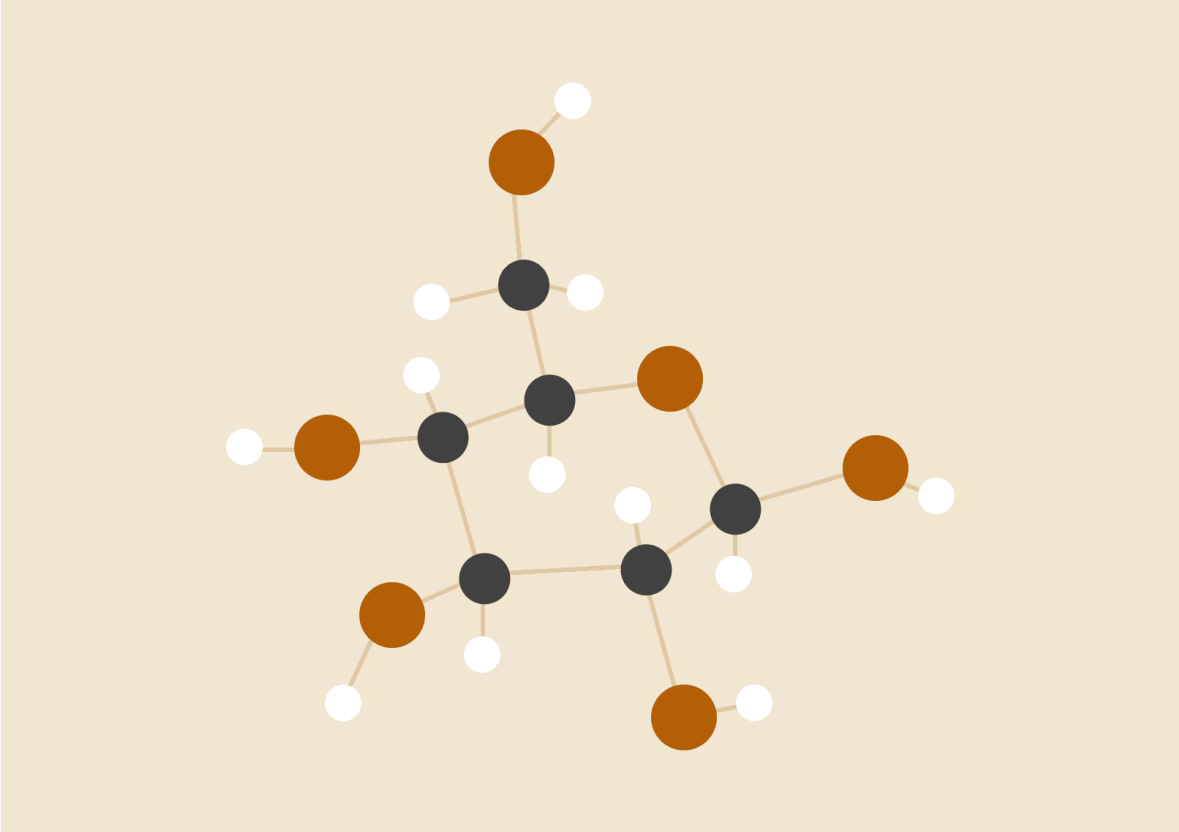
CSTR Domain Adaptation and Fault Diagnosis



Shivam Adbute

210107078

CL 653

Date of submission- 29th march 2024

Project Overview

Introduction

Automatic fault diagnosis systems are an important component for fault tolerance in modern control loops. Nonetheless, the training of such diagnosis systems can be costly or even dangerous since faulty data need to be collected by driving the process to dangerous conditions. A possible solution to the said problem is training an automatic diagnosis system solely on simulation data. However, due to modeling errors, the data acquired may not reflect real process data. This is characterized by a change in the probability distributions upon which data is sampled. This problem is known in the literature as domain adaptation or cross-domain fault diagnosis in our context. Thus this work analyzes the cross-domain diagnosis problem through the point of view of optimal transport.

Objective

We apply our methodology in a case study concerning the continuous stirred tank reactor (CSTR) system. Our contributions are threefold:

1. we perform a comparative study concerning feature extraction and domain adaptation algorithms

2. we analyze the relation between wrongful model specification and the distance between source and target distributions, and its impact on classification performance

3. we analyze the impact of modeling errors in the quality of optimal transport plans, and the influence of this latter factor into classification performance

Description

Theoretical Background:

- **Automatic Fault Diagnosis Systems:** These systems analyze process data to detect and diagnose faults in control systems, enhancing their fault tolerance and overall reliability.
- **Domain Adaptation:** Domain adaptation focuses on transferring knowledge from a source domain (e.g., simulation data) to a target domain (e.g., real process data), despite differences in their distributions.
- **Optimal Transport Theory:** Optimal transport, also known as Wasserstein distance or earth mover's distance, provides a mathematical framework for measuring the discrepancy between probability distributions and finding an optimal mapping between them.

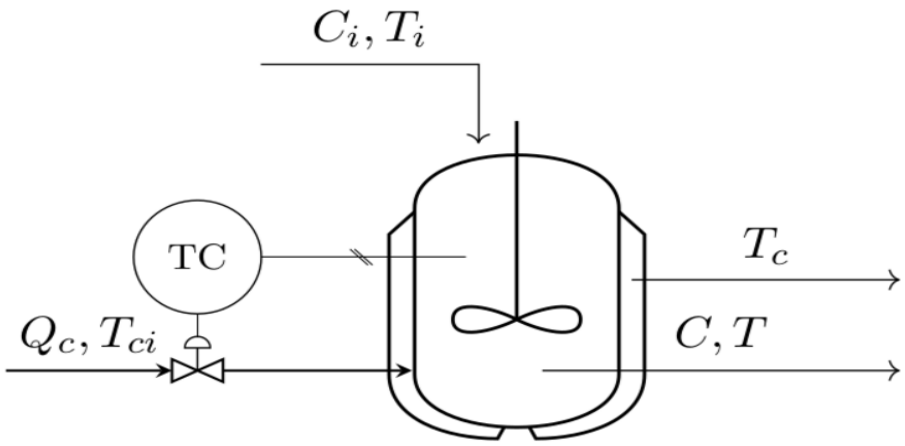
Problem Statement:

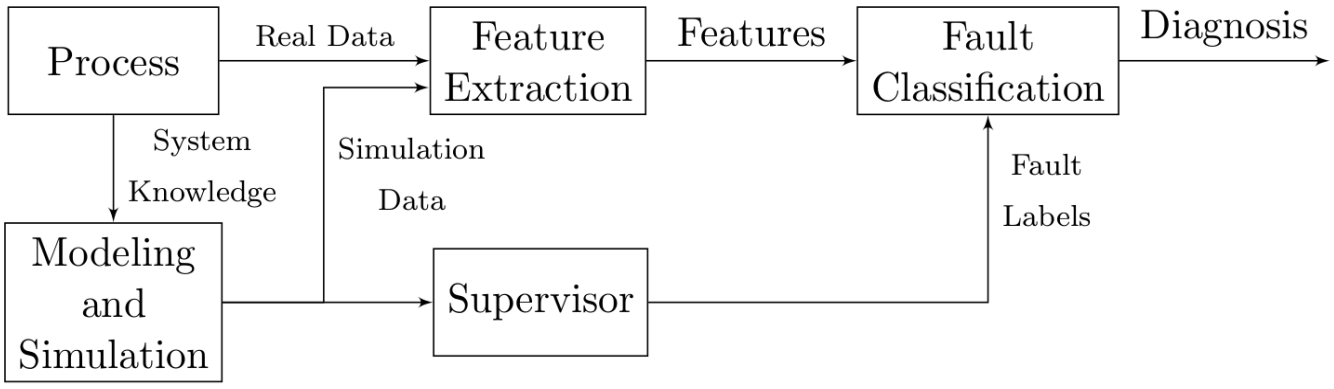
The main problem addressed in this work is the cross-domain fault diagnosis problem, where a fault diagnosis system trained on simulation data needs to accurately detect faults in real-world processes despite differences in their underlying distributions

Significance of Addressing the Issue:

- **Safety:** By accurately diagnosing faults in real-time, automatic fault diagnosis systems contribute to the safety of industrial processes and minimize the risk of accidents.
- **Reliability:** Robust fault diagnosis systems enhance the reliability of control loops, reducing downtime and improving overall system performance.
- **Cost-effectiveness:** By leveraging simulation data and mitigating the need for collecting real faulty data, the proposed methodology offers a cost-effective

Block flow diagram





Structure of my AI/ML model

Data source and description

This data was collected by the Department of Teleinformatics Engineering, Federal University of Ceará, Brazil,conducting a 200 min. Analysis on an exothermic reactor. This dataset contains a set of simulations of a known benchmark in the fault diagnosis for the chemical processes community, i.e The Continuous Stirred Tank Reactor is Dynamic by nature. This system carries an exothermic reaction A -> B, and a feedback loop for controlling the reactor's temperature. There are 7 measured variables,

- Concentration of A in the inlet flow,
- Temperature of the inlet flow,
- Temperature of the inlet coolant flow,
- Coolant flow-rate,
- Concentration of B in the outlet flow,
- Temperature of the outlet flow,
- Temperature of the outlet coolant flow.

This is a 2860 x 1404 matrix. Each row contains an independent simulation of the CSTR process. The first 1400 columns contain 200 measurements from 7 process variables. The last 4 columns contain: class labels, domain labels, and the simulation parameter noise and reaction order. The goal is to predict a set of 12 faults from these 7 variables, measured throughout 200 minutes, at a 1 minute rate.

To prepare my data for analysis ,here are the anticipated data preprocessing steps:

- Address missing values (sensor errors) using imputation or interpolation.
- Normalize data (scale variations) using min-max scaling or z-score normalization.
- Segment data (optional, for LSTMs) into smaller time chunks based on operations.
- Augment data (for domain adaptation) to create variations and improve generalizability.

Strategies for AI/ML Model Development

For my CSTR domain adaptation and fault diagnosis project, a 1D Convolutional Neural Network (CNN) is a strong choice due to the following reasons:

- **Time Series Data:** CNNs are adept at extracting relevant features from sequential data like time series. In this case, the 1D CNN will analyze the sequence of measurements from the 7 variables across 200 minutes.
- **Feature Extraction:** CNNs can automatically learn informative features from the raw data without the need for manual feature engineering. This is crucial as the fault patterns might be complex and not easily defined manually.
- **Fault Detection:** By learning these features, the CNN can effectively identify patterns indicative of specific fault types within the time series data.

1d CNN Model Training The training process includes the following stages:

- **Data Preprocessing:** normalization, and potential segmentation into smaller time subsequences.
- **Model Architecture:** the 1d CNN model will have several convolutional and pooling layers to extract features and dimensionality reductions. Moreover, layers with activation functions such as ReLU and fully connected layers will be added to classify 12 fault categories.
- **Training Parameters**: hyperparameters, such as the number of filters and kernel, can be tuned during the training process..

To evaluate my CSTR fault diagnosis model, use accuracy, F1-score, and a confusion matrix. Accuracy measures overall performance, F1-score balances precision and recall for various faults, and the confusion matrix identifies issues with specific fault classes.

Validating my CNN model with k-fold cross-validation to assess performance on unseen data within the same domain. Utilize hyperparameter tuning to prevent overfitting.

Deployment Strategy for CSTR Fault Diagnosis CNN

Deployment Plan:

1. **Cloud-Based Deployment:** Consider deploying the model on Google Cloud Platform (GCP. It offers scalable computing resources, fault tolerance, and easy integration with other services.

2. **API Integration:** Develop a RESTful API that encapsulates the model's functionality. This API will receive real-time sensor data from the CSTR system, process it through the model, and return the predicted fault diagnosis.

3. **User Interface:** Depending on the needs, a user interface (UI) can be developed.

Maintenance and Updates:

- **Monitoring Performance:** Continuously monitor the model's performance in production using metrics like accuracy and F1-score. This helps identify potential performance degradation.

- **Version Control:** Implement a version control system to track changes made to the model and deployment environment. This facilitates rollbacks if necessary and ensures a clear audit trail for future reference.

By following these steps,I will effectively deploy my CNN model for real-world CSTR fault diagnosis, enabling integration with existing systems, user interaction, and ongoing maintenance for long-term success.

Scalability and Performance Optimization

Scalability:

As your CSTR operation grows or data collection becomes more extensive, here's how to scale the CNN model:

- **Hardware Upgrades:** Utilize hardware with higher processing power like GPUs or TPUs specifically designed for deep learning workloads. Cloud platforms often offer flexible resource allocation, allowing you to scale compute resources up or down based on demand.
- **Model Architecture Optimization:** Explore techniques like model pruning or quantization. Pruning removes redundant connections within the CNN, while quantization reduces the precision of weights and activations, both leading to smaller model size and faster inference times.

Performance Optimization:

To squeeze the best performance out of your CNN:

- **Hyperparameter Tuning:** Refine hyperparameter tuning beyond the initial training phase. Techniques like Bayesian Optimization can be used for more efficient hyperparameter search, potentially leading to further accuracy improvements.
- **Data Augmentation:** For larger and more complex datasets, explore advanced data augmentation techniques. This can involve generating synthetic variations of existing data points, enriching the training data and potentially improving the model's ability to generalize to unseen scenarios.
- **Early Stopping:** Implement early stopping during training to prevent overfitting. This technique monitors the model's performance on a validation set and stops training once validation accuracy plateaus or starts to decline.

Open-Source Tools

Core Libraries:

- **TensorFlow:** This powerful deep learning framework will be used to build, train, and evaluate the 1D CNN model for fault diagnosis..
- **Scikit-learn:** This versatile machine learning library will be used for data preprocessing tasks such as feature normalization, data splitting, and potentially model evaluation.

Data Visualization:

- **NumPy:** Fundamental for numerical computations and array manipulation, providing the core data structures for handling sensor data and model output.
- **Pandas:** Enables convenient data cleaning, exploration, and analysis, allowing you to organize and understand the CSTR dataset effectively.
- **Seaborn:** Simplifies the creation of informative and visually appealing plots for visualizing data distributions, model performance, and insights..
- **Matplotlib:** Offers a comprehensive plotting library for more customized visualizations, enabling fine-grained control over the creation of informative and compelling figures.

Collectively, these open-source tools provide a comprehensive and robust foundation for developing, evaluating, and deploying a high-performance CNN model for CSTR fault diagnosis, with a specific focus on domain adaptation.

Real-World Application and Impact of CSTR Fault Diagnosis

This project tackles a crucial challenge in the chemical process industry: pinpointing faults in CSTR systems early and accurately. The developed CNN model offers a practical solution that integrates seamlessly into real-world scenarios:

Application:

1. **Continuous Monitoring:** The model will be integrated into existing CSTR monitoring systems. It will continuously receive real-time sensor data streams, including temperature, flow rates, and concentration measurements.
2. **Intelligent Diagnosis:** The CNN analyzes this data in real-time, promptly identifying any developing faults within the CSTR.
3. **Early Intervention:** Early detection enables timely corrective actions to be taken. This prevents equipment damage, production stoppages, and potential safety hazards.

Impact:

- **Chemical Manufacturing Companies:**
    - Improved CSTR uptime translates to higher production efficiency.
    - Reduced maintenance costs due to early fault detection and preventive measures.
- **Safety Personnel:**
    - Early identification of potential safety hazards allows for proactive measures to ensure safe plant operation, minimizing risks to workers.

Conclusion

This research investigated optimal approaches for addressing distributional shift in domain adaptation tasks, specifically focusing on CSTR fault diagnosis. The key findings are:

- **Optimal Domain Adaptation:** Our analysis suggests that **optimal transport-based domain adaptation** is the most effective technique for tackling the distributional shift problem in this context.
- **Modeling Error and Distributional Distance:** We established a positive correlation between the degree of modeling error and the distance between source and target domain distributions.
- **Performance and Distributional Distance:** Our experiments confirm theoretical predictions, demonstrating a correlation between a larger distance between source and target distributions and a decrease in classification performance.
- **Modeling Error and Mass Transfer:** Furthermore, we discovered that high modeling error can negatively impact the transportation plan, causing unintended mass transfer between different fault classes, ultimately harming classification performance.

These findings offer valuable insights for developing robust domain adaptation methods in CSTR fault diagnosis and potentially other scenarios with similar distributional shifts. Future work can explore advanced transport-based techniques and mitigation strategies to address the negative effects of modeling error on classification performance.

References:

[1] Pilario, K.E.S. and Cao, Y. "Canonical Variate Dissimilarity Analysis for Process Incipient Fault Detection," IEEE Transactions on Industrial Informatics, 2018. DOI: 10.1109/TII.2018.2810822

[2] Pilario, K.E.S., Cao, Y., and Shafiee, M. "Mixed Kernel Canonical Variate Dissimilarity Analysis for Incipient Fault Monitoring in Nonlinear Dynamic Processes". Comput. Chem. Eng. 123, pp. 143-154, 2019. DOI: 10.1016/j.compchemeng.2018.12.027

.