温州肯恩大学
WENZHOU-KEAN UNIVERSITY

CPS 5745: Interactive Data Visualization

Dr. Hamza Djigal

These projects aim to provide students with hands-on experience in both interactive data visualization and integration with advanced technologies in AI, IoT, Edge, and Cloud computing.

1. AI-Powered Real-Time Object Detection Dashboard

Technology and Tools:

- **Libraries**: OpenCV (video processing), TensorFlow or PyTorch (AI models)
- **Backend**: Flask or Django for data management
- **Visualization**: Plotly/Dash or Streamlit for real-time interactive dashboards

Project Outline:

1. **Set up the object detection model** (YOLO, MobileNet, etc.) with OpenCV for live video feed processing.
2. **Build the dashboard** using Dash/Streamlit to visualize:
    - Object detections in real time with bounding boxes
    - Frequency of detected objects over time
    - Heatmaps showing movement of objects over time
3. **Integrate model and dashboard** so users can see live updates and historical trends.

Poster Presentation:

- **Introduction**: Briefly describe the use of AI in real-time object detection.
- **Objective**: Demonstrate the power of AI to extract meaningful insights from visual data in real time.
- **Technical Workflow**: Outline model training (if applicable), the real-time processing pipeline, and dashboard integration.
- **Results**: Show screenshots of the dashboard with sample detection results and insights.
- **Future Scope**: Extending to multiple camera feeds or adding advanced analytics (e.g., object tracking).

---

2. Edge Device Performance Monitoring System

Technology and Tools:

- **Libraries**: psutil (for system performance metrics), Flask or Django (for backend)
- **Visualization**: Plotly/Dash for real-time metrics display
- **Data Source**: Raspberry Pi or other edge devices (real or simulated)

Project Outline:

1. **Set up performance metric collection** using psutil to gather data (CPU/GPU usage, memory, power).
2. **Simulate multiple edge devices** or connect real devices like Raspberry Pi.
3. **Create an interactive dashboard** to show:
    - Real-time metrics per device
    - Alerts for high resource usage
    - Historical trend charts for performance metrics

Poster Presentation:

- **Introduction**: Explain edge computing and the importance of monitoring device health.
- **Objective**: Demonstrate efficient resource tracking and management on edge devices.
- **Architecture Diagram**: Show the data flow from device to dashboard.
- **Results and Insights**: Display examples of real-time monitoring and alerts.
- **Future Enhancements**: Implement predictive analytics for performance bottlenecks.

---

3. IoT Sensor Data Visualization for Smart Home

Technology and Tools:

- **Libraries**: Pandas (data manipulation), MQTT (data collection)
- **Backend**: Flask or Django to manage sensor data
- **Visualization**: Bokeh or Plotly for interactive and real-time visualizations

Project Outline:

1. **Simulate or collect sensor data** (temperature, humidity, occupancy) from IoT devices.
2. **Set up an MQTT broker** to handle real-time data collection.
3. **Develop a dashboard** with:
    - Live readings and trend charts
    - Room-by-room data filtering options
    - Anomaly detection with alerts for unusual conditions

Poster Presentation:

- **Introduction**: Describe IoT applications for smart homes.

- **Objective**: Real-time monitoring of environmental conditions for optimal comfort and safety.
- **Technical Workflow**: Explain MQTT data flow and dashboard setup.
- **Sample Visuals**: Show room-level views and data filtering features.
- **Next Steps**: Add predictive maintenance or energy-saving suggestions.

---

4. Cloud Service Latency and Uptime Analysis Tool

Technology and Tools:

- **Libraries**: Requests (API calls), Pandas (data manipulation)
- **Visualization**: Streamlit or Plotly for latency trends and interactive analysis

Project Outline:

1. **Set up API calls** to gather latency and uptime data from cloud providers.
2. **Create a data pipeline** to regularly collect and store performance metrics.
3. **Develop a dashboard** with:
    - Regional latency comparisons
    - Historical uptime trends
    - Filtering options by service or provider

Poster Presentation:

- **Introduction**: Describe the importance of monitoring cloud services.
- **Objective**: Compare performance and uptime of various cloud services.
- **Data Flow Diagram**: Show data collection from APIs to dashboard.
- **Sample Charts**: Visuals comparing latencies and uptimes.
- **Future Scope**: Include more granular metrics or expand to more cloud providers.

---

5. Real-Time Traffic Flow Visualization with AI

Technology and Tools:

- **Libraries**: Scikit-learn or TensorFlow for traffic prediction model, Plotly/Dash or Leaflet for mapping
- **Backend**: Flask for model integration

Project Outline:

1. **Collect or simulate real-time traffic data** for a city or region.
2. **Train a simple ML model** to predict future traffic flow.

3. **Build an interactive dashboard** with:
    - Real-time and predicted traffic conditions
    - Interactive heatmaps and filtering options

Poster Presentation:

- **Introduction**: Explain traffic prediction applications in smart cities.
- **Objective**: Use AI to visualize and predict traffic flow in real time.
- **Data Pipeline**: Show how data is collected, processed, and visualized.
- **Visuals**: Display sample heatmaps and prediction accuracy.
- **Future Work**: Expand to multiple cities or add congestion forecasting.

---

6. Edge AI Device Failure Prediction and Visualization

Here are some practical student projects for an interactive data visualization course with applications in AI, edge computing, cloud computing, and IoT. Each project idea involves Python for data processing and visualization and provides a unique learning opportunity.

1. AI-Powered Real-Time Object Detection Dashboard

- **Objective**: Develop a dashboard that visualizes real-time object detection from a video feed (e.g., webcam or security camera).
- **Details**: Use a pre-trained deep learning model (e.g., YOLO or MobileNet) to detect objects in the video feed, then visualize detections on a dashboard. Display real-time data about object frequency, position, and movement trends.
- **Technologies**: Python (OpenCV for video processing), TensorFlow or PyTorch (for object detection model), Plotly/Dash or Streamlit (for interactive visualization).

2. Edge Device Performance Monitoring System

- **Objective**: Create a visualization tool that tracks and visualizes edge device performance metrics like CPU/GPU usage, memory, and power consumption.
- **Details**: Simulate or collect data from multiple edge devices (e.g., Raspberry Pi or other low-power devices) and display metrics on an interactive dashboard, highlighting areas of high resource usage or potential performance bottlenecks.
- **Technologies**: Python (for data gathering and processing), Flask or Django (for backend), D3.js or Plotly for visualization.

3. IoT Sensor Data Visualization for Smart Home

- **Objective**: Develop a real-time monitoring system that displays data from simulated or real IoT sensors in a smart home setup (temperature, humidity, light, occupancy).
- **Details**: Visualize live sensor data with historical trends and alert features for anomalies (e.g., sudden temperature spikes). Allow users to filter data by room or sensor type.

- **Technologies**: Python (Pandas for data manipulation, MQTT for IoT data simulation), Plotly/Dash or Bokeh for visualization.

4. Cloud Service Latency and Uptime Analysis Tool

- **Objective**: Visualize the performance and uptime of different cloud services (AWS, Azure, Google Cloud) over time.
- **Details**: Gather latency data from different regions or services, then create interactive visualizations that allow users to see uptime and latency trends and compare providers.
- **Technologies**: Python (Requests for API calls, Pandas for data analysis), Flask/Dash, D3.js, or Matplotlib for visualizations.

5. Real-Time Traffic Flow Visualization with AI

- **Objective**: Use AI models to predict and visualize traffic flow in real time based on historical traffic data.
- **Details**: Collect traffic data from a real-time source or simulate data, then use a simple machine learning model to predict future traffic flow. Visualize current and predicted traffic conditions with an interactive heatmap.
- **Technologies**: Python (Scikit-learn for ML model, Pandas for data manipulation), Plotly/Dash or Leaflet for mapping and visualization.

6. Edge AI Device Failure Prediction and Visualization

- **Objective**: Predict potential hardware failures in edge AI devices and visualize their status on an interactive dashboard.
- **Details**: Use historical performance data to predict failures using machine learning algorithms, then visualize the probability of failure along with device health and performance metrics.
- **Technologies**: Python (TensorFlow/Scikit-learn for predictive model), Pandas for data processing, Streamlit or Bokeh for visualization.

7. Environmental Monitoring with IoT and AI for Anomaly Detection

- **Objective**: Develop a system that uses AI to detect anomalies in environmental data (e.g., air quality, temperature) collected from IoT sensors.
- **Details**: Use simulated IoT data for environmental metrics and apply an anomaly detection algorithm to identify unusual patterns. Visualize real-time and historical data with alerts for anomalies.
- **Technologies**: Python (Scikit-learn for anomaly detection, Pandas for data manipulation), Plotly/Dash or D3.js for interactive visualizations.

8. Cloud Resource Optimization Visualization Tool

- **Objective**: Visualize the usage patterns and optimize resource allocation for a cloud environment.

- **Details**: Collect and analyze cloud resource usage metrics (e.g., storage, bandwidth, CPU). Build a dashboard with visualizations to identify underused and overused resources and provide recommendations for cost optimization.
- **Technologies**: Python (Requests for data collection, Pandas for analysis), Streamlit or Flask/Dash for visualization, Plotly for interactive plots.

9. Energy Consumption Monitoring for IoT Devices

- **Objective**: Visualize energy consumption of IoT devices in real time.
- **Details**: Collect energy usage data from IoT devices or simulate data, displaying it on a dashboard with real-time charts. Include options to filter by device type and view aggregated statistics.
- **Technologies**: Python (Pandas for data processing, MQTT for simulated IoT data), Plotly/Dash or D3.js for interactive visualizations.

10. AI-Driven Predictive Maintenance Dashboard for Industrial IoT

- **Objective**: Create a dashboard that visualizes predictive maintenance insights for industrial IoT equipment.
- **Details**: Collect or simulate equipment data (e.g., vibration, temperature) and train a predictive model to anticipate failures. Visualize current status, predicted health, and historical trends for each machine in an interactive format.
- **Technologies**: Python (TensorFlow or Scikit-learn for predictive model), Flask or Streamlit, Plotly for visualization.

Technology and Tools:

- **Libraries**: Scikit-learn (predictive model), Pandas for data processing
- **Visualization**: Streamlit or Bokeh for device health tracking

Project Outline:

1. **Collect performance data** (e.g., temperature, usage) and build a dataset for predictive modeling.
2. **Train a predictive model** to estimate failure risk.
3. **Create an interactive dashboard** that shows:
    - Device health status and alerts for potential failures
    - Historical trends and predictive risk

Poster Presentation:

- **Introduction**: Describe predictive maintenance for edge devices.

- **Objective**: Predict device failure risk based on health metrics.
- **Model Flow**: Show how data is processed, modeled, and visualized.
- **Sample Dashboard**: Visuals of device health and predictions.
- **Next Steps**: Improve model accuracy with larger datasets.

---

7. Environmental Monitoring with IoT and AI for Anomaly Detection

Technology and Tools:

- **Libraries**: Scikit-learn (for anomaly detection), Pandas for data handling
- **Visualization**: Dash or Plotly for real-time data display

Project Outline:

1. **Simulate environmental data** from IoT sensors.
2. **Apply anomaly detection** using clustering or statistical methods.
3. **Build a dashboard** with:
    - Real-time sensor data
    - Anomaly alerts with a focus on temperature, air quality, etc.

Poster Presentation:

- **Introduction**: Discuss environmental monitoring and anomaly detection.
- **Objective**: Monitor and detect anomalies in environmental data.
- **Data and Model Flow**: Explain data collection, processing, and visualization.
- **Visualization Examples**: Show real-time readings and anomaly alerts.
- **Future Scope**: Add machine learning for more sophisticated detections.

---

8. Cloud Resource Optimization Visualization Tool

Technology and Tools:

- **Libraries**: Pandas (data manipulation), Requests for API data gathering
- **Visualization**: Plotly, Dash, or Streamlit

Project Outline:

1. **Collect resource usage data** from cloud services.
2. **Analyze data** to identify underused and overused resources.
3. **Build a visualization tool** with:
    - Usage charts by resource type
    - Recommendations for optimization

Poster Presentation:

- **Introduction**: Explain the need for cloud resource optimization.
- **Objective**: Visualize cloud resource use and suggest optimizations.
- **Process Flow**: Outline data collection, processing, and dashboard setup.
- **Sample Dashboard**: Show usage charts and recommendations.
- **Next Steps**: Implement automated optimization recommendations.

---

9. Energy Consumption Monitoring for IoT Devices

Technology and Tools:

- **Libraries**: Pandas for data manipulation, MQTT for data simulation
- **Visualization**: Bokeh or Plotly for real-time data visualization

Project Outline:

1. **Simulate energy data** for IoT devices.
2. **Set up real-time data processing** with MQTT or similar protocol.
3. **Develop a dashboard** with:
    - Live energy usage per device
    - Historical consumption trends

Poster Presentation:

- **Introduction**: Discuss energy monitoring in IoT.
- **Objective**: Visualize real-time and historical energy data.
- **Architecture**: Explain data collection and visualization pipeline.
- **Sample Visuals**: Show device-level energy use and trends.
- **Future Work**: Add predictive analytics for energy optimization.

---

10. AI-Driven Predictive Maintenance Dashboard for Industrial IoT

Technology and Tools:

- **Libraries**: TensorFlow or Scikit-learn (predictive model), Flask or Streamlit
- **Visualization**: Plotly or Bokeh for trend analysis and predictions

Project Outline:

1. **Collect data** (temperature, vibration) from industrial IoT devices.
2. **Train a predictive model** to estimate equipment health.

3. **Build an interactive dashboard** with:
    - Equipment health scores
    - Predictions for potential failures

Poster Presentation:

- **Introduction**: Describe the importance of predictive maintenance.
- **Objective**: Predict maintenance needs based on device health.
- **Model and Data Flow**: Show data, model, and visualization pipeline.
- **Dashboard Example**: Display health trends and predictions.
- **Next Steps**: Add more predictive analytics and expand to other equipment.
