Alumno: Jorge Luis Toral Gamez

Repositorio: https://github.com/Jorge-1501/Redes-Neuronales.git

Código de la primera tarea

In [2]:
```
# Librerías

import mnist_loader
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:
```
import network as nw
```

In [4]:
```
# Descarga de información

training_data, validation_data, test_data = \
mnist_loader.load_data_wrapper()
```

In [3]:
```
# Entrenamiento de red

net = nw.Network([784, 30, 10])
net.SGD(training_data, 30, 10, 0.01, test_data=test_data)
```

```
Época 0: 1103 / 10000
Época 1: 1356 / 10000
Época 2: 1561 / 10000
Época 3: 1848 / 10000
Época 4: 2122 / 10000
Época 5: 2365 / 10000
Época 6: 2563 / 10000
Época 7: 2741 / 10000
Época 8: 2901 / 10000
Época 9: 3066 / 10000
Época 10: 3234 / 10000
Época 11: 3382 / 10000
Época 12: 3539 / 10000
Época 13: 3670 / 10000
Época 14: 3828 / 10000
Época 15: 3999 / 10000
Época 16: 4243 / 10000
Época 17: 4466 / 10000
Época 18: 4660 / 10000
Época 19: 4800 / 10000
Época 20: 4930 / 10000
Época 21: 5042 / 10000
Época 22: 5145 / 10000
Época 23: 5226 / 10000
Época 24: 5338 / 10000
Época 25: 5461 / 10000
Época 26: 5589 / 10000
Época 27: 5699 / 10000
Época 28: 5814 / 10000
Época 29: 5914 / 10000
```

Código de la segunda tarea

Implementación de optimizador SGD con momento

La siguiente línea se realizó porque estuve trabajando en otra carpeta y no podía ver las actualizaciones que hacía en network.py

In [8]:
```
from importlib import reload
import network as nw
reload(nw)
```

Out[8]:
```
<module 'network' from '/home/jorge_t/Documentos/Física/Redes Neuronales/Proyecto_1/Redes-Neuronales/Digitos/src/network.py'>
```

In [17]:
```
net2 = nw.Network([784, 30, 10])

# Se dejó el nombre de SGD para no alterar todo el código, pero es SGD con m
net2.SGD_M(training_data, 30, 10, 0.005, 0.95, test_data=test_data)
```

```
Época 0: 1094 / 10000
Época 1: 1284 / 10000
Época 2: 1287 / 10000
Época 3: 1272 / 10000
Época 4: 1242 / 10000
Época 5: 1261 / 10000
Época 6: 1239 / 10000
Época 7: 1232 / 10000
Época 8: 1224 / 10000
Época 9: 1204 / 10000
Época 10: 1198 / 10000
Época 11: 1196 / 10000
Época 12: 1192 / 10000
Época 13: 1187 / 10000
Época 14: 1181 / 10000
Época 15: 1167 / 10000
Época 16: 1159 / 10000
Época 17: 1162 / 10000
Época 18: 1162 / 10000
Época 19: 1154 / 10000
Época 20: 1149 / 10000
Época 21: 1147 / 10000
Época 22: 1148 / 10000
Época 23: 1149 / 10000
Época 24: 1152 / 10000
Época 25: 1146 / 10000
Época 26: 1145 / 10000
Época 27: 1143 / 10000
Época 28: 1149 / 10000
Época 29: 1151 / 10000
```

In [8]:
```
net2 = nw.Network([784, 30, 10])

# Se dejó el nombre de SGD para no alterar todo el código, pero es SGD con m
net2.SGD_M(training_data, 30, 10, 0.0005, 0.95, test_data=test_data)
```

```
Época 0: 1120 / 10000
Época 1: 1143 / 10000
Época 2: 1146 / 10000
Época 3: 1197 / 10000
Época 4: 1240 / 10000
Época 5: 1264 / 10000
Época 6: 1281 / 10000
Época 7: 1305 / 10000
Época 8: 1325 / 10000
Época 9: 1358 / 10000
Época 10: 1366 / 10000
Época 11: 1376 / 10000
Época 12: 1397 / 10000
Época 13: 1406 / 10000
Época 14: 1421 / 10000
Época 15: 1425 / 10000
Época 16: 1441 / 10000
Época 17: 1452 / 10000
Época 18: 1464 / 10000
Época 19: 1467 / 10000
Época 20: 1470 / 10000
Época 21: 1478 / 10000
Época 22: 1494 / 10000
Época 23: 1506 / 10000
Época 24: 1512 / 10000
Época 25: 1520 / 10000
Época 26: 1525 / 10000
Época 27: 1535 / 10000
Época 28: 1550 / 10000
Época 29: 1560 / 10000
```

Se logró encontrar una predicción lenta, pero sin aparente convergencia. Después de varias pruebas no logré modificar a una mejor predicción este optimizador, sin embargó logré que mantuviera un crecimiento de predicciones contra los demás intentos con rebotes de valores. Al final otuvimos una reducción del **80.5 %** de los valores de predicción

Al aumentar η empezaban a rebotar los valores y al reducir tardaban en avanzar. Por el lado de momento, tanto al reducir como bajar del valor propuesto $momentum = 0.95$ la cantidad de predicciones se reducía.

RMSprop

In [19]:
```
net3 = nw.Network([784, 30, 10])

net3.RMSprop(training_data, 30, 10, 0.001, 0.9, 1e-8, test_data=test_data)
```

```
Época 0: 1130 / 10000
Época 1: 1528 / 10000
Época 2: 1618 / 10000
Época 3: 1819 / 10000
Época 4: 1939 / 10000
Época 5: 1915 / 10000
Época 6: 1969 / 10000
Época 7: 1979 / 10000
Época 8: 2078 / 10000
Época 9: 2045 / 10000
Época 10: 2096 / 10000
Época 11: 2014 / 10000
Época 12: 2095 / 10000
Época 13: 2194 / 10000
Época 14: 2147 / 10000
Época 15: 2102 / 10000
Época 16: 2085 / 10000
Época 17: 2115 / 10000
Época 18: 2168 / 10000
Época 19: 2188 / 10000
Época 20: 2096 / 10000
Época 21: 2059 / 10000
Época 22: 2117 / 10000
Época 23: 2228 / 10000
Época 24: 2194 / 10000
Época 25: 2207 / 10000
Época 26: 2188 / 10000
Época 27: 2180 / 10000
Época 28: 2189 / 10000
Época 29: 2177 / 10000
```

In [23]:
```
net4 = nw.Network([784, 30, 10])

net4.RMSprop(training_data, 30, 10, 0.0001, 0.90, 1e-8, test_data=test_data)
```

```
Época 0: 852 / 10000
Época 1: 887 / 10000
Época 2: 948 / 10000
Época 3: 1004 / 10000
Época 4: 1038 / 10000
Época 5: 1056 / 10000
Época 6: 1100 / 10000
Época 7: 1130 / 10000
Época 8: 1172 / 10000
Época 9: 1192 / 10000
Época 10: 1228 / 10000
Época 11: 1255 / 10000
Época 12: 1303 / 10000
Época 13: 1345 / 10000
Época 14: 1368 / 10000
Época 15: 1411 / 10000
Época 16: 1443 / 10000
Época 17: 1460 / 10000
Época 18: 1476 / 10000
Época 19: 1472 / 10000
Época 20: 1494 / 10000
Época 21: 1544 / 10000
Época 22: 1580 / 10000
Época 23: 1614 / 10000
Época 24: 1639 / 10000
Época 25: 1666 / 10000
Época 26: 1647 / 10000
Época 27: 1689 / 10000
Época 28: 1790 / 10000
Época 29: 1771 / 10000
```

In [27]:
```
net5 = nw.Network([784, 30, 10])

net5.RMSprop(training_data, 30, 10, 0.001, 0.9, 1e-7, test_data=test_data)
```

```
Época 0: 1094 / 10000
Época 1: 1115 / 10000
Época 2: 1472 / 10000
Época 3: 1696 / 10000
Época 4: 1968 / 10000
Época 5: 2022 / 10000
Época 6: 2102 / 10000
Época 7: 2205 / 10000
Época 8: 2206 / 10000
Época 9: 2171 / 10000
Época 10: 2291 / 10000
Época 11: 2239 / 10000
Época 12: 2093 / 10000
Época 13: 2274 / 10000
Época 14: 2292 / 10000
Época 15: 2326 / 10000
Época 16: 2244 / 10000
Época 17: 2290 / 10000
Época 18: 1971 / 10000
Época 19: 1938 / 10000
Época 20: 2056 / 10000
Época 21: 2029 / 10000
Época 22: 1967 / 10000
Época 23: 1893 / 10000
Época 24: 1865 / 10000
Época 25: 1967 / 10000
Época 26: 1997 / 10000
Época 27: 2019 / 10000
Época 28: 1870 / 10000
Época 29: 1819 / 10000
```

In [5]:
```
net6 = nw.Network([784, 30, 10])

net6.RMSprop(training_data, 50, 10, 0.001, 0.9, 1e-10, test_data=test_data)
```

```
Época 0: 5544 / 10000
Época 1: 7337 / 10000
Época 2: 7909 / 10000
Época 3: 8275 / 10000
Época 4: 8475 / 10000
Época 5: 8688 / 10000
Época 6: 8783 / 10000
Época 7: 8824 / 10000
Época 8: 8860 / 10000
Época 9: 8886 / 10000
Época 10: 8927 / 10000
Época 11: 8933 / 10000
Época 12: 8984 / 10000
Época 13: 8992 / 10000
Época 14: 8980 / 10000
Época 15: 9017 / 10000
Época 16: 9047 / 10000
Época 17: 9075 / 10000
Época 18: 9063 / 10000
Época 19: 9074 / 10000
Época 20: 9095 / 10000
Época 21: 9093 / 10000
Época 22: 9107 / 10000
Época 23: 9109 / 10000
Época 24: 9109 / 10000
Época 25: 9107 / 10000
Época 26: 9098 / 10000
Época 27: 9132 / 10000
Época 28: 9142 / 10000
Época 29: 9120 / 10000
Época 30: 9132 / 10000
Época 31: 9141 / 10000
Época 32: 9142 / 10000
Época 33: 9161 / 10000
Época 34: 9164 / 10000
Época 35: 9156 / 10000
Época 36: 9174 / 10000
Época 37: 9171 / 10000
Época 38: 9186 / 10000
Época 39: 9167 / 10000
Época 40: 9168 / 10000
Época 41: 9172 / 10000
Época 42: 9179 / 10000
Época 43: 9184 / 10000
Época 44: 9188 / 10000
Época 45: 9192 / 10000
Época 46: 9221 / 10000
Época 47: 9197 / 10000
Época 48: 9204 / 10000
Época 49: 9221 / 10000
```

Después de varios intentos modificando los parámetros de logró encontrar, con $\eta = 0.001$, $decay\_rate = 0.9$ y un $\epsilon = 10 ^{-10}$, una predicción de $9221/1000$ una mejora increible respecto a SGD del **76.8 %**. Esto también puede deberse a que inició con un valor de predicción mucho más alto.

En las celda anteriores no está marcada la modificación que realicé a $decay\_rate$ porque decidí dejar fijo ese parámetro después de experimentar y solo ver rebotar los valores.

Capa soft Max

In [11]:
```
from importlib import reload
import network as nw
reload(nw)
```

Out[11]:
```
<module 'network' from '/home/jorge_t/Documentos/Física/Redes Neuronales/Proyecto_1/Redes-Neuronales/Digitos/src/network.py'>
```

SGD con momento

In [12]:
```
net7 = nw.Network([784, 30, 10])

net7.SGD_M(training_data, 30, 10, 0.0005, 0.95, test_data=test_data)
```

```
Época 0: 954 / 10000
Época 1: 1004 / 10000
Época 2: 1078 / 10000
Época 3: 1287 / 10000
Época 4: 1224 / 10000
Época 5: 1246 / 10000
Época 6: 1319 / 10000
Época 7: 1399 / 10000
Época 8: 1569 / 10000
Época 9: 1725 / 10000
Época 10: 1814 / 10000
Época 11: 1921 / 10000
Época 12: 2175 / 10000
Época 13: 2271 / 10000
Época 14: 2367 / 10000
Época 15: 2580 / 10000
Época 16: 2691 / 10000
Época 17: 2894 / 10000
Época 18: 2956 / 10000
Época 19: 3031 / 10000
Época 20: 3079 / 10000
Época 21: 3151 / 10000
Época 22: 3257 / 10000
Época 23: 3486 / 10000
Época 24: 3509 / 10000
Época 25: 3538 / 10000
Época 26: 3562 / 10000
Época 27: 3594 / 10000
Época 28: 3627 / 10000
Época 29: 3642 / 10000
```

Hubo una notable mejora en la predicción utilizando softmax en lugar de una activación sigmoid. Específicamente tenemos un **incremento de predicciones del 133 %**

RMSprop

Al ver el poco mejoramiento con 50 épocas respecto a las 30 de volvió a reducir a 30. Los demás parámetros se mantuvieron igual a la última prueba con RMSprop.

In [13]:
```
net8 = nw.Network([784, 30, 10])

net8.RMSprop(training_data, 30, 10, 0.001, 0.9, 1e-10, test_data=test_data)
```

```
Época 0: 3398 / 10000
Época 1: 3939 / 10000
Época 2: 4314 / 10000
Época 3: 4732 / 10000
Época 4: 5073 / 10000
Época 5: 5581 / 10000
Época 6: 5880 / 10000
Época 7: 6157 / 10000
Época 8: 6240 / 10000
Época 9: 6392 / 10000
Época 10: 6280 / 10000
Época 11: 6510 / 10000
Época 12: 6520 / 10000
Época 13: 6561 / 10000
Época 14: 6513 / 10000
Época 15: 6616 / 10000
Época 16: 6723 / 10000
Época 17: 6646 / 10000
Época 18: 6636 / 10000
Época 19: 6602 / 10000
Época 20: 6690 / 10000
Época 21: 6581 / 10000
Época 22: 6678 / 10000
Época 23: 6661 / 10000
Época 24: 6670 / 10000
Época 25: 6620 / 10000
Época 26: 6696 / 10000
Época 27: 6633 / 10000
Época 28: 6638 / 10000
Época 29: 6619 / 10000
```

In [16]:
```
net9 = nw.Network([784, 30, 10])

net9.RMSprop(training_data, 30, 10, 0.001, 0.9, 1e-9, test_data=test_data)
```

```
Época 0: 4579 / 10000
Época 1: 5525 / 10000
Época 2: 5965 / 10000
Época 3: 6510 / 10000
Época 4: 6876 / 10000
Época 5: 7203 / 10000
Época 6: 7462 / 10000
Época 7: 7620 / 10000
Época 8: 7786 / 10000
Época 9: 7877 / 10000
Época 10: 7950 / 10000
Época 11: 8012 / 10000
Época 12: 8025 / 10000
Época 13: 8092 / 10000
Época 14: 8107 / 10000
Época 15: 8186 / 10000
Época 16: 8222 / 10000
Época 17: 8259 / 10000
Época 18: 8258 / 10000
Época 19: 8293 / 10000
Época 20: 8279 / 10000
Época 21: 8286 / 10000
Época 22: 8325 / 10000
Época 23: 8344 / 10000
Época 24: 8356 / 10000
Época 25: 8393 / 10000
Época 26: 8355 / 10000
Época 27: 8394 / 10000
Época 28: 8427 / 10000
Época 29: 8412 / 10000
```

Respecto a la época 29 con la función de activación sigmoid tenenmos una pérdida de **7.76 %** en los valores de predicción. Esto se ve afectado por el punto inicial también. Pareciera tener más potencial de predicción esta activación, softmax, para este caso de estudio, pero se necesita experimentar más con los parámetros.

La combinación entre RMSprop y softmax da un aprendizaje más rápido, sin embargo al implementar esta nueva activación se necesita modificar los parámetros para evitar empezar a rebotar o crecer muy lento. Después de aproximadamente 10 pruebas no logré configurar bien los parámetros para evitar estancarse en mínimos locales o empezar a rebotar con los valores.