Práctica 5

Multiperceptrón – Parte 2

Objetivos

El objetivo de esta práctica es comprender el funcionamiento del perceptrón multicapa

Temas

- Parada temprana
- Regularización L1 y L2

Lectura

Material de Lectura: Capítulos 2 y 3 del libro Neural Networks and Deep Learning.

Ejercicio 1

Utilizando los ejemplos del archivo ***AUTOS.csv*** genere un modelo utilizando un multiperceptrón para predecir el precio del auto (atributo ***price***) y la cantidad de millas por galón en ruta (***MPG-highway***) en función del resto de los atributos. Recuerde completar los valores faltantes, utilizar normalización y dividir el dataset en entrenamiento y validación.
Realice 20 ejecuciones independientes de la configuración seleccionada para respaldar sus afirmaciones referidas a la precisión obtenida tanto para el conjunto de entrenamiento como para el de testeo. Utilice un máximo de 1000 épocas e implemente una parada temprana con paciencia de 10.

Complete la siguiente tabla y realice un análisis de los valores obtenidos.

Optimizador	Función activación	Épocas Promedio	R2 Promedio
SGD	tanh		
	sigmoid		
	ReLU		
	LeakyReLU		
RMSProp	tanh		
	sigmoid		
	ReLU		
	LeakyReLU		
Adam	tanh		
	sigmoid		
	ReLU		
	LeakyReLU		

Donde

- **Épocas Promedio** es el número de épocas promedio en el que se detuvo el entrenamiento
- **R2 Promedio** es el promedio de la métrica R2 para la predicción en cada entrenamiento.

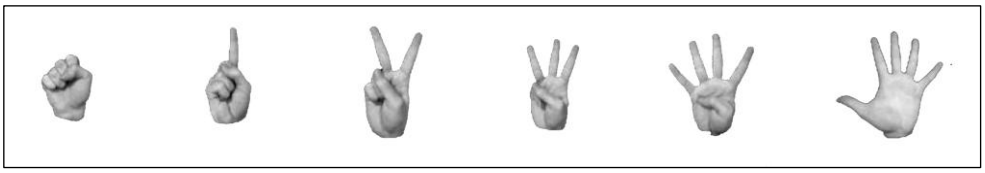
Ejercicio 2

Utilice el dataset ***CCPP.csv*** que contiene datos de una central de ciclo combinado para generar una red neuronal capaz de predecir la energía a producir. Realice un análisis del efecto de la regularización comparando 3 versiones del modelo: sin regularización, con regularización L1 y con regularización L2.
Utilice la siguiente configuración: 2 capas ocultas (30 y 20 neuronas) con función relu, optimizador sgd, división en datos de entrenamiento, validación (0.4 del entrenamiento) y prueba. Aplique distintas métricas sobre el modelo y sobre los pesos y calcule el tiempo de entrenamiento. Entrene los modelos durante 150 iteraciones y calcule el tiempo de entrenamiento.
Luego, responda las siguientes preguntas:

- Cómo es el tiempo de entrenamiento respecto del modelo con regularización respecto de sin regularización. ¿Por qué?
- Observando las primeras épocas de las gráficas de las 3 funciones de pérdida de los entrenamientos de los modelos, indique las diferencias y justifique.
- Grafique los pesos de la última capa oculta y realice algunas métricas sobre los pesos (suma y promedio absolutos, desviación, etc.). Marque las diferencias entre los 3 modelos e indique porque se dan.
- La regularización es una herramienta útil para prevenir problemas de sobreajuste. ¿Percibe sobreajuste en los modelos? ¿En que beneficia o perjudica la regularización?

Ejercicio 3

El conjunto de datos ***Fingers*** consiste en una serie de imágenes de 64x64 píxeles con fondo negro donde en su centro se encuentra un mano que muestra una cantidad de dedos que va desde 0 a 5. La versión original de este conjunto de imágenes se encuentra en https://www.kaggle.com/koryakinp/fingers



Muchas veces se utilizan técnicas de procesamiento de imágenes para obtener características representativas (features extraction) de los objetos dentro de una imagen con el objetivo de simplificar el problema, reducir la dimensionalidad y/o reducir el costo de procesamiento. Bibliotecas como ***OpenCV*** o ***SciKit-Learn*** proveen funciones que permiten procesar imágenes y obtener valores estadísticos que caracterizan los objetos.

a) Utilice la función ***regionprops*** de ***SciKitLearn*** con algunas imágenes de ejemplo del dataset para experimentar con las distintas características que extrae de la imagen de la mano.
b) Implemente un script que convierta las imágenes en las carpetas **"test"** y **"train"** del dataset **"Fingers"** en dos archivos (uno por carpeta) separados por comas (csv). De todas las características que provee ***regionprops***, tienen potencial aquellas que son independientes o se pueden independizar de las unidades (píxeles). Algunas de estas características pueden ser:
    - ***filled_area***: cantidad de píxeles que contiene región (podría interpretarse como píxeles cuadrados).
    - ***major_axis_length***: longitud (en píxeles) del eje mayor de la elipse que mejor se ajusta a la región.
    - ***minor_axis_length***: longitud (en píxeles) del eje menor de la elipse que mejor se ajusta a la región.
    - ***perimeter***: cantidad de píxeles que forman el contorne de la región.
    - **eccentricity**: excentricidad de la elipse de mejor ajuste, cerca de 0 es un círculo, mientras que cerca de 1 es una elipse más "larga".
    - ***solidity***: razón entre la cantidad de píxeles de la región original y de la región convexa. Para generar una región convexa se completan los pixeles de forma de eliminar regiones cóncavas de una figura. La región convexa de una estrella de 5 puntas se convertirá en un pentágono al completarla.
    - ***extent***: razón entre pixeles de la región original y el rectángulo que la contiene (bounding box).

c) Entrene un modelo de multiperceptrón a partir del archivo de entrenamiento generado en el punto anterior.
d) Reflexione acerca de las propiedades geométricas generadas a partir de un objeto/región de una imagen. ¿Cómo cree que afectaría la rotación, translación y escalado de un objeto/región dentro de la imagen?