Reporte 05

Jairo Andres Saavedra Alfonso

Universidad de los Andes

1. Objetivos

- Implementar una red neuronal.
- Observar el proceso de aprendizaje en una red neuronal.

2. Estructura de la red neuronal

En esta semana se continuo con la implementación de un red neuronal básica con el objetivo de clasificar espectros. La estructura de la red neuronal implementada es simple; consiste en una capa de entrada (*input layer*), una capa escondida (*hidden layer*) y una capa de salida (*output layer*).

Inicialmente se escogió aleatoriamente un *set* de 2000 espectros reconocidos como entrenamiento. Estos flujos poseen una dimensión de 433 píxeles. Por lo tanto, La capa de entrada tiene una dimensión de 433 neuronas las cuales reciben el logaritmo del flujo de los espectros reconocidos. La capa escondida o *hidden layer* posee una dimensión de 16 neuronas, correspondiente a múltiplos de 4. Finalmente, la capa de salida tiene dimensión 4 neuronas que indica la clase que se predijo.

3. Entrenamiento de la red neuronal

Antes de proceder a entrenar la red neuronal fue necesario realizar un proceso de escalamiento de los datos. La implementación se realizo mediante *Pytorch* con ayuda del modulo de redes neuronales *nn*.

Se creo la clase *Neural_Network* para trabajar el proceso de aprendizaje. En la clase anteriormente mencionada se ajustaron los parámetros correspondientes a las dimensiones del capas, estas dimensiones me determinan las matrices de pesos. También tenemos una función *forward* encargada realizar la convolución de las matrices pesos con nuestra matriz de espectros mediante la función de activación *sigmoid* con el objetivo de obtener nuevos pesos.

Finalmente, se tiene una función *backward* encargada de actualizar y optimizar los pesos para minimizar la perdida con respecto a nuestros pesos en cada entrenamiento. Esta ultima función es vital en el proceso de aprendizaje pues es la que nos indica si estamos o no teniendo evolución con respecto a cada ronda de entrenamiento. El proceso de entrenamiento se realizo sobre 100 épocas para un solo *batch*.

4. Rendimiento de la red neuronal.

Con el animo de determinar, independientemente de lo predicho, si la red neuronal esta aprendiendo o no con los datos de entrada. Se evalúa el gradiente de la función de perdida, *CrossEntropyLoss*, en las épocas que la red fue entrenada. Conforme el numero de entrenamientos aumenta, la función de perdida, una medida cuantitativa de cuanto difiere la predicción del valor real, decrece (Ver Figura 1). Esto indica que para cada ronda de entrenamiento, la propagación de los pesos ajustan un modelo que predice la clase del espectro.

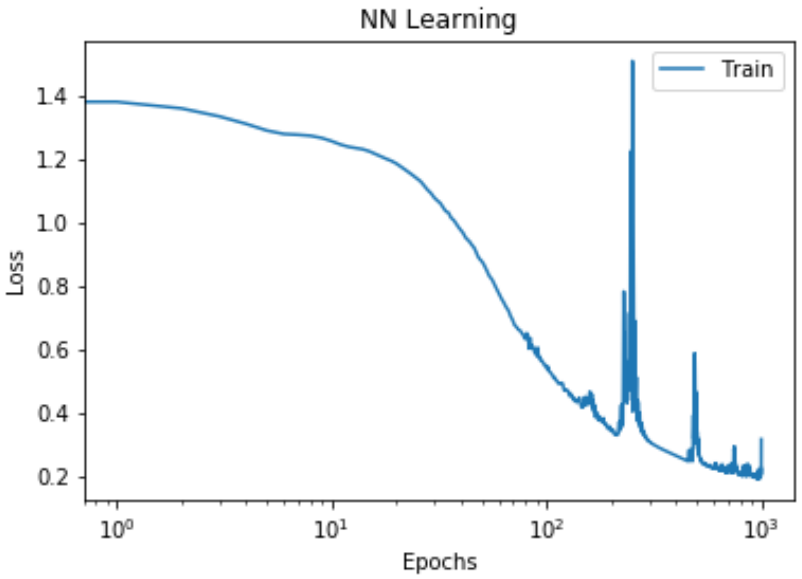

Figura 1. Función de perdida v.s. épocas

Se evaluó el modelo de la red sobre un conjunto de 1000 espectros con cada clase balanceada. Mediante una matriz de confusión podemos observar la fracción de espectros clasificados como Estrellas, Galaxias, QSO y QSO BAl con respecto a la etiqueta verdadera, ver Figura 2.

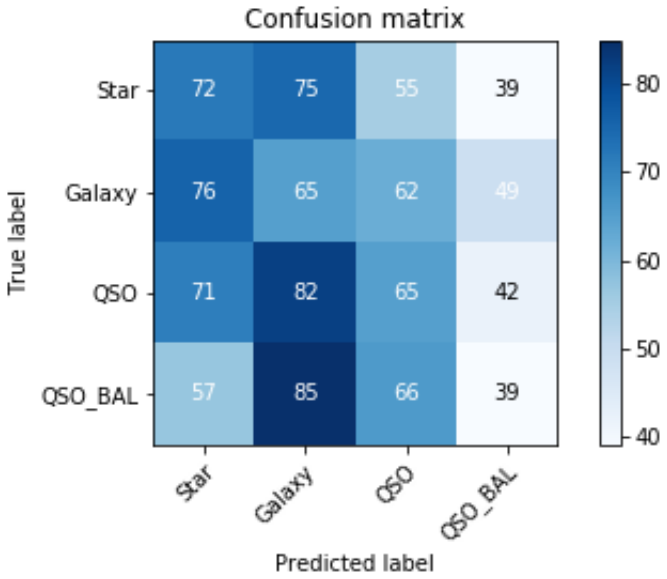

Figura 2. Matriz de confusión para *test*.

Se puede observar que la red neural implementada tiene problemas clasificando los causares tipo BAL, sin embargo, se clasificaron 39 QSO BAL efectivamente.

5. Tareas pendientes

- Profundizar sobre el funcionamiento, estructura y evaluación de redes neuronales.
- Implementar un clasificador exitosamente.

6. Repositorio Github

En este repositorio se pueden ver todos los reportes semanales y le notebook. https://github.com/MrX1997/Reportes-Proyecto-de-Monograf-a

Referencias

1. Busca G., Balland C., (2018).: QuasarNET: Human-level spectral classification and redshifting with Deep Neuronal Networks. Retrieve from: https://arxiv.org/abs/1808.09955
2. Paris, I., et al., (2017), Astron. Astrophys..: The Sloan Digital Sky Survey Quasar Catalog: Twelfth data release. , 597, A79