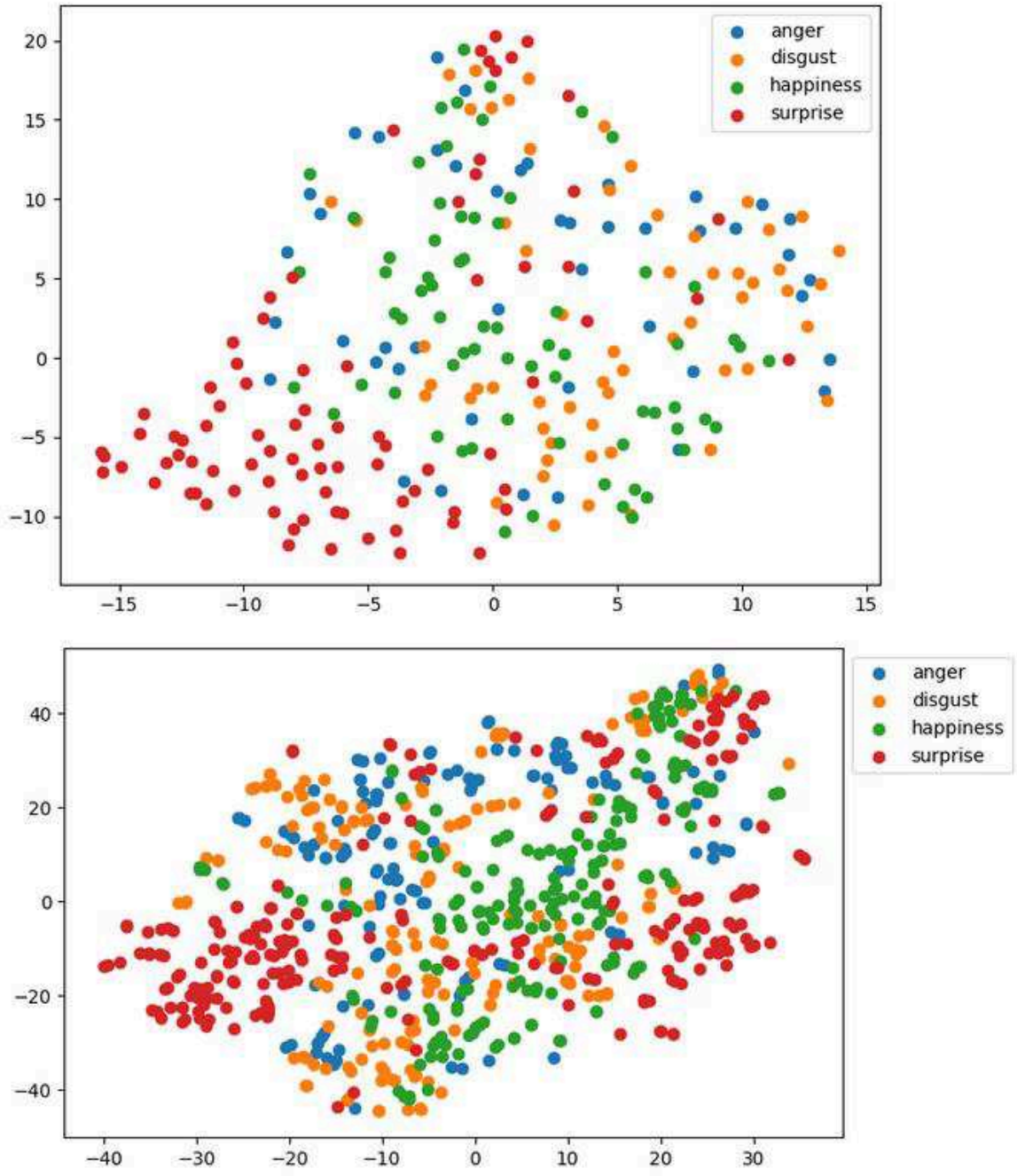
7.2 ESPAÇOS VETORIAIS DAS CARACTERÍSTICAS

Para se obter uma melhor compreensão de como as diferentes emoções podem ter suas características não muito distantes de outras emoções, foram geradas algumas visualizações de como ficaram os espaços vetoriais em 2D. A seguir estão exemplos gerados a partir de conjunto de dados sem aumento de dados e com aumento de dados para fim de comparação e verificar que com uma massa de dados maior, pode-se observar algumas divisões mais nítidas entre as emoções.

Para gerar tais gráficos foi utilizado o método de redução de dimensionalidade t-SNE, em inglês, t-Distributed Stochastic Neighbor Embedding. O t-SNE minimiza as divergências de duas distribuições, uma distribuição que representa as similaridades de pares dos pontos de entrada e outra distribuição que representa as similaridades de pares de pontos correspondentes em um espaço de baixa dimensionalidade.

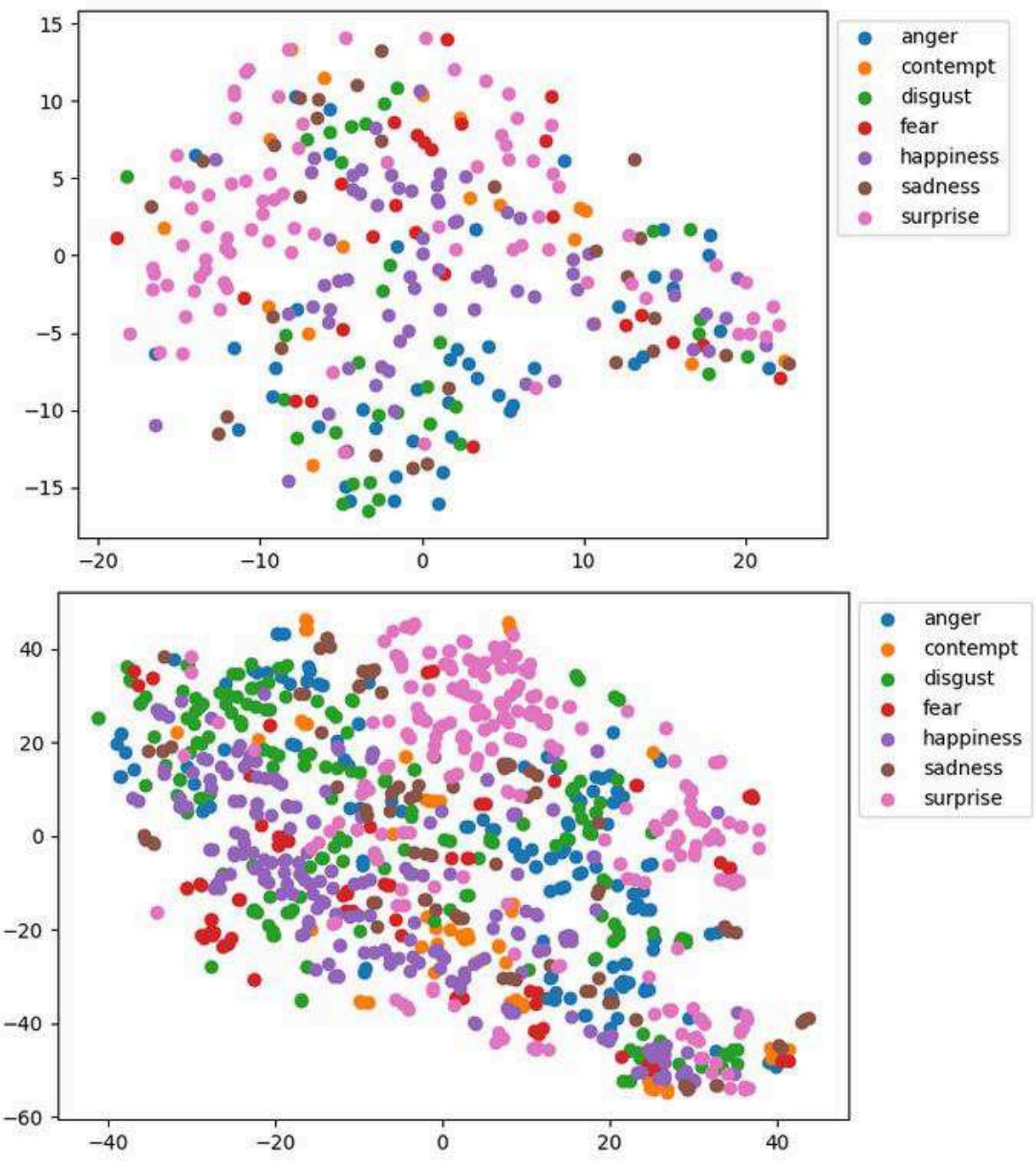
Desta maneira, o t-SNE mapeia dados multi-dimensionais para um espaço de 2 ou 3 dimensões e procura por padrões nos dados observando clusters de pontos. Porém, as características não são mais identificáveis, desta forma, os gráficos a seguir são apenas para se obter uma ideia do quão esparsas ou densas são as emoções.

Figura 13 - Raiva, Desgosto, Felicidade e Surpresa.



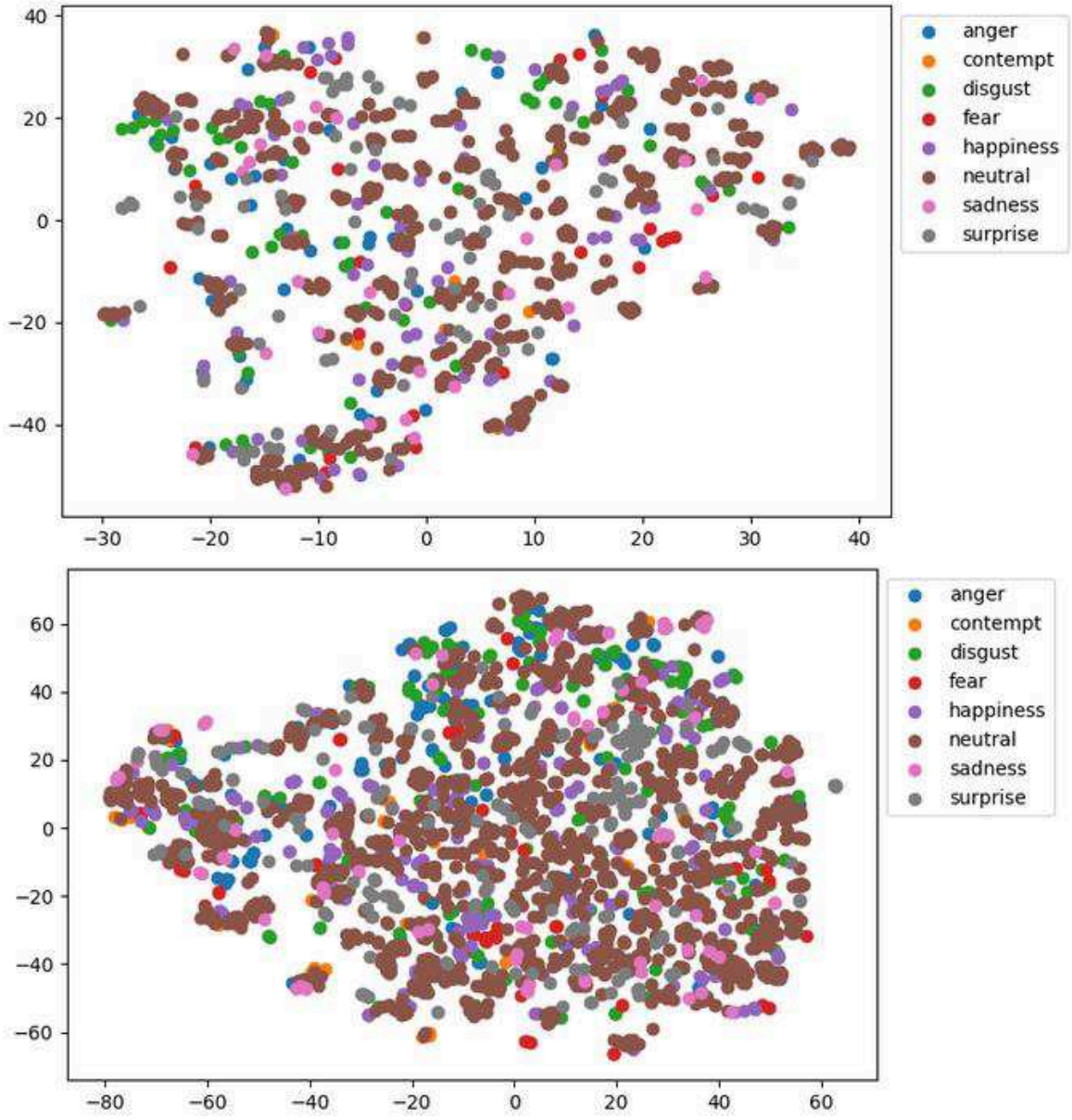
Fonte: O autor (2019)

Figura 14 - Todas as 7 emoções, sem neutro



Fonte: O autor (2019)

Figura 15 - Todas as 7 emoções com neutro



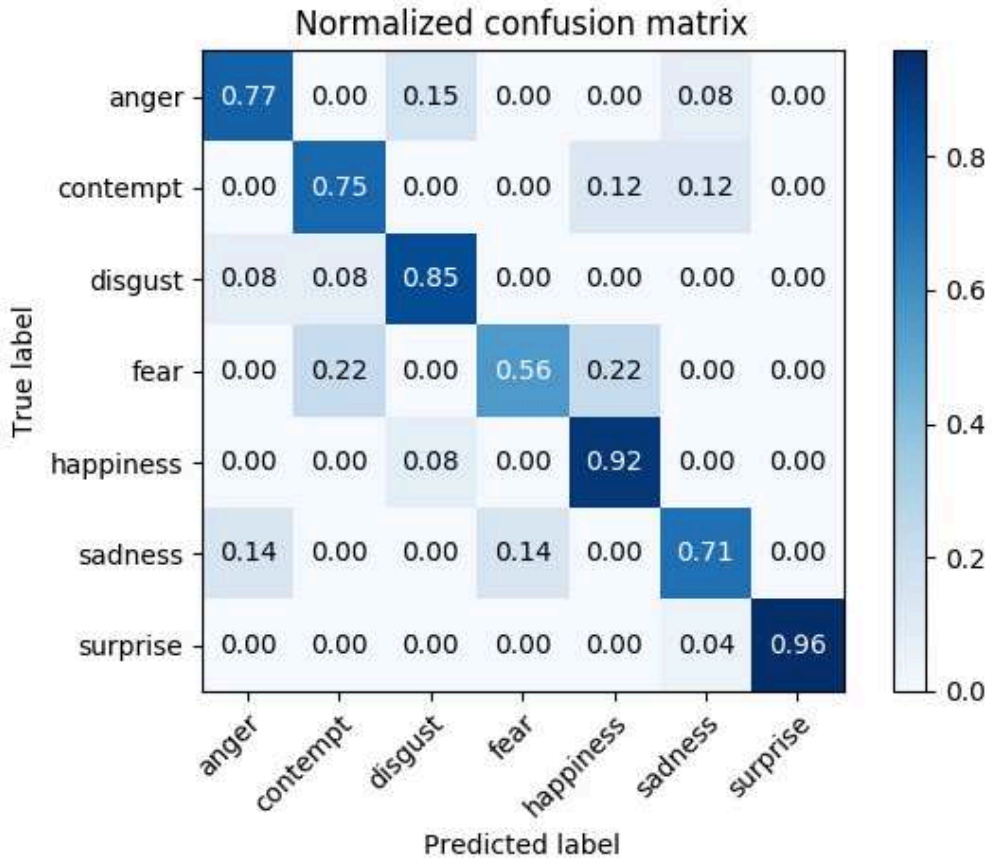
Fonte: O autor (2019)

É possível observar que o aumento do número de imagens no conjunto trouxe uma melhoria na separação das classes e desempenho na identificação. Como pode ser visto na última imagem, a emoção neutra é muito esparsa, devido a isso ela foi retirada, mesmo se fossem escolhidas imagens aleatórias da emoção neutra, a fim de equilibrar com a quantidade das demais emoções, não seria obtida muita melhora, devido, novamente, a sua dispersão.

7.3 CLASSIFICAÇÃO

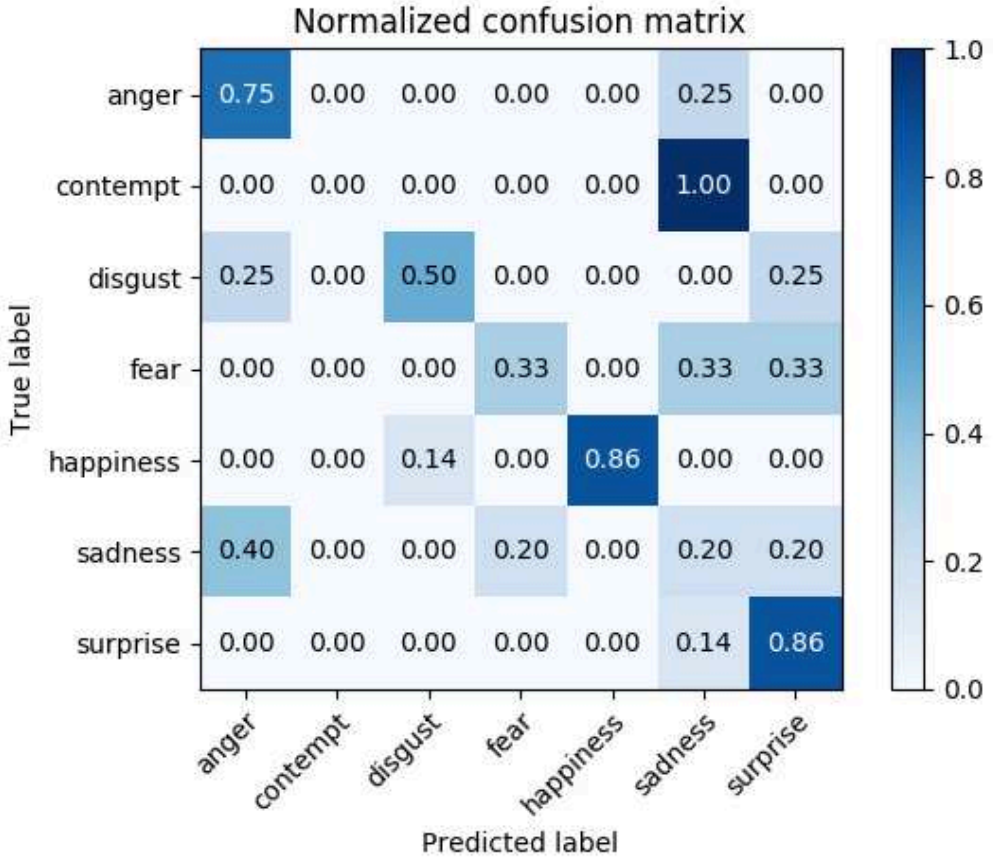
A partir dos modelos treinados são geradas matrizes de confusão, que como objetivo obter uma visualização dos resultados da classificação da rede. . No eixo horizontal estão as classes esperadas pelo classificador e no eixo vertical está a saída do classificador. Com as respostas é possível calcular a razão entre acertos e erros do classificador e verificar quais as classes que mais se distanciam das demais e quais classes não obtiveram um bom resultado.

Figura 16 - Matriz confusão da MLP B



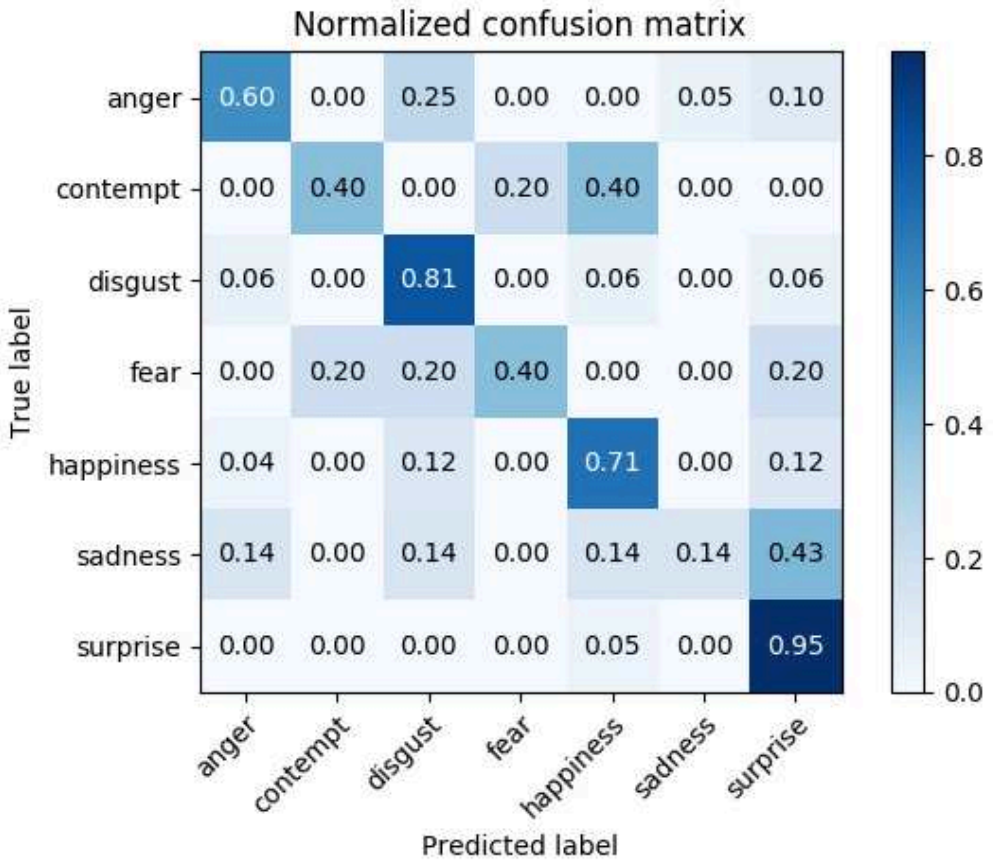
Fonte: O autor (2019)

Figura 17 - Matriz confusão MLP B sem aumento de dados



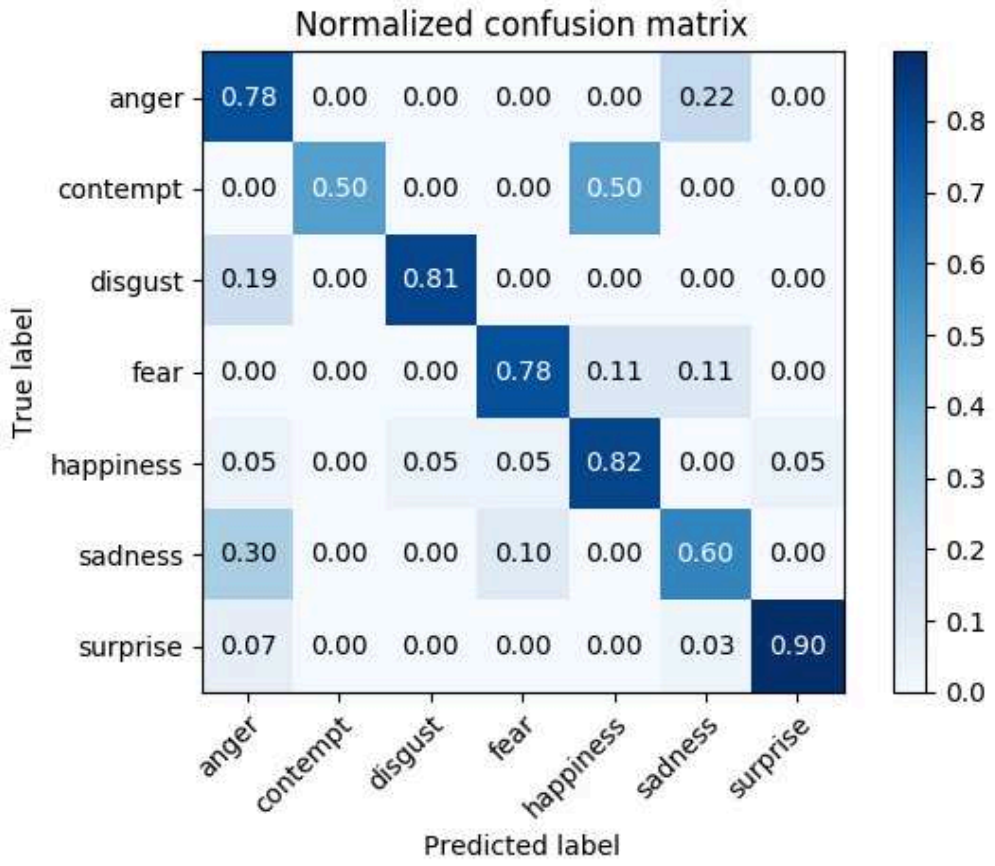
Fonte: O autor (2019)

Figura 18 - Matriz confusão da SVM A



Fonte: O autor (2019)

Figura 19 - Matriz confusão da combinação dos dois classificadores



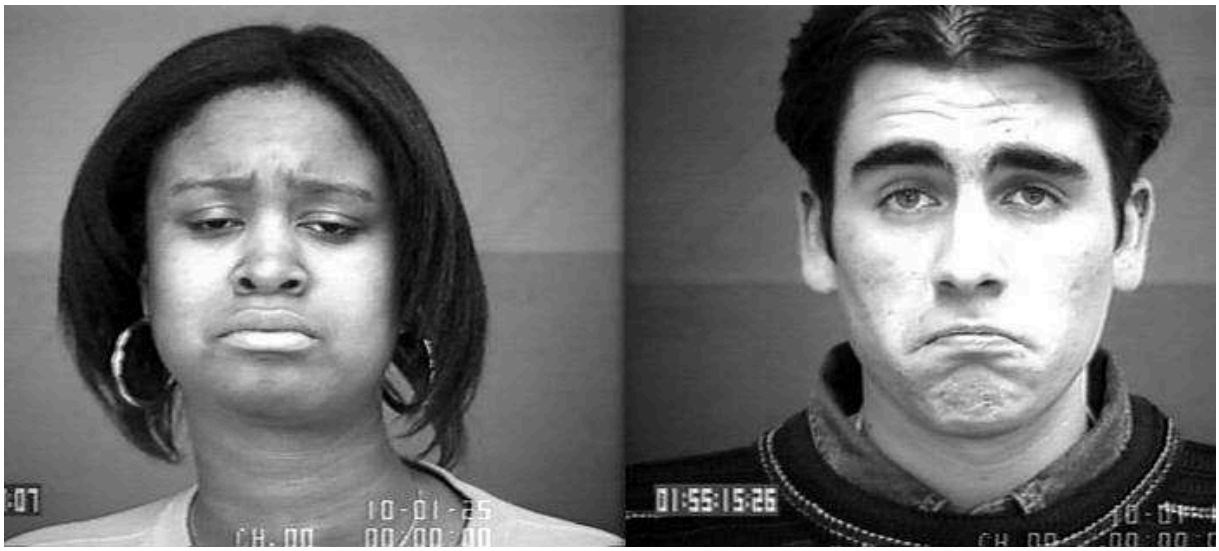
Fonte: O autor (2019)

Com as matrizes, junto com o espaço vetorial, é possível verificar que algumas emoções possuem um grande distinção, como a felicidade, já a tristeza não obteve um bom resultado. Isto pode ser comparado com os gráficos do espaço vetorial das características. Onde as emoções que se aglomeram em clusters obtiveram os melhores resultados na matriz confusão, diferentemente das emoções mais esparsas no gráfico, como a tristeza, que aparece muito espalhada.

A terceira matriz confusão é combinação entre a rede MLP e a SVM. Considera-se a resposta da rede que atribui maior grau de certeza. Como pode se observar, em algumas emoções houve uma melhora em comparação a uma só rede, mas em outros houve uma piora, pois as vezes uma rede pode atribuir uma certeza muito alta para a resposta errada, geralmente no SVM.

Como pode ser visto no exemplo a seguir, as duas imagens são da emoção tristeza, mas apresentam um visual bem distinto.

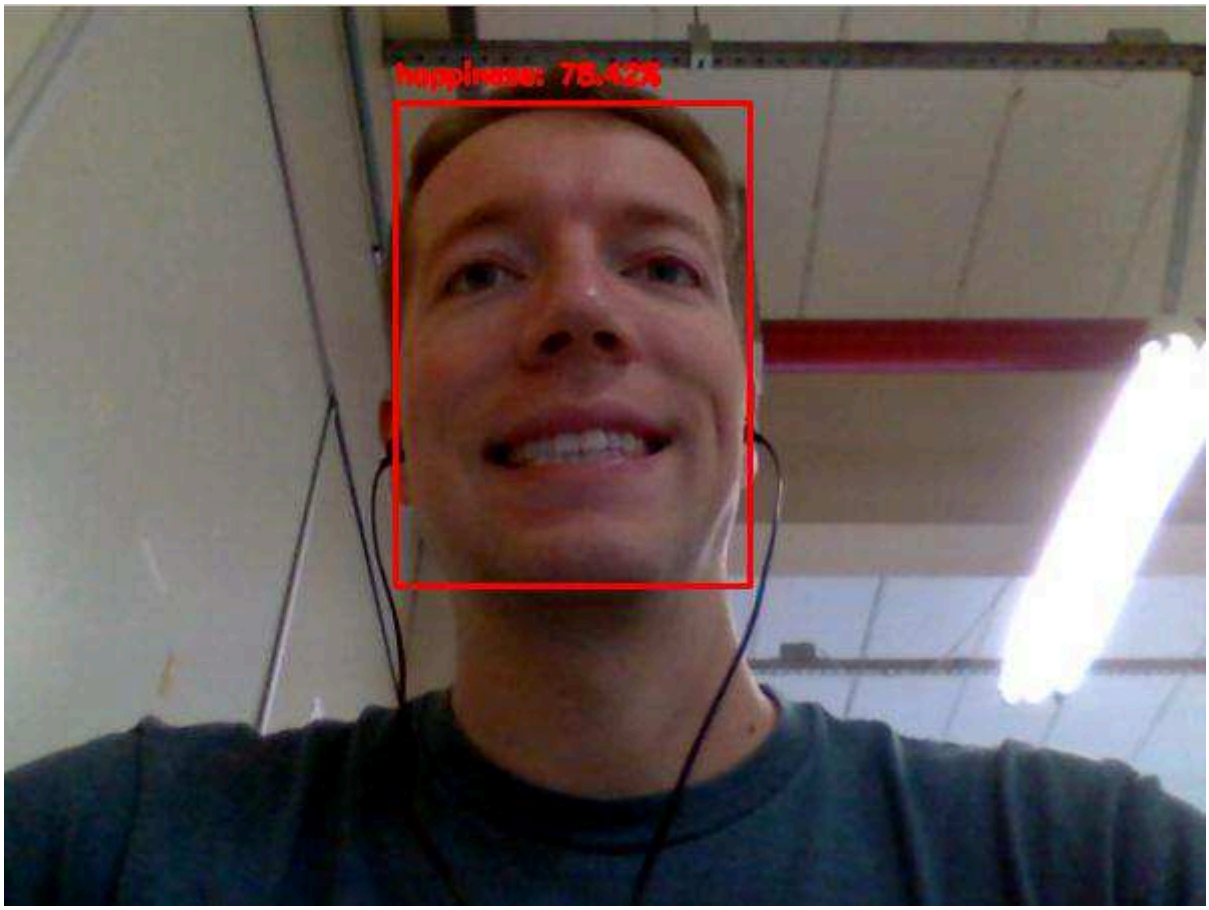
Figura 20 - Comparação imagens rotulados como tristeza



Fonte: Adapatado pelo autor (2019)

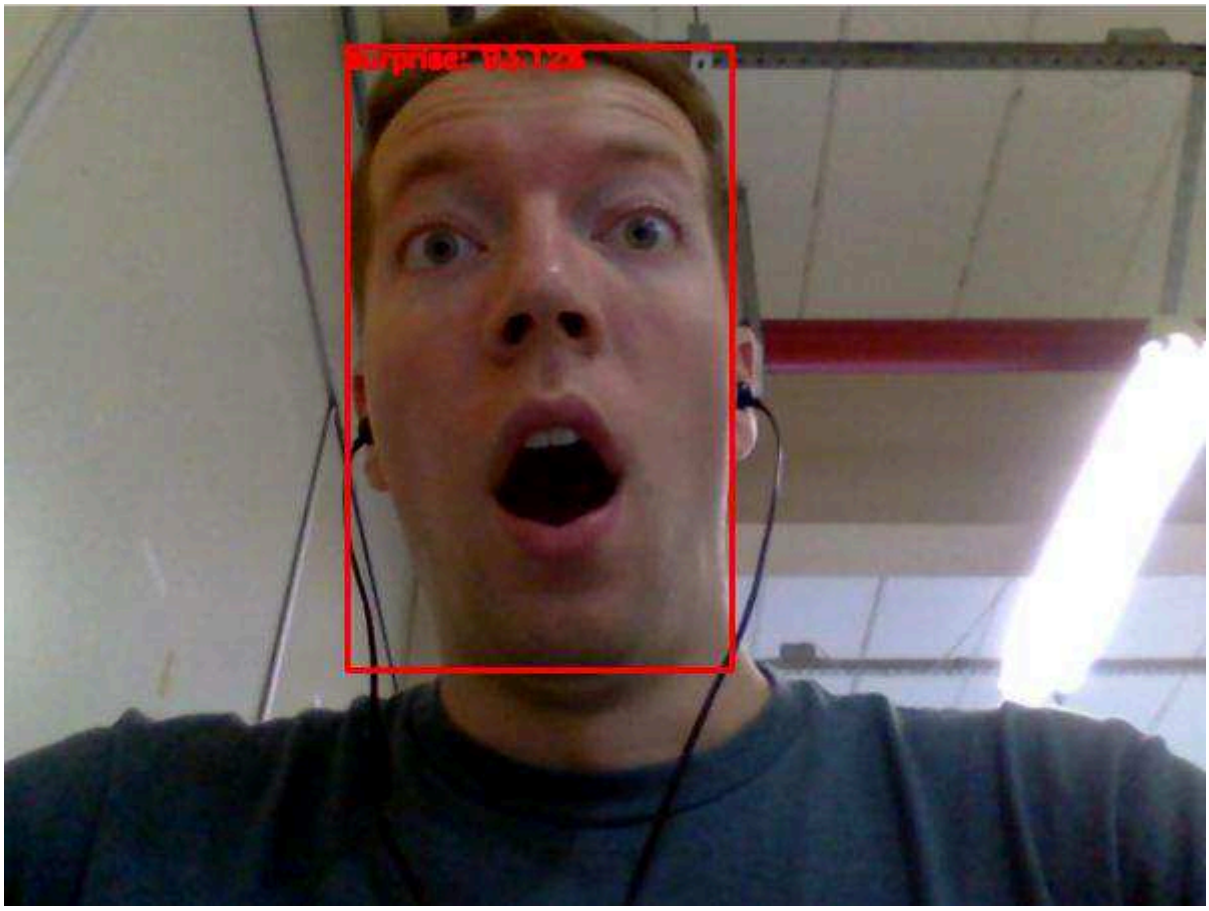
Além de testes com o próprio dataset de treinamento, foram realizados testes com a câmera do computador, os quais retornam resultados razoáveis, com taxas de acertos acima de 65%. Para este teste foram geradas 3 imagens do autor em cada uma das 7 emoções e identificadas pela rede.

Figura 21 - Teste de reconhecimento de felicidade



Fonte: O autor (2019)

Figura 22 - Teste de reconhecimento de surpresa



Fonte: O autor (2019)

8 **CONCLUSÃO**

Este trabalho teve como objetivo analisar o desempenho de redes de aprendizado supervisionado para a classificação de imagens individuais ou frames de vídeos de entrada em diferentes classes de emoções previamente treinadas.

Com o objetivo de fornecer um embasamento e indícios da necessidade da ferramenta desenvolvida, foi realizada uma pesquisa na literatura com base nas implementações de reconhecimento de emoções por meio das mais diversas técnicas.

Com base na leitura de trabalhos que abrangem a classificação de emoções, tanto por meio de imagens ou outros tipos de dados, ficou clara a dificuldade de se obter conjunto de dados de tamanho satisfatórios, previamente rotulados por completo ou parcialmente, devido a questões de direitos de imagens das pessoas envolvidas. Revisando a literatura foram visualizadas e demonstradas as principais características das redes SVM e MLP, que são os tipos mais utilizados nos trabalhos estudos, obtendo-se assim os pontos fracos e fortes de cada uma..

Durante as etapas de desenvolvimento, implementação e testes do trabalho foi notado que é de extrema importância possuir equipamento que tenha um certo poder de processamento. Pois, mesmo com bibliotecas otimizadas do OpenCV e scikit learn, alguns passos podem levar horas para serem concluídos. Existem ferramentas gratuitas, como o Google Cloud, que podem ser utilizadas para etapas de treinamento que demandem muito tempo em equipamentos com menos poder computacional.

O uso do detector de faces do OpenCV e do extrator de características do OpenFace, trouxeram bons resultados. Um dos motivos é o fato que as duas ferramentas usaram enormes conjuntos de dados e grandes clusters para os seus respectivos treinamentos. Sendo que o classificador do OpenCV já vem junto com a biblioteca, assim sendo de fácil acesso. Porém o extrator de características requer algumas bibliotecas adicionais e deve ser obtido individualmente. Podendo-se assim, este trabalho, focar na etapa de treinamento e classificação das emoções, assim como também, na etapa de preparação das imagens e na obtenção e interpretação dos resultados.

Os dois tipos de aprendizados de máquinas supervisionados testados, MLP e SVM, mostraram seus pontos positivos e negativos de cada uma. As redes MLPs obtiveram melhores métricas de assertividade na classificação das emoções, porém podem levar muito mais tempo para serem treinadas, algumas das redes levaram mais de 2 horas para um conjunto de menos de 3000 imagens, e dependem de um conjunto de dados maior para obter um desempenho satisfatório. Já as redes SVM, se mostraram muito mais eficientes para o treinamento, sendo geradas com bastante velocidade. Porém, conforme o tamanho do conjunto de dados é maior e, no caso do trabalho, a dispersão de algumas classes cresce, a rede tem dificuldades para conseguir gerar hiperplanos que gerem menos erros, ou seja, que englobam o menor número de dados não pertencentes a esta classe.

8.1 TRABALHOS FUTUROS

Um dos maiores problemas enfrentados neste trabalho foi o pequeno número de imagens disponíveis para os treinamentos, deste modo, futuramente deve-se aumentar o número de imagens de emoções, até mesmo de outras emoções. Segundo (EKMAN; FRIESEN, 2003), a processo de identificação de emoções realizado no cérebro analisa os últimos microssegundos da mudança nos músculos faciais, assim seria interessante um trabalho que levasse em conta mais de um frame para uma única identificação.

Mesmo realizando diversos testes com vários parâmetros diferentes nos dois tipos de rede, deve-se testar outros tipos de redes e outros extratores de características.

REFERÊNCIAS

ABDEL-HAMID, Ossama et al. Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition. In: **2012 IEEE international conference on Acoustics, speech and signal processing (ICASSP)**. IEEE, 2012. p. 4277-4280.

BOUHABBA, E. M.; SHAFIE, A. A.; AKMELIAWATI, R. Support vector machine for face emotion detection on real time basis. In: **2011 4th International Conference on Mechatronics (ICOM)**. IEEE, 2011. p. 1-6.

Deep learning: How OpenCV's blobFromImage works. Pyimagesearch. Disponível em: <https://www.pyimagesearch.com/2017/11/06/deep-learning-opencvs-blobfromimage-works/>. . Acesso em: 22 maio 2019.

EKMAN, Paul; FRIESEN, Wallace V. **Unmasking the face: A guide to recognizing emotions from facial clues**. Ishk, 2003.

EKMAN, Rosenberg. **What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)**. Oxford University Press, USA, 1997.

FAIRHURST, Michael; ERBILEK, Meryem; LI, Cheng. Enhancing the forensic value of handwriting using emotion prediction. In: **2nd International Workshop on Biometrics and Forensics**. IEEE, 2014. p. 1-6.

FAN, Yuchao et al. Automatic emotion variation detection using multi-scaled sliding window. In: **2014 International Conference on Orange Technologies**. IEEE, 2014. p. 232-236.

JAGADISWARY, D.; APPASAMI, G.; RAJESH, S. Eye features normalization and face emotion detection for human face recognition. In: **2011 International Conference on Electronics, Communication and Computing Technologies**. IEEE, 2011. p. 64-68.

JARRETT, Kevin et al. What is the best multi-stage architecture for object recognition?. In: **2009 IEEE 12th international conference on computer vision**. IEEE, 2009. p. 2146-2153.

LEE, Honglak et al. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: **Proceedings of the 26th annual international conference on machine learning**. ACM, 2009. p. 609-616.

LEON, E. et al. Real-time physiological emotion detection mechanisms: Effects of exercise and affect intensity. In: **2005 IEEE Engineering in Medicine and Biology 27th Annual Conference**. IEEE, 2006. p. 4719-4722.

LUCEY, Patrick et al. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In: **2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops**. IEEE, 2010. p. 94-101.

NACHAMAI, M. et al. A comprehensive survey on features and methods for speech emotion detection. In: **2015 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)**. IEEE, 2015. p. 1-6.

OH, Byung-Hun; HONG, Kwang-Seok. A study on facial components detection method for face-based emotion recognition. In: **2014 International Conference on Audio, Language and Image Processing**. IEEE, 2014. p. 256-259.

PANTIC, Maja; ROTHKRANTZ, Leon JM. Toward an affect-sensitive multimodal human-computer interaction. **Proceedings of the IEEE**, v. 91, n. 9, p. 1370-1390, 2003.

PUN, Thierry; GERIG, Guido; RATIB, Osman. Image analysis and computer vision in medicine. **Computerized Medical Imaging and Graphics**, v. 18, n. 2, p. 85-96, 1994.

RUSSELL, Stuart J.; NORVIG, Peter. **Artificial intelligence: a modern approach**. Malaysia; Pearson Education Limited,, 2016.

SANCHEZ-MENDOZA, David; MASIP, David; LAPEDRIZA, Agata. Emotion recognition from mid-level features. **Pattern Recognition Letters**, v. 67, p. 66-74, 2015.

SCHROFF, Florian; KALENICHENKO, Dmitry; PHILBIN, James. Facenet: A unified embedding for face recognition and clustering. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. 2015. p. 815-823.

SU, Bo-Hao et al. A spoken dialogue system with situation and emotion detection based on anthropomorphic learning for warming healthcare d. In: **2014 International Conference on Orange Technologies**. IEEE, 2014. p. 133-136.

TSAI, Ching-Chih; CHEN, You-Zhu; LIAO, Ching-Wen. Interactive emotion recognition using support vector machine for human-robot interaction. In: **2009 IEEE International Conference on Systems, Man and Cybernetics**. IEEE, 2009. p. 407-412.

VAPNIK, Vladimir. **The nature of statistical learning theory**. Springer science & business media, 2013.

WITZE, Alexandra.. **Artificial intelligence nails predictions of earthquake aftershocks**. Nature. 10.1038/d41586-018-06091-z. 2018

APÊNDICE A - Fontes

```
from imutils import paths
import numpy as np
import argparse
import imutils
import pickle
import cv2
import os
from sklearn.preprocessing import LabelEncoder
from sklearn.svm import SVC
from sklearn.neural_network import MLPClassifier

protoPath = "det/deploy.prototxt"
modelPath = "det/res10_300x300_ssd_iter_140000.caffemodel"
detector = cv2.dnn.readNetFromCaffe(protoPath, modelPath)
extractor = cv2.dnn.readNetFromTorch("openface_nn4.small2.v1.t7")
confidenceDefault = 0.5
imagePaths = list(paths.list_images("datasetDA"))
facesVectors = []
facesLabels = []
total = 0

def DetectFace(image):

        name = image.split(os.path.sep)[-2]
        image = cv2.imread(image)
        image = imutils.resize(image, width=600)
        h = image.shape[0]
        w = image.shape[1]

        image_blob = cv2.dnn.blobFromImage(cv2.resize(image, (300, 300)), 1.0, (300, 300),
                (124.96, 115.97, 106.13), swapRB=False, crop=False)

        detector.setInput(image_blob)
        detection = detector.forward()

        return detection, h, w, image, name

def FaceValidation(detection, h, w, image, name):

                i = np.argmax(detection[0, 0, :, 2])
                confidence = detection[0, 0, i, 2]

                if confidence > confidenceDefault:
                        x1 = int(detection[0, 0, i, 3] * w)
                        y1 = int(detection[0, 0, i, 4] * h)
```

```
                        x2 = int(detection[0, 0, i, 5] * w)
                        y2 = int(detection[0, 0, i, 6] * h)
                        face = image[y1:y2, x1:x2]

                        faceH = face.shape[0]
                        faceW = face.shape[1]

                        if faceW < 20 or faceH < 20:
                                return

                        face_blob = cv2.dnn.blobFromImage(face, 1.0 / 255, (96, 96), (0, 0, 0),
swapRB=True, crop=False)
                        extractor.setInput(face_blob)
                        vec = extractor.forward()

                        facesLabels.append(name)
                        facesVectors.append(vec.flatten())
                        global total
                        total += 1

def Train(network):

        labelEnc = LabelEncoder()
        labels = labelEnc.fit_transform(facesLabels)
        print("labels {}".format(labels))
        print("[INFO] training model...")
        recognizer = None

        if(network == "CNN_128"):
                recognizer = MLPClassifier(solver='adam',
                        hidden_layer_sizes=(128, 128),
                        activation='relu',
                        max_iter = 5000,
                        verbose=True,
                        tol=1e-4)
        if(network == "SVM_Linear"):
                recognizer = SVC(C=1.0, kernel="linear", probability=True)

        if(network == "SVM_Poly"):
                recognizer = SVC(C=1.0, kernel="poly", probability=True)

        recognizer.fit(facesVectors, labels)

        return labelEnc, recognizer

def main():
```

```
        for (i, image) in enumerate(imagePaths):
                detection, h, w, image, name = DetectFace(image)
                if ( len(detection) == 0 ):
                        continue
                FaceValidation(detection, h, w, image, name)
        network = "CNN_128"
        labelEnc, recognizer = Train(network)

        f = open("recognizer{}.pickle".format(network), "wb")
        f.write(pickle.dumps(recognizer))
        f.close()

        f = open("labelEnc{}.pickle".format(network), "wb")
        f.write(pickle.dumps(labelEnc))
        f.close()

if __name__ == '__main__':
    main()
```

```
from imutils.video import VideoStream
from imutils.video import FPS
import numpy as np
import argparse
import imutils
import pickle
import time
import cv2
import os

def DetectFace(image):

        image = imutils.resize(image, width=600)
        height = image.shape[0]
        width = image.shape[1]

        image_blob = cv2.dnn.blobFromImage(cv2.resize(image, (300, 300)), 1.0, (300, 300),
                (124.96, 115.97, 106.13), swapRB=False, crop=False)

        detector.setInput(image_blob)
        detections = detector.forward()

        return detections, height, width, image
```

```
def FaceValidation(detection, height, width, frame, i):


        confidence = detection[0, 0, i, 2]
        vec = None
        if confidence > confidenceDefault:
                x1 = int(detection[0, 0, i, 3] * width)
                y1 = int(detection[0, 0, i, 4] * height)
                x2 = int(detection[0, 0, i, 5] * width)
                y2 = int(detection[0, 0, i, 6] * height)
                face = frame[y1:y2, x1:x2]

                faceH = face.shape[0]
                faceW = face.shape[1]

                if faceW < 20 or faceH < 20:
                        return

                face_blob = cv2.dnn.blobFromImage(face, 1.0 / 255, (96, 96), (0, 0, 0),
swapRB=True, crop=False)
                extractor.setInput(face_blob)
                vec = extractor.forward()
                preds = recognizer.predict_proba(vec)[0]
                j = np.argmax(preds)
                proba = preds[j]
                name = le.classes_[j]

                text = "{}: {:.2f}%".format(name, proba * 100)
                y = y1 - 10 if y1 - 10 > 10 else y1 + 10
                cv2.rectangle(frame, (x1, y1), (x2, y2),
                        (255, 0, 255), 2)
                cv2.putText(frame, text, (x1, y),
                        FONT_HERSHEY_PLAIN, 0.5, (255, 0, 255), 2)

        return frame

while True:
        frame = vs.read()
        detections, height, width, image = DetectFace(frame)

        for i in range(0, detections.shape[2]):
                frame = FaceValidation(detections, height, width, frame, i)

        cv2.imshow("Frame", frame)
        key = cv2.waitKey(1) & 0xFF
```

```
cv2.destroyAllWindows()
vs.stop()
```

APÊNDICE B – Artigo

RECONHECIMENTO DE EMOÇÕES EM VÍDEO[1]

ALEXANDRE BEHLING[2]

RESUMO

O reconhecimento de emoções em faces humanas apresenta uma série de utilizações no campo de saúde, para análise comportamental, por exemplo. É uma tarefa altamente desafiadora pois necessita ter um bom desempenho, em questão de tempo e de assertividade, mas sem necessitar de grandes clusters de computação, deste modo, podendo funcionar em pequenos dispositivos. Este trabalho apresenta uma abordagem inicial para o desenvolvimento de um classificador de emoções faciais por meio de frames de um vídeo utilizando redes neurais convolucionais e máquinas de vetores de suporte. A proposta analisa um stream de vídeo e a partir da detecção de um rosto passa este rosto para o classificador de emoções, de modo que alcance uma taxa de assertividade aceitável para as emoções conhecidas. De modo a validar o classificador de emoções faciais, foram geradas matrizes de confusão com a emoção classificada pelo classificador e a real emoção presente no vídeo. Foram obtidos resultados de 80% acurácia para a identificação de emoções.

Palavras-chave: Detecção de emoções, Análise facial, Rede Neural Convolucional, Máquina de Vetores de Suporte

1 **INTRODUÇÃO**

Trabalhos que focam na detecção e identificação de emoções utilizando a face se propõem a fazer um computador ser hábil a identificar o estado emocional de uma pessoa. E este campo vem atraindo muita atenção nos últimos anos devido a seu imenso potencial para ser aplicado em vários campos, tal como robótica (TSAI et al., 2009), medicina (LEON, E. et al, 2005) e até mesmo aplicações investigativas (FAIRHURST, ERBILEK, LI, 2014) .

---

[1]Artigo apresentado ao curso de Ciências da Computação, da UNIVERSIDADE FEDERAL DE SANTA CATARINA.

[2]UNIVERSIDADE FEDERAL DE SANTA CATARINA

Segundo (SANCHEZ-MENDOZA et al, 2015), o comportamento dinâmico da face dispõe de uma enorme fonte de informações para a transmissão e caracterização das emoções. Assim como em outras formas de comunicação, muitas informações podem ser inferidas da mensagem original por meio das expressões faciais do seu emissor. Tais características fazem das diversas etapas do reconhecimento facial um problema desafiador e interessante para a pesquisa e a indústria de visão computacional.

Segundo (EKMAN; FRIESEN, 2003) o corpo não tem um movimento específico para o medo ou raiva, porém existem padrões faciais específicos para cada emoção. Se alguém está nervoso, seu corpo pode tentar omitir esta emoção, mas é muito difícil esconder emoções faciais.

Diversas formas e métodos de visão computacional foram abordados por diferentes pesquisadores em suas buscas por uma abordagem que fosse eficiente e permitisse a detecção das emoções faciais.

Dados estes argumentos, o presente trabalho tem como intuito o desenvolvimento de um sistema que seja capaz de reconhecer emoções por meio de redes neurais artificiais, como Redes Neurais Convolucionais e Máquinas de Vetores de Suporte, de forma rápida e eficiente, visando o seu uso em equipamentos com menor poder de processamento, sem depender de grandes clusters para processamento. Assim provendo um modelo pré-treinado para a fase de reconhecimento.

2 **DESENVOLVIMENTO**

2.1 DATA AUGMENTATION

O desempenho de redes neurais de aprendizagem profunda geralmente melhora conforme a quantidade de dados disponíveis vai aumentando (RUSSELL; NORVIG, 2016). Aumento de dados é uma técnica para criar artificialmente novos dados de treinamento a partir de dados de treinamento existentes. Isso é feito aplicando técnicas específicas de domínio a exemplos dos dados de treinamento que criam exemplos de treinamento novos e diferentes.

O aumento de dados de imagem é talvez o tipo mais conhecido de aumento de dados e envolve a criação de versões transformadas de imagens no conjunto de

dados de treinamento que pertencem à mesma classe da imagem original. As transformações incluem uma variedade de operações do campo de manipulação de imagens, como mover para os lados, inversões, zooms, adições de ruídos e filtragens.

Figura 1 - Visualização do aumento de dados

Fonte: O autor (2019)

2.2 EXTRAÇÃO DO VETOR DE CARACTERÍSTICAS

Nesta etapa serão gerados os vetores de características das faces que serão utilizados posteriormente para o treinamento e classificação dos grupos de emoções. A partir da região da face na imagem original é gerada outra imagem blob, desta vez apenas da face, não mais da imagem inteira, e a imagem mantém as mesmas cores, apenas alterando questões de tamanho e aspecto.

Logo em seguida a região da face é colocada como entrada do extrator do OpenFace. Ao fim, é gerado um vetor 128-D em espaço euclidiano que descreve a face em questão. Esse é o vetor que é utilizado para a classificação e identificação da emoção. Na etapa de treinamento, esse passo não é realizado quando se busca apenas a identificação, são adicionados o nome da classe de emoção em um vetor e o vetor de classificação em outro, para a parte do realizar o treinamento.

2.3 RECONHECIMENTO DE EMOÇÕES

Finalmente com os modelos treinados é possível realizar a identificação das emoções em vídeo. Para isso são realizados quase os mesmos passos utilizados na etapa de treinamento dos modelos. A etapa de reconhecimento é feita sobre frames de vídeo, e não sobre imagens individuais, como no treinamento, logo são necessários outros passos para pode extrair os frames do vídeo. E isto é obtido por meio da biblioteca imutils, que disponibiliza diversas ferramentas para manipulação de dispositivos.

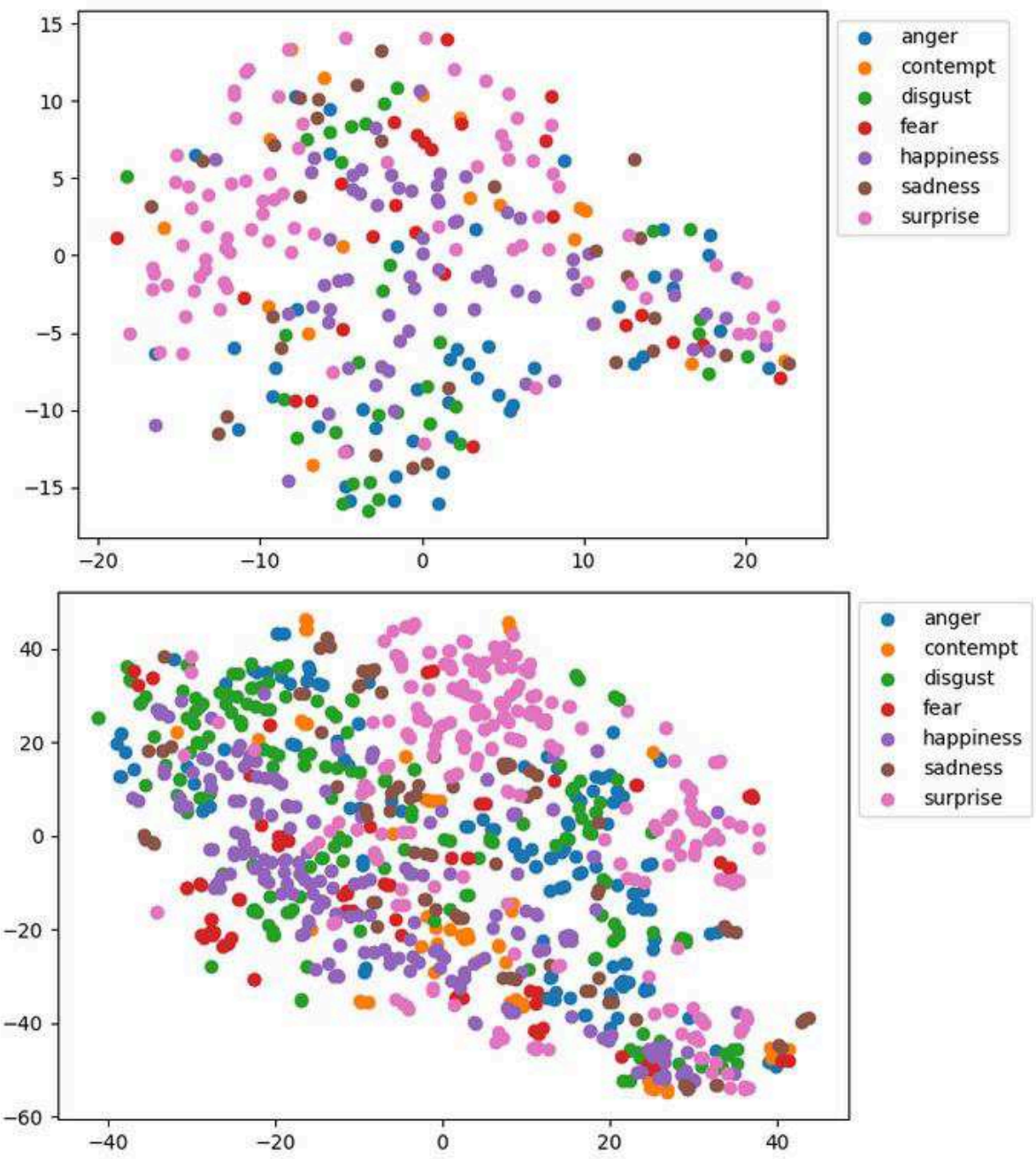
A região passa então pelo extrator de características e gera o seu vetor de 128-D para ser comparado os grupos de emoções. Esse vetor é então inserido como entrada no classificador previamente treinado. Na saída são dadas as probabilidades da face estar em cada um das classes presentes no classificador, logo a classe com maior probabilidade se torna a saída principal. A saída do classificador é um número que é decodificado pelo Descritor de Classes, este por sua vez, que retorna qual é o grupo de emoções que aquela face pertence.

3 **RESULTADOS**

3.1 ESPAÇOS VETORIAIS DAS CARACTERÍSTICAS

Para se obter uma melhor compreensão de como as diferentes emoções podem ter suas características não muito distantes de outras emoções, foram geradas algumas visualizações de como ficaram os espaços vetoriais em 2D. A seguir estão exemplos gerados a partir de conjunto de dados sem aumento de dados e com aumento de dados para fim de comparação e verificar que com uma massa de dados maior, pode-se observar algumas divisões mais nítidas entre as emoções.

Figura 2 - Todas as 7 emoções, sem neutro, com e sem aumento de dados



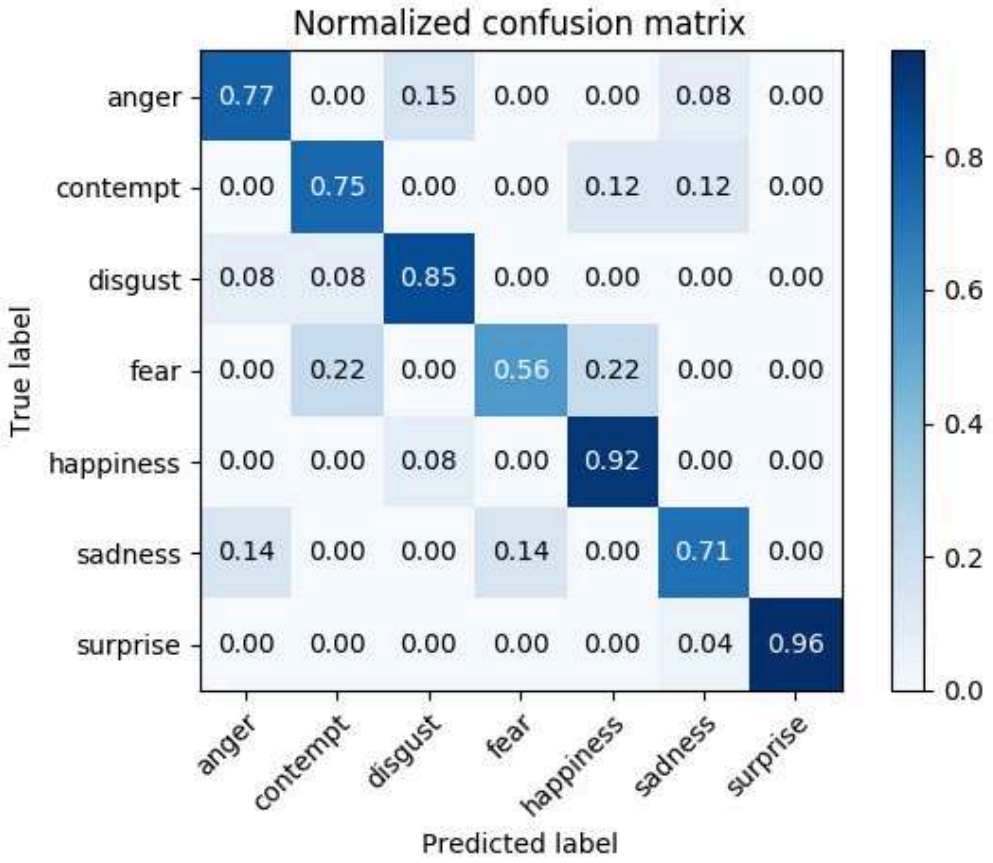
Fonte: O autor (2019)

3.2 CLASSIFICAÇÃO

A partir dos modelos treinados são geradas matrizes de confusão, que como objetivo obter uma visualização dos resultados da classificação da rede. . No eixo horizontal estão as classes esperadas pelo classificador e no eixo vertical está a saída do classificador. Com as respostas é possível calcular a razão entre acertos e

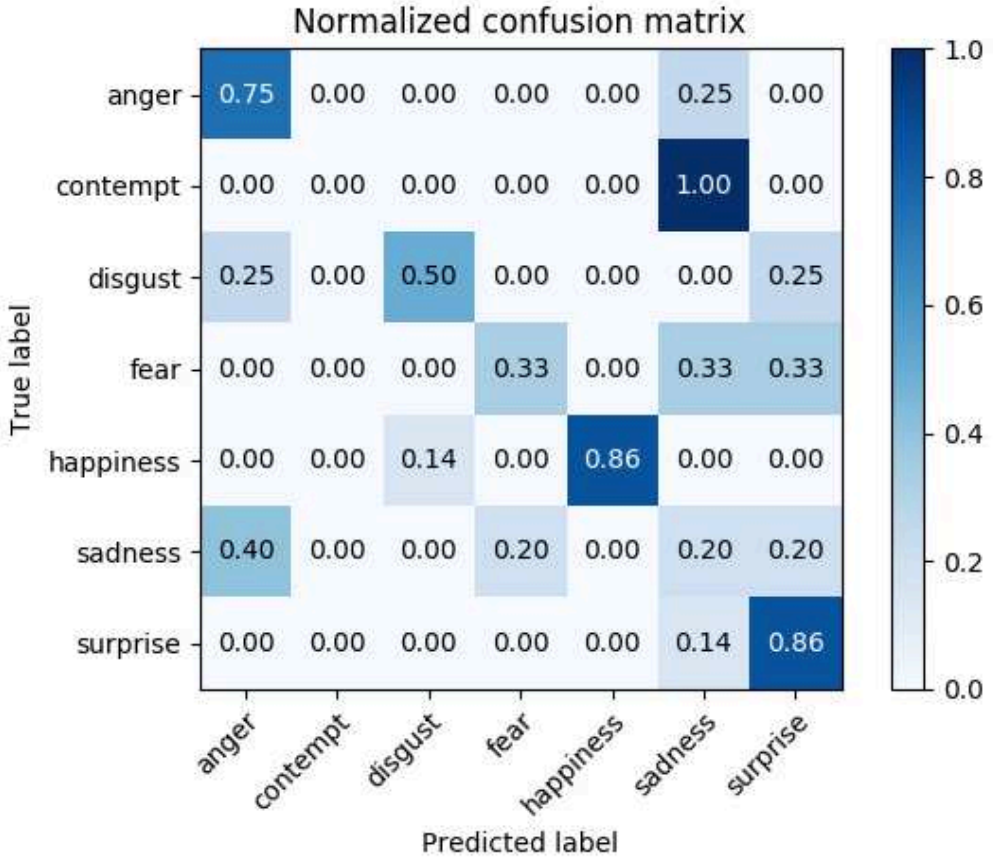
erros do classificador e verificar quais as classes que mais se distanciam das demais e quais classes não obtiveram um bom resultado.

Figura 3 - Matriz confusão da CNN



Fonte: O autor (2019)

Figura 4 - Matriz confusão CNN sem aumento de dados



Fonte: O autor (2019)

4 **CONCLUSÃO**

Este trabalho teve como objetivo analisar o desempenho de redes de aprendizado supervisionado para a classificação de imagens individuais ou frames de vídeos de entrada em diferentes classes de emoções previamente treinadas.

Com o objetivo de fornecer um embasamento e indícios da necessidade da ferramenta desenvolvida, foi realizada uma pesquisa na literatura com base nas implementações de reconhecimento de emoções por meio das mais diversas técnicas.

O uso do detector de faces do OpenCV e do extrator de características do OpenFace, trouxeram bons resultados. Um dos motivos é o fato que as duas ferramentas usaram enormes conjuntos de dados e grandes clusters para os seus

respectivos treinamentos. Sendo que o classificador do OpenCV já vem junto com a biblioteca, assim sendo de fácil acesso. Porém o extrator de características requer algumas bibliotecas adicionais e deve ser obtido individualmente. Podendo-se assim, este trabalho, focar na etapa de treinamento e classificação das emoções, assim como também, na etapa de preparação das imagens e na obtenção e interpretação dos resultados.

Os dois tipos de aprendizados de máquinas supervisionados testados, CNN e SVM, mostraram seus pontos positivos e negativos de cada uma. As redes CNNs obtiveram melhores métricas de assertividade na classificação das emoções, porém podem levar muito mais tempo para serem treinadas, algumas das redes levaram mais de 2 horas para um conjunto de menos de 3000 imagens, e dependem de um conjunto de dados maior para obter um desempenho satisfatório. Já as redes SVM, se mostraram muito mais eficientes para o treinamento, sendo geradas com bastante velocidade. Porém, conforme o tamanho do conjunto de dados é maior e, no caso do trabalho, a dispersão de algumas classes cresce, a rede tem dificuldades para conseguir gerar hiperplanos que gerem menos erros, ou seja, que englobam o menor número de dados não pertencentes a esta classe.

ABSTRACT

The recognition of emotions in human faces presents a series of uses in the field of healthcare, for behavioral analysis, for example. It is a highly challenging task because it needs to perform well, in a matter of time and assertiveness, but without the need for large computing clusters, so that it can run on small devices. This work presents an initial approach for the development of a facial emotion classifier through the frames of a video using convolutional neural networks and support vector machines. The proposal analyzes a video stream and from the detection of a face passes this face to the emotion classifier so that it reaches an acceptable assertiveness rate for the known emotions. In order to validate the facial emotion classifier, matrices of confusion were generated with the emotion classified by the classifier and the real emotion present in the video.

Keywords: Emotion Detection, Facial Analysis, Convolutional Neural Network, Support Vector Machine

REFERÊNCIAS

EKMAN, Paul; FRIESEN, Wallace V. **Unmasking the face: A guide to recognizing emotions from facial clues**. Ishk, 2003.

EKMAN, Rosenberg. **What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)**. Oxford University Press, USA, 1997.

FAIRHURST, Michael; ERBILEK, Meryem; LI, Cheng. Enhancing the forensic value of handwriting using emotion prediction. In: **2nd International Workshop on Biometrics and Forensics**. IEEE, 2014. p. 1-6.

LEON, E. et al. Real-time physiological emotion detection mechanisms: Effects of exercise and affect intensity. In: **2005 IEEE Engineering in Medicine and Biology 27th Annual Conference**. IEEE, 2006. p. 4719-4722.

LUCEY, Patrick et al. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In: **2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops**. IEEE, 2010. p. 94-101.

RUSSELL, Stuart J.; NORVIG, Peter. **Artificial intelligence: a modern approach**. Malaysia; Pearson Education Limited,, 2016.

SANCHEZ-MENDOZA, David; MASIP, David; LAPEDRIZA, Agata. Emotion recognition from mid-level features. **Pattern Recognition Letters**, v. 67, p. 66-74, 2015.

TSAI, Ching-Chih; CHEN, You-Zhu; LIAO, Ching-Wen. Interactive emotion recognition using support vector machine for human-robot interaction. In: **2009 IEEE International Conference on Systems, Man and Cybernetics**. IEEE, 2009. p. 407-412.