Reporte de Tarea 5: Funciones, Modelos personalizados y Ecuaciones Diferenciales.

1.- Al crear la capa que transformaba las imágenes de color a escala de grises intente probar que esta funcionaba con la base de datos MNIST, pero esta es una base de datos que posee únicamente imágenes a blanco y negro, por lo que no serviría para probar la capa, entonces pensé en cual otra base de datos usar, pero finalmente concluí que una forma más fácil de ver si la capa funcionaba era generar una imagen a color haciendo uso de la librería numpy y posteriormente hacer pasar dicha imagen por el modelo, al guardar la imagen resultante se observó que efectivamente la capa creada funcionaba correctamente.

2.- Para resolver esta parte de la tarea me base en el código: “ModeloyCapaPersonalizado.ipynb” compartido a través de colab, haciendo los cambios pertinentes en las funciones requeridas y en los intervalos de dichas funciones. Al realizar diferentes pruebas se observó que las redes funcionaban igual de bien al reducir a 200 el número de neuronas de una capa (originalmente tenían 600), además de reducir el número de épocas a 1000. El entrenamiento de las redes se realizó en el intervalo [-1,1] pero las gráficas se hicieron en el intervalo [-1.5,1.5] para poder apreciar de mejor forma las diferencias entre las aproximaciones y los valores exactos.

3.- En esta parte decidí buscar en internet de qué forma se podía entrenar la red modificando los coeficientes del polinomio requerido, esto se logró haciendo que estos coeficientes fueran los pesos de una capa diseñada, la cual no tenía sesgos. Al trabajar únicamente con esta capa formada por los coeficientes, se obtuvo originalmente una mala aproximación, al aumentar la tasa de aprendizaje a 0.01 (originalmente se tenía 0.0001) la aproximación mejoro mucho, observando que tenía su mayor diferencia con el valor exacto (prácticamente 10%) en x=1. Reduje el número de épocas a 100, teniendo el mismo resultado que si lo dejaba en 1000. Trate de hacer más cambios en el optimizador, los puntos de colocación, el número de épocas, la tasa de aprendizaje, la función de costo, pero sin importar que tanto los modificara (incluso de manera drástica), no logre que la red mejorara más.

4.- Para esta parte me base en el código: “ODEsolver.ipynb” compartido a través de colab, haciendo únicamente cambios pertinentes en las funciones requeridas y en los intervalos de dichas funciones.

a) En primera instancia la función era considerablemente una buena aproximación, excepto en “el centro” (x=0) y en “los extremos” (x >4 y x <-4). Al cambiar diferentes parámetros en la red, se notó que lo único que mejoraba a esta era el aumento de capas, realice esto hasta que note que la red empeoraba si agregaba más, probé distinto número de neuronas hasta que pude apreciar cual era la mejor combinación, al final la red con la que decidí quedarme era una buena aproximación al valor real, siendo las mayores diferencias nuevamente los extremos (la mayor diferencia con el valor exacto fue de prácticamente 0.3).

b) Para esta parte (para cada condición inicial) tuve que agregar modificaciones para poder escribir la segunda derivada de “y” respecto a “x”, esto basándome en el código de “PDESolver.ipynb” compartido a través de colab, al realizar diversas pruebas se notó que la red mejoraba mucho si se aumentaba el número de puntos de colocación (se aumentó a 1000) y si se colocaban pocas capas (una de 150, dos de 10 y la final de 1 neurona), logrando aproximaciones muy buenas.