Redes Neuronales Profundas: Trabajo Práctico 1

Respuestas

El código realizado para la resolución de los ejercicios está disponible en github (branch *lpineda*)

Ejercicio 1

Para leer el dataset completo se genera recursivamente una lista de *strings* con las rutas de las imágenes. Luego, estas son leídas utilizando la función *reshape_image*, la cual descarta una fila o una columna si el alto o el ancho tienen dimensiones impares, y centra la imagen en un rectángulo negro de las medidas dadas. Dado que el dataset completo no cabe en memoria (más de 50GB), solo se lee un subdirectorio.

Ejercicio 2

En este ejercicio se toman imágenes directamente del dataset, sin el procesamiento realizado en el ejercicio 1. Dado que se deben extraer parches en posiciones aleatorias de la imagen, la subimagen podría ser seleccionada de uno de los bordes negros, lo cual no agregaría información relevante.

Por otro lado, para normalizar la imagen con $\mu = 0$ y $\sigma^2 = 1$, a cada pixel de la imagen se le sustrae la media y se lo divide por la desviación estandar. Esto es, para una imagen I:

$$\forall p_{i,j} \in I, \qquad (p_{i,j} - \mu_I) \div \sigma_I^2$$

El resultado de este procesamiento se puede observar a continuación:

La imagen transformada es inteligible puesto que al normalizar cada canal, los valores enteros de los pixeles en el rango $[0, 255]$ son transformados a valores flotantes, y al tener una distribución de probabilidad con $\mu = 0$ es

Figura 1: Imagen original.

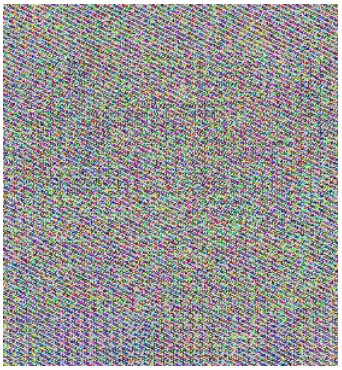


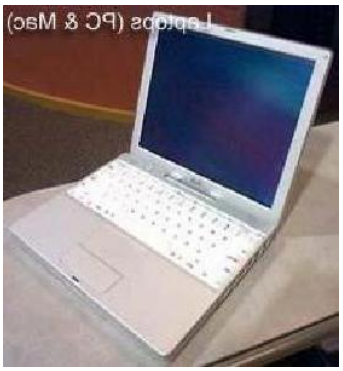
Figura 2: $\mu = 0 \quad \sigma^2 = 1$

de esperar la existencia de valores negativos. Estos nos son transformados correctamente por la librería utilizada[1].

El hecho que la imagen sea inteligible no quiere decir que la información se halla perdido. Esto se puede comprobar al cuantizar los valores y transformarlos nuevamente al rango $[0, 255]$ mediante la función:

$$f(x) = 255\frac{x - min(I)}{max(I) - min(I)}$$

donde nuevamente I es una imagen dada. Esta función está implementada bajo el nombre *quantize*, y el resultado de la transformación de la figura 2 (en su versión de valores flotantes) puede observarse en la siguiente imagen:



[1]Python Imaging Library (PIL): http://www.pythonware.com/products/pil/

Ejercicio 3

Las imagenes naturales (las imagenes que los humanos estamos acostumbrados a ver, como paisajes, animales, etc) contienen información redundante dada la alta correlación entre pixeles adjacentes. Si utilizamos un dataset de imágenes para entrenar algún modelo, lo ideal sería que las entradas contengan la menor cantidad de información repetida posible; el objetivo de aplicar *whitening* sobre cada imagen de un dataset es hacer que los datos con los que entrenamos la red neuronal no tengan información redundante.

Dados m vectores de características $\mathbf{x}$ (nuestro dataset) con media cero $\mu_{\mathbf{x}_i} = 0$, la matriz de covarianza está dada por:

$$\mathbf{\Sigma} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}\mathbf{x}^T$$

Si queremos que los datos no estén correlacionados, debemos transformar el dataset de forma que su matriz de covarianza $\mathbf{\Sigma}$ sea una matriz diagonal. Esta transformación se puede expresar lograrse pre-multiplicando cada vector de características por la matriz $\mathbf{\Phi}^T$, donde las columnas de $\mathbf{\Phi}$ son los eigenvectores de la matriz de covarianza $\mathbf{\Sigma}$. Podemos expresar la matriz de covarianza diagonalizada como:

$$\mathbf{\Phi}^T \mathbf{\Sigma} \mathbf{\Phi} = \mathbf{\Lambda}$$

donde $\mathbf{\Lambda}$ es una matriz de ceros con los eigenvalores de la matriz de covarianza en su diagonal. Luego, la matriz de covarianza de $\mathbf{y} = \mathbf{\Phi}^T \mathbf{x}$ es la matriz diagonal $\mathbf{\Lambda}$.

Aplicar *whitening* a un dato consiste en hacer que su matriz de covarianza tenga la forma $\mathbf{\Lambda} = \alpha \mathbf{I}$. Usando el hecho que:

$$\mathbf{\Lambda}\mathbf{\Lambda}^{-1} = 1 \rightarrow \mathbf{\Lambda}^{-1} = \mathbf{\Lambda}^{-1/2} \mathbf{I} \mathbf{\Lambda}^{-1/2} \rightarrow \mathbf{I} = \mathbf{\Lambda}^{-1/2} \mathbf{\Lambda} \mathbf{\Lambda}^{-1/2}$$

y reemplazando $\mathbf{\Lambda}$ tenemos que:

$$\mathbf{\Lambda}^{-1/2} \mathbf{\Phi}^T \mathbf{\Sigma} \mathbf{\Phi} \mathbf{\Lambda}^{-1/2} = \mathbf{I}$$

Finalmente, la transformación $\mathbf{z}$ para *blanquear* los datos esta dada por:

$$\mathbf{z} = \mathbf{\Lambda}^{-1/2} \mathbf{\Phi}^T \mathbf{\Sigma} \mathbf{x}$$

Esto se puede comprobar al calcular la matriz de covarianza de $\mathbf{z}$

$$\mathbf{\Sigma} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{z}\mathbf{z}^T = \frac{1}{m} \sum_{i=1}^{m} \mathbf{\Lambda}^{-1/2} \mathbf{\Phi}^T \mathbf{x}\mathbf{x}^T \mathbf{\Phi} \mathbf{\Lambda}^{-1/2} = \mathbf{I}$$