

Universidad Galileo
FISICC
Seminario de Investigación de Operaciones

Digit Recognizer

José Pérez 14006048
Carlos Montiel 15000552
Diego Aldana 12002322

Guatemala, 11 de mayo de 2019

Introducción

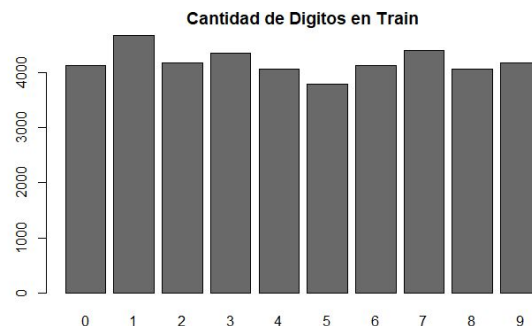
Cuando un dígito es escrito a mano y con cualquier tipo de material (tinta, ladrillo, carbón) su color no es totalmente homogéneo incluso cuando la tinta es muy oscura, existe un tipo de color tipo gradiente descendiente a lo largo de sus bordes, este se da debido a que el material con el que se escribió tuvo una fuerza aplicada discontinua, es por esto que se hace fácil reconocer los límites del trazo. El objetivo del proyecto es identificar correctamente dígitos de un conjunto de datos de decenas de miles de manuscritos utilizando herramientas como redes neuronales simples y convolucionales.

Definición de la Competencia

En esta competencia, el objetivo principal es identificar correctamente los dígitos escritos a mano alzada o bien, identificar un trazo y expresarlo en un dígito en trazo a un dígito virtual como una imagen.

Estrategias de Resolución

Sabemos por la descripción de la data que los datos están en matrices de 28x28 haciendo un total de 784 píxeles expresados en escala de grises con un valor que varía entre 0 y 255. El set de datos contiene una columna label la cual describe el dígito que se visualiza en la matriz pero no influye en el análisis de los píxeles por lo cual se procedió a eliminarla.



Se visualizó la cantidad de dígitos que había en el dataset de trainesto antes de eliminar la columna label y se procedió a realizar una gráfica de barras con la cantidad de los digitos del 0 al 9.

Se normalizará todos los valores de píxel a un valor en $[0,1]$ para el mejor manejo de los datos

Se transpone la matriz de píxeles para que los datos sean compatibles con la librería MXNet

Fundamentos teóricos.

Librería mxnet:

mxnet es un marco de software de aprendizaje profundo de código abierto, es utilizado para entrenar e implementar redes neuronales profundas. Es escalable y permite una capacitación rápida en modelos y es compatible con un modelo de programación flexible y múltiples lenguajes de programación.

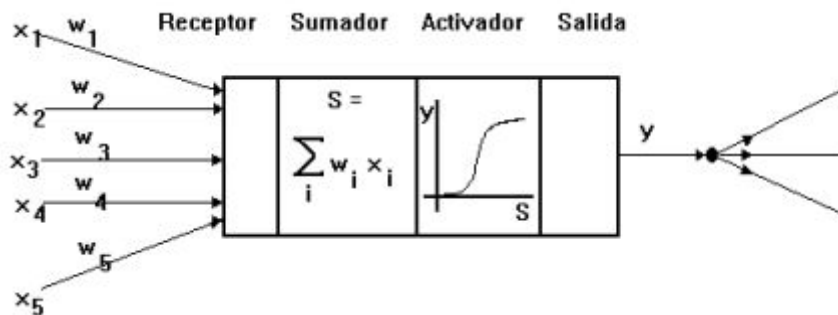
Redes neuronales Artificiales:

Las redes neuronales artificiales son un modelo computacional vagamente inspirado en el comportamiento observado en su homólogo biológico. Consiste en un conjunto de unidades, llamadas neuronas artificiales, conectadas entre sí para transmitirse señales. La información de entrada atraviesa la red neuronal produciendo unos valores de salida.

El objetivo de la red neuronal es resolver los problemas de la misma manera que el cerebro humano, aunque las redes neuronales son más abstractas. Las redes neuronales actuales suelen contener desde unos miles a unos pocos millones de unidades neuronales

Neurona Artificial

La neurona artificial es una unidad procesadora con cuatro elementos funcionales:

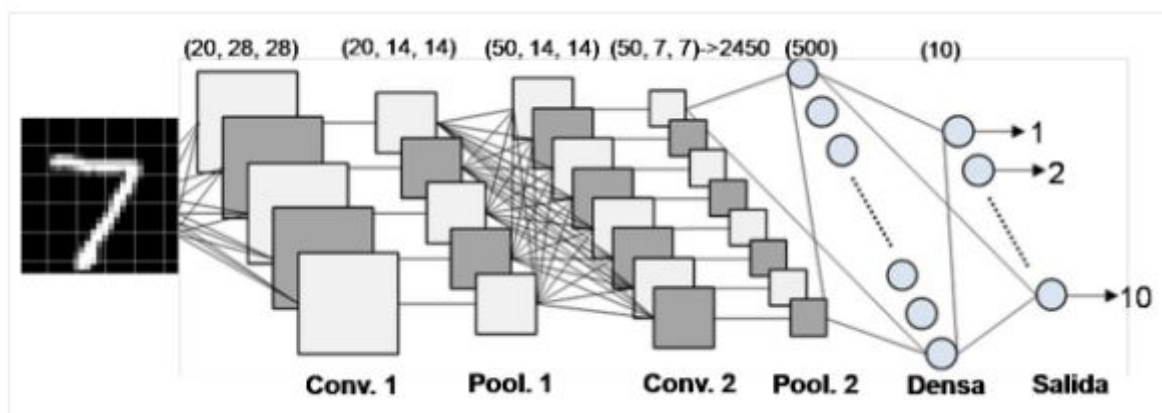


- 1) **El elemento receptor**, a donde llegan una o varias señales de entrada X_i , que generalmente provienen de otras neuronas y que son atenuadas o amplificadas cada una de ellas con arreglo de un factor de peso W_i que constituye la conectividad entre la neurona fuente de donde provienen y la neurona destino en cuestión.
- 2) **Elemento sumador**, es el encargado de efectuar la suma algebraica ponderada de las señales de entrada, ponderando las de acuerdo con su peso.
- 3) **Función activadora**, este elemento aplica una función lineal o no lineal de umbral (con frecuencia es una función escalón o una curva logística) a la salida del sumador para decidir si la neurona se activa, disparando una salida o no.
- 4) **Elemento de salida**, es el que produce la señal, de acuerdo con el elemento anterior que constituye la salida de la neurona.

Redes neuronales Convolucionales:

Las redes neuronales convolucionales consisten en múltiples capas de filtros convolucionales de una o más dimensiones. Después de cada capa, por lo general se añade una función para realizar un mapeo causal no-lineal.

Como redes de clasificación, al principio se encuentra la fase de extracción de características, compuesta de neuronas convolucionales y de reducción de muestreo. Al final de la red se encuentran neuronas de perceptrón sencillas para realizar la clasificación final sobre las características extraídas. La fase de extracción de características se asemeja al proceso estimulante en las células de la corteza visual. Esta fase se compone de capas alternas de neuronas convolucionales y neuronas de reducción de muestreo. Según progresan los datos a lo largo de esta fase, se disminuye su dimensionalidad, siendo las neuronas en capas lejanas mucho menos sensibles a perturbaciones en los datos de entrada, pero al mismo tiempo siendo estas activadas por características cada vez más complejas.



En la versión basada en fotografías de la red neuronal convolucional implementada para el reconocimiento de dígitos que se muestra en la imagen anterior. En el sistema, la salida de cada una de las capas consiste en un conjunto de imágenes de salida o planos llamados “mapas de características”, que se componen de conjuntos de neuronas. Las neuronas que pertenecen a un mapa están conectadas únicamente con neuronas en mapas de la capa siguiente a través de campos de proyección (máscaras de convolución).

Modelos Generados.

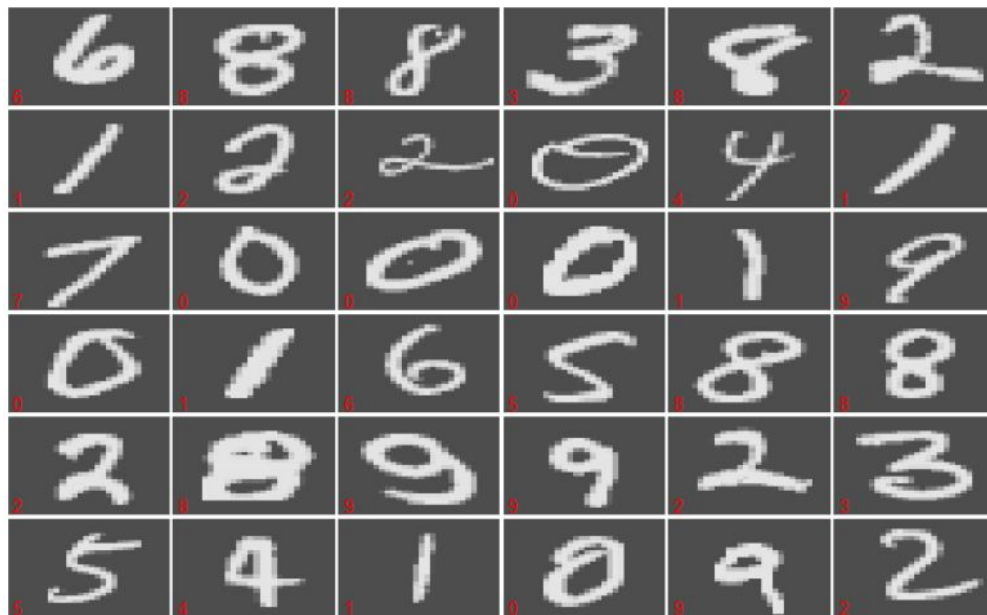
para nuestro análisis se procedió a realizar 2 modelos, el primero de ellos en el cual se realiza una iteración a la red neuronal dio los siguientes resultados:



el cual fue corrido durante 6.86 segundos con un accuracy de 0.36535714, el accuracy tan bajo es debido al número de iteraciones que se otorgó al modelo, en este caso fue 1.

Modelo Final.

El modelo de convolución de red neuronal con





Resultados Finales.

11.20 horas de entrenamiento:

```
[1] "Training took: 40325.2 seconds"
```

```
[817] Train-accuracy=0.996904769114086
[818] Train-accuracy=0.996857152098701
[819] Train-accuracy=0.997142865544274
[820] Train-accuracy=0.996952389677366
[821] Train-accuracy=0.997452391755013
[822] Train-accuracy=0.997428578989846
[823] Train-accuracy=0.997119053488686
[824] Train-accuracy=0.997119052779107
[825] Train-accuracy=0.997238104542096
[826] Train-accuracy=0.997238105251676
```

259	Akash Srivastava		0.99728	10	8h
260	Weifa Gan		0.99714	15	1mo

Con el accuracy de nuestro resultado final el cual no fue posible subir a kaggle por un error en la tabla de resultados se colocará aproximadamente en el 260

Ranking de Kaggle.

2.1 horas de entrenamiento:

4 horas de entrenamiento:

753	Diego Aldana		0.99342	2
-----	--------------	---	---------	---

8 horas de entrenamiento:

629	Jose perez		0.99457	1
-----	------------	---	---------	---

Conclusiones y Recomendaciones.

- Para utilizar la librería mxnet en R es necesario descargar la librería e instalarla de un repositorio utilizando:

```
cran <- getOption("repos")  
cran["dmlc"] <-  
"https://s3-us-west-2.amazonaws.com/apache-mxnet/R/CRAN/"  
options(repos = cran)  
install.packages("mxnet",dependencies = T)  
library(mxnet)
```
- Es necesario realizar con tiempo el entrenamiento de la red neuronal ya que entre más tiempo esté entrenando el accuracy mejorará lo cual se verá reflejado en el resultado final.