22 - Prática: Reconhecimento de Emoções com :TensorFlow 2.0 e Python (III)
Higor Miller Grassi

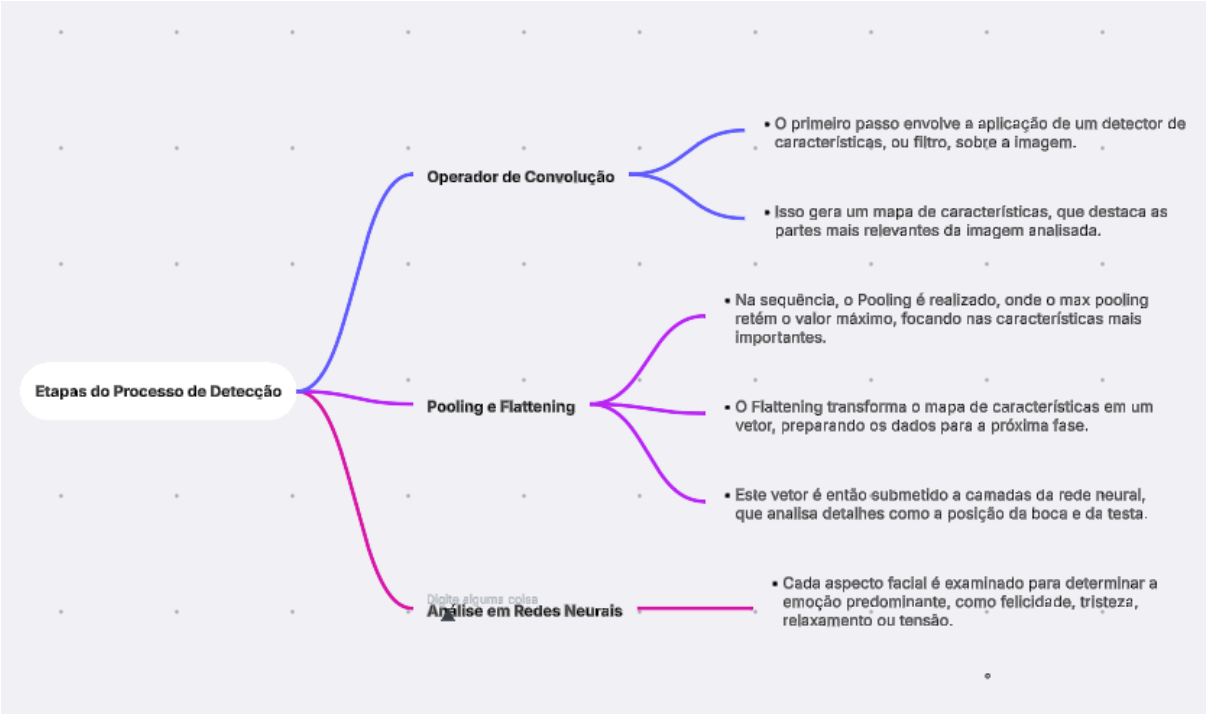
Descrição da Atividade: A atividade abordou o reconhecimento de emoções, focando em como detectar expressões faciais em imagens e vídeos. Foi explicado o processo de convolução, pooling e flattening para extração e processamento de características faciais, além da utilização de funções de ativação como ReLU, Sigmoid e Softmax.

Seção 2

Detecção de emoções e expressões faciais: Analisar imagens e/ou vídeos e indicar qual a emoção predominante a partir de características faciais. As expressões faciais fornecem informações essenciais e desempenham um papel crucial na interação entre as pessoas, servindo como uma forma importante de comunicação não verbal, fornecendo uma mensagem. Dito isso, a técnica mais usada para trabalhar com expressões faciais é o Deep Learning e Redes Neurais Convolucionais.

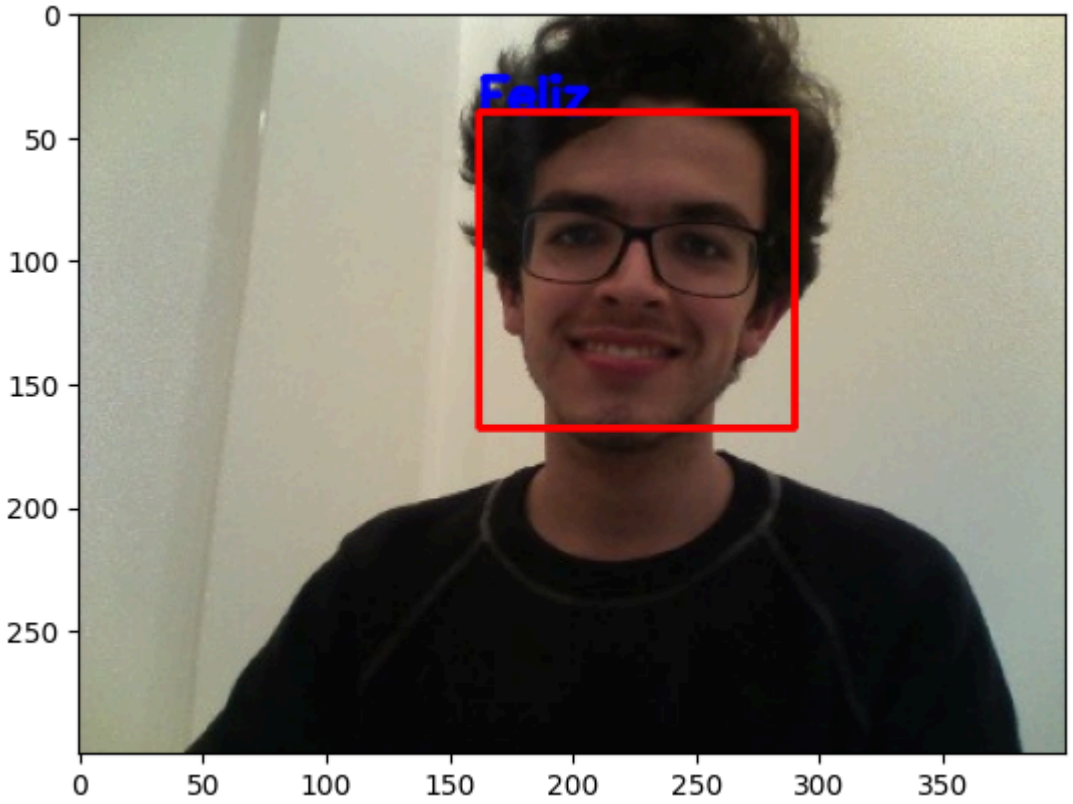
Aplicações: Como citado, nos carros inteligentes alertando os condutores distraídos, verificando o seu rosto e o estado em que se encontra, e quando analisa e percebe que o motorista está sonolento, alertar o condutor

Etapas: Operador de convolução aplica o detector de características (filtro/feature detector) e multiplicará pela imagem de entrada, e no final teremos o mapa de características(feature mapa), focando apenas nas partes principais da imagem. Depois temos a segunda etapa, o Pooling, aplicando uma fórmula, o max pooling voltará o valor máximo da aplicação, focando em características mais importantes. Na etapa três temos o Flattening, que transformara o mapa em um vetor, e por último, submeter este vetor encontrada em camadas da rede neural, assim como na imagem a seguir, onde ocorre a quebra de pixels para a análise separa, onde analisa a boca para ver se está feliz ou triste, aberta ou fechada, a testa se está relaxada ou franzida entre outros aspectos da imagem e no final a resposta



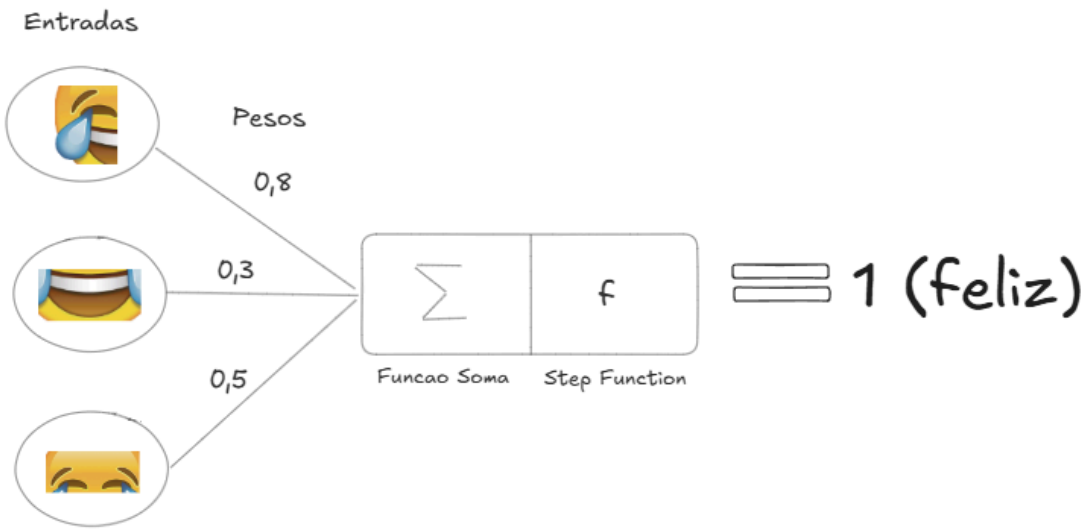
Seção 3:

A seção 3 foi prática, aprendemos como mexer com imagens para reconhecer emoções, funções que fazem, ajustes de imagem para facilitar a rede neural e também as funções de ativações. Também, aprendemos a mexer com vídeo, sobre como reconhecer em vídeos, assim como na imagem a seguir, onde foi o resultado das emoções rodadas.

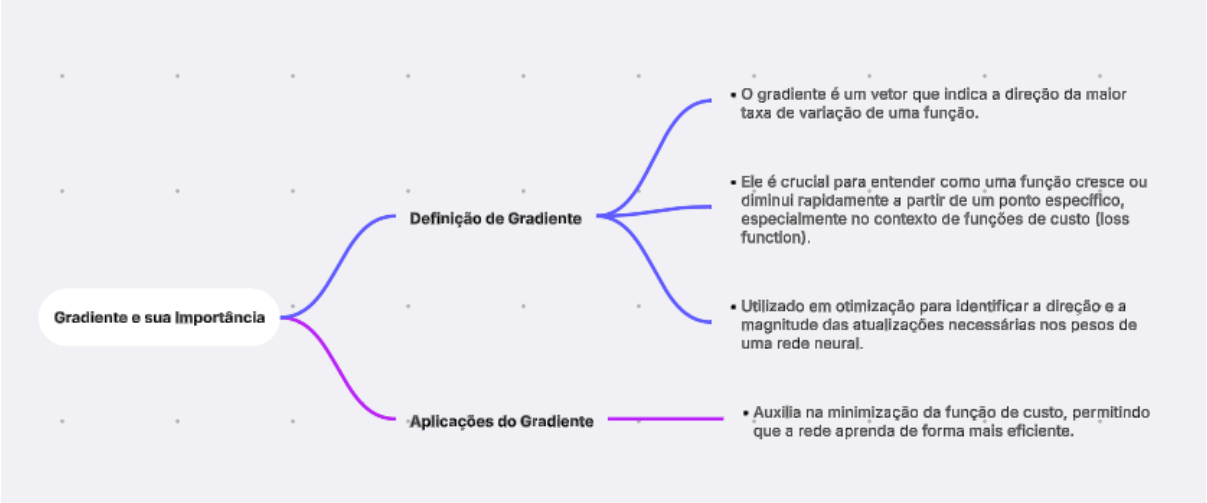


Seção 4:

Neurônio artificial: Um neurônio artificial é o componente fundamental de uma rede neural, simulando o comportamento de um neurônio biológico, processando informações para gerar uma saída, onde cada neurônio recebe múltiplas entradas (inputs), que são valores numéricos, como características de dados ou pixels de uma imagem. Essas entradas são multiplicadas por pesos que determinam a importância de cada entrada e além disso, um termo de bias é adicionado à soma ponderada das entradas para melhorar a flexibilidade do modelo, assim como na imagem a seguir, onde os pesos define o que é mais importante ou não para a saída, onde o mesmo é igual a 1, que neste caso, feliz, com ajuda do gradiente para ajustar o peso de forma a minimizar o erro.

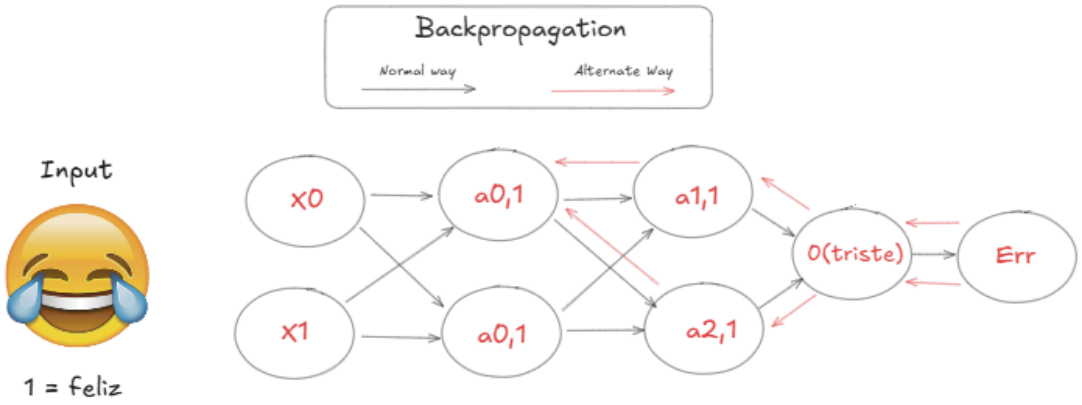


Gradiente: O gradiente é um vetor que aponta na direção de maior taxa de variação de uma função, ele indica a direção em que uma função cresce ou diminui mais rapidamente a partir de um ponto específico, com um custo minimo (loss function).



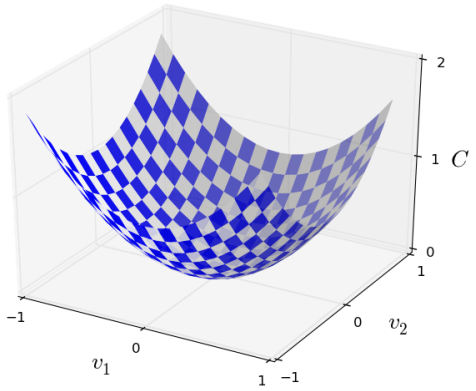
Delta: O cálculo do delta em redes neurais é usado para ajustar os pesos durante o backpropagation, para a camada de saída, o delta é a diferença entre a saída real e a prevista, multiplicada pela derivada da função de ativação, já nas camadas ocultas, o delta é o erro propagado da camada seguinte, ajustado pelos pesos e pela derivada da função de ativação, esse processo permite que a rede aprenda e minimize

Back Propagation: Algoritmo utilizado para ajustar os pesos de uma rede neural durante o treinamento, ele calcula o erro da saída da rede e propagando esse erro de volta através das camadas e em cada iteração, os pesos são ajustados com base no erro calculado, utilizando as derivadas da função de ativação, até que a rede minimize o erro e melhore a precisão das previsões, assim como na imagem a seguir, o input foi feliz, e a saída foi 0, ou seja, triste, nisso a backpropagation funciona a modo de ajustar os pesos a fim de minimizar os erros .



Bias: é um valor adicional que ajuda a ajustar a função de ativação de um neurônio, permitindo que a rede neural aprenda padrões complexos, agora o erro é a diferença entre a previsão da rede e o valor real, usado para ajustar os pesos durante o treinamento.

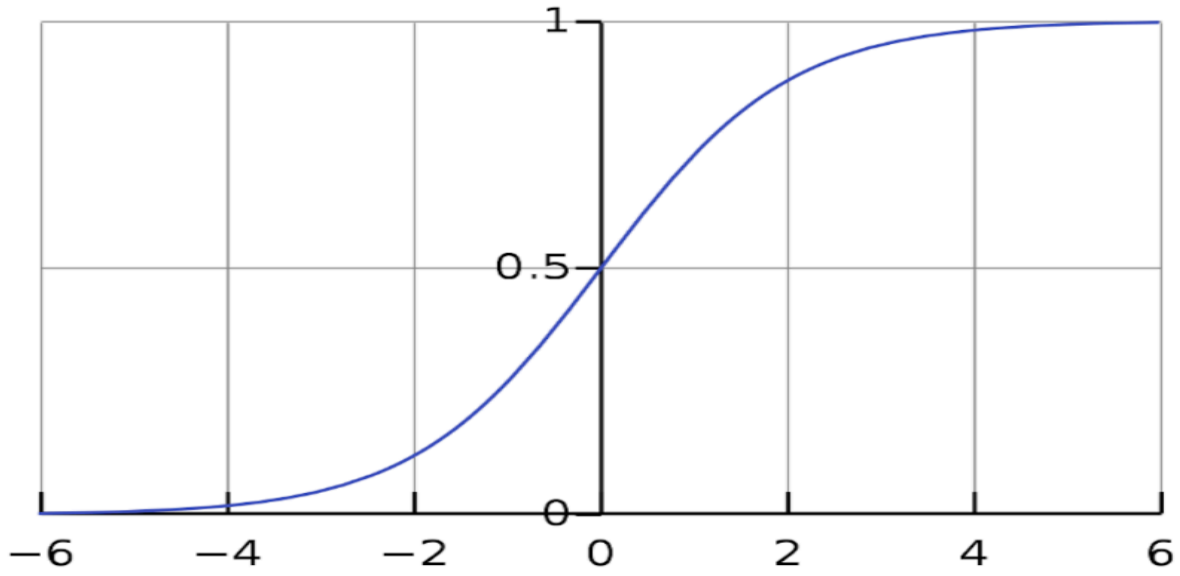
Descida do gradiente estocástica (SGD): é um algoritmo de otimização que ajusta os pesos iterativamente, calculando o gradiente do erro e atualizando os pesos na direção oposta, os parâmetros como a taxa de aprendizado, número de épocas e tamanho do lote influenciam a eficiência e performance do treinamento da rede neural, com a curvatura da descida do gradiente representada no gráfico a seguir.



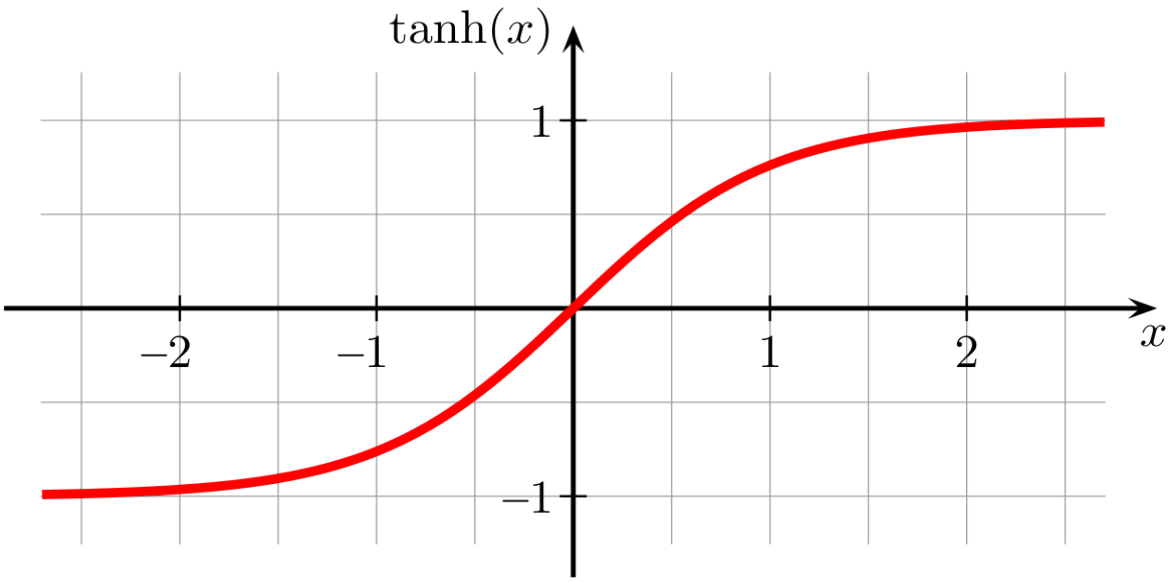
Funções de Ativação:

-Step Function (Função Passo): é uma função de ativação simples que retorna um valor fixo (normalmente 0 ou 1) dependendo se a entrada é maior ou menor que um limiar definido, sendo usada para decidir se um neurônio é ativado ou não, mas tem limitações, como não ser diferenciável, o que dificulta seu uso em redes neurais modernas.

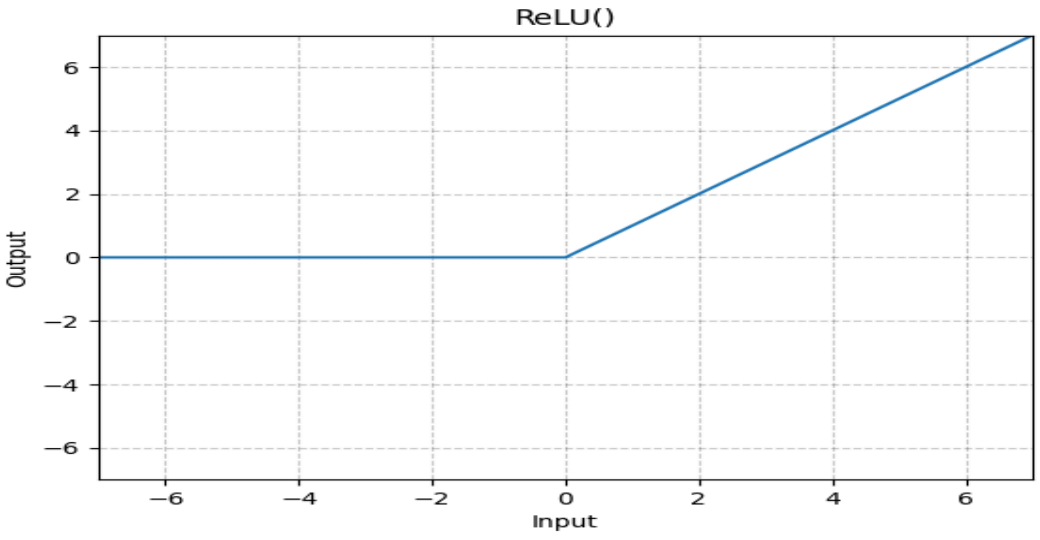
-**Sigmoid:** é uma função de ativação que mapeia qualquer valor de entrada para um intervalo entre 0 e 1, de forma suave e contínua, onde ele é a base do logaritmo natural. A sigmoide é usada em redes neurais para modelar probabilidades, mas tem como limitação o desvanecimento do gradiente, que pode dificultar o treinamento de redes profundas, a função é representada na próxima imagem.



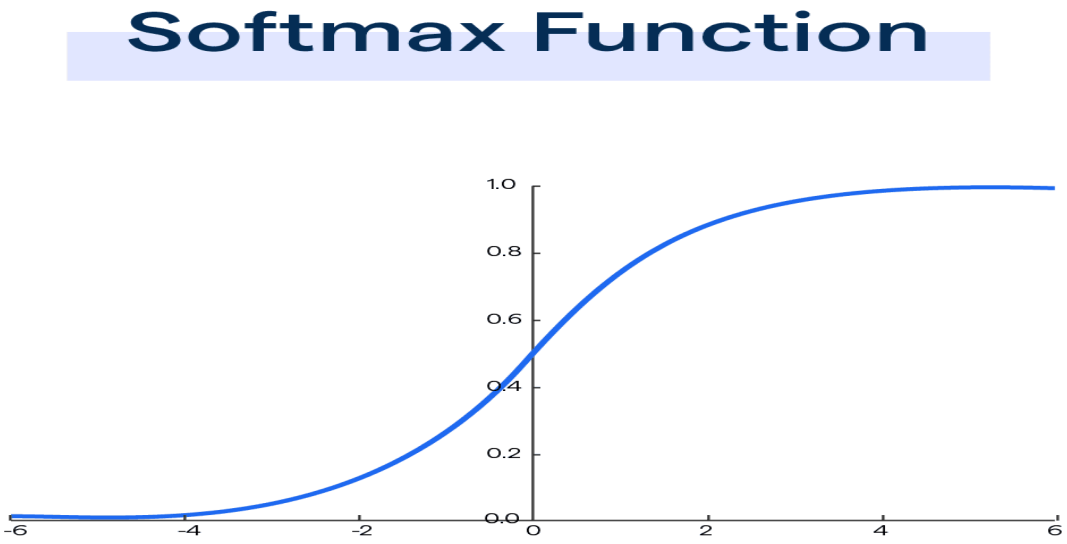
-**Tangente Hiperbólica (tanh)**: é uma função de ativação similar à sigmoide, mas com um intervalo de saída entre -1 e 1, em vez de 0 e 1, mais eficaz que a sigmoide em muitos casos, pois tem uma média de 0 para valores de entrada próximos de 0, o que pode ajudar na convergência do modelo. No entanto, assim como a sigmoide, ela também pode sofrer com o problema de desvanecimento do gradiente quando usada em redes neurais profunda



-**ReLU (Rectified Linear Unit)** é uma função de ativação que retorna 0 para entradas negativas e o próprio valor para entradas positivas, sendo rápida de calcular e ajuda a evitar o problema de desvanecimento do gradiente, mas pode levar à "morte" de neurônios, onde certos neurônios não atualizam mais seus pesos.



-**Função Softmax** é uma função de ativação usada principalmente em problemas de classificação multiclasse, convertendo um vetor de valores em uma distribuição de probabilidade, ou seja, a soma das saídas será sempre 1, onde cada valor de saída representa a probabilidade de uma classe, sendo que a classe com a maior probabilidade será a escolhida como a previsão do modelo, sendo muito utilizada na camada de saída de redes neurais para tarefas de classificação.



Conceito	O que é / faz	Aplicação / Importância
Convolução	Extrai características relevantes da imagem aplicando filtros (feature detectors).	Identifica partes importantes do rosto (ex: olhos, boca) para reconhecimento de emoções.
Pooling	Reduz a dimensionalidade mantendo as informações mais relevantes (ex: MaxPooling).	Aumenta eficiência do modelo, focando em traços mais marcantes.
Flattening	Transforma os mapas de características em vetores para entrada na rede neural.	Prepara os dados extraídos da imagem para classificação.
Funções de Ativação	Decidem se um neurônio "dispara" ou não com base no valor de entrada.	ReLU, Sigmoid, Softmax, Tanh — ajudam a aprender padrões não lineares nas imagens.
Backpropagation	Ajusta os pesos da rede neural com base no erro, propagando-o de volta.	Permite que a rede aprenda com os erros, melhorando sua precisão a cada iteração.
Gradiente e Delta	O gradiente mostra como minimizar o erro; o delta calcula a correção para cada neurônio.	Fundamentais para o processo de otimização dos pesos.
Bias	Valor extra somado à entrada para melhorar a flexibilidade do modelo.	Permite à rede aprender padrões mais complexos.
SGD	Algoritmo que atualiza os pesos com base no gradiente calculado de maneira amostral.	Torna o treinamento mais rápido e eficiente.
Aplicações práticas	Reconhecimento de emoções em rostos, como detectar sonolência em motoristas.	Segurança, saúde, marketing e interação homem-máquina.

Conclusão: O reconhecimento de emoções por meio de técnicas de deep learning e redes neurais convolucionais tem demonstrado grande potencial na análise de expressões faciais em imagens e vídeos, onde a prática e compreensão de conceitos como convolução, pooling e funções de ativação são fundamentais para o JUdesenvolvimento de modelos precisos e eficientes, e além de tudo isso, o uso de funções como ReLU e Softmax, juntamente com o processo de backpropagation e otimização por gradiente, permite a melhoria contínua das redes neurais, tornando-as ferramentas poderosas para diversas aplicações no reconhecimento de emoções e outras áreas, como saúde, segurança e marketing.