Redes Neuronales Profundas: Trabajo Práctico 2

Respuestas

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

Ejercicio 1

El ejercicio tutorial propuesto consiste en definir una función $f(x, y) = x^2 + 2xy + y^2$ en Theano. La única modificación debe realizarse en la variable "out".

Para el ejemplo de regresion lineal, se separa el dataset en mini-batches para ir entrenando el modelo de forma progresiva, utilizando subconjuntos de datos. Con el dataset de motos y aviones de Caltech101, las imagenes son leidas y apiladas como vectores en una matriz que representa el dataset, y cada clase se representa con un vector de 0 o un 1. Luego el entrenamiento se realiza por épocas, finalizando cuando el error de entrenamiento no varia más que cierto umbral en las últimas 50 épocas o por máximo de iteraciones. La evolución del error de entrenamiento y test se puede observar en la figura 1

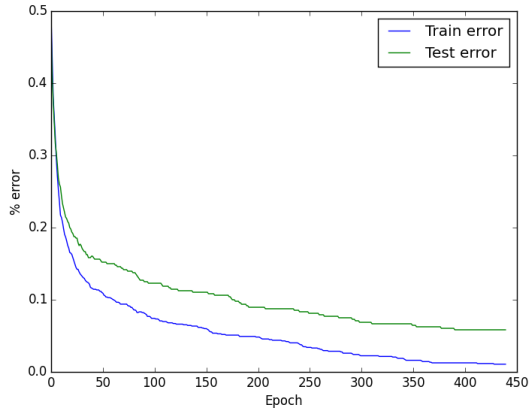

Figura 1: Error de train y test

Finalmente, para la tercera parte de este ejercicio se debe agregar al model una capa oculta de 100 neuronas con activación ReLU. Para esto es necesario definir una nueva matriz de pesos y un nuevo vector de bias para esta capa oculta. Además, se debe modificar la función de actualización para que calcule los delta usando los gradientes de la capa oculta. El entrenamiento se realiza de la misma manera que en el ejemplo de regresión lineal. La evolución del error de entrenamiento y test se puede observar en la figura 2

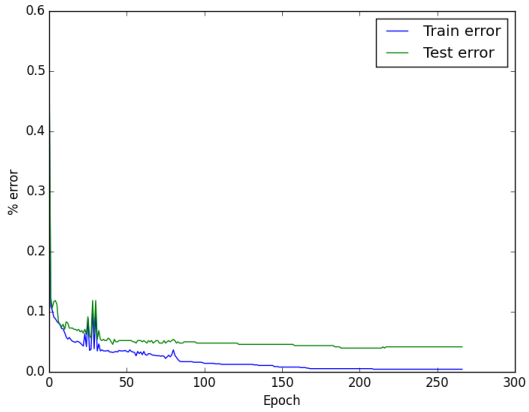

Figura 2: Error de train y test de modelo con capa oculta de 100 neuronas

Ejercicio 2

En la figura 3 se muestra la evolución del entrenamiento de la red neuronal con el dataset completo de 60000 ejemplos. El modelo está compuesto por dos capas de 512 neuronas, un dropout de 0,2 y activación ReLU, y una capa de salida de 10 neuronas y activación softmax.

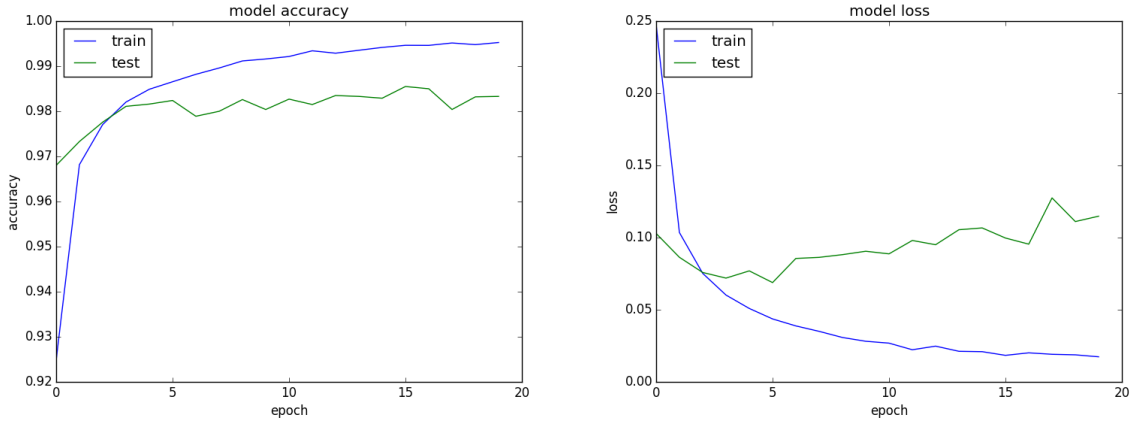

Figura 3: Ejemplo de Redes Neuronales de Keras con todos los patrones

En la figura 4 se muestra la evolución del entrenamiento de la red neuronal con el 25 % de los ejemplos del dataset y la misma red neuronal. En este caso la precisión del model bajo un 2 % respecto del entrenado con el dataset completo.

En la figura 5 se muestra la evolución del entrenamiento de la red neuronal con el 25 % de los ejemplos del dataset y una red neuronal con una sola capa de 512 neuronas con activación ReLU y dropout de 0,2.

En la figura 6 se muestra la evolución del entrenamiento de la red neuronal con el 25 % de los ejemplos del dataset y una red neuronal con una sola capa de 512 neuronas con activación sigmoidea y dropout de 0,2.

En la figura 7 se muestra la evolución del entrenamiento de la red neuronal con el 25 % de

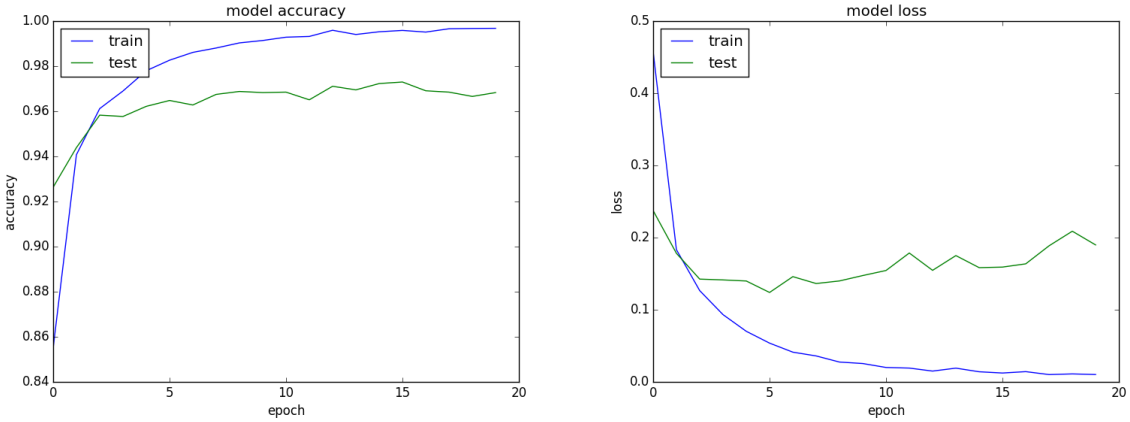

Figura 4: Utilizando el 25 % de los datos para entrenamiento

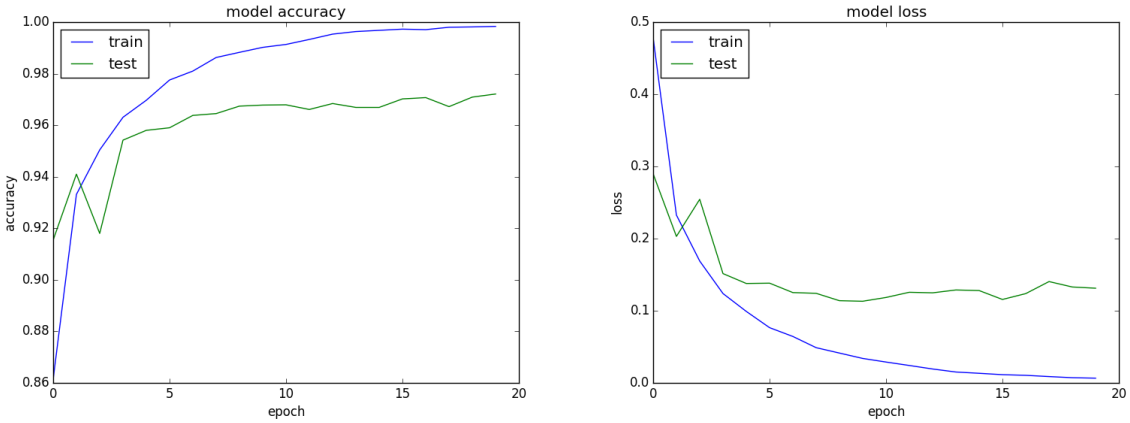

Figura 5: Utilizando solo una capa con activación ReLU

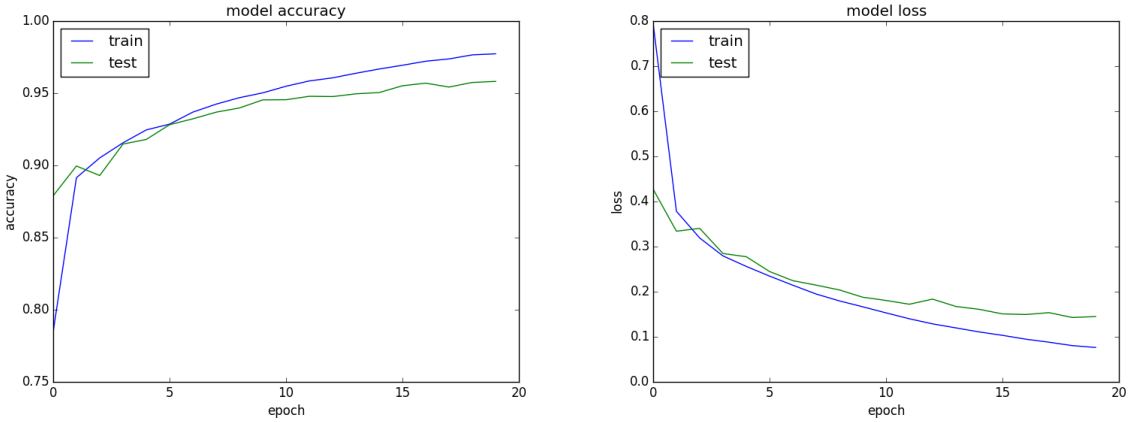

Figura 6: Utilizando solo una capa con activación sigmoidea

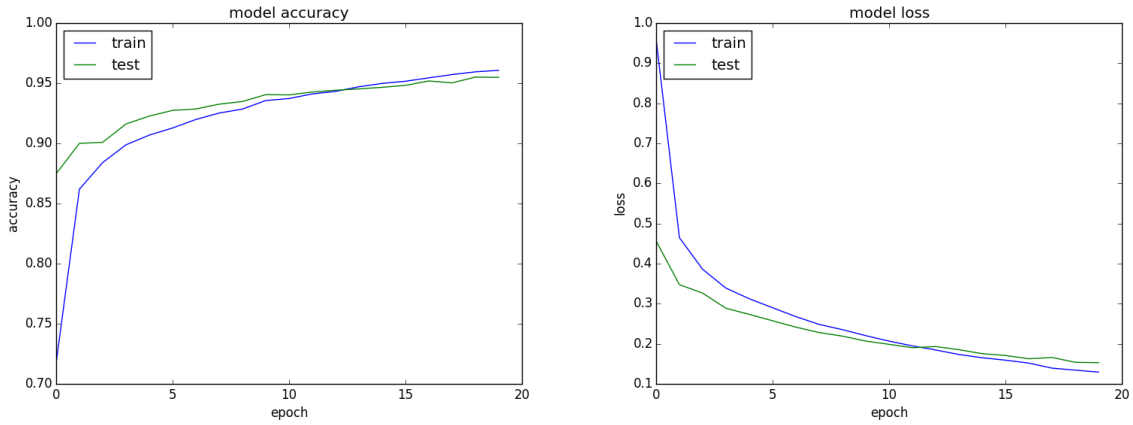

Figura 7: Utilizando solo una capa con activación sigmoidea y dropout de 0.5

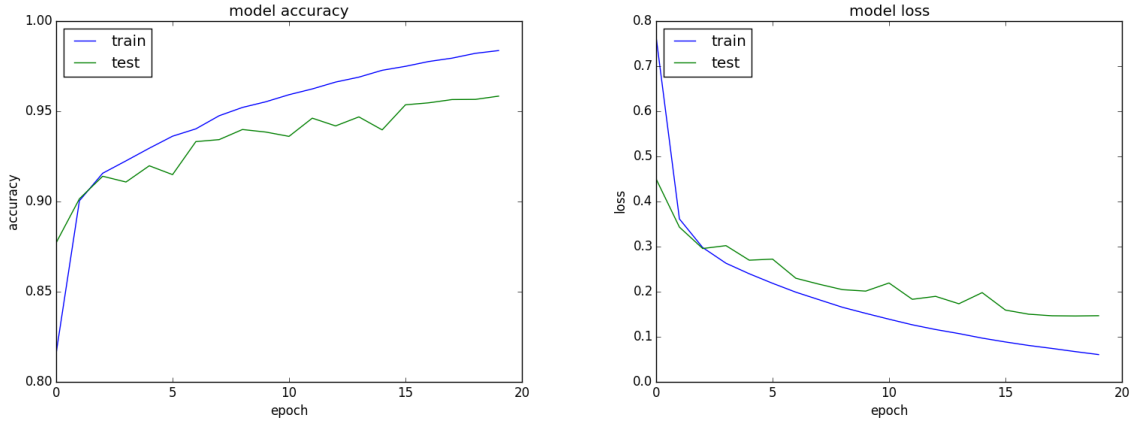

Figura 8: Utilizando solo una capa con activación sigmoidea pero sin dropout

los ejemplos del dataset y una red neuronal con una sola capa de 512 neuronas con activación sigmoidea y dropout de 0,5.

En la figura 8 se muestra la evolución del entrenamiento de la red neuronal con el 25 % de los ejemplos del dataset y una red neuronal con una sola capa de 512 neuronas con activación sigmoidea.

Trabajo práctico 1 Las funciones del ejercicio (b) implementadas con primitivas de Keras pueden encontrarse en el repositorio GitHub.

Finalmente, el formato h5 permite salvar el modelo entrenado usando un formato que soporta una cantidad ilimitada de tipos de datos. De esta forma, podemos distribuir en un archivo un modelo entrenado, sin necesidad de recalcularlo, o guardar modelos intermedios en las distintas épocas de entrenamiento.