Práctica 12: red neuronal

Alumno: José Adrian Garcia Fuentes
Profesor: Satu Elisa Schaeffer

Universidad Autónoma de Nuevo León. Facultad de Ingeniería Mecánica y Eléctrica.
16/mayo/2021

Resumen

Reconocer dígitos de imágenes pequeñas en blanco y negro con una red neuronal, utilizando un perceptrón que separe las entradas verdaderas y falsas.

Palabras Claves: *Red neuronal.*

1. Introducción

La última práctica es una demostración básica de aprendizaje a máquina: se requiere reconocer dígitos de imágenes pequeñas en blanco y negro con una red neuronal. El elemento básico de una red neuronal es un perceptrón que esencialmente es un hiperplano (una línea si nos limitamos a dos dimensiones) que busca colocarse en la frontera que separa las entradas verdaderas y las entradas falsas. La dimensión d del perceptrón es el largo del vector x que toma como entrada, y su estado interno se representa con otro vector w que contiene sus pesos [1].

2. Objetivo

Estudia de manera sistemática el desempeño de la red neuronal en términos de su puntaje F para los diez dígitos en función de las tres probabilidades asignadas a la generación de los dígitos, variando a las tres en un experimento factorial adecuado [1].

3. Metodologia

La metodología empleada se realizó a través de Rstudio [2] llevando a cabo los pasos señalados en la *Práctica 12: red neuronal* [1], a partir del código en el repositorio de Schaeffer [3], se realizaron modificaciones. El código completo de la metodología empleada se encuentra en el repositorio de GitHub [4].

4. Resultados

A continuación se muestran los cambios al codigo, en la figura 1 se muestra un dígito en blanco y negro.

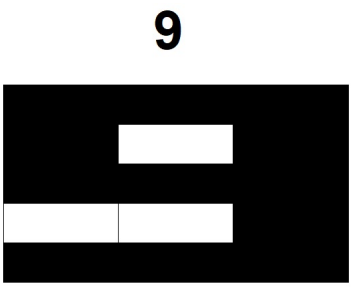

Figura 1: Dígito blanco y negro 3 x 5.

```
n_prob <- c(0.995, 0.98, 0.95, 0.30)
g_prob <- c(0.920, 0.90, 0.85, 0.50)
b_prob <- c(0.002, 0.01, 0.05, 0.80)

prom_fscore <- c()

for (iter in 1:4) {
```

```
  modelos <- read.csv("digits.txt", sep=" ",
    header=FALSE, stringsAsFactors=F)
  modelos[modelos=='n'] <- n_prob[iter]
  modelos[modelos=='g'] <- g_prob[iter]
  modelos[modelos=='b'] <- b_prob[iter]
```

```
precision <- diag(contadores) / rowSums(
    contadores)
  recall <- diag(contadores) / colSums(
    contadores)[1:(ncol(contadores)-1)]
  fscore <- (2 * precision * recall) / (
    precision + recall)
  prom_fscore[iter] <- mean(fscore)
}
prom_fscore
```

5. Reto 1

Extiende y entrena la red neuronal para que reconozca además por lo menos doce símbolos ASCII adicionales, aumentando la resolución de las imágenes a 5 x 7 de lo original de 3 x 5 (modificando las plantillas de los dígitos acorde a este cambio).

Se modificó el codigo de la tarea base cambiando las dimensiones de la resolución de la imagen a 7 x 5 (ver figura 2, 3, 4).

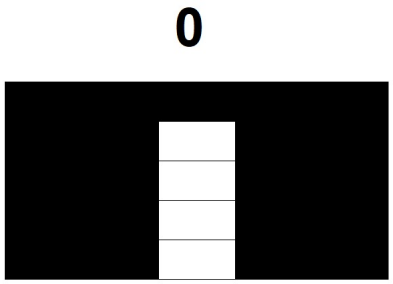

Figura 2: Dígito blanco y negro 7 x 5.

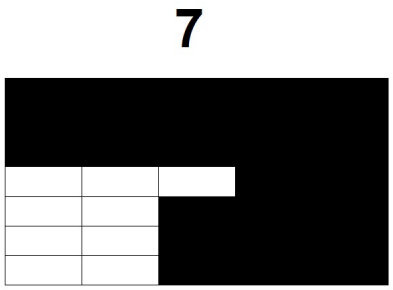

Figura 3: Dígito blanco y negro 7 x 5.

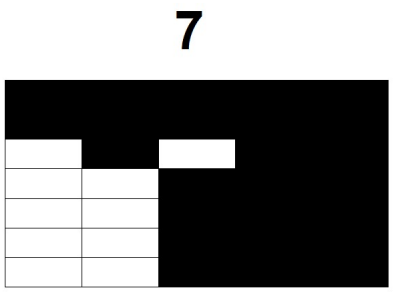

Figura 4: Dígito blanco y negro 7 x 5.

```
modelos <- read.csv("digitos2.csv", sep=",",
   header=FALSE, stringsAsFactors=F)
modelos[1,1] <- "n"

modelos[modelos=='n'] <- 0.995
modelos[modelos=='g'] <- 0.92
modelos[modelos=='b'] <- 0.002

r <- 7
c <- 5
dim <- r * c

tasa <- 0.15
tranqui <- 0.99

tope <- 11

```

6. Reto 2

Agrega ruido *sal-y-pimienta* en las entradas para una combinación ngb con la cual la red desempeña bien; este tipo de ruido se genera cambiando con una probabilidad los pixeles a blanco o negro (uniformemente al azar entre las dos opciones). Estudia el efecto de en el desempeño de la red (no importa si se hace esto con la red de la tarea base o la red extendida del primer reto).

A partir del código del primer reto se añadió el siguiente código para estudiar el efecto de desempeño en la red y se obtuvo el gráfico *fscore* de *pr* (ver figura 5 un ejemplo de dígito se muestra en la figura 6

```
 pr <- seq(0,.5,.025)
prom_fscore <- c()
iter = 1

for (prob in pr) {
```

```
s <- (runif(dim) > prob)
    for (m in 1:length(s)) {
      if (s[m] == FALSE) {pixeles[m]<- !
    pixeles[m]}
    }
    correcto <- binario(d, n)
```

```
iter = iter + 1
}
prom_fscore
plot(pr, prom_fscore, type= "l")
```

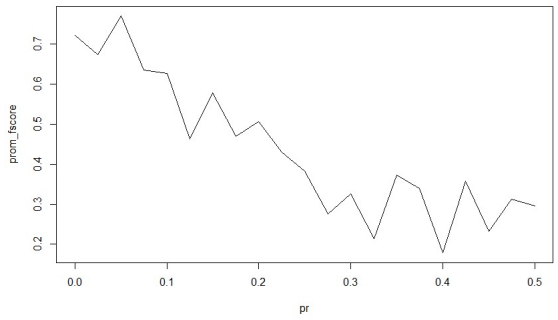

Figura 5: Gráfico f score.

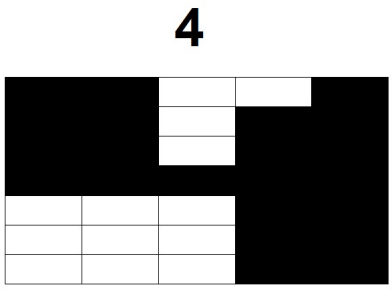

Figura 6: Dígito blanco y negro 3 x 5.

7. Conclusión

El valor *Fscore* depende de la probabilidad con la que aparezcan los colores y estos pixeles dominantes predominan los colores negro y blanco.

Referencias

[1] E. Schaeffer, "Práctica 12: red neuronal," mayo 2021. https://https://elisa.dyndns-web.com/teaching/comp/par/p12.html.

[2] J. J. Allaire, "Rstudio," mayo 2021. https://rstudio.com.

[3] E. Schaeffer, "Práctica 12: red neuronal," mayo 2021. https://github.com/satuelisa/Simulation/tree/master/NeuralNetwork.

[4] J. A. Garcia mayo 2021. https://github.com/fuentesadrian/SIMULACION-DE-NANOMATERIALES/tree/main/Tarea%2012.