Intel® Unnati
Data-Centric Labs in Emerging Technologies

A Report on

Power Manager Telemetry

Submitted for the Intel Unnati Industrial Training Program 2024

Agile Innovators

L Shriya Reddy 1NT21IS082
Rahul KC 1NT21EC106

Under the Guidance of

Dr. Ramachandra A C
Professor
Dept. of Electronics and Communication Engineering

NITTE EDUCATION TRUST | NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

YELAHANKA, BENGALURU- 560063

Introduction

In the era of increasing energy consumption and environmental concerns, optimizing power usage in electronic devices has become a critical challenge. This project focuses on utilizing machine learning techniques to analyze and predict power consumption patterns in laptop systems, specifically through the lens of power manager telemetry. By systematically collecting and analyzing power consumption data from various components, we aim to enhance energy efficiency and reduce unnecessary power usage. This report outlines the objectives, methodology, technologies used, and expected outcomes of the project.

Problem Statement

The primary goal of this project is to optimize the power consumption of systems by analyzing and measuring power usage patterns. This involves the collection of power consumption data, identification of inefficiencies, and implementation of strategies to enhance energy efficiency.

Solution Overview

The proposed solution leverages open-source tools to measure power consumption data from various system components (CPU, memory, NIC, TDP). By analyzing this data and identifying inefficiencies, the project aims to implement dynamic strategies for optimizing power consumption.

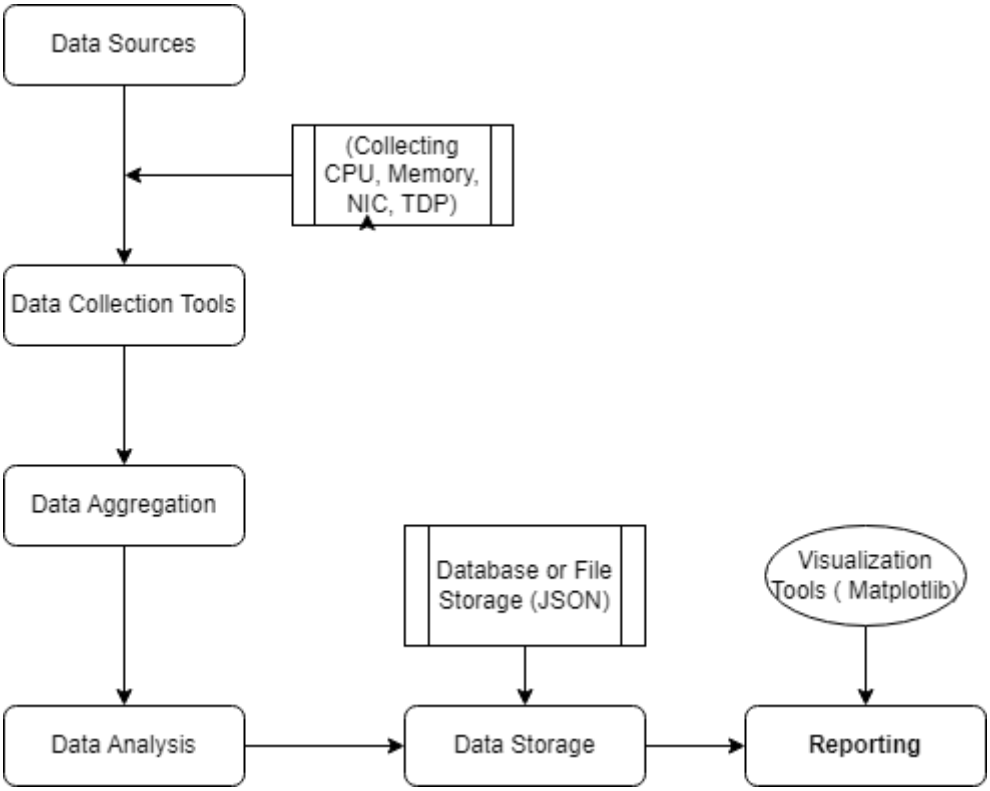
Key Features

- Real-time Power Monitoring: Collect and display real-time power consumption data.
- Historical Data Analysis: Analyze historical power consumption to identify usage patterns.
- Optimization Suggestions: Provide actionable recommendations for power usage optimization.

Methodology

- Data Collection: Utilize open-source tools such as Powerstat and Powertop to gather power consumption data.
- Data Pre-processing: Clean and pre-process the collected data for further analysis.
- Data Analysis: Analyze historical data to identify patterns and inefficiencies.
- Optimization Implementation: Use insights from the analysis to suggest and implement power-saving strategies.

- Implementation: Implement the optimization strategies and monitor their effectiveness continuously.



Technologies Used

Data Collection:

- **Powerstat and Powertop**: Monitors power consumption metrics for various components.
- **Custom Scripts**: Developed to collect system metrics (e.g., `psutil` for CPU, memory, NIC data).

Data Preprocessing:

- **Pandas**: For data manipulation and analysis.
- **NumPy**: For numerical operations and handling arrays.

Data Storage:

- **JSON Files**: Format for storing collected and processed data.

Data Analysis:

- **Matplotlib/Seaborn**: For data visualization and pattern identification.

Advantages and Limitations of Power Manager Telemetry

Advantages

1. Energy Efficiency: Optimized power usage leads to more efficient energy use and extended device lifespan.
2. Cost Savings: Reduces electricity bills and operational costs.
3. Performance Optimization: Balances performance and power usage for optimal results.
4. Environmental Impact: Contributes to lower carbon emissions and resource conservation.
5. Data-Driven Decisions: Enables informed optimization strategies based on detailed telemetry data.

Limitations

1. Data Accuracy: Requires precise sensor calibration and accurate data collection methods.
2. Data Volume: Generates large volumes of data, necessitating substantial storage capacity and processing power.
3. Real-time Processing: Can be computationally intensive and may introduce latency issues.
4. Security Concerns: Requires robust security measures to protect sensitive telemetry data.
5. Implementation Complexity: Requires specialized knowledge and skills for setup and maintenance.
6. Cost: Involves upfront investment and ongoing maintenance expenses.

Code

File: **`power_telemetry.py`**

```
python
Copy code
import psutil
import time
import json

def collect_power_data(interval=5):
    utilization_log = []
    while True:
        timestamp = time.time()
        cpu_percent = psutil.cpu_percent()
        memory_info = psutil.virtual_memory()
        bytes_sent = psutil.net_io_counters().bytes_sent
        bytes_recv = psutil.net_io_counters().bytes_recv
```

```
        power_cpu = cpu_percent * 0.05  # Example conversion factor
        power_memory = memory_info.percent * 0.2  # Example conversion
factor
        power_nic = (bytes_sent + bytes_recv) * 0.0001  # Example
conversion factor
        total_power = power_cpu + power_memory + power_nic

        utilization = {
            "timestamp": timestamp,
            "utilization": {
                "cpu_percent": cpu_percent,
                "memory_percent": memory_info.percent,
                "bytes_sent": bytes_sent,
                "bytes_recv": bytes_recv
            },
            "power": {
                "power_cpu": power_cpu,
                "power_memory": power_memory,
                "power_nic": power_nic,
                "total_power": total_power
            }
        }

        utilization_log.append(utilization)
        with open('utilization_log.json', 'a') as f:
            f.write(json.dumps(utilization) + '\n')

        time.sleep(interval)

if __name__ == "__main__":
    collect_power_data()
```

File: `plot_power_data.py`

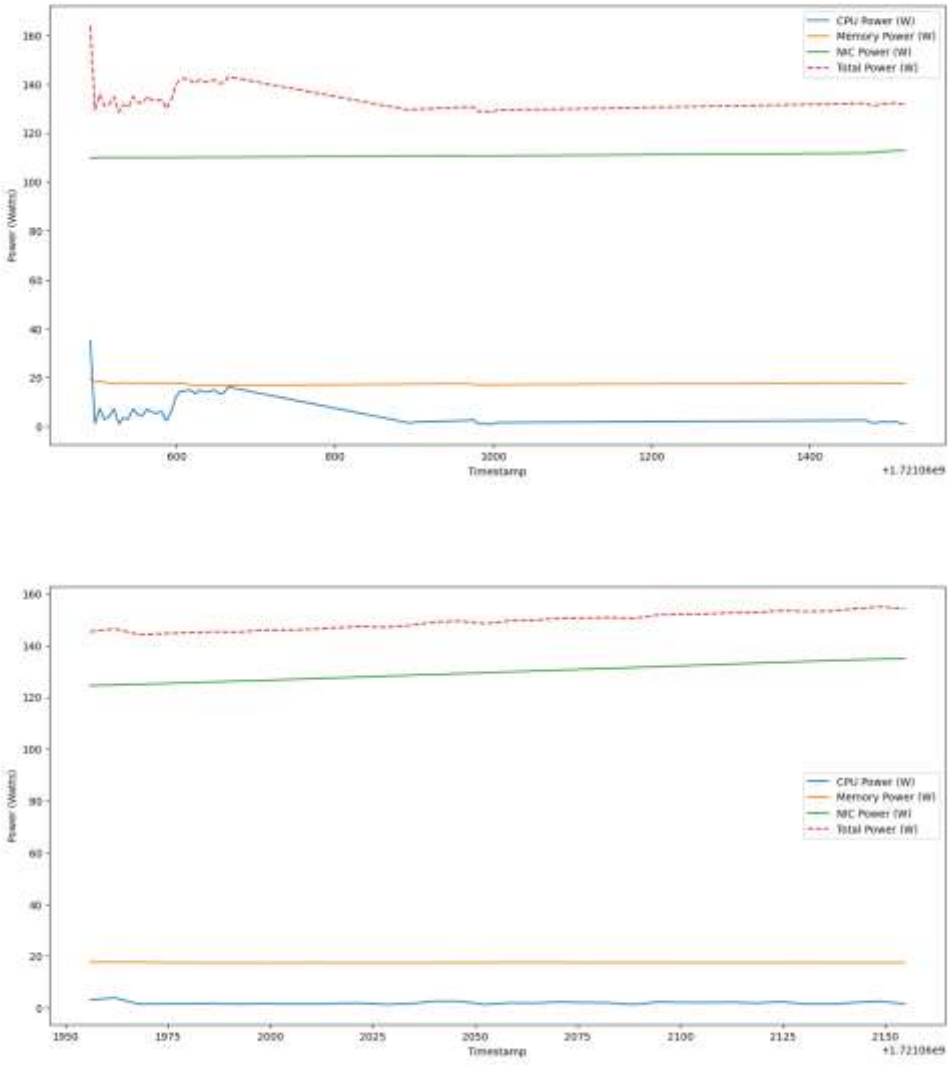
```
python
Copy code
import matplotlib.pyplot as plt
import json

# Load data from the JSON file
with open('utilization_log.json') as f:
    data = [json.loads(line) for line in f]

# Extract relevant data for plotting
timestamps = [entry['timestamp'] for entry in data]
cpu_power = [entry['power']['power_cpu'] for entry in data]
memory_power = [entry['power']['power_memory'] for entry in data]
nic_power = [entry['power']['power_nic'] for entry in data]
total_power = [entry['power']['total_power'] for entry in data]

# Plotting the power utilization
plt.figure(figsize=(12, 6))
plt.plot(timestamps, cpu_power, label='CPU Power')
plt.plot(timestamps, memory_power, label='Memory Power')
plt.plot(timestamps, nic_power, label='NIC Power')
plt.plot(timestamps, total_power, label='Total Power')
plt.xlabel('Timestamp')
plt.ylabel('Power Consumption (W)')
plt.title('Power Utilization Over Time')
plt.legend()
plt.show()
```

Results



The chart below displays the power utilization over time for various components such as CPU, memory, and NIC. This visualization helps in identifying patterns and potential inefficiencies in power usage, providing a basis for optimization strategies.

Conclusion

This project effectively demonstrates the application of machine learning algorithms in optimizing power consumption within laptop systems. By gathering detailed power usage data and employing advanced analytical techniques, the project can predict future consumption patterns and uncover inefficiencies. The resulting solution not only facilitates real-time monitoring and historical data analysis but also provides actionable recommendations for enhancing energy efficiency. This approach has the potential for extension to other systems and devices, contributing to overall energy savings and improved operational performance.