Trabajo Práctico Nº 3

Bugnon, Leandro Pineda, Leandro Rodriguez, Tadeo

El desarrollo de las redes neuronales profundas, mediante la incorporación de capas convolutivas, ha significado un avance sustancial en el área de aprendizaje supervisado de imagenes. Por ejemplo, la red neuronal profunda esquematizada en la Figura logra una exactitud 99,25 % sobre un conjunto de datos de 10 clases que representan digitos manuscritos (MNIST).

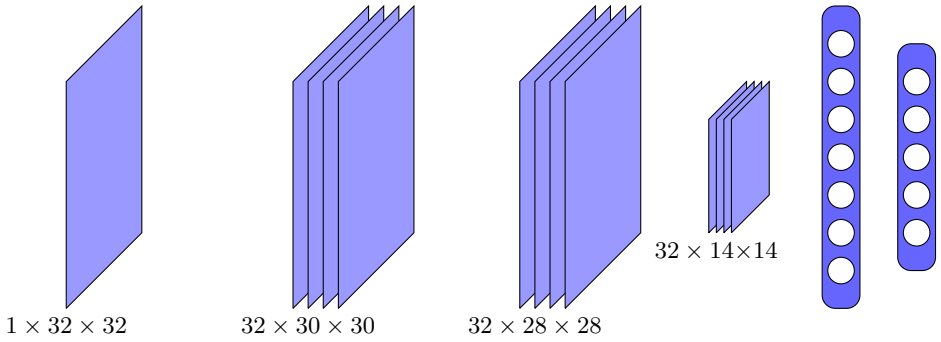

Figura 1: Red neuronal profunda MNIST. 99,25 % de exactitud.

Considerando el ejemplo de los digitos, se desea hacer lo mismo con caracteres de un texto manuscrito. Dicho conjunto de datos fue extraído de imagenes de textos completos, que fueron posteriormente segmentados. Dicha segmentación es aproximada, pues no es posible dividir cada uno de los caracteres en forma apropiada. Por otra parte, los textos fueron escritos por diferentes individuos, lo que implica un dataset de gran tamaño y diversidad. A su vez, se optado por separar los autores, un conjunto será destinado al entrenamiento/validación y otro exclusivamente al testeo. De esta manera el clasificador tendrá el desafio de clasificador ejemplos que en ningún momento utilizo durante se entrenamiento, ni si similares que hubieren provenido del mismo escritor.

A tal fin, se parte del modelo propuesto para MNIST con la modificaciones necesarias para ser utilizada en este conjunto de datos. En particular, se modifica la última capa de neuronas cuya arquitectura esta definida por la cantidad de clases. En este caso existen 91 clases en lugar de las 10 del ejemplo anterior. Los resultados de este sencillo modelo pueden ser apreciados en la Figura 2. Lo obtenido difiere significamente de lo logrado en MNIST – por la cantidad de clases y el desbalance de las mismas, entre otros motivos– y representa un punto de partida sobre el cual realizar modificaciones para así lograr una mejor performance. Por lo tanto, la estrategia consistirá en modificar individualmente cada uno de los paraámetros del modelo, indagando el efecto de cada uno en la performance general. Finalmente, a partir del conocimiento inferido, se buscara exhaustivamente una arquitectura apropiada para el problema de interés.

En primer lugar, la estrategia que se adopta consiste en realizar un preprocesamiento de los datos. En particular se opta por realizar un escalado de los datos, asi como una transformación en escala de grises. A tal fin se utilizan las herramientas de preprosamiento de Keras. Esta rutina logra una mejora en la performance, de alrededor de 5 puntos con respecto al modelo original, como se puede apreciar al comparar la Figura 3 con la anterior. Asimismo, este procedimiento es normal, por lo que se decide incluirla en el resto de las realizaciones del método.

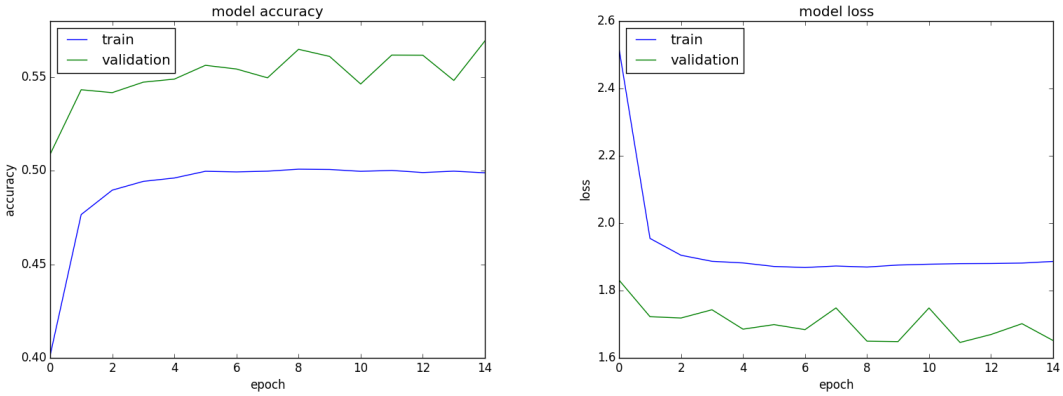

Figura 2: Exactitud y perdida. Red basada en MNIST.

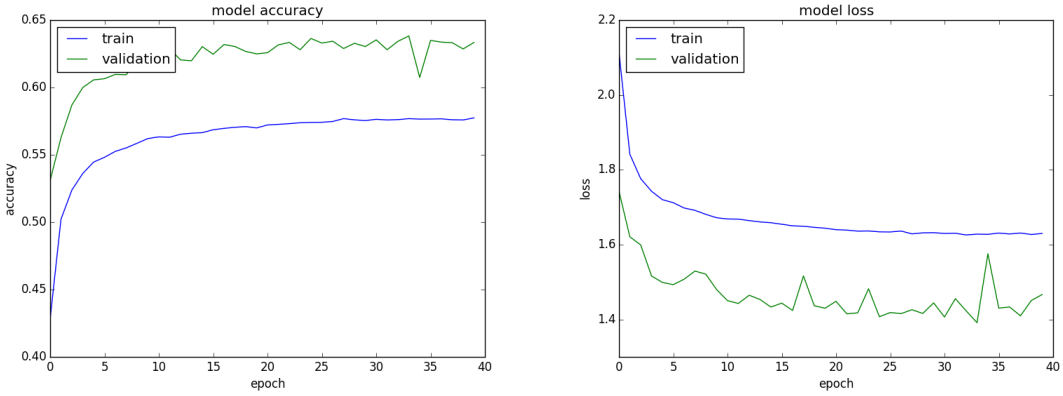

Figura 3: Exactitud y perdida. Aplicando preprocesamiento.

Por otra parte, resulta de interés analizar el comportamiento de la red, con un porcentaje de los datos de entrenamiento. Esta observación nos permite prevenir defectos en el modelo como es el caso del overfitting o similares. En este caso se utilizó un 25 % de los datos disponibles para entrenamiento. Como se ve en la Figura 4, esta disminución no representa una mejoría del desempeño respecto al caso anterior. Esta situación puede entenderse debido a la presencia del Dropout, el cual previene de los inconvenientes mencionados.

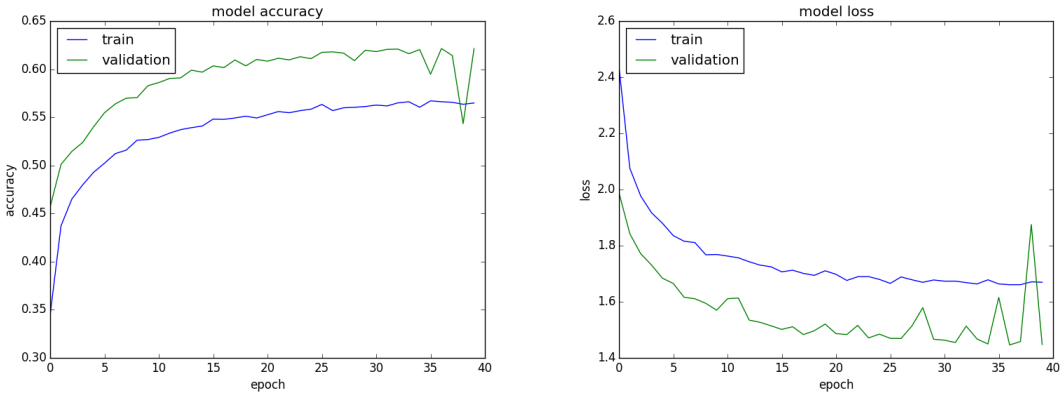

Figura 4: Exactitud y perdida. Red original utilizando un 25 % de los datos de entrenamiento.

Uno de los parámetros mas influyentes de una red neuronal profunda es la arquitectura que toman sus capas convolutivas. La cantidad de filtros y el tamaño de los mismos así como el número de capas convolutivas juegan papel preponderante en el diseño y funcionamiento de este tipo de red. En particular se medirá el desempeño de estas, en el caso extremo de una sola capa convolucional y en un caso con 5 capas convolutivas. En ambos, las capas poseen 32 filtros de 3×3 pixeles. Estan colocados de manera secuencial, y seguidos por una capa de Maxpooling, en forma similar al modelo original.

El resultado de ambas se puede apreciar en las Figuras 5 y 6. Allí se observa que la sustracción de una capa respecto al modelo original implica, una perdida de alrededor de 5 puntos porcentuales de exactitud. En cambio, la adición de capas permite aumentar en 5 puntos, respecto al modelo con preprocesamiento.

El otro parámetro mas significativo de este tipo de red, se trata del clasificador mismo, las capas completamente conectadas al final de la red. Estas representan casi la totalidad de parámetros que deben ser ajustados en el modelo, influyendo no solo en el desempeño sino tambieén en el tiempo de entrenamiento. El modelo actual, consta de una sola capa oculta con 100 neuronas. La estrategia que se adopta, y cuyos resultados se muestran en Figura 7, es reducir la cantidad neuronas a 32. Por otro lado se desea conocer el efecto de tener mayor cantidad de capas ocultas. En la Figura 8 se ve la performance de una red con 3 capas ocultas de 256, 128 y 64 neuronas cada una. Claramente ambas opciones representan un decaimiento en la calidad de la clasificación, siendo la mas significativa para el caso de la disminución de neuronas.

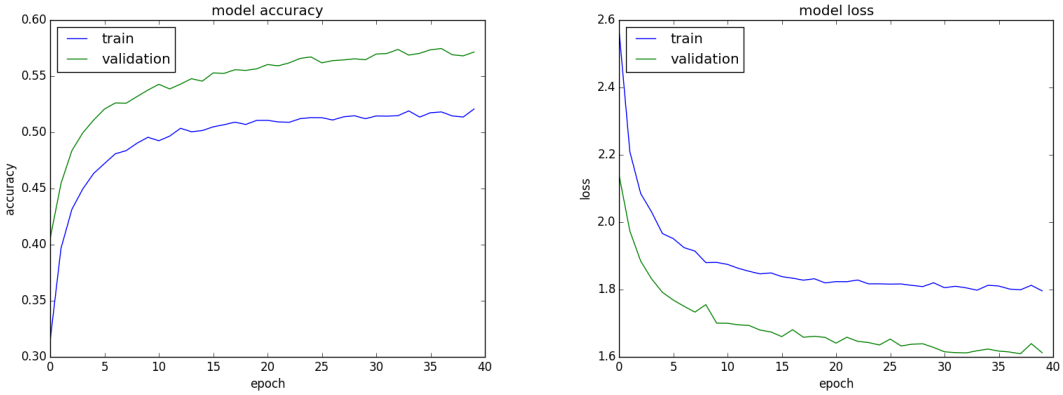

Figura 5: Exactitud y perdida. Red con una sola capa convolutiva

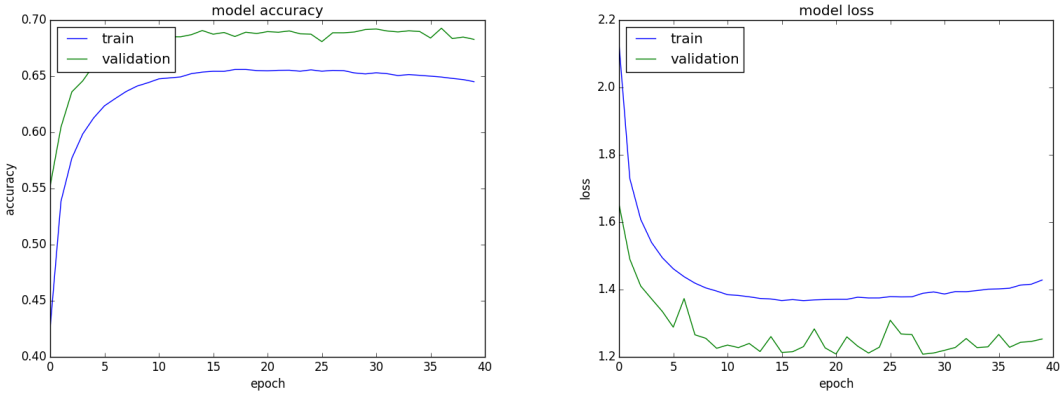

Figura 6: Exactitud y perdida. Red con 5 capas convolutivas.

Otra elección que puede influir en el funcionamiento del modelo es la función de activación escogida. En la red base, todas las activaciones son del tipo $ReLU$. Otra posibilidad es utilizar funciones sigmoideas es decir la función:

$$\text{sigmoid}(x) = \frac{1 + \tanh x}{2} = \frac{e^x}{e^x + e^{-x}}.$$

El efecto de esta sustitución se puede apreciar en la Figura 9 donde se observa un fuerte decaimiento en la calidad del clasificador, desempeñandose aún peor que el modelo sin preprocesamiento. De esta manera se puede afirmar que la RELU será una mejor opción en el momento de construir un modelo con el mayor desempeño posible para esta dataset.

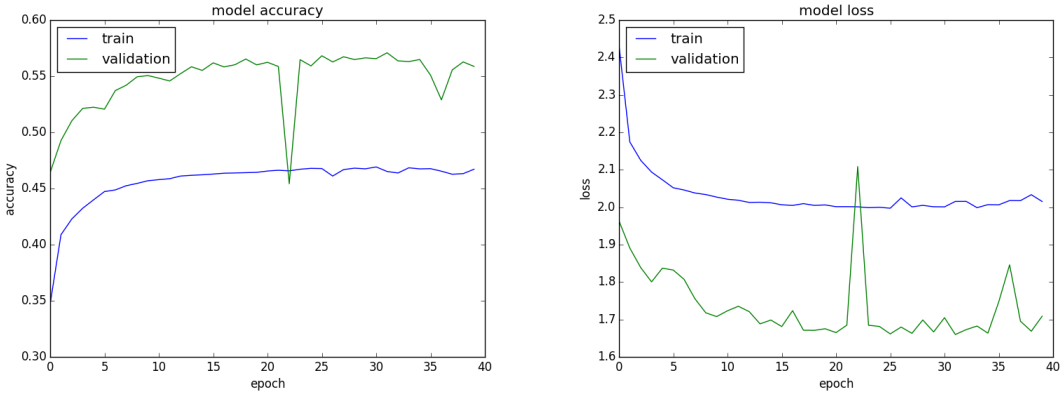

Figura 7: Exactitud y perdida. Red con una capa totalmente conectada(32).

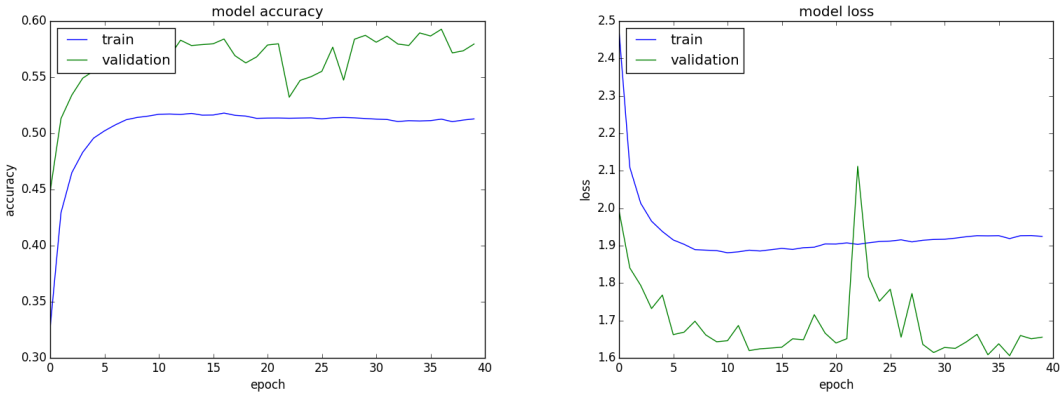

Figura 8: Exactitud y perdida. Red con 3 capas totalmente conectadas(256,128,64).

Finalmente, otra característica que se desea analizar es el efecto del Dropout. Este tiende a disminuir los errores por overfitting, sin embargo, resulta de interés conocer si el nivel es apropiado o, incluso, si su efecto es en realidad negativo en la cosntrucción del modelo.

Las curvas en las Figuras 10 y 11 dejan en claro que aumentar el Dropout empeora el comportamiento del modelo, mientras que eliminarlo significa un aumento significativo respecto al modelo base pero, posiblemente, sensible al overfitting

En virtud de lo expuesto previamente se desarrollo una red profunda con la arquitectura propuesta en la Figura . En todos los casos se utilizo como función de activación una $ReLU$ tradicional. Asimismo, al inicio de la red los filtros poseen un tamaño de 3×3, mientras que en las capas totalmente conectadas se aplico un Dropout al nivel de 0,3. Por otra parte las capas de MaxPooling reducen el tamaño de la entrada a su mitad.

Los niveles alcanzados para esta arquitectura pueden observarse Figura 12 , donde se logra un máximo alrededor del 79 %.

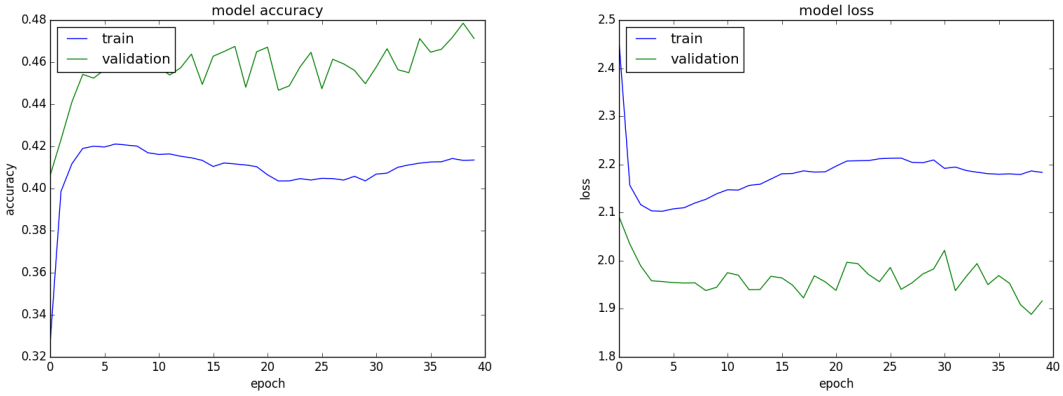

Figura 9: Exactitud y perdida. Red con funciones sigmoideas en la activación.

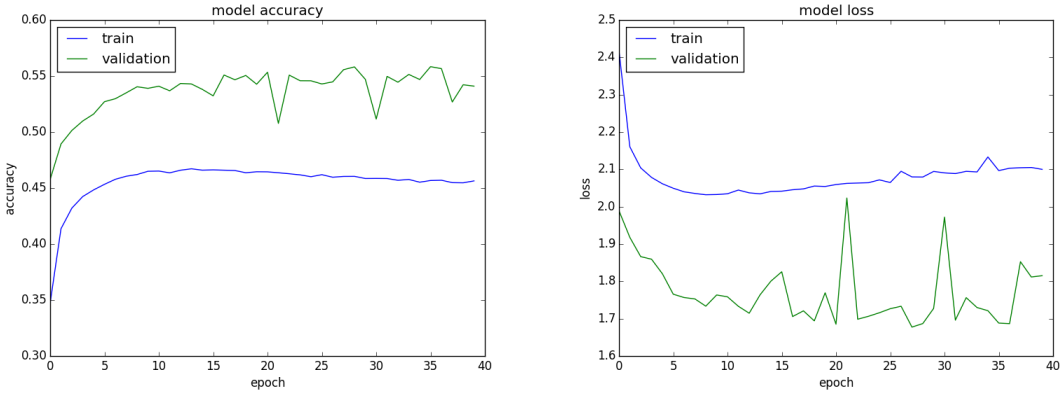

Figura 10: Exactitud y perdida. Red con nivel de Dropout aumentado.

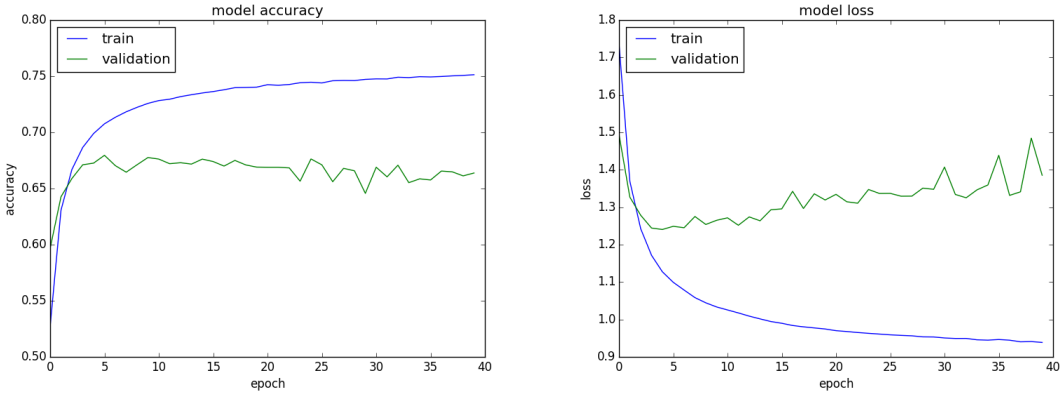

Figura 11: Exactitud y perdida. Red sin efecto de Dropout.

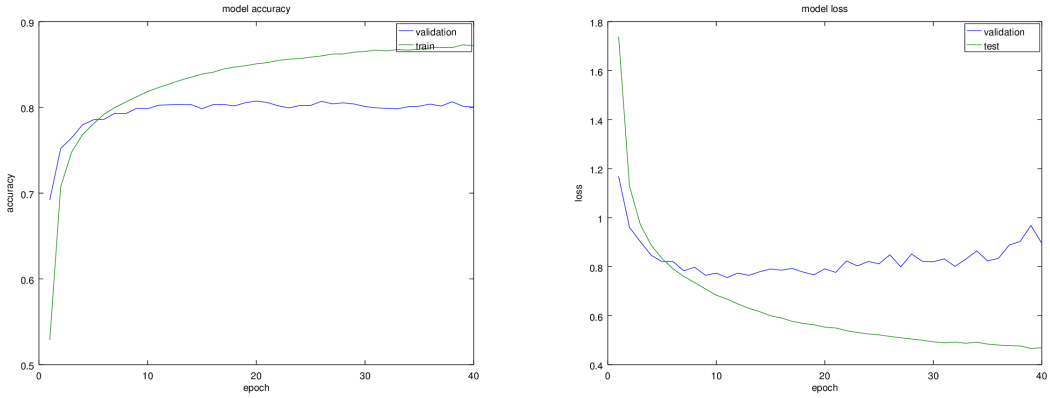

Figura 12: Exactitud y perdida. Modelo propuesto.

```
Convolution2D(32)
BatchNormalization()
Convolution2D(32)
MaxPooling2D()
Convolution2D(32)
Convolution2D(32)
MaxPooling2D()
Convolution2D(32)
Convolution2D(64)
Convolution2D(128)
Convolution2D(128)
MaxPooling2D()
Flatten()
Dense(2048)
Dense(1024)
Dense(512)
Dense(91)
softmax()
```

Figura 13: Arquitectura propuesta.