

**UNIVERSIDAD DEL ISTMO
FACULTAD DE INGENIERÍA**



**“TAREA 7. DETECCIÓN DE EMOCIONES CON REDES CONVOLUCIONALES”
SISTEMAS INTELIGENTES**

**DIEGO VALLEJO
CARNÉ 2020-1283**

Guatemala, 6 de octubre de 2024

Introducción

La detección de emociones en imágenes faciales es un problema crucial dentro del ámbito de la inteligencia artificial y el reconocimiento de patrones. Las expresiones faciales son una de las formas más inmediatas y universales de comunicación no verbal, y su correcta interpretación puede ser utilizada en una amplia gama de aplicaciones, desde sistemas de asistencia y bienestar, hasta marketing, seguridad y entretenimiento. Sin embargo, identificar las emociones de manera precisa en imágenes es un desafío, ya que las expresiones pueden variar considerablemente entre individuos, culturas y situaciones contextuales. Las técnicas de inteligencia artificial, en particular las Redes Neuronales Convolucionales (CNN), han demostrado un gran potencial para abordar este problema, al aprender a identificar patrones en los datos visuales que son difíciles de detectar manualmente. Este reporte se enfoca en el diseño y desarrollo de redes CNN para la detección automática de emociones en imágenes faciales, con el objetivo de mejorar la precisión y la robustez del reconocimiento emocional.

Conjunto de Datos

El conjunto de datos utilizado para este estudio contiene una colección extensa de imágenes faciales etiquetadas con siete emociones diferentes: **Angry, Disgust, Fear, Happy, Neutral, Sad** y **Surprise**. Estas emociones están representadas por miles de imágenes en blanco y negro de tamaño 48x48 píxeles. El dataset está equilibrado de manera que todas las emociones tienen una cantidad representativa de ejemplos, lo cual es crucial para evitar sesgos en el modelo. Cada imagen contiene una única cara, centrada y alineada para facilitar el proceso de entrenamiento. Este conjunto de datos proporciona una base sólida para entrenar redes neuronales, ya que abarca una variedad de expresiones faciales y garantiza una diversidad en las muestras, permitiendo que el modelo generalice bien a nuevas imágenes y entornos.

Procesamiento de Datos

El preprocesamiento de los datos es una etapa crucial para asegurar que las imágenes se encuentren en un formato adecuado para ser procesadas por el modelo de red neuronal. A continuación, se describen los pasos realizados en el preprocesamiento de las imágenes, utilizando las librerías de **ImageDataGenerator** de Keras:

Carga de Imágenes:

Las imágenes se cargan desde dos directorios: `train_dir` (para el conjunto de entrenamiento) y `test_dir` (para el conjunto de validación). Cada directorio contiene subcarpetas correspondientes a las diferentes clases de emociones, las cuales están organizadas de acuerdo a las etiquetas mencionadas anteriormente (Angry, Disgust, Fear, etc.).

```
train_dir = "./train"
test_dir = "./test"
```

Redimensionamiento:

Todas las imágenes se redimensionan a un tamaño de 48x48 píxeles, que es la entrada esperada por el modelo. Esta reducción de dimensiones ayuda a disminuir la complejidad computacional y garantiza que todas las imágenes tengan el mismo tamaño.

```
img_size = 48
```

Procesamiento de Datos:

Las imágenes son convertidas a escala de grises mediante el parámetro `color_mode="grayscale"`, reduciendo la información de color a una única canal. Esto es apropiado dado que el color no es un factor determinante en la detección de emociones faciales, y permite un procesamiento más eficiente.

Las imágenes son normalizadas dividiendo los valores de los píxeles por 255 (`rescale = 1./255`). Esto convierte los valores originales de los píxeles, que están en el rango de $[0, 255]$, a un rango de $[0, 1]$. La normalización es importante para acelerar el entrenamiento y mejorar el rendimiento del modelo, ya que los modelos de redes neuronales suelen entrenarse mejor con valores en un rango más pequeño.

Para mejorar la capacidad de generalización del modelo, se aplican técnicas de aumento de datos (data augmentation) en las imágenes de entrenamiento. Estas técnicas incluyen:

- Desplazamiento Horizontal y Vertical: Se permiten pequeños desplazamientos de la imagen en ambas direcciones, hasta un 10% del ancho y altura de la imagen original (`width_shift_range = 0.1` y `height_shift_range = 0.1`).
- Flip Horizontal: Se aplica volteo horizontal (`horizontal_flip = True`) para aumentar la diversidad de las imágenes, simulando expresiones faciales en diferentes posiciones.

Estas transformaciones generan nuevas variaciones de las imágenes, lo que ayuda a reducir el sobreajuste al introducir más ejemplos únicos al conjunto de entrenamiento.

Se utiliza el parámetro `validation_split = 0.2` para dividir el conjunto de datos en dos partes: el 80% se utiliza para el entrenamiento y el 20% restante para la validación. Esta técnica permite evaluar el rendimiento del modelo durante el entrenamiento en un subconjunto de datos que no se utiliza para ajustar los parámetros.

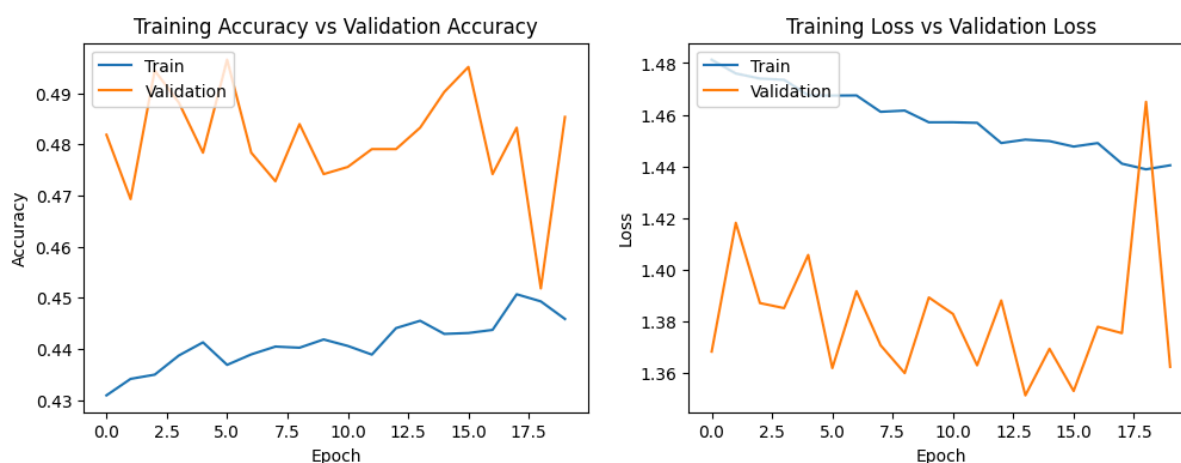
```
train_datagen = ImageDataGenerator(  
    width_shift_range = 0.1,  
    height_shift_range = 0.1,  
    horizontal_flip = True,  
    rescale = 1./255,  
    validation_split = 0.2  
)  
validation_datagen = ImageDataGenerator(rescale = 1./255,  
    validation_split = 0.2)  
  
train_generator = train_datagen.flow_from_directory(directory =  
train_dir,  
    target_size =  
(img_size,img_size),  
    batch_size = 64,  
    color_mode =  
"grayscale",  
    class_mode =  
"categorical",  
    subset = "training"  
)  
validation_generator = validation_datagen.flow_from_directory(  
directory = test_dir,
```

```
target_size = (img_size,img_size),  
  
batch_size  
= 64,  
  
color_mode  
= "grayscale",  
  
class_mode  
= "categorical",  
  
subset =  
"validation"  
  
)
```

Modelo 1: Red CNN de Profundidad Moderada

El Modelo 1 es una red de profundidad moderada diseñada para capturar las características esenciales de las imágenes faciales, utilizando dos capas convolucionales con 32 y 64 filtros respectivamente, seguidas de una normalización por lotes, max-pooling y dropout. Después de aplanar las características extraídas, se emplea una capa densa con activación **softmax** para clasificar las emociones en siete categorías. Este modelo fue entrenado durante 20 épocas utilizando la función de pérdida **categorical_crossentropy** y el optimizador **Adam**. Durante el entrenamiento, se observó una disminución progresiva en la función de pérdida y un aumento en la precisión, tanto en los datos de entrenamiento como de validación, indicando que el modelo fue efectivo para aprender las características relevantes de las emociones faciales.

Resultados Modelo 1

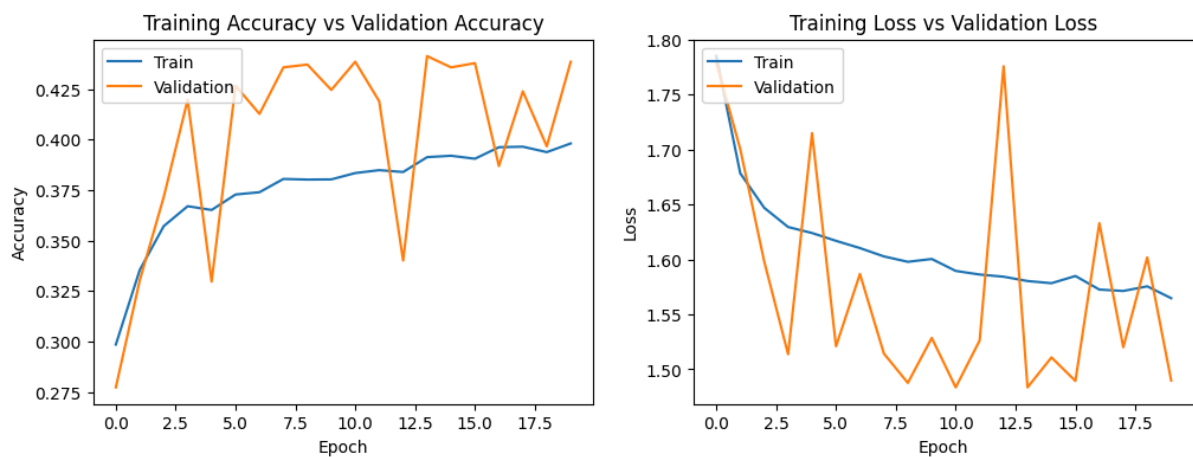


Los resultados del entrenamiento reflejan un rendimiento moderado de ambos modelos en la detección de emociones. Para el Modelo 1, se logró una precisión de entrenamiento final del **46.44%** y una precisión de validación del **48.53%**, lo que indica una capacidad moderada para clasificar correctamente las emociones en las imágenes de prueba.

Modelo 2: Red CNN Profunda

El Modelo 2 es más complejo, utilizando una sola capa convolucional con 512 filtros, lo que le permite capturar características más detalladas. Al igual que el Modelo 1, incorpora

una capa de normalización, max-pooling y dropout, seguida de una capa de aplanamiento y una capa densa con **softmax** para la clasificación. Este modelo fue entrenado durante 64 épocas bajo las mismas condiciones de pérdida y optimización. Aunque más profundo, el aumento en la precisión no fue sustancialmente superior al Modelo 1, lo que sugiere que la mayor complejidad no siempre implica un mejor rendimiento, especialmente cuando se trata de conjuntos de datos moderados. Sin embargo, el modelo mostró una buena capacidad de aprendizaje a lo largo del entrenamiento.



El Modelo 2 mostró un desempeño inferior, con una precisión de entrenamiento final del **40.85%** y una precisión de validación del **43.85%**. Estos resultados sugieren que, a pesar de su mayor complejidad, **el Modelo 2 no fue capaz de generalizar mejor que el Modelo 1**, lo que resalta la importancia de optimizar las arquitecturas y parámetros de los modelos para mejorar su capacidad de clasificación en tareas de detección de emociones.

Resultados con Imagen Personal

Modelo 1

Al probar con una imagen que representa tristeza y otra que refleja felicidad, el **Modelo 1** clasificó ambas imágenes como "Surprise". Este resultado indica que el modelo pudo haber interpretado ciertas características faciales en mis expresiones como indicadores de sorpresa, a pesar de que las imágenes estaban destinadas a representar emociones opuestas. La confusión entre emociones sugiere que el modelo podría beneficiarse de una mayor capacidad para aprender las sutilezas en las características faciales que diferencian las emociones, así como de un ajuste adicional en su arquitectura y parámetros de entrenamiento.

Modelo 2

En contraste, el **Modelo 2** presentó una clasificación diferente, identificando ambas imágenes como "Angry". Este resultado pone de relieve la tendencia del modelo a asociar las características faciales de mis expresiones con la emoción de ira, lo que sugiere una falta de sensibilidad hacia las variaciones emocionales más sutiles. La discrepancia en las predicciones con respecto al Modelo 1 resalta la complejidad del reconocimiento de emociones en imágenes faciales y la necesidad de ajustar y optimizar el modelo para mejorar su precisión en la detección de diferentes emociones, especialmente aquellas que pueden parecerse visualmente.

Imágenes Utilizadas:



Screenshots de Resultados:

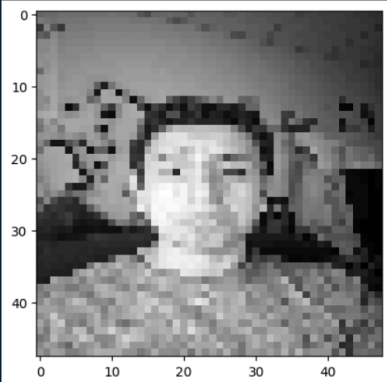
Modelo 1

Modelo 1

```
img_path1 = './triste_yo.jpeg'
predict_image(model1, img_path1)
```

✓ 0.3s Python

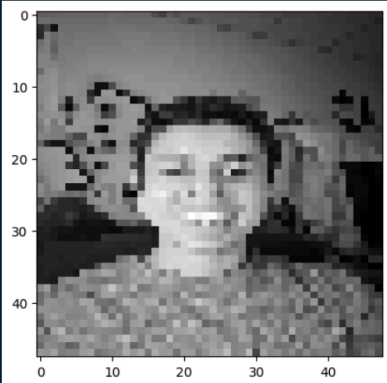
Shape de la imagen: (48, 48)
1/1 — 0s 19ms/step
Resultados de la predicción: [0.0, 0.0, 0.0, 0.0, 6.281339e-27, 0.0, 1.
Predicción: Surprise



```
img_path2 = './feliz_yo.jpeg'
predict_image(model1, img_path2)
```

[54] ✓ 0.2s Python

Shape de la imagen: (48, 48)
1/1 — 0s 18ms/step
Resultados de la predicción: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0]
Predicción: Surprise



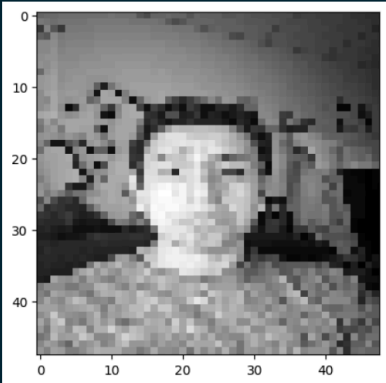
Modelo 2

Modelo 2

```
img_path1 = './triste_yo.jpeg'
predict_image(model2, img_path1)
```

✓ 0.2s Python

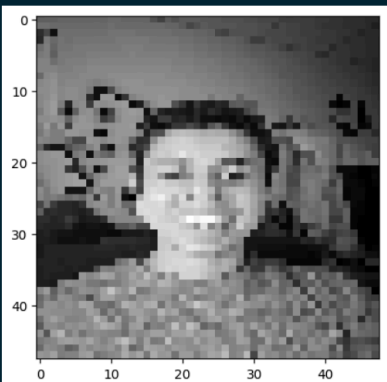
Shape de la imagen: (48, 48)
1/1 — 0s 17ms/step
Resultados de la predicción: [1.0, 0.0, 0.0, 4.03911e-27, 0.0, 0.0, 0.0
Predicción: Angry



```
img_path2 = './feliz_yo.jpeg'
predict_image(model2, img_path2)
```

[56] ✓ 0.3s Python

Shape de la imagen: (48, 48)
1/1 — 0s 18ms/step
Resultados de la predicción: [1.0, 0.0, 1.07585374e-26, 0.0, 0.0, 0.0,
Predicción: Angry



Referencias

Emotion detection. *Kaggle* [en línea], 2020. Disponible en:
<https://www.kaggle.com/datasets/ananthu017/emotion-detection-fer>.