REDES DE NEURONAS ARTIFICIALES

**REDES DE NEURONAS ARTIFICIALES:**

**PRÁCTICA 2 PARTE 2**

Pablo Salcedo Arroyo

100408773

Jorge Sousa Herranz

100408805

Ángel Del Viejo Casillas

100405898

****

**ÍNDICE**

1. **INTRODUCCIÓN**

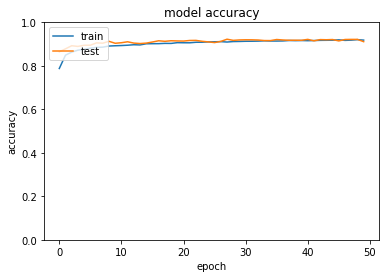
En este documento vamos a explicar todo lo que hemos realizado en la parte 2 de la práctica 2 de la asignatura. En esta segunda parte vamos a comparar el funcionamiento de un perceptrón multicapa y el de una Red de Neuronas Convolucionales(CNN) en la clasificación de imágenes, el PM es una red más genérica que ya hemos estudiado en la parte 1 de esta práctica y en la práctica 1 de la asignatura, mientras que la CNN es una red más orientada a este tipo de trabajo. Vamos a buscar los mejores modelos de ambos tipos y vamos a compararlos para ver cuál ha realizado la tarea de clasificación de mejor manera.

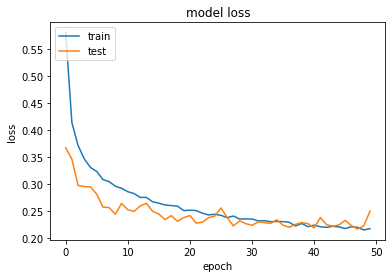
1. **Parte 2:**
   1. **Pruebas**

| **CNN** | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **N.º de capas** | **N.º filtros por capa** | **Tamaño del kernel por capa** | **Dropout por capa** | **N.º de ciclos** | **loss** | **sparse\_categorical\_accuracy** | **val\_loss** | **val\_sparse\_categorical\_accuracy** |
| **1** | 32 | 3x3 | 0.25 | 10 | 0.2357 | 0.9125 | 0.2429 | 0.9112 |
| **1** | 32 | 3x3 | 0.25 | 30 | 0.2367 | 0.9114 | 0.2446 | 0.9111 |
| **1** | 32 | 3x3 | 0.5 | 10 | 0.2387 | 0.9114 | 0.2409 | 0.9144 |
| **1** | 32 | 3x3 | 0.5 | 30 | 0.2098 | 0.9211 | 0.2345 | 0.9177 |
| **1** | 32 | 5x5 | 0.5 | 30 | 0.1981 | 0.9257 | 0.2369 | 0.9164 |
| **1** | 32 | 5x5 | 0.5 | 50 | 0.1953 | 0.9275 | 0.2373 | 0.9162 |
| **1** | 32 | 5x5 | 0.75 | 30 | 0.2678 | 0.9007 | 0.2397 | 0.9120 |
| **1** | 32 | 5x5 | 0.75 | 50 | 0.2460 | 0.9088 | 0.2312 | 0.9160 |
| **1** | 64 | 3x3 | 0.5 | 100 | 0.2188 | 0.9180 | 0.2401 | 0.9141 |
| **1** | 64 | 5x5 | 0.5 | 100 | 0.2234 | 0.9164 | 0.2484 | 0.9093 |
| **1** | 128 | 3x3 | 0.5 | 100 | 0.2064 | 0.9236 | 0.2489 | 0.9118 |
| **1** | 128 | 5x5 | 0.5 | 100 | 0.2062 | 0.9225 | 0.2464 | 0.9155 |
| **1** | 256 | 3x3 | 0.5 | 100 | 0.2370 | 0.9107 | 0.2534 | 0.9093 |
| **1** | 256 | 5x5 | 0.5 | 100 | 0.2205 | 0.9187 | 0.2522 | 0.9116 |
| **2** | 32 | 3x3  3x3 | 0.5  0.5 | 50 | 0.3037 | 0.8871 | 0.2423 | 0.9129 |
| **2** | 32 | 3x3  5x5 | 0.5  0.5 | 50 | 0.3210 | 0.8811 | 0.2474 | 0.9088 |
| **2** | 32 | 5x5  5x5 | 0.5  0.5 | 50 | 0.3094 | 0.8871 | 0.2419 | 0.9112 |
| **2** | 64 | 3x3  3x3 | 0.5  0.5 | 50 | 0.2622 | 0.9034 | 0.2239 | 0.9199 |
| **2** | 64 | 3x3  5x5 | 0.5  0.5 | 50 | 0.2722 | 0.9006 | 0.23 | 0.9123 |
| **2** | 64 | 5x5  5x5 | 0.5  0.5 | 50 | 0.2588 | 0.9040 | 0.2266 | 0.9177 |
| **2** | 128 | 3x3  3x3 | 0.5  0.5 | 50 | 0.2217 | 0.9167 | 0.2192 | 0.9205 |
| **2** | 128 | 3x3  5x5 | 0.5  0.5 | 50 | 0.2325 | 0.9145 | 0.228 | 0.9190 |
| **2** | 128 | 5x5  5x5 | 0.5  0.5 | 50 | 0.2199 | 0.9192 | 0.2213 | 0.9225 |
| **2** | 256 | 3x3  3x3 | 0.5  0.5 | 50 | 0.1995 | 0.9270 | 0.2152 | 0.9262 |
| **2** | 256 | 3x3  5x5 | 0.5  0.5 | 50 | 0.2075 | 0.9229 | 0.2259 | 0.9199 |
| **2** | 256 | 5x5  5x5 | 0.5  0.5 | 50 | 0.2144 | 0.9195 | 0.2218 | 0.9207 |
| **3** | 32 | 3x3  3x3  3x3 | 0.5  0.5  0.5 | 50 | 0.6141 | 0.7799 | 0.5079 | 0.8369 |
| **3** | 32 | 4x4  3x3  2x2 | 0.5  0.5  0.5 | 50 | 0.49 | 0.8240 | 0.3456 | 0.8697 |
| **3** | 32 | 5x5  4x4  1x1 | 0.5  0.5  0.5 | 50 | 0.4782 | 0.8259 | 0.3555 | 0.8717 |
| **3** | 32 | 5x5  5x5  1x1 | 0.5  0.5  0.5 | 50 | 0.4596 | 0.8344 | 0.3452 | 0.8763 |
| **3** | 32 | 4x4  4x4  1x1 | 0.5  0.5  0.5 | 50 | 0.5278 | 0.8052 | 0.3892 | 0.8568 |
| **3** | 32 | 5x5  2x2  2x2 | 0.5  0.5  0.5 | 50 | 0.5103 | 0.8138 | 0.3766 | 0.8558 |
| **3** | 64 | 3x3  3x3  3x3 | 0.5  0.5  0.5 | 50 | 0.4624 | 0.8355 | 0.3580 | 0.8742 |
| **3** | 64 | 4x4  3x3  2x2 | 0.5  0.5  0.5 | 50 | 0.4064 | 0.8543 | 0.3207 | 0.8885 |
| **3** | 64 | 5x5  5x5  1x1 | 0.5  0.5  0.5 | 50 | 0.5197 | 0.8076 | 0.3737 | 0.8594 |
| **3** | 128 | 3x3  3x3  3x3 | 0.5  0.5  0.5 | 50 | 0.3864 | 0.8620 | 0.3014 | 0.8867 |
| **3** | 128 | 4x4  3x3  2x2 | 0.5  0.5  0.5 | 50 | 0.3446 | 0.8752 | 0.2536 | 0.9056 |
| **3** | 128 | 5x5  4x4  1x1 | 0.5  0.5  0.5 | 50 | 0.3381 | 0.8773 | 0.2695 | 0.8992 |
| **3** | 128 | 5x5  5x5  1x1 | 0.5  0.5  0.5 | 50 | 0.3257 | 0.8806 | 0.2661 | 0.9043 |
| **3** | 256 | 3x3  3x3  3x3 | 0.5  0.5  0.5 | 50 | 0.3148 | 0.8960 | 0.2459 | 0.9034 |
| **3** | 256 | 4x4  3x3  2x2 | 0.5  0.5  0.5 | 50 | 0.3210 | 0.8816 | 0.2473 | 0.9081 |
| **3** | 256 | 5x5  4x4  1x1 | 0.5  0.5  0.5 | 50 | 0.3285 | 0.88 | 0.2638 | 0.9006 |
| **3** | 256 | 5x5  5x5  1x1 | 0.5  0.5  0.5 | 50 | 0.3031 | 0.8888 | 0.2525 | 0.9069 |

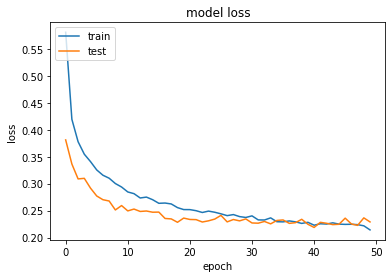
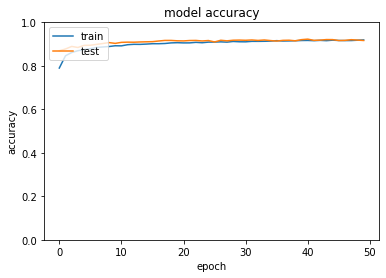
En las primeras pruebas con 1 capa y 32 filtros por capa queríamos sacar el mejor valor del Dropout, ya que debido al límite de páginas no podemos hacer todas las combinaciones posibles por lo que hemos decidido sacar el mejor valor y dejarlo fijo para el resto de experimentos y lo mismo para el número de ciclos. Después con 2 capas queríamos ver los resultados con capas muy homogéneas en cuanto al kernel, es decir los 2 valores iguales o un poco diferentes, y ya después con 3 capas comparar experimentos con capas iguales, muy parecidas o con mucha diferencia en los tamaños de los kernel.

Tras todos estos experimentos de la CNN hemos concluido que los mejores valores son los siguientes, que además vamos a usar como modelo de entrada para el PM para obtener la mejor tasa de aprendizaje y obtener el mejor modelo final y poder realizar la comparación más completa posible:

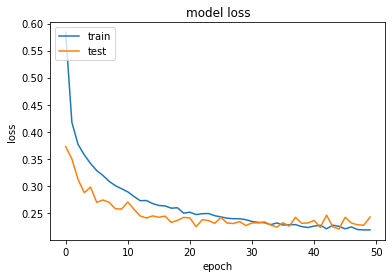
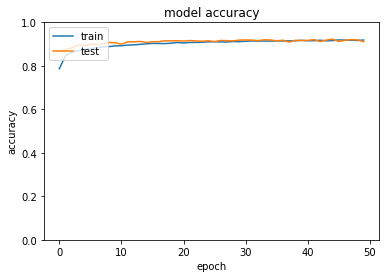
* 2 capas con un dropout de 0.5 en cada capa y 50 ciclos con los siguientes tamaños de kernel y filtros por capa:
  + **128 filtros, 3x3, 3x3**



* + **128 filtros, 5x5, 5x5**



* **256 filtros, 3x3, 3x3**



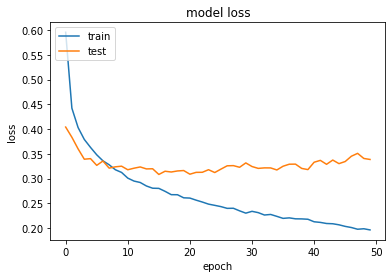
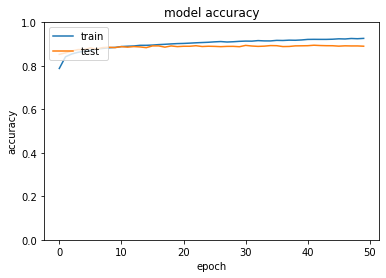
Analizando estas gráficas y la tabla podemos observar que prácticamente todos los experimentos de 2 capas y 64 o más filtros van a producir los mismos resultados por lo que podríamos considerar cualquier experimento entre esos como el mejor.

La precisión no varía mucho salvo en los primeros ciclos, cuando aún no tiene el modelo más exacto y va ajustando los parámetros hasta encontrar un modelo bastante preciso. Mientras tanto el fallo vemos que en la fase de entrenamiento, se produce una oscilación, eso pasa porque el criterio de parada es que ejecute todos los ciclos, entonces al seguir ejecutando el modelo seguirá buscando mejores modelos, pero algunos de ellos obtienen una tasa de fallo mayor.

Podemos concluir que el mejor número de capas es 2 y cuantos más filtros mejor por capa mejores resultados vamos a obtener. También cuando las capas tienen kernel de igual tamaño siempre va a ser preferible a distinto kernel en cada capa. Solo vamos a probar los 3 mejores modelos de la CNN como entrada para el PM, para poder realizar una mejor comparación entre ambos modelos, al hacerlo entre los mismos modelos, y analizar las diferencias en las mismas condiciones.

| **MODELO DE ENTRADA PM** | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Nombre del modelo de entrada** | **N.º de capas** | **N.º filtros por capa** | **Tamaño del kernel por capa** | **Dropout por capa** | **Ciclos** |
| **1** | **2** | 128 | 3x3  3x3 | 0.5  0.5 | 50 |
| **2** | **2** | 128 | 5x5  5x5 | 0.5  0.5 | 50 |
| **3** | **2** | 256 | 3x3  3x3 | 0.5  0.5 | 50 |

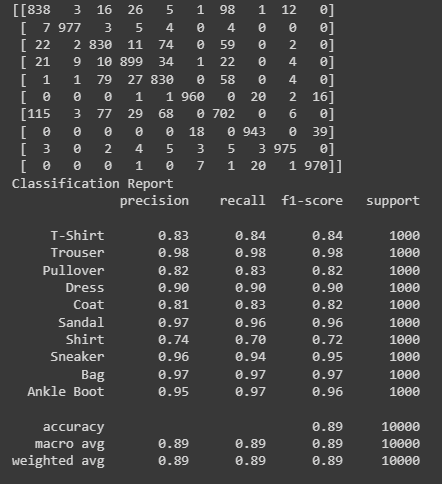
| **PERCEPTRÓN MULTICAPA** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Modelo** | **Tasa de aprendizaje** | **N.º ciclos** | **loss** | **sparse\_categorical\_accuracy** | **val\_loss** | **val\_sparse\_categorical\_accuracy** |
| **1** | **0.0001** | 50 | 0.1863 | 0.9451 | 0.3386 | 0.8903 |
| **0.001** | 50 | 0.1934 | 0.9328 | 0.3242 | 0.9101 |
| **0.005** | 50 | 0.2805 | 0.8944 | 0.3198 | 0.8836 |
| **0.01** | 50 | 0.2827 | 0.8872 | 0.3223 | 0.8904 |
| **2** | **0.0001** | 50 | 0.2246 | 0.9167 | 0.2434 | 0.9112 |
| **0.001** | 50 | 0.2606 | 0.9031 | 0.3089 | 0.8901 |
| **0.005** | 50 | 0.3279 | 0.8832 | 0.3238 | 0.8855 |
| **0.01** | 50 | 0.3359 | 0.8774 | 0.3355 | 0.8770 |
| **3** | **0.0001** | 50 | 0.2802 | 0.8956 | 0.3084 | 0.8917 |
| **0.001** | 50 | 0.2950 | 0.8902 | 0.3210 | 0.8859 |
| **0.005** | 50 | 0.2093 | 0.9217 | 0.3289 | 0.8934 |
| **0.01** | 50 | 0.1977 | 0.9258 | 0.3510 | 0.8915 |



| **MODELO FINAL** | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Modelo** | **Tasa de aprendizaje** | **N.º ciclos** | **N.º neuronas** | **loss** | **sparse\_categorical\_accuracy** |
| **2** | **0.0001** | 50 | 25  25 | 0.2385 | 0.9134 |
| 25  50 | 0.2360 | 0.9122 |
| 25  100 | 0.2162 | 0.9203 |
| 50  50 | 0.1595 | 0.9427 |
| 50  100 | 0.1472 | 0.9460 |
| 50  150 | 0.1448 | 0.9477 |
| 100  100 | 0.083 | 0.9711 |
| 100  150 | 0.0781 | 0.9729 |
| 150  150 | 0.0483 | 0.9837 |

Para el modelo final podemos concluir que si seguimos aumentando el número de neuronas de las capas vamos a seguir mejorando el modelo, pero no podemos considerar que es mejor el PM que la CNN, ya que no tenemos el error de validación.

Si observamos los resultados del PM vemos que la tasa de error es superior a la de CNN, eso es algo bastante lógico, ya que CNN es una red especializada en clasificación de imágenes por lo que sería muy complicado encontrar un modelo mejor a pesar de las limitaciones que tiene la CNN en esta práctica.

****

Aquí podemos observar la matriz de confusión, en la que en la diagonal principal se encuentran todos los verdaderos positivos de cada categoría y vemos que son todos valores muy altos y muy pocas predicciones mal realizadas. En cuanto al classification report podemos ver que todas las categorías tienen un f1-score por encima de 0.8, lo que indica un buen modelo, salvo 1, que es Shirt(Camisa), que es entendible, ya que se puede parecer y confundir con un abrigo o un jersey en imágenes de 28x28.

1. **Conclusiones parte 2:**

Esta segunda parte nos ha servido para trabajar y comprender el tratamiento de imágenes por parte de una red neuronal y poder comparar el rendimiento de una red específica como CNN o una más genérica como el Perceptrón Multicapa. También nos hemos dado cuenta de que el PM es una red bastante potente, ya que la diferencia con la CNN en cuanto al error de validación es bastante pequeña considerando que es una red genérica con muchos usos.