Lista de Tareas de Deep Learning

4 de octubre de 2022

1. Aproximador Universal

En toda esta sección se entrenarán redes neuronales de 1 sola capa oculta entrenando con gradiente descendente clásico. Se buscará apreciar las bondades de una red como aproximador universal (no nos importa por ahora la capacidad de generalización).

1.1. Clasificación

1. Generar una base de datos de una XOR con todas las posibles combinaciones de ± 1 (4 casos). Asignar los labels correspondientes (1 si ambas entradas son iguales, 0 si son diferentes).

2. Entrenar una red neuronal con activación ReLU que alcance 100 % de accuracy. ¿Cuál es la mínima dimensión de la unidad oculta para lograr esto?

3. Repetir con activación sigmoide. Extraer conclusiones.

1.2. Regresión

1. Generar una base de datos de la función $f(x, y, z) = \sin(x) + \cos(y) + z$. Para ello barra una grilla de 20 puntos para cada coordenada ($0 \leq x < 2\pi$, $0 \leq y < 2\pi$ y $0 \leq z \leq 1$) y arme una base de datos con las 8000 combinaciones posibles.

2. Entrenar una red neuronal con activación ReLU e indique el error cuadrático medio. Grafique $f(x, x, x)$ y comparela con la salida del regresor barriendo x.

3. Repetir con activación sigmoide. Extraer conclusiones.

2. Optimizadores

Hacer el tutorial del notebook "Optimizador.ipynb". Preste atención a los detalles de implementación.

3. Regularización

3.1. Autoencoder

Con FASHION-MNIST construir un autoencoder (utilizar la base de datos como entrenamiento/validación).

1. Al finalizar reportar error cuadrático medio de validación.
2. Guardar el modelo en un archivo h5 (lo van a necesitar mas tarde).
3. A partir del error cuadrático construya un detector de anomalías (sin volver a entrenar). Reportar el Equal-Error-Rate en el conjunto de datos resultante de combinar los datos de validación de FASHION-MNIST con los de MNIST (20000 muestras en total).
4. Obtenga los valores de las unidades de menor dimensión de su autoencoder tanto para entrenamiento como para validación (FASHION-MNIST). Con ellos construya una nueva base de datos y guarde los data-frames.

3.2. Clasificación

CIFAR-10 (está en keras) contiene imágenes RGB de 32×32 (en total dimensión 3072) para hacer clasificación de objetos. Las clases son

- airplane
- automobile
- bird
- cat
- deer
- dog
- frog
- horse
- ship
- truck

1. Observar algunos ejemplos de imágenes de CIFAR. Una buena opción para ésto es utilizar imshow de pyplot.
2. Construir un clasificador utilizando la base de datos como entrenamiento/validación.
3. Reportar el accuracy de validación.

3.3. Regresión

El archivo "molinos.csv" contiene datos de potencias acumuladas por un parque eólico para los diferentes vientos. La columna "Velocity" contiene el módulo del viento en ese instante y la comulna "Direction" el ángulo de la velocidad medido en sentido horario ubicando el cero en vientos que provienen del norte. Finalmente las columnas "P" contiene las potencias acumuladas por cada molino.

1. Pasar las velocidades a coordenadas cartesianas.

2. Entrenar un regresor que estime la velocidad del viento (dos dimensiones cartesianas) en función de las potencias. Para ello separar los datos como crea conveniente.

3. Crear una script que levante el modelo entrenado y permita probarlo. Tiene que devolver el error cuadrático medio (básicamente esto significa que hay un set de datos que no les paso que lo voy a usar para probar su algoritmo). Tenga en cuenta que el formato de los datos será el mismo que este dataset (módulo y fase en sentido horario con respecto al viento del norte).