Detección de Sonrisas en Imágenes con Modelos de Aprendizaje Profundo

Santiago Mora Cruz
Gabriel Reynoso Escamilla
María P. Rodríguez-Muñoz
Guillermo Villegas Morales

Noviembre de 2024

1 Introduction

Los modelos de redes neuronales son capaces de clasificar objetos que por otro métodos de *Mavhine Learning* sería más complejo. El proposito de este trabajo es entrenar dos modelos de aprendizaje profundo o *Deep Learning* (DP) para la detección de sonrisas en fotos de rostros humanos, y evaluar su efectividad para la realización de esta tarea. Los modelos por evaluar serán: una Red Neuronal Convolucional, CNN, por sus siglas en inglés, una arquitectura de red basada en capas convolutivas, series de filtros entrenables que abstraen la información, usualmente usadas en problemas de detección de imagenes, sonidos, entre otros [1]. Y un *Autoencoder*, un tipo de red neuronal la cual comprime la información y la decodifica para reconstruir la imagen, estas son útiles entre otros propositos para detección anomalías [2].

2 Metodología

Se trabajó con la bases de datos *LFWcrop* y Smiles Dataset [3], la primera es una versión recortada de *Labeled Faces in the Wild* y la segunda una clasificación de si en dichos rostros la persona sonreía o no.

2.1 Preprocesamiento

Cada imagen es cargada y preprocesada para cumplir con los requisitos del modelo. Primero, las imágenes se redimensionan a 64x64 píxeles, un tamaño común que reduce la complejidad computacional, manteniendo suficiente información visual para la clasificación. Luego los valores de los píxeles son normalizados, asegurando que todos los valores se encuentren en un rango entre 0 y 1. Después, las listas que contienen las imágenes procesadas y sus etiquetas se convierten en arreglos de NumPy. Las etiquetas se transforman adicionalmente mediante codificación one-hot, donde cada etiqueta binaria se representa como un vector.

Figure 1: Imagen de persona sonriendo

Por último se barajan las filas de los datos y se dividen en x_train, y_train, x_test, y_test.

2.2 Construcción del Modelo CNN

Se utilizó la API *Sequential* de Keras para construir un modelo convolucional. La arquitectura de la red neuronal consiste en las siguientes capas:

1. Capa convolucional de 32 kernels de 3x3 con función de activación *ReLu* (Rectified Linear Unit), recibiendo un *input* de 64x64x3.
2. Capa de *Max Pooling* de 2x2, con *stride* de 2.
3. Capa convolucional de 64 kernels de 3x3 con una función de activación *ReLu*.
4. Capa de *Max Pooling* de 2x2, con *stride* de 2. con función de activación *ReLu*.
5. Capa de aplanamiento (*flatten*) para transformar el tensor en un vector.
6. Capa densa con 128 unidades con función de activación *Relu*.
7. Capa de salida con 2 unidades con función de activación *Softmax*, para clasificar correctamente.

El modelo se compila usando el optimizador *adam*. Se utiliza la función de pérdida *categorical_crossentropy*, adecuada para problemas de clasificación multiclase con etiquetas codificadas one-hot. Además, se especifica la métrica *accuracy* para evaluar el rendimiento del modelo durante el entrenamiento. Este modelo consiste de 1,625,410 parámetros (entrenables).

Finalmente se entrenó el modelo en 10 épocas, y un *batch_size* de 32, con un *validation_split* de 0.15.

2.3 Construcción del modelo Autoencoder

De igual manera se utilzó la API *Sequential* de Keras, en primer lugar se codifica la imagen hasta reducir sus dimensiones considerablemente, para posteriormente reconstruirla. Para esto se diseñó la red de la siguiente manera, usando *padding* = *same* para preservar el tamaño original de la imágen:

1. Capa convolucional de 32 kernels de 5x5 con una función de activación *ReLu*.
2. Capa de *Max Pooling* de 2x2, con *stride* de 2.
3. Capa convolucional de 16 kernels de 5x5 con una función de activación *ReLu*.
4. Capa de *Max Pooling* de 2x2, con *stride* de 2.
5. Capa de *Up Samplind* de 2x2, con *stride* de 2.
6. Capa convolucional de 16 kernels de 5x5 con una función de activación *ReLu*.
7. Capa de *Up Samplind* de 2x2, con *stride* de 2.
8. Capa convolucional de 32 kernels de 5x5 con una función de activación *ReLu*.
9. Capa Convolucional de 3 kernels de 3x3 con una función de activación lineal.

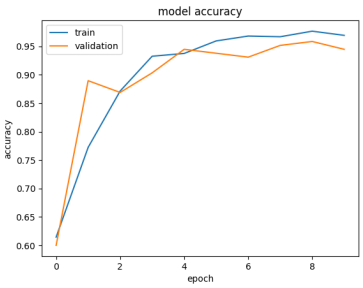
El modelo se compiló con un optimizador de Descenso de Gradiente Estocastico. Una función de pérdida y métrica de Errores Medios Cuadrados, para identificar los errores de reconstrucción. Este modelo consiste en 35,363 parémtros entrenables.

El resultado de este modelo es una reconstrucción, por lo que se entrenó solamente con imágenes con sonrisa, esto para usar los errores de reconstrucción para detectar aquellas con anomalías o no sonrisa.

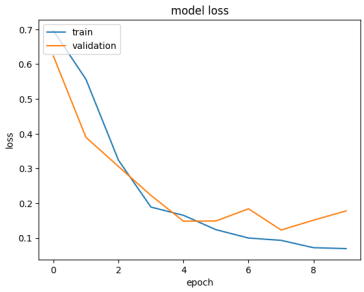
3 Experimentos y Resultados

3.1 Primer Modelo con Capas Convolucionales

Al entrenar la red convolucional, se obtuvieron las siguientes gráficas respecto al entrenamiento del modelo, y su evaluación en el set de entrenamiento y de validación:



(a) Primer Modelo: Loss de los datos entrenados a comparación del Loss de los datos de validación a través de las iteraciones.



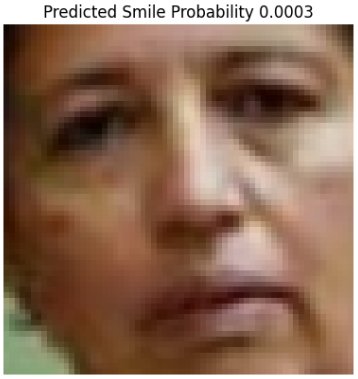
(b) Primer Modelo: Loss de los datos entrenados a comparación del Loss de los datos de validación a través de las iteraciones.

Figure 2: Gráficas sobre el entrenamiento del primer modelo.

Además de las exactitudes mostradas en la gráfica, que sugieren que la red funciona de manera bastante adecuada, se destaca que la exactitud en el set de prueba (mismo que se usó para evaluar la otra red) fue de 94.60%.

Estos fueron algunas predicciones que se realizaron de parte del primer modelo en rostros humanos.

(a) Probabilidad de que la persona este sonriendo: %89.93



(b) Probabilidad de que la persona este sonriendo: %0.03

Figure 3: Predicciones del modelo convolucional

En cuanto al segundo modelo, el autoencoder, algunos ejemplos de las reconstrucciones sobre el dato de entrenamiento se encuentran a continuación.

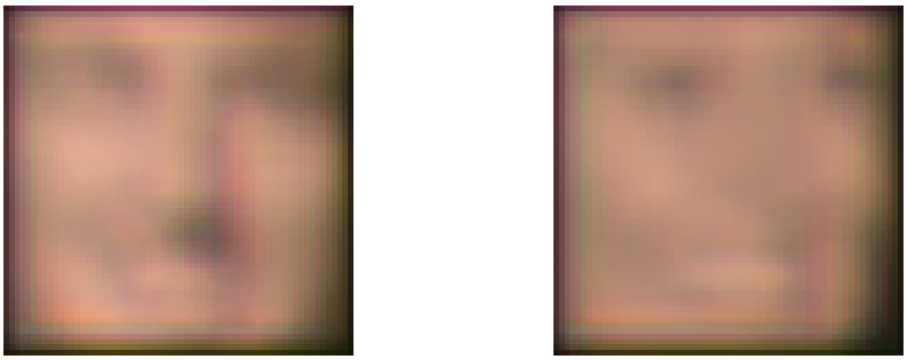


Figure 4: Reconstrucción de imagénes del (segundo) modelo autoencoder.

Se evaluó el error de reconstrucción de las generadas por el autoencoder, comparándolas con las imágenes originales utilizando la distancia de Wasserstein (también conocida como distancia de la tierra trasladada o "Earth Mover's Distance") para medir las diferencias entre las distribuciones (histogramas) de las imágenes originales y sus respectivas reconstrucciones. De esta forma se puede evaluar qué tan bien el modelo logra preservar las características de las imágenes originales, ya que en teoría la distancia de Wasserstein es buena para medir diferencias sutiles en la reconstrucción.

Una vez determinada esta distancia para cada imagen, se obtuvieron los cuantiles de las distancias de todas las imágenes en el conjunto de entrenamiento, y se definió un *threshold* entre el cuantil 0.9 y 1.0. Finalmente, se clasifican las imágenes como sonrientes o no sonrientes basado en si están por encima o debajo, respectivamente, de este *threshold*.

Para comparar los resultados de ambos modelos, se obtuvieron las matrices de confusión para cada modelo, con las categorías: "Non-Smile" (no sonriente) y "Smile" (sonriente).

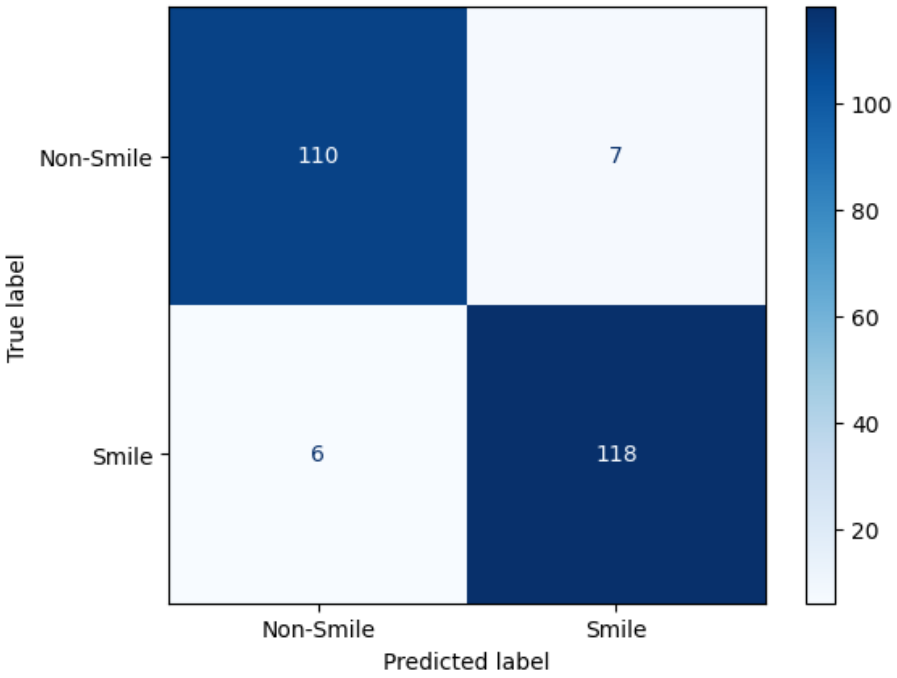


Figure 5: Matriz de Confusión del Modelo Convolucional

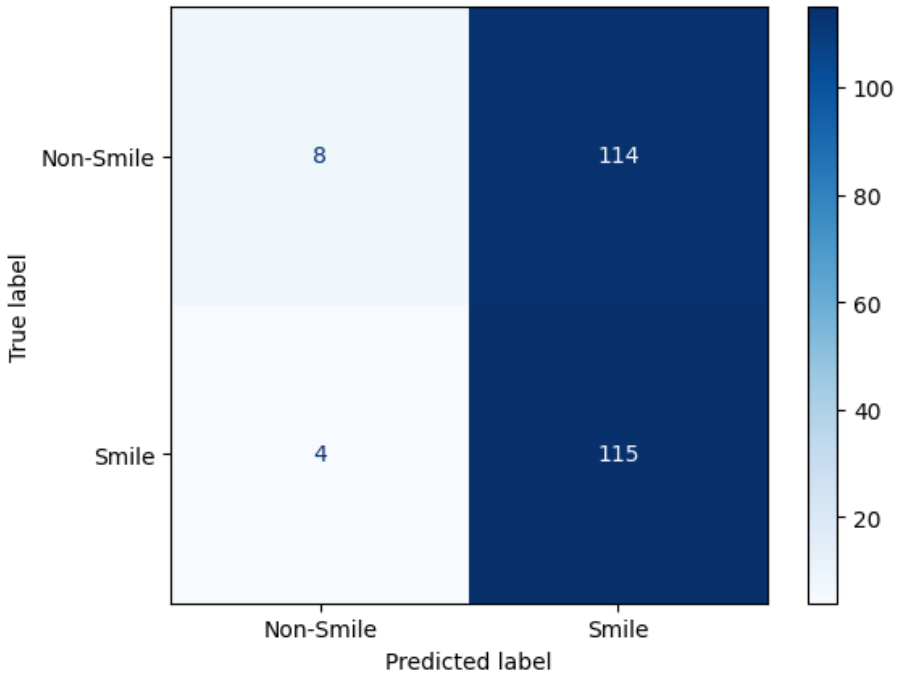


Figure 6: Matriz de Confusión del Modelo Autoencoder

3.2 Resultados

La tabla 1 muestra las métricas de evaluación relevantes para cada modelo.

En exactitud y recall para la clase no sonriente, el modelo de convolución es superior. Sin embargo, en recall para la clase sonriente, el modelo autoencoder es superior.

Tomando en cuenta que clasificó al 91.4% de las observaciones como sonrientes, es seguro asumir que el modelo es inferior al convolucional para clasificar a las personas, y que posiblemente este desbalance se debe a que únicamente se entrenó con datos de personas sonrientes.

	Accuracy	Recall (sonriente)	Recall (no sonriente)
CNN	0.9460	0.9516	0.9401
Autencoder	0.5103	0.9663	0.0634

Table 1: Comparación de métricas de clasificación de ambos modelos

4 Conclusiones

En este caso en modelo CNN realizó buenas predicciones con un accuracy de 0.94 y un recall por clase de 0.95 y 0.94 para sonrisa y no sonrisa respectivamente. Sin embargo el Autoencoder no fue capaz de hacer predicciones, con un accuracy de 0.51 y un recall por clase de 0.96 y 0.06, es decir clasificando prácticamente todas las imágenes como sonrisa. Para la solución de este problema fue más útil el CNN. Sin embargo, no se descarta la posibilidad de que con una arquitectura distinta y un acercamiento al problema diferente, sería posible utilizar un autoencoder para determinar cuando hay una sonrisa presente en la imagen.

References

[1] D. Scherer, A. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," in *Artificial Neural Networks – ICANN 2010* (K. Diamantaras, W. Duch, and L. S. Iliadis, eds.), (Berlin, Heidelberg), pp. 92–101, Springer Berlin Heidelberg, 2010.

[2] D. Bank, N. Koenigstein, and R. Giryes, *Autoencoders*, pp. 353–374. Cham: Springer International Publishing, 2023.

[3] O. Arigbabu, "Dataset for smile detection from face images," 2017.