perplexity

Tarea a realizar:

- Carga y explora el conjunto de datos "Individual household electric power consumption dataset" (disponible en UCI Machine Learning Repository). Este conjunto de datos contiene mediciones del consumo de energía eléctrica en un hogar durante un período de cuatro años.
- Realiza un preprocesamiento adecuado de los datos, incluyendo la limpieza de valores faltantes, la normalización y la preparación de los datos para su uso en modelos de series temporales.
- Implementa un modelo de Red LSTM para predecir el consumo de energía eléctrica.
- Implementa un modelo de Red GRU para realizar la misma tarea.
- Evalúa el rendimiento de ambos modelos utilizando métricas como el Error Cuadrático Medio (RMSE) y el Error Absoluto Medio (MAE).
- Compara los resultados obtenidos por los modelos LSTM y GRU, analizando sus fortalezas y debilidades en este contexto específico.

En definitiva, se debe elaborar un informe conciso, usando un cuaderno Jupyter, que resuma los hallazgos, incluyendo gráficos que visualicen las predicciones de ambos modelos y una tabla comparativa de las métricas de evaluación

Here's how to approach the task of analyzing the household electric power consumption dataset, building LSTM and GRU models, and comparing their performance.

1. Data Loading and Exploration

- **Download the Dataset:** Obtain the dataset from the UCI Machine Learning Repository[1]. The file is named `"household_power_consumption.txt"` and is approximately 127MB[1].
- **Load the Data:** Use pandas to load the data. Specify the separator as `";"`, handle missing values represented by `"?"`, and parse the "Date" and "Time" columns into a single datetime column.

```
import pandas as pd
# Define column names
column_names = ['Date', 'Time', 'Global_active_power', 'Global_reactive_power', 'Volt
                'Global_intensity', 'Sub_metering_1', 'Sub_metering_2', 'Sub_metering

# Load the dataset
data = pd.read_csv('household_power_consumption.txt', sep=';', header=0, names=columr
                    low_memory=False, na_values=['?'], parse_dates={'Datetime': ['Da
                    infer_datetime_format=True)
```

```
# Set 'Datetime' as index
data = data.set_index('Datetime')
```

- **Explore the Data:** Examine the data types, missing values, and summary statistics.

2. Data Preprocessing

- **Handle Missing Values:** The dataset has missing values, approximately 1.25% of the rows[2]. Impute these using appropriate methods such as mean or median imputation, or more sophisticated techniques like interpolation[2].

```
# Impute missing values with the mean
for col in data.columns:
    data[col] = data[col].fillna(data[col].mean())
```

- **Normalize the Data:** Scale the data using techniques like MinMaxScaler to improve model performance.

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
data['Global_active_power'] = scaler.fit_transform(data[['Global_active_power']])
```

- **Prepare Time Series Data:** Convert the data into a format suitable for time series forecasting. This typically involves creating sequences of data points to use as input features and corresponding target values.

3. LSTM Model Implementation

- **Define the Model:** Create an LSTM model using a deep learning framework like TensorFlow or PyTorch. The model should consist of LSTM layers followed by dense layers for prediction.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense

# Define the LSTM model
model_lstm = Sequential()
model_lstm.add(LSTM(50, activation='relu', input_shape=(X_train.shape[^1], X_train.sh
model_lstm.add(Dense(1))
model_lstm.compile(optimizer='adam', loss='mse')
```

- **Train the Model:** Train the LSTM model on the prepared data. Use a portion of the data for validation to monitor performance during training.

```
# Train the LSTM model
history_lstm = model_lstm.fit(X_train, y_train, epochs=50, batch_size=72, validation_
```

4. GRU Model Implementation

- **Define the Model:** Create a GRU model similar to the LSTM model.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import GRU, Dense

# Define the GRU model
model_gru = Sequential()
model_gru.add(GRU(50, activation='relu', input_shape=(X_train.shape[^1], X_train.shap
model_gru.add(Dense(1))
model_gru.compile(optimizer='adam', loss='mse')
```

- **Train the Model:** Train the GRU model using the same data and training parameters as the LSTM model.

```
# Train the GRU model
history_gru = model_gru.fit(X_train, y_train, epochs=50, batch_size=72, validation_da
```

5. Model Evaluation

- **Make Predictions:** Use the trained LSTM and GRU models to make predictions on the test data.

```
# Make predictions with LSTM
y_pred_lstm = model_lstm.predict(X_test)

# Make predictions with GRU
y_pred_gru = model_gru.predict(X_test)
```

- **Evaluate Performance:** Calculate RMSE and MAE to evaluate the performance of both models.

```
from sklearn.metrics import mean_squared_error, mean_absolute_error
import numpy as np

# Calculate RMSE and MAE for LSTM
rmse_lstm = np.sqrt(mean_squared_error(y_test, y_pred_lstm))
mae_lstm = mean_absolute_error(y_test, y_pred_lstm)

# Calculate RMSE and MAE for GRU
rmse_gru = np.sqrt(mean_squared_error(y_test, y_pred_gru))
mae_gru = mean_absolute_error(y_test, y_pred_gru)

print(f'LSTM RMSE: {rmse_lstm:.4f}, MAE: {mae_lstm:.4f}')
print(f'GRU RMSE: {rmse_gru:.4f}, MAE: {mae_gru:.4f}')
```

6. Comparison and Analysis

- **Compare Metrics:** Compare the RMSE and MAE values obtained for the LSTM and GRU models.
- **Analyze Strengths and Weaknesses:** Discuss the strengths and weaknesses of each model based on the evaluation metrics and any observed behavior during training. LSTM networks are better at capturing long-term dependencies but are more computationally expensive, while GRU networks train faster and often perform comparably[2].

7. Report and Visualization

- **Jupyter Notebook:** Create a Jupyter Notebook to document the entire process, including code, results, and analysis.
- **Visualizations:** Include plots visualizing the predictions of both models against the actual values.
- **Comparative Table:** Summarize the evaluation metrics in a table.

Metric	LSTM	GRU
RMSE	value	value
MAE	value	value

⁂

1. https://www.machinelearningmastery.com/how-to-load-and-explore-household-electricity-usage-data/
2. https://archive.ics.uci.edu/dataset/235/individual+household+electric+power+consumption