Entrega No. 2: Reconocedor de Lenguaje de Señas usando Machine Learning

Responsables

Jair Silva Herrera

Leonardo Castro Piracoa

2024

INTRODUCCIÓN

El proyecto a desarrollar corresponde a un modelo predictivo de lenguaje de señas mediante el uso de un modelo de redes neuronales convolucionales, el cual es entrenado con un data set que contiene imágenes de todas las letras del abecedario. Durante el desarrollo del proyecto se prueban una serie de variables de la red neuronal convolucional para escoger el de mejor desempeño.
El modelo es probado con varias imágenes incluida una captura de cámara.

ESTRUCUTRA DE NOTEBOOKS

Para la obtención del modelo de reconocimiento de lenguaje de señas, en primera instancia se hace un notebook denominado ***01_Entrenamiento_Obtencion_modelo.ypynb***, el cual contiene todo lo concerniente a la obtención del modelo de redes neuronales convolucionales. El Notebook está dividido en 3 partes: la primera es obtención del Dataset, la segunda es el preprocesado de imágenes y la ultima es la parte de obtención del modelo, entrenamiento y análisis de resultados.
El segundo Notebook denominado ***02. Ejercicio_Reconocimiento.ipynb*** se trata de un código en Python que ejecuta un modelo previamente guardado, con el fin de hacer unas predicciones de prueba.

SOLUCIÓN A PROBLEMA PLANTEADO

El objetivo del proyecto es realizar un reconocedor de lenguaje de señas, usando machine learning. Para el desarrollo del reconocedor se usará un modelo neuronal de capas convolucionales, usando un adecuado dataset de imágenes, que posteriormente será preprocesado para obtener resultados óptimos. En el desarrollo del proyecto se usan herramientas como el dropout y pooling para mejorar el desempeño del modelo, además de evitar el sobre ajuste; además se varían unos parámetros del modelo para escoger la mejor opción de predicción.

ARQUITECTURA Y PREPROCESADO

Inicialmente se obtiene el Dataset que se va a usar para entrenar el modelo y además probarlo. El dataset contiene carpetas de entrenamiento (TRAIN), y prueba (TEST), y dentro de estas hay subcarpetas de todas las letras del abecedario con varias imágenes con la seña de la respectiva letra. Las imágenes de entrenamiento son aproximadamente 5996 por letra, mientras que para las de prueba se tienen 4 imágenes de prueba.
La segunda parte corresponde al pre procesamiento de imágenes, con lo cual se busca mejorar las imágenes iniciales para entrenar el modelo y ajustar el tamaño. Para el preprocesamiento se usa *ImageDataGenerator*, el cual sirve para cambiar el valor de RGB que originalmente es de 0 a 255, a valores de 0 a 1 con el comando *rescale*, además de agregar efectos a las imágenes como *horizontal_flip* que sirve para obtener imágenes con efecto espejo (imágenes reflejadas sobre un eje), y los comandos *shear_range* y *zoom_range*, que sirven para obtener imágenes con distinta inclinación y acercamiento. El preprocesamiento sirve para obtener una mayor cantidad de imágenes contemplando otras variaciones en las imágenes.

En la tercera parte se obtiene el modelo usando la opción de una, dos o tres capas convolucionales de acuerdo a la preferencia del usuario. Para la primera capa se usan 32 filtros de 7*7, para la segunda 64 filtros de 5*5, y para la ultima 128 filtros de 3*3. Por otro lado, también se da la opción de cambiar la cantidad de filtros mediante un multiplicador. Para el ejercicio se probarán 6 modelos y se escogerá el de mejor rendimiento.
Adicionalmente se aplica pooling en todas las capas convolucionales para hacer reducción de parámetros mediante la selección de aquellos parámetros con mejores características; además se aplica dropout en la tercera capa convolucional y en la capa densa antes de la salida, con el fin de evitar el sobreajuste apagando ciertas neuronas.

RESULTADOS E ITERACIONES

Con el fin de obtener el modelo con las predicciones mas acertadas, se van a probar los siguientes modelos:

ID	Modelo	Cantidad de Capas convolucionales	Componentes adicionales	Dimensiones Filtros
1	1 capa	1	Maxpooling 2*2	32 filtros de 7*7
2	1 Capa / mitad Filtros	1	Maxpooling 2*2	16 filtros de 7*7
3	2 capas	2	Capa 1: Maxpooling 2*2	32 filtros de 7*7
			Capa 2: Maxpooling 2*2	64 filtros de 5*5
4	2 capas / mitad filtros	2	Capa 1: Maxpooling 2*2	16 filtros de 7*7
			Capa 2: Maxpooling 2*2	32 filtros de 5*5
5	3 capas	3	Capa 1: Maxpooling 2*2	32 filtros de 7*7
			Capa 2: Maxpooling 2*2	64 filtros de 5*5
			Capa 3: Dropout de 0,3 / Maxpooling 2*2	128 filtros de 3*3
6	3 capas / mitad filtros	3	Capa 1: Maxpooling 2*2	32 filtros de 7*7
			Capa 2: Maxpooling 2*2	64 filtros de 5*5
			Capa 3: Dropout de 0,3 / Maxpooling 2*2	128 filtros de 3*3

Cada uno de los modelos se entrenan con 25 épocas, tamaño de batch 32 y 32 pasos por época. A continuación, se muestran los resultados de accuracy para los datos de entrenamiento y prueba:

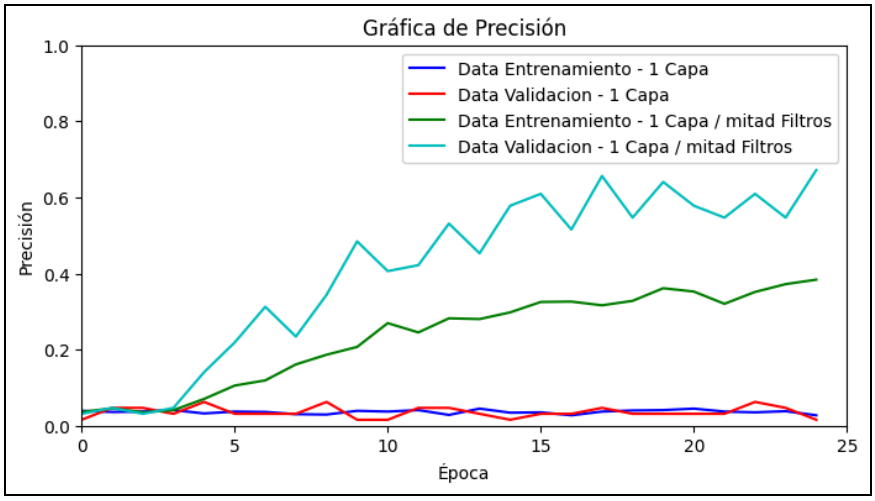

Ilustración 1. Accuracy de entrenamiento y prueba por época, modelos de 1 capa.

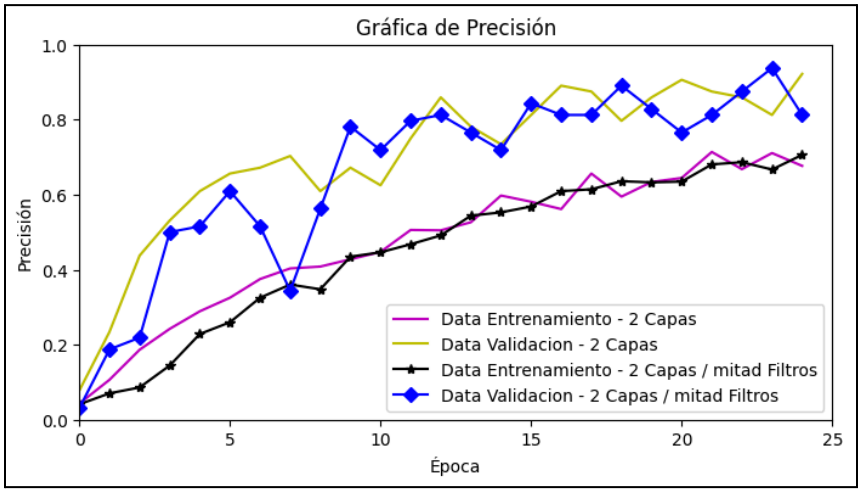

Ilustración 2. Accuracy de entrenamiento y prueba por época, modelos de 2 capas.

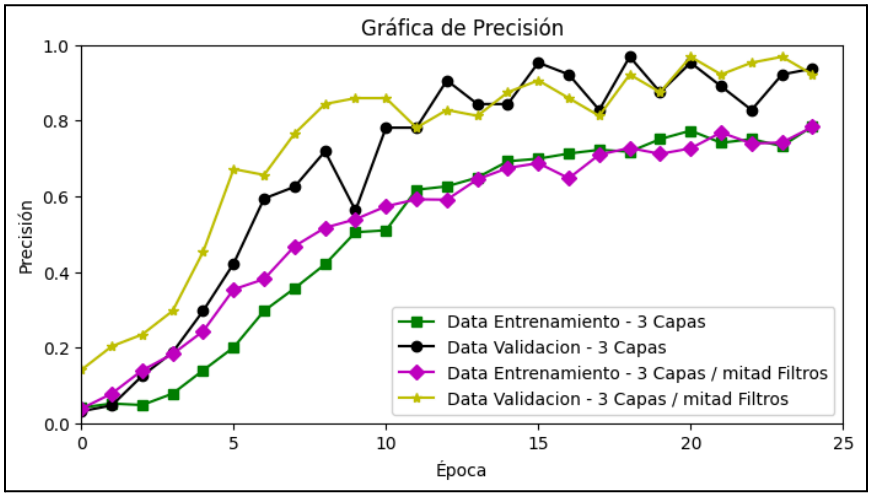

Ilustración 3. Accuracy de entrenamiento y prueba por época, modelos de 3 capas.

De acuerdo a las gráficas se puede ver que en el modelo de una capa que usa todos los filtros no hay mejora en el accuracy, y cuando se usa la mitad de los filtros se obtiene mejora en el accuracy sin ser significativamente alto. Para los modelos que usan 2 y 3 capas convolucionales se obtiene una mejoría notable, siendo la ultima la de mejor comportamiento.

Una observación particular en todos los modelos es que el accuracy de prueba es mayor que el de entrenamiento, lo cual es opuesto a lo que generalmente se observa en los modelos; posiblemente el comportamiento observado se da por que la partición de los datos es altamente desigual, ya que usualmente se usa el 70 – 80% de los datos para entrenamiento y el resto para prueba, mientras que para el caso práctico se tienen 5996 imágenes por letra para entrenamiento, contra 4 imágenes por letra para prueba.

Otro método para decidir sobre el mejor modelo, es el uso de matrices de confusión, para lo cual se hace nuevamente la predicción de las imágenes de prueba y se obtienen las respectivas matrices. A continuación, se ilustran los respectivos resultados:

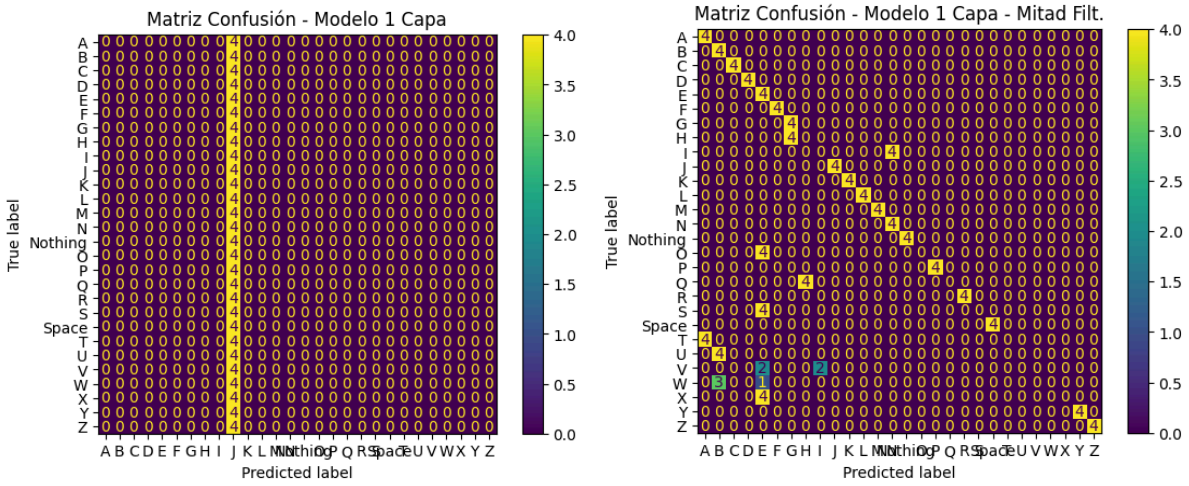

Ilustración 4. Matrices de confusión modelos de una capa.

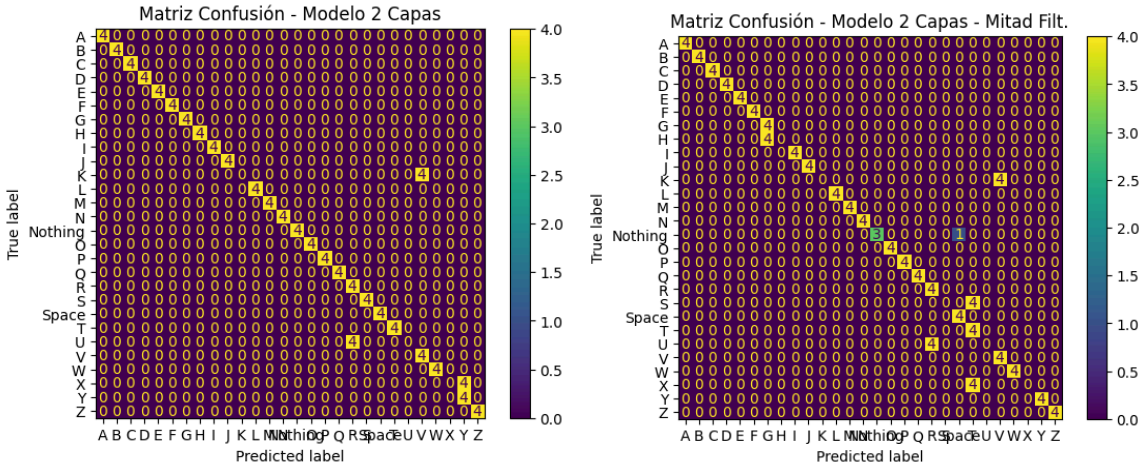

Ilustración 5. Matrices de confusión modelos de dos capas.

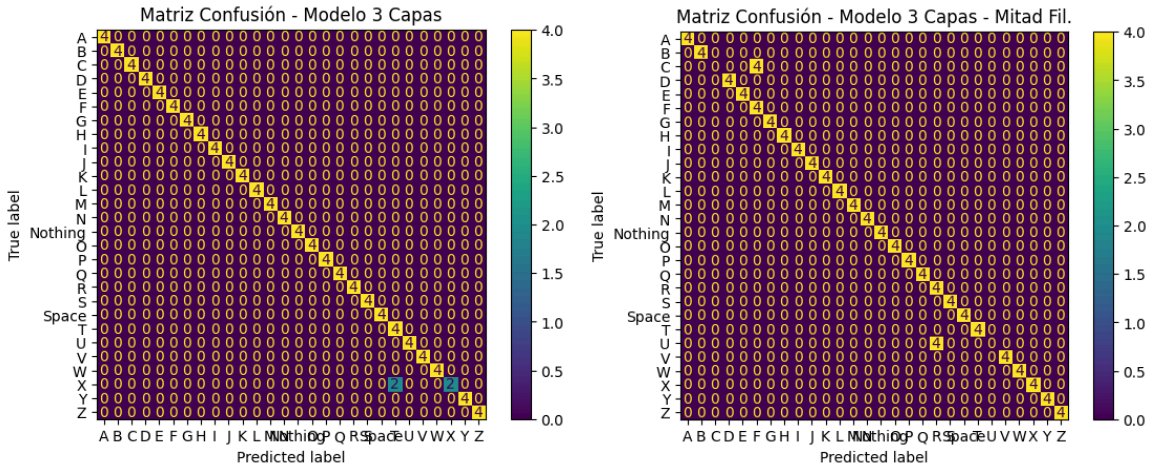

Ilustración 6. Matrices de confusión modelos de 3 capas

Finalmente, las matrices de confusión ilustran una tendencia similar a lo obtenido con las gráficas de accuracy, sin embargo, muestran un panorama más selectivo ya que se observa que cuando se usa el modelo de 3 capas convolucionales se obtiene el mejor resultado respecto a predicción de letras.
Una vez se define que el modelo con el mejor comportamiento es el de 3 capas con todos los filtros, se guarda el modelo para ser usado en otras predicciones.

PREDICCIONES

Después de seleccionar el mejor modelo se procedió a probar su funcionamiento en la predicción con imágenes nuevas. Para ello se tomaron tres fotografías nuevas con la señal correspondiente a la letra L:

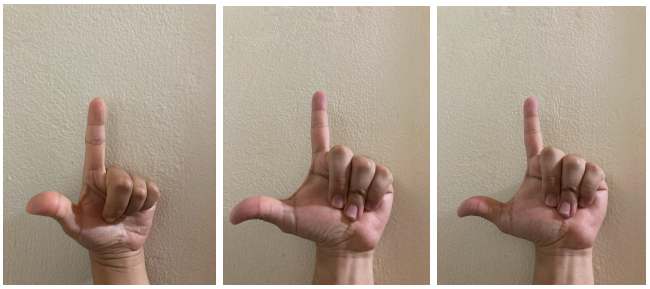

Ilustración 7. Imagenes nuevas Señal de Letra L

También se tomaron dos imágenes propias del conjunto de datos originales y se realizó la predicción utilizando el modelo entrenado:

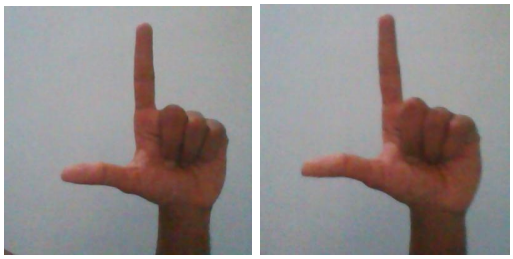

Ilustración 8. Imágenes del Conjunto de Datos Originales Señal de Letra L

Con el conjunto de imágenes anterior se procedió a realizar la predicción utilizando el modelo a continuación se resumen los resultados:

Imagen	Etiqueta Real	Etiqueta Predicha
Imagen Nueva 1	L	D
Imagen Nueva 2	L	D
Imagen Nueva 3	L	D
Imagen DataSet Original 1	L	L
Imagen DataSet Original 2	L	L

En la tabla anterior se resumen los resultados de predicción para imágenes nuevas y con imágenes propias del DataSet original, en términos generales se observa que la predicción falla para imágenes nuevas y es acertada para imágenes del conjunto original. Lo anterior permite inferir que el modelo se encuentra sobreajustado para el conjunto de datos originales y le hace falta generalización para nuevas imágenes.

BIBLIOGRAFIA

[1] I. A. Adeyanju, O. O. Bello and M. Adegboye, "Machine learning methods for sign language recognition: A critical review and analysis," *Intelligent Systems with Applications,* vol. 12, 2021.

[2] A. MEDIA, "AI MEDIA," [Online]. Available: https://www.ai-media.tv/knowledge-hub/insights/sign-language-alphabets. [Accessed 7 4 2024].

[3] K. Lea, "Kaggle," [Online]. Available: https://www.kaggle.com/datasets/kirlelea/spanish-sign-language-alphabet-static. [Accessed 3 4 2024].