**DuocUC**

Escuela de Informática y Telecomunicaciones



**Métricas De Rendimiento**

**“SignAI”**

**Equipo 1**

**Jenniffer Coñuel / Mattias Gonzalez / Ariel Silva**

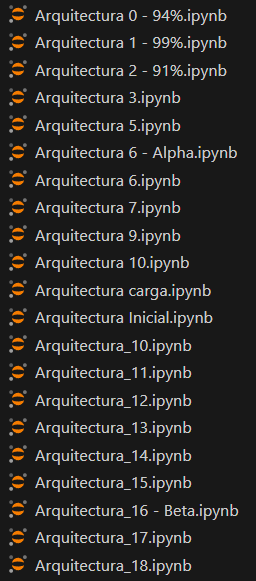
Capstone 002D

Profesora: Jazna Meza & Juan Pablo Mellado

**Fase 1: Experimentación**

Se diseñarán modelos CNN (redes neuronales convolucionales) y se probarán distintas arquitecturas para escoger la que tenga mejor desempeño en **set de pruebas** (y que no peque de **overfitting**) además de una buena generalización y buen desempeño en casos reales. Se espera un mínimo de **80% de accuracy** en set de pruebas y al menos un **50% en casos reales**.

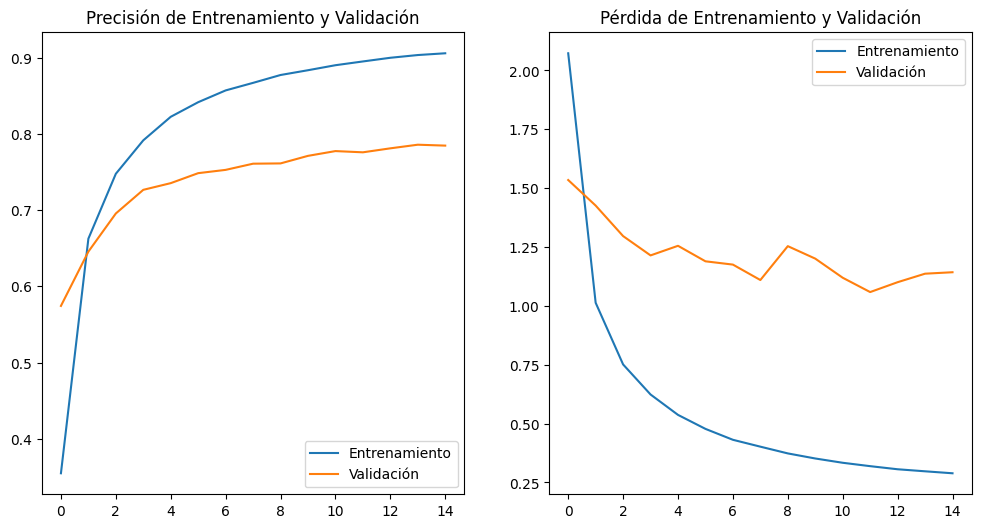
Se utilizará un **20% del set de datos** para destinarlos a pruebas (*set de pruebas*) y un aproximado de **240 imágenes de casos reales**, considerando diferentes escenarios, iluminación, nacionalidad, etcétera.

**Fase 2: Ejecución**

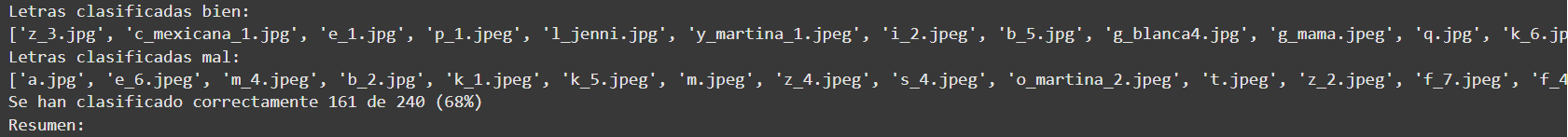
Se hicieron 18 arquitecturas diferentes con un dataset de 223.074 imágenes redimensionadas a una resolución 200x200.

Cabe destacar que cada arquitectura requirió entre 1 a 3 horas en los primeros intentos (*arquitecturas más sencillas*) y entre 8 a 10 horas (*en arquitecturas más profundas*) de ejecución para poder tener resultados tangibles.

La mejor arquitectura en cuanto a las métricas propuestas es la **16**, ya que esta obtuvo un accuracy del **90.56%** en el **set de pruebas** y aproximadamente un **69% en casos reales**.



Casos reales:



Con esto, podemos concluir que el modelo cumple con creces las expectativas para ser considerado la Base que dé paso a futuras mejoras e implementaciones.

**Fase 3: Intentos de arquitecturas**

| **Arquitectura 5**  Este modelo fue descartado desde un comienzo, en la época 4 ya que alcanzó un accuracy del 100%, lo cual consideramos irreal y las siguientes épocas nos dieron la razón, al haber fluctuaciones tan mínimas entre el **100%** y **96%**, es casi seguro que está **sobre ajustado**. | **Arquitectura 6**    Este modelo, al igual que el anterior fue descartado por un posible **overfitting**, aunque su pico máximo fue un **97%**. **¿Por qué overfitting?** El modelo no generalizaba para nada bien y únicamente detectaba las señas del set de datos o letras muy específicas con muchos arreglos (iluminación, fondo, etc). |
| --- | --- |
| **Arquitectura 7**  Al igual que anteriores modelos, este fue rápidamente descartado. Alcanzó un **94%** de **accuracy**, pero no generalizaba absolutamente nada. | **Arquitectura 9**    Éste modelo alcanzó un **81.6%** de **Accuracy**, lo cual está dentro del umbral de lo aceptable según lo estipulado, lastimosamente lo descartamos porque al igual que los otros, no clasificaba señas del todo bien. |
| **Arquitectura 10**  Esta arquitectura alcanzó un **70.8%** de **accuracy**, lo cual es bastante aceptable, lastimosamente, al igual que las demás arquitecturas, no generaliza bien. | **Arquitectura 11**  Esta arquitectura se descartó de forma inmediata debido a que no alcanzó el mínimo establecido (**80%**), logrando un modesto **75%** de **accuracy**. |
| **Arquitectura 12**  **Error**  Esta arquitectura no se pudo ejecutar debido a que daba error OOM (Out of Memory). Por ende se descartó, más no su diseño (archivo notebookpy). El archivo se conservó a modo de registro y porque no sabemos si es que tendrá buen o mal desempeño debido a la falta de recursos (entornos de ejecución) | **Arquitectura 13**  Esta arquitectura alcanzó el mínimo esperado, con un 83.2%, lo malo, es que este no daba buenos resultados en el ambiente de pruebas reales |
| **Arquitectura 14**  Esta arquitectura logró un 82.4% de accuracy y es la primera que empezó a detectar casos reales, pero de una forma ineficiente, logrando solo un 27% de aciertos en casos reales. | **Arquitectura 15**  Esta arquitectura logró 87.9% de accuracy y fue la segunda arquitectura en mostrar buenos resultados en casos reales, lastimosamente logró sólo un 30%. Esta arquitectura se descartó ya que la siguiente obtuvo resultados mucho mejores. |
| **Arquitectura 16**    Esta fue la arquitectura definitiva, la cual dejamos como modelo fijo para nuestro servicio beta. Alcanzó un accuracy del 91% y un 69% de casos reales clasificados correctamente. | **Arquitectura 17**  Esta arquitectura fue una variación de la anterior, por eso los gráficos son similares. Aunque es idéntica y lo único que cambió fue que se añadió una capa densa antes de la capa de output. Los resultados en casos reales fueron similares, únicamente bajando 1% y detectando señas que la anterior no, y también NO detectando señas que la anterior si. |
| **Arquitectura 18**  **ERROR**  Esta arquitectura no se pudo ejecutar porque se intentó subir la resolución del dataset a 400x400 pixeles y es por eso la omisión del gráfico. Creemos que podría dar un resultado ligeramente mejor que el anterior, lastimosamente se descartó por*OOM* (*Out of Memory*). | |

**Fase 4: Posibles Mejoras**

Mientras se realizaban las pruebas con casos reales, en las letras N, M, A, S y T. Se encontró que NO hay muchas clasificaciones exitosas, con un modesto 4 como máximo.

El modelo falla al clasificar estas señas ya que son muy similares las unas con las otras, habiendo únicamente cambios sutiles en la posición del dedo pulgar.

Las señas problemáticas son las siguientes:

| **Letra N** | **Letra M** | **Letra A** | **Letra S** | **Letra T** |
| --- | --- | --- | --- | --- |

Además de haber señas problemáticas que no son del todo generalizadas si no es con un fondo sin mucho ruido visual, hay señas las cuales detecta sin problema alguno independientemente del fondo, persona, iluminación, etc. Cómo las que se verán a continuación:

| **Letra L** | **Letra W** | **Letra Y** | **Letra U** | **Letra V** |
| --- | --- | --- | --- | --- |

Entre varias otras señas.

Posibles mejoras al modelo que hemos estado investigando y que planteamos son:

* Subir la resolución del Dataset de 200x200 a 400x400.
* Aumentar capas convolucionales.

Descartamos por ahora la segunda, ya que el modelo con las capas actuales es capaz de aprender patrones complejos en las muestras. Por ende, se recomienda a futuro subir la resolución del dataset de 200x200 a 400x400 o más, ésto para que el modelo tenga más espacio en la imagen para extraer características, por muy sutiles que sean.

Ésta opción significa un aumento en el costo computacional del modelo, por ende, este podría tardar mucho más en entrenarse o dar el error ***OOM*** (*Out of Memory*).

Es ideal que si se considera esta posibilidad, se cuente con un equipo más potente para realizar la tarea, ya que con el actual no se pudo.

El equipo actual consta de:

* GPU RTX 2060 6GB
* CPU Intel(R) Core(TM) i5-10300H 2.50GHz
* 16GB RAM