Aprendizaje Profundo: Práctica #2 – Modelos Recurrentes para Señales

M.I. Juan José Cárdenas Cornejo
jj.cardenascornejo@ugto.mx

***Resumen*—Esta práctica guía al estudiante en el diseño, entrenamiento y evaluación de tres arquitecturas recurrentes (RNN, LSTM y GRU) para un problema de análisis de señales de motor. Se abordan tareas de clasificación (13 clases de falla) y de regresión, bajo el enfoque IMRA (Introducción–Metodología–Resultados–Análisis).**

***Index Terms*—Aprendizaje profundo, RNN, LSTM, GRU, series de tiempo, señales de motor, clasificación, regresión.**

I. Introducción

Contexto: Las Redes Neuronales Recurrentes (RNN) son la arquitectura por excelencia para modelar datos secuenciales y series de tiempo, gracias a su capacidad de mantener un .estado.o "memoria"de eventos pasados. En esta práctica, se comparan las **RNN simples (Elman)**, **LSTM (Long Short-Term Memory)** y **GRU (Gated Recurrent Unit)**.

Problema. Analizar señales de corriente de un motor para:

1. **Clasificación:** Identificar el estado de salud del motor entre 13 clases (1 sano + 4 niveles de falla $\times$ 3 fases).
2. **Regresión:** Generar al menos otras 500 muestras por clase.

Objetivos de aprendizaje.

- Implementar y entrenar RNN, LSTM y GRU para la clasificación y la regresión de señales.
- Comprender el preprocesamiento de señales para RNNs (segmentación en ventanas, normalización).
- Diseñar y evaluar **variantes** (como mínimo dos por modelo), p.ej., bidireccionalidad, apilamiento (stacking), dropout.
- Analizar métricas (Accuracy, F1-Score, RMSE, R^2) y las curvas de entrenamiento para ambas tareas.

Preguntas e hipótesis. Formule preguntas (p.ej., ¿Supera LSTM a la RNN simple en dependencias largas? ¿Mejora la bidireccionalidad el F1-Score en fallas incipientes?) y una hipótesis por cada variante.

II. Metodología

II-A. Datos y partición

Conjunto de datos: Describir brevemente las señales de motor. Indicar las 13 clases (Sano, Falla F1-A, Falla F2-A, ..., Falla F4-C). Describir el *target* de regresión.
Partición: entrenamiento/validación/prueba (p.ej., 70/15/15) con semilla fija. ¡Asegurarse de no mezclar segmentos del mismo ciclo de motor entre particiones!
Preprocesamiento (clave para RNNs):

1. **Normalización:** p.ej., StandardScaler (media 0, desviación estándar 1) o MinMaxScaler (rango [0, 1]), ajustado *solo* con los datos de *train*.
2. **Segmentación:** Crear ventanas deslizantes (secuencias) de longitud fija (p.ej., 1024). La entrada a la red será (Batch, 1024, N_features).

II-B. Modelos (base)

Para todas las redes, la salida de la capa recurrente (el último estado oculto h_T o la secuencia completa) se conecta a una capa Lineal (Densa) final.

RNN (base). 1-2 capas de ‘nn.RNN‘ (con $\tanh$ o ‘relu‘); 1 capa FC. Variantes: Incrementar profundidad, cambiar activación.
LSTM (base). 1-2 capas ‘nn.LSTM‘; 1 capa FC. Variantes: ‘bidirectional=True‘, añadir Dropout recurrente, apilar más capas.
GRU (base). 1-2 capas ‘nn.GRU‘; 1 capa FC. Variantes: ‘bidirectional=True‘, apilar más capas.

II-C. Protocolo de entrenamiento (común)

Épocas: 80. **Batch size:** 64. **Optimizador:** Adam (weight_decay=1e-5). **LR inicial:** 0.001.
Criterion (Clasificación): Cross-Entropy Loss.
Criterion (Regresión): MSE Loss (o MAE Loss).
Semillas: fijadas para reproducibilidad.

II-D. Variantes (objetivo obligatorio)

Proponer al menos **dos variantes por modelo** (RNN, LSTM, GRU). Justificar la expectativa. Ejemplos:

- *Arquitectura:* LSTM apilada (2 capas); GRU Bidireccional, cambio en el numero de elementos ocultos por capa para la celdas (modelo lstm, gru).
- *Regularización:* Añadir Dropout a la(s) capa(s) FC; añadir Dropout recurrente (en LSTM/GRU).
- *Entrenamiento:* Probar optimizador SGD con momento vs Adam.

II-E. Métricas e histórico

Clasificación: Accuracy, F1 macro, matriz de confusión. **Regresión:** RMSE (Root Mean Sq. Error), MAE (Mean Abs. Error), R^2 (Coef. de Determinación).
Curvas: Pérdida y métrica principal (train/val) para ambas tareas. **Costo:** tiempo por época y # de parámetros. Guardar **checkpoint** del mejor modelo por métrica de validación (F1 o RMSE).

III. Resultados

III-A. Tarea 1: Clasificación (13 Clases)

Tabla I: Base (Clasificación): desempeño y costo.

Modelo	Params (M)	t/época (s)	Val Acc	Val F1	Test Acc	Test F1
RNN	–	–	–	–	–	–
LSTM	–	–	–	–	–	–
GRU	–	–	–	–	–	–

III-B. Tarea 2: Regresión (p.ej., RUL)

Tabla II: Base (Regresión): desempeño y costo.

Modelo	Params (M)	t/época (s)	Val RMSE	Val R^2	Test RMSE	Test R^2
RNN	–	–	–	–	–	–
LSTM	–	–	–	–	–	–
GRU	–	–	–	–	–	–

III-C. Impacto de las variantes

Tabla III: Variantes por modelo: cambio y efecto en Test F1 (Clasif.) / Test RMSE (Regres.).

Modelo	Variante	Cambio clave	ΔTest F1	ΔTest RMSE
RNN	+Profunda	$1 \rightarrow 2$ capas	+0.0X	-X.X
LSTM	+Bi-dir	'bidirectional=True'	+0.0X	-X.X
GRU	+Stack	$1 \rightarrow 2$ capas	+0.0X	-X.X

III-D. Curvas y matrices

Figura 1: Curvas de pérdida (Clasificación) (train/val).

Figura 2: Curvas de pérdida (Regresión, MSE) (train/val).

Figura 3: Matriz de confusión (Test) del mejor modelo de clasificación.

Figura 4: Gráfico de dispersión (Test) del mejor modelo de regresión (Predicho vs. Real).

IV. Análisis

Discuta si las **hipótesis** se confirman. Analice:

- **RNN vs LSTM/GRU:** ¿Se observó el problema del gradiente desvaneciente en la RNN simple (p.ej., estancamiento en el entrenamiento)?
- **Costo/Beneficio:** Comparar GRU vs LSTM. ¿Justifica LSTM su mayor costo computacional (más parámetros) con mejores métricas en este problema?
- **Clasif. vs Regres.:** ¿Qué tarea fue más difícil para los modelos? ¿Las variantes (p.ej., bidireccionalidad) ayudaron tanto a una tarea como a la otra?
- **Errores (Clasif.):** Analizar la matriz de confusión. ¿Qué fallas se confunden entre sí (p.ej., fallas leves con estado sano, o fallas en diferentes fases)?
- **Errores (Regres.):** Analizar el gráfico de dispersión. ¿El modelo es preciso en todo el rango, o falla más en valores extremos (p.ej., RUL muy bajo)?

V. Entregables

Informe (PDF) con filosofia IMRA con 1–2 pág. por sección, tablas/figuras y **apéndice** con hardware y tiempos.
Fijar **semillas**, documentar versiones y comandos de entrenamiento.

VI. Rúbrica (100 pts)

- Correctitud técnica (30): 3 modelos funcionales (c/u para 2 tareas); preprocesamiento de secuencias correcto.
- Diseño experimental (25): modelos base claros; variantes realistas; control de hiperparámetros.
- Análisis crítico (25): interpretación de métricas (F1 y RMSE); validación de hipótesis; comparación RNN/LSTM/GRU.
- Comunicación IMRA (15): claridad, figuras/tablas, redacción académica.
- Orden y reproducibilidad del código (5): modular, claro.