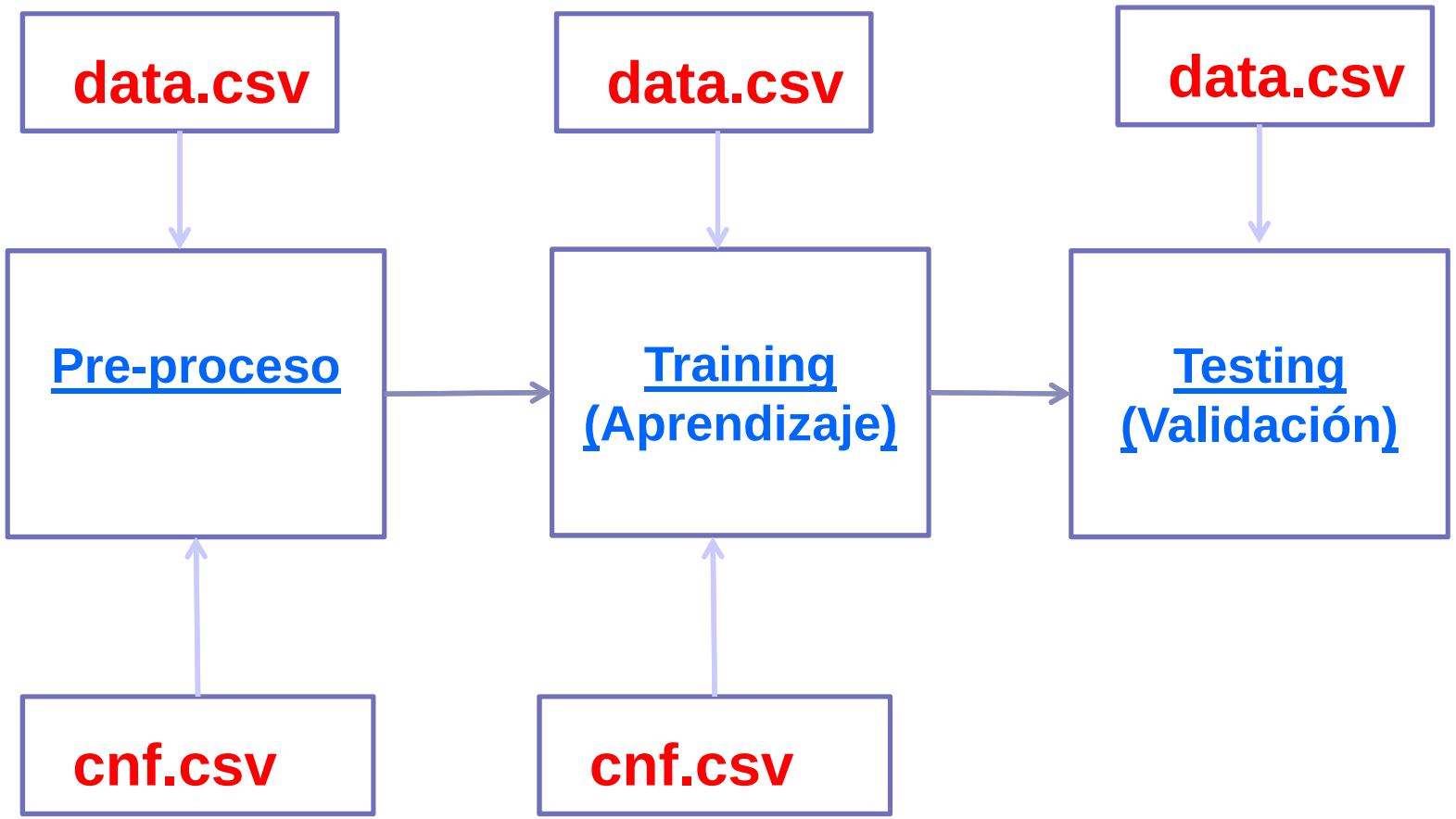
TAREA #3: Deep Learning

Deep AutoEncoder
+
Algoritmo mAdam

OBJETIVO

- Implementar y evaluar el rendimiento de un modelo de Deep-AutoEncode-Softmax con aprendizaje mAdam para un problema de clasificación de datos obtenidos desde un conjunto de sensores.

Etapas del Modelo:



Pre-proceso: ppr.py

- Formato File: class**.csv** :
  - N-filas.
  - d-columnas.
- Donde:
  - **filas** :
    - Números de muestras.
  - **columnas**:
    - Número de variables.

Pre-proceso: ppr.py

- **Para cada archivo de Clases Haga:**
  - Segmentar cada Columna (variable):
    - Usar '**N**' Frame de tamaño '**L**' muestras (valores).
      - Calcular la Amplitud de Fourier para cada Frame.
        - Seleccionar las primeras (**L/2**) muestras de la Amplitud Fourier.
      - Crear nueva data de entrada para el Modelo:
        - Formato Matriz : **(N-Frame, L/2**)
    - Apilar la matriz previa para cada columna
    - Crear matriz de etiqueta binaria para cada clase.
    - Apilar la matriz de etiquetas binarias para cada clase.

Pre-proceso: ppr.py

- Normalizar la Data de entrada :

$$x = \frac{(x - x_{\min})}{(x_{\max} - x_{\min})} \times (b - a) + a, \quad a = 0.01 \qquad b = 0.99$$

- Re-ordenar aleatoriamente las posiciones de la data de entrada y la data etiquetas binarias.

- Dividir la data re-ordenada previa:
  - Data Training : p %
  - Data Testing : (1-p)%

Pre-proceso: ppr.py

- Crear archivo de training: **train.csv**
  - Xe: data de entrada
  - Ye: data de etiquetas binarias
- Crear archivo de testing: **test.csv**
  - Xv: data de entrada
  - Yv: data de etiquetas binarias.

trn.py

- Cargar datos de configuración.
- Carga datos de Training.
- Ajustar el Modelo Deep Learning usando:
  - Deep-AutoEncoder con algoritmo mAdam
  - Softmax con algoritmo mAdam.
- Crear archivos de pesos:
  - **wdae.npz**
- Crear archivo de costo Softmax: **costo.csv**.
  - M-filas por una columna (número de iteraciones)

Función de Activación : DAE

- 1. ReLu:

$$f(x) = \begin{cases} x, & x > 0 \\ 0, & x \leq 0 \end{cases}, x \in \Re^d$$

- 2. L-ReLu:

$$f(x) = \begin{cases} 0.01\,x, & x < 0 \\ x, & x \geq 0 \end{cases}, x \in \Re^d$$

- 3. ELU:

$$f(x) = \begin{cases} a\left(e^x - 1\right), & x \leq 0 \\ x, & x > 0 \end{cases}, x \in \Re^d$$

- 4. SELU:

$$f(x) = \lambda \times \begin{cases} a\left(e^x - 1\right), & x \leq 0 \\ x, & x > 0 \end{cases}$$
$$x \in \Re^d,\ \lambda = 1.0507\ ,\ a = 1.6732$$

- 5. Sigmoidal:

$$f(x) = \frac{1}{1 + e^{-z}},\ x \in \Re^d$$

tst.py

- Cargar data de test.
- Cargas peso entrenados.
- Realizar proceso forward del Deep Learning.
  - Crear archivo de Matriz de Confusión**:**
    - **cmatriz.csv**.
  - Crear archivo F-scores:
    - **fscores.csv**
    - (***m+1***)-filas por 1-columa
      - Fila *(m+1)* representa el F-scores promedio de las ***m***-clases.

Configuración: cnf_sae.csv

- Línea 1: Número de Clases : 5
- Línea 2: Número de Frame : 100
- Línea 3: Tamaño de Frame : 1024
- Línea 4: Porcentaje Training : 0.8
- Línea 5: Func. Activación Encoder : 1
- Línea 6: Max. Iteraciones : 60
- Línea 7: Tamaño miniBatch : 32
- Línea 8: Tasa Aprendizaje : 0.001
- Línea 9: Nodos $Encoder_1$. : 192
- Línea 10: Nodos $Encoder_2$. : 128
- Línea 11: Nodos $Encoder_3$. : 64
- ….

Configuración: cnf_softmax.csv

- *Parámetros* :

- Línea 1: Max. Iteraciones : 300
- Línea 2: Tasa Aprendizaje : 0.01
- Línea 3: Tamaño miniBatch : 32
- ….

OBSERVACIÓN #1:

- El tiempo máximo ejecución de Pre-proceso y entrenamiento del DAE es:
  - 3 Minutos

OBSERVACIÓN #2:

- Si un Grupo no Cumple con los requerimientos funcionales y no-funcionales, entonces la nota máxima será igual a 1,0 (uno coma cero).

ENTREGA

- Sábado : 24/Junio/2023
  - Hora : 23 horas
  - Lugar : Aula Virtual del curso

- **Lenguaje Programación:**
  - Python version: 3.7.6 window (anaconda)
    - Numpy/Panda.