Análisis de una MLP (*Multi-Layer Perceptron*)

Objetivos:

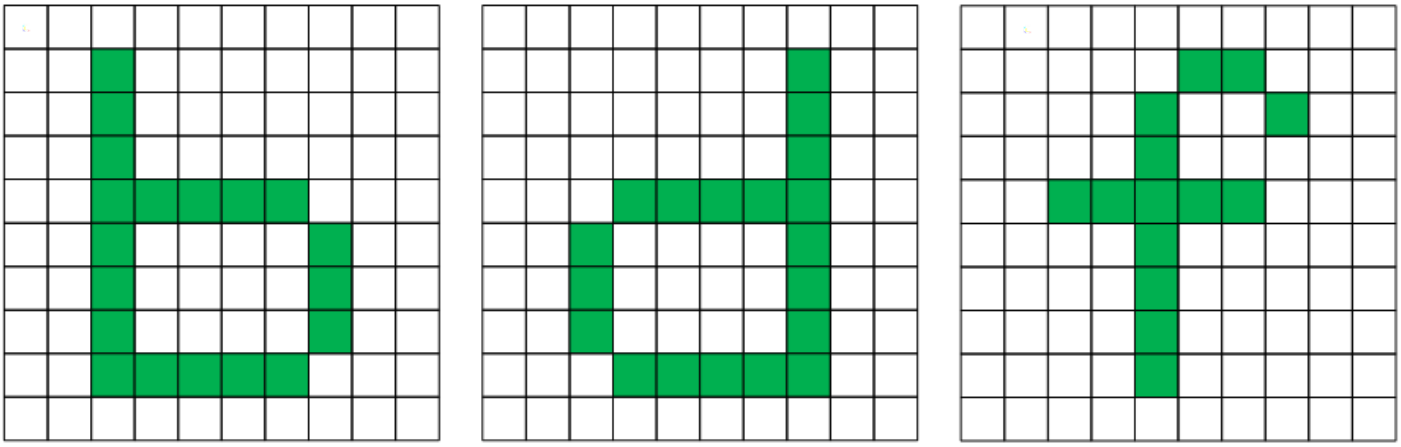
1. Implementar el algoritmo *MLP*.
2. Evaluar la precisión (MSE, error de entrenamiento y validación) de una MLP teniendo en cuenta distintas configuraciones: cantidad de capas, cantidad de neuronas, funciones de activación
3. Elaborar un informe completo en base a las pruebas realizadas.

Descripción del problema:

Este trabajo consiste en implementar el algoritmo *MLP* que permita, dado un dataset en R^2, parametrizar la cantidad de capas, neuronas y funciones de activación con los que se entrenará la red neuronal. La idea es desarrollar una aplicación que defina la arquitectura de la red (con 3 salidas, cada una asociada a un patrón de entrada), tome los datos de diferentes datasets, entrene el modelo y devuelva los resultados de clasificación (MSE, error de entrenamiento y validación).

La implementación deberá contar también con una interfaz de usuario para el ingreso de un patrón distorsionado (determinado por el usuario), que será clasificado según alguno de los patrones aprendidos, mostrando los resultados obtenidos**.**

Los patrones a detectar y clasificar estarán contenidos en una matriz de 10x10 que contendrán las letras b, d, f como se ve en las siguientes figuras:



Datasets

- El grupo de trabajo deberá generar 3 datasets que contengan 100, 500 y 1000 ejemplos. El 10% deberán ser patrones sin distorsionar y el resto con una distorsión del 1% al 30%. Los Datasets deberán ser representativos a la hora de definir la distribución de los ejemplos de entrenamiento.

Requerimientos mínimos para el entrenamiento

- Por cada Dataset deberán construirse tres conjuntos de validación con 10%, 20% y 30% de los ejemplos. El conjunto de validación debe ser representativo del Dataset de entrenamiento.
- 1 o 2 capas ocultas.
- De 5 a 10 neuronas por capa.
- Funciones de activación: lineal y sigmoidal.
- Coeficiente de aprendizaje entre 0 y 1.
- Término momento entre 0 y 1.

Requerimientos mínimos para el reconocimiento

- Patrón distorsionado de 0% a 30% generado de manera automática o manual.

Requerimientos mínimos de pruebas para el informe

- Se deberán realizar como mínimo las siguientes pruebas para cada uno de los datasets con conjuntos de validación de 10%, 20% y 30% de patrones:
  - 1 capa oculta de 5 neuronas, función de transferencia lineal, coeficiente de aprendizaje 0,5 y término momento 0,5.
  - 1 capa oculta de 10 neuronas, función de transferencia lineal, coeficiente de aprendizaje 0,5 y término momento 0,5.
  - 2 capas ocultas (primera capa de 5 neuronas, segunda capa de 5 neuronas), función de transferencia lineal, coeficiente de aprendizaje 0,5 y término momento 0,5.
  - 2 capas ocultas (primera capa de 10 neuronas, segunda capa de 10 neuronas), función de transferencia lineal, coeficiente de aprendizaje 0,5 y término momento 0,5.

- Repetir las mismas pruebas con término momento 0,9.

Multi-Layer Perceptron

Multi-Layer Perceptron (*MLP*)

Las redes multicapa son aquellas que disponen de conjuntos de neuronas agrupadas en varios niveles (2, 3, etc.)

Normalmente todas las neuronas de una capa reciben señales de entrada de otra capa anterior, más cercana a las entradas de la red, y envían las señales de salida a una capa posterior, más cercana a la salida de la red. A estas conexiones se las denomina *conexiones hacia adelante o feedforward.*

Sin embargo, en un gran número de estas redes también existe la posibilidad de conectar las salidas de las neuronas de capas posteriores a las entradas de las capas anteriores, a estas conexiones se les denomina *conexiones hacia atrás o feedback.*

Hilera, J. R. Martinez, V. J. (2000) Redes Neuronales Artificiales. Fundamentos, modelos y aplicaciones. Alfaomega.

Se deberá entregar:

- Código fuente (que incluya documentación interna) y ejecutable del programa con los IDEs de desarrollo usados y todas las librerías necesarias.
- Datasets utilizados para cada entrenamiento con su correspondiente descripción.
- Informe completo (archivo e impreso) del desarrollo del trabajo que contenga:
    1. Introducción.
    2. Descripción del/los algoritmo/s de desarrollado/s.
    3. Descripción de la solución implementada (explicando los problemas encontrados).
    4. Simulaciones realizadas y resultados obtenidos.
    5. Conclusiones.
    6. Referencias.

Consideraciones adicionales:

- Se deberá contar con una interfaz de usuario que permita la total operabilidad de la aplicación.
- Las interfaces deberán ser amigables (se aceptarán solamente entornos gráficos) e intuitivas (menú contextual de guía).
- El código debe estar totalmente documentado/comentado.
- El algoritmo debe ser enteramente desarrollado por los alumnos.
- Debe ser una aplicación de escritorio.
- El informe del trabajo deberá seguir el formato de LNCS (Lecture Notes in Computer Science) de Springer Verlag. La URL de donde se pueden bajar las plantillas del informe es:

http://preview.springer.com/gp/computer-science/lncs/conference-proceedings-guidelines

En esa página se encuentra, además de las plantillas, una guía para dar formato al informe.

- El informe no deberá exceder las 10 carillas (no incluir la impresión del código fuente).
- Para la nota final del Informe se tendrán en cuenta aspectos como: prolijidad, redacción y ortografía.
- La información de cada fuente utilizada deberá ser citada. En caso que la referencia no coincida con el desarrollo o haya algún tipo de plagio, el trabajo práctico será automáticamente desaprobado sin derecho a recuperatorio.

Condiciones generales:

Fecha límite de entrega	09/11/2020 a las 23.59 hs. (*)
Fecha de coloquio grupal	A convenir con los alumnos.
Calificación	Promedio ponderado de informe, programa y coloquio.

(*) Luego de esta fecha y hora se aceptarán los trabajos hasta el 16/11/2020 (23.59), pero sin la posibilidad del recuperatorio.