# Progress Report

## Research Project Title- Domain Adaptation for Equipment Fault Detection

Supervisor - Professor Sparsh Mittal, ECE Department, IIT Roorkee [Google-Scholar-Profile] &
Dr. R. Sai Chandra Teja [Google Scholar Profile]

**Week 1**---------------------------------------------
Signal data→ time series, frequency variation
Traditional practice is to convert the signal to image→ representations-> image form— `MFCC, Spectrogram,

1. Translate time series data into representations→ Learn the features from representations and evaluate—>what types of representations are easy to learn
2. Basic representation→ architecture-module-new which will help extract / learn from basic representation 66k truncated to 66 (427x**13**x129, 427x128x129)--> Convolution- spatial dimensions, Involution- channels
3. Dual paths→ one path on spatial and other paths on channels effective learning→

4. Domain adaptation→ steel defect images model train, wood defect images, dual path → sound files path, rattling→ vibration
5. Multi modal→ learning both sound and train

Summary→
Datasets 2,3-> which have the same label features-> rattling, fixed, sound data, vibration data, thermal data
Cracks-> sound, vibration, thermal

Paper referred for Domain Adaptation
https://ieeexplore.ieee.org/document/8794530

-x-x-x-x-x-x-x-x-x

**Week 2**-------------------------------------------------

- Worked on Condition Monitoring for Bearing Faults detection of Rotatory Machines.

- Started Working on a dataset provided by Teja Sir:

Mendeley Data

## Sounds of valves in heating systems for classification and condition monitoring

Published: 21 September 2022 | Version 2 | DOI: 10.17632/y6fkrybb32.2
Contributors: Primož Potočnik, Lučka Vodopivec, Egon Susič

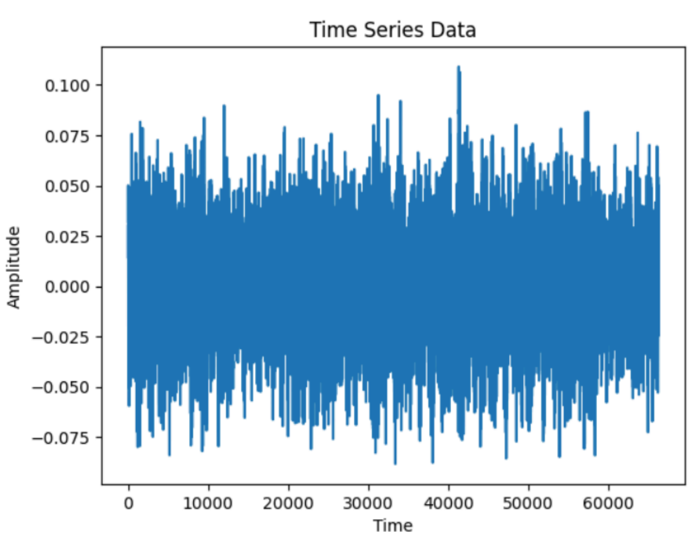
### Description

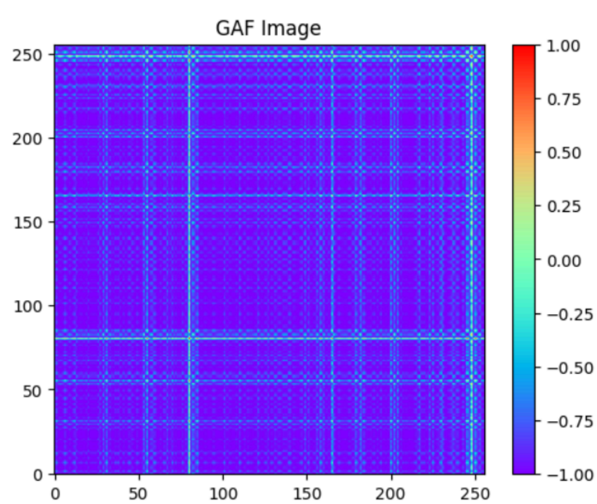
Dataset contains 427 sounds of valves installed and operating in district heating systems.

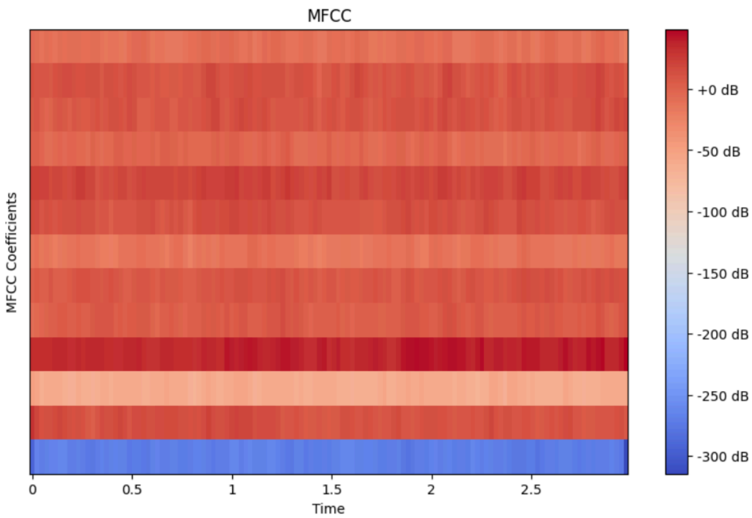
Tables in Excel and Matlab format contain categories for each sound as follows:
1) FlowNoise, 2) Rattling, 3) Whistling, 4) Cavitation.

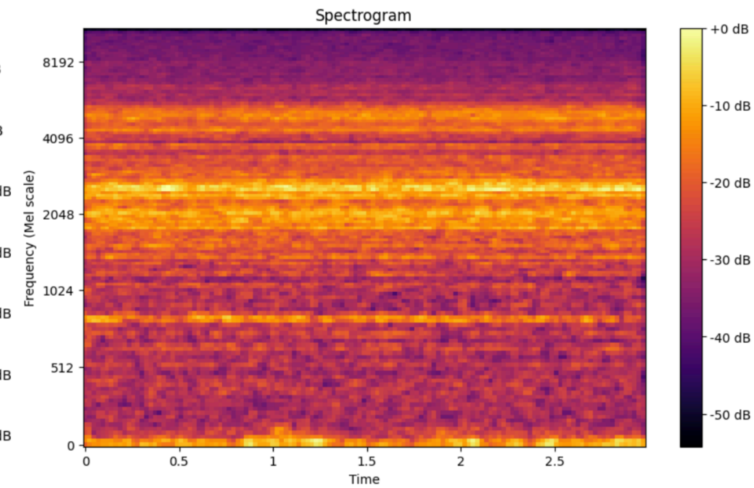
- 

- Converted Acoustic time series data into image data Using three methods:
  - Gramian Angular Field
  - Mel-frequency cepstral coefficients (MFCCs)
  - Mel-spectrogram

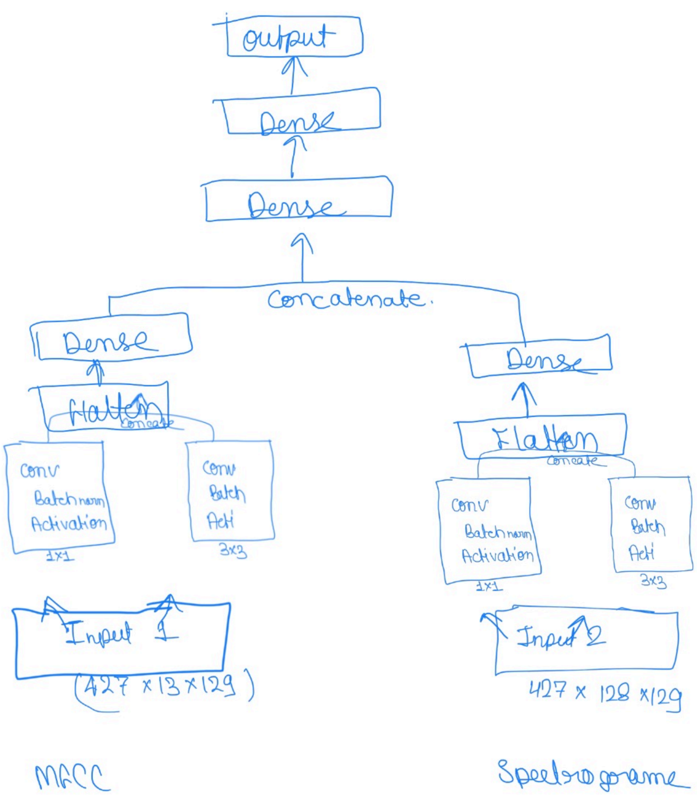





- Using these Images as a feature vector, I tried to apply a classification model to this dataset.
- I made an inception module like CNN architecture for classification using MFCC and Spectrograme as two input features.



- Got an accuracy of **98%** on this dataset with 427 samples with a training time of approx **1 minute** using a 2GB GPU on the local device.

The idea behind the above experiment is to test whether we can work on Multi-Model Methods where we can fuse different modes of Condition Monitoring for classification.

## Week 3---------------------------------------------

**Focused on these areas:**

- **Domain Adaptation** and **Attention-based** effective feature extraction from different acoustic, vibration, and motor current's time series data.
- Effective representation of different Modal data into a single mode eg, image.
- Choosing an effective DNN Architecture for classification.

➔ Started Working On Domain Adaptation:

- ◆ Applying a classifier-constrained Domain Adaptation paper on two different datasets.

**Paper:** https://ieeexplore.ieee.org/document/8794530

# Classifier-Constrained Deep Adversarial Domain Adaptation for Cross-Domain Semisupervised Classification in Remote Sensing Images

Wenxiu Teng, *Student Member, IEEE*, Ni Wang, Huihui Shi, Yuchan Liu, and Jing Wang

***Abstract*—This letter presents a classifier-constrained deep adversarial domain adaptation (CDADA) method for cross-domain semisupervised classification in remote sensing (RS) images. A deep convolutional neural network (DCNN) is used to build feature representations to describe the semantic content of scenes before the adaptation process. Then, adversarial domain adaptation is used to align the feature distribution of the source and the target. Specifically, two different land-cover classifiers are used as a discriminator to consider land-cover decision boundaries between classes and increase their distance to separate them from the original land-cover class boundaries. The generator then creates robust transferable features far from the original land-cover class boundaries under the classifier constraint. The experimental results of six scenarios built from three benchmark RS scene data sets (AID, Merced, and RSI-CB data sets) are reported and discussed.**

***Index Terms*—Cross-domain classification, deep convolutional neural networks (DCNNs), domain adaptation (DA), generative adversarial networks (GANs), remote sensing (RS).**

used as a generic image representation. Therefore, almost all these works focus on the methods of acquiring strong image representations by transferring a pretrained DCNN to their tasks.

However, RS images are inevitably affected by various human and natural factors, such as sensors, camera perspectives, geographic locations, seasons, and weather conditions. Therefore, when the source data set is far from the target data set (also known as data shift), transferring strategies for pretrained DCNNs is likely to yield unsatisfactory results. Domain adaptation (DA) can be helpful to solve this problem [9].

DA, as it pertains to transfer learning (TL), is the process of adapting one or more source domains for transferring information to improve the performance of a target learner [10]. The DA method for deep features is called deep DA, and it can generally be categorized into discrepancy-based or adversarial-

**CDADA Network Architecture:**

[https://drive.google.com/file/d/1_27dPiROPs3lBJE6mkMw-WcT4wERWkYe/view?usp=sharing](https://drive.google.com/file/d/1_27dPiROPs3lBJE6mkMw-WcT4wERWkYe/view?usp=sharing)

## Week 4-----------------------------------------------

### Dataset Working With:

| Paderborn Bearing Dataset (2016) | |
|---|---|
| Dataset Paper: https://mb.uni-paderborn.de/fileadmin-mb/kat/PDF/Veroeffentlichungen/20160703_PHME16_CM_bearing.pdf<br><br>Dataset: http://groups.uni-paderborn.de/kat/BearingDataCenter/ | |
| Modes | Classes Used |
| ● Acoustic<br>● Vibration<br>● Motor Current<br>● Temperature | ● Normal Bearing<br>● Outer Bearing Faults<br>● Inner Bearing Faults |
| **Mandeley Bearing Dataset (2023)** | |
| Dataset Paper: https://www.sciencedirect.com/science/article/pii/S2352340923001671<br><br>Dataset: https://data.mendeley.com/datasets/ztmf3m7h5x/4 | |
| Modes | Classes Used |
| ● Acoustic<br>● Vibration<br>● Motor Current<br>● Temperature | ● Normal Bearing<br>● Outer Bearing Faults<br>● Inner Bearing Faults |

### Paderborn Dataset:

In summary, the main characteristic of the data set are:

- Synchronously measured motor currents and vibration signals with high resolution and sampling rate of 26 damaged bearing states and 6 undamaged (healthy) states for reference.
- Supportive measurement of speed, torque, radial load, and temperature.
- Four different operating conditions (Table 6).
- 20 measurements of 4 seconds each for each setting, saved as a matlab file with a name consisting of the code of the operating condition and the four-digit bearing code (e.g. N15_M07_F10_KA01_1.mat).
- Systematic description of the bearing damage by uniform fact sheets (according to the categorization in section 2 - Categorization of Bearing Damage).

Table 10. Categorization of data sets for healthy bearings and bearings with real damages.

| Healthy (Class 1) | Outer ring damage (Class 2) | Inner ring damage (Class 3) |
|---|---|---|
| K001 | KA04 | KI04 |
| K002 | KA15 | KI14 |
| K003 | KA16 | KI16 |
| K004 | KA22 | KI18 |
| K005 | KA30 | KI21 |

## Mandeley Dataset:

---- Description of vibration file format ----
Vibration data file contains five columns namely ‘Time Stamp’, ‘x_direction_housing_A’, ‘y_direction_housing_A’, ‘x_direction_housing_B’, and ‘y_direction_housing_B’. The unit of the vibration is ‘gravitational constant (g)’.

aaaaNm_bbbb_cccc.mat : This file includes vibration data of the condition of "bbbb" with "cccc" of severity and "aaaa" load condition.

---- Description of acoustic file format ----
Acoustic data file contains two columns namely ‘Time Stamp’, and ‘values’. The unit of the acoustic is ‘Pascal (Pa)’.

aaaaNm_bbbb_cccc.mat : This file includes acoustic data of the condition of "bbbb" with "cccc" of severity and "aaaa" load condition.

---- Description of temperature, and motor current file format ----
Temperature and motor current data file contain six columns namely ‘Time Stamp’, ‘Temperature_housing_A’, ‘Temperature_housing_B’, ‘U-phase’, ‘V-phase’, and ‘W-phase’. The unit of the temperature and motor current are ‘Celsius (°C), and ‘ampere (A)’, respectively.

aaaaNm_bbbb_cccc.tdms : This file includes temperature and motor current data of the condition of "bbbb" with "cccc" of severity and "aaaa" load condition.

- ➔ Started applying CDADA Domain adaptation, applying code to train on the Mendeley dataset to build an effective feature extractor which will be tested on the Paderborn dataset.

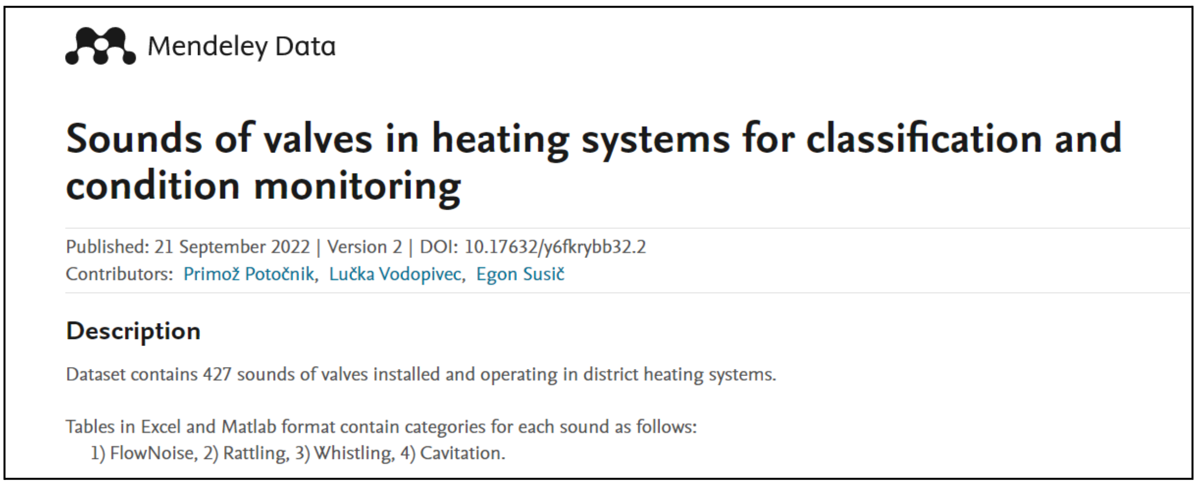
**Week 5**---------------------------------------------

## **Problems Worked Upon**:

Problem-1

- Tried different ways towards multi-modality

Began working on the dataset given by Teja Sir:



Mendeley Data

**Sounds of valves in heating systems for classification and condition monitoring**

Published: 21 September 2022 | Version 2 | DOI: 10.17632/y6fkrybb32.2
Contributors: Primož Potočnik, Lučka Vodopivec, Egon Susič

**Description**

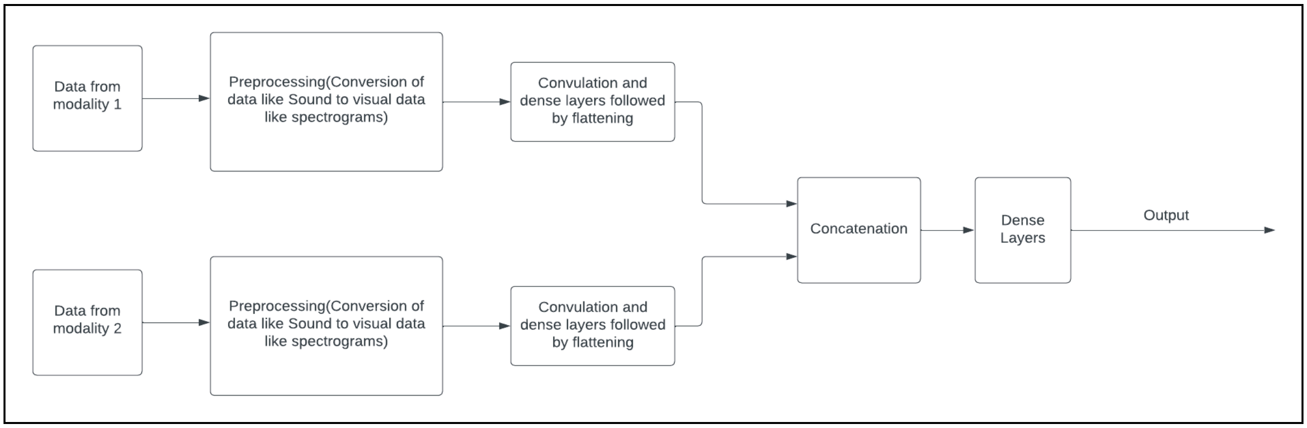
Dataset contains 427 sounds of valves installed and operating in district heating systems.

Tables in Excel and Matlab format contain categories for each sound as follows:
1) FlowNoise, 2) Rattling, 3) Whistling, 4) Cavitation.

Link: https://data.mendeley.com/datasets/y6fkrybb32

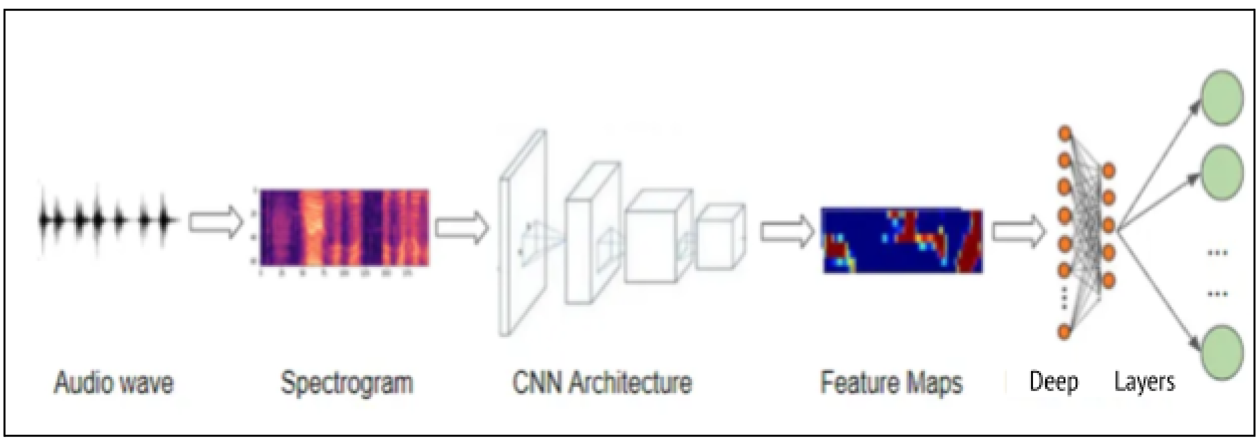
Aim- To find different kinds of representations of sounds and similarly for vibration data, so that these representations can be used for multimodal representations and computing.

Approach- Representing time series data like acoustic data and similar vibration data into different visual representations so that they can be convoluted and then concatenated after which they will be passed through dense layers.




But firstly, a single modality audio deep learning sound classification was performed to gauge the scope of improvement using multimodality.

For the audio classification: audio was firstly represented by a mel-spectrogram which was then passed through convolution layers for feature extraction followed by deep layers.




Result - An accuracy of 96% was obtained using mel spectrogram which indicated that the data set chosen was too simple for applying any kind of multimodality. Hence Teja Sir instructed us to work individually in different directions, giving us different areas to work on such as:-

- Worked towards domain adaptation i.e. feature alignment of source and target domains. Mr. Teja Sir also shared a research paper for starting reference. Link: https://ieeexplore.ieee.org/document/8794530

## Week 6-------------------------------------------

- Researched new and effective ways of representing time series data whose training can be less computationally expensive.

**Problem**: Exploring new time series-based representations.

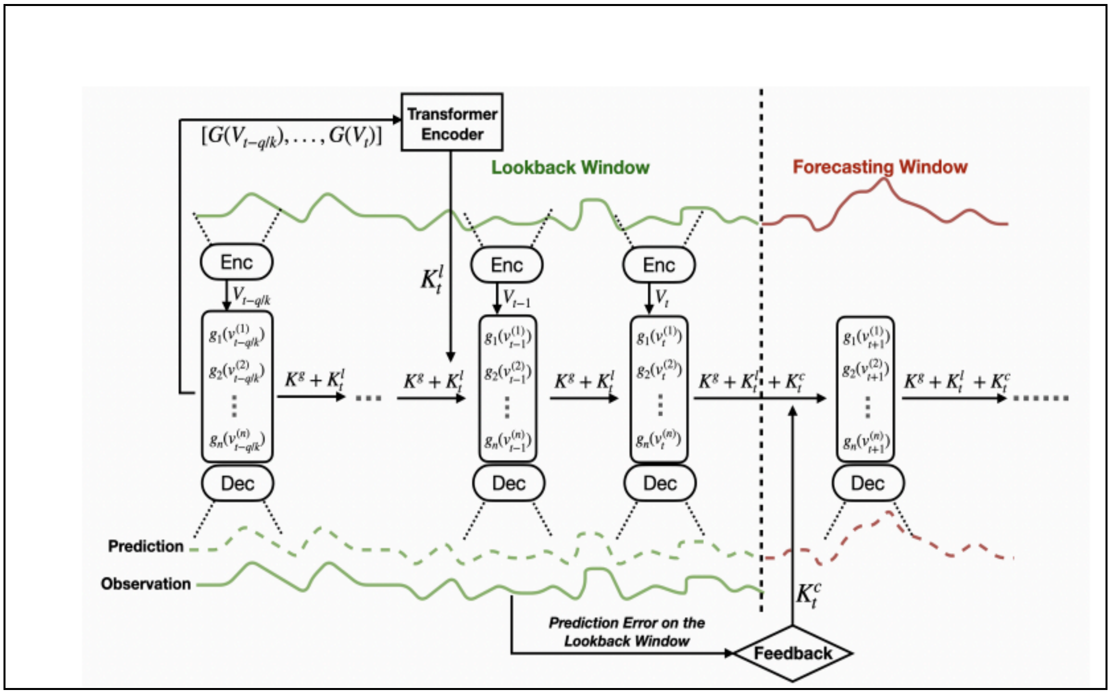
- Came across two recent research papers which can prove to be useful.

**Paper-1** Koopman Neural Forecaster For Time Series With Temporal Distribution Shifts

Link: https://paperswithcode.com/paper/koopman-neural-forecaster-for-time-series

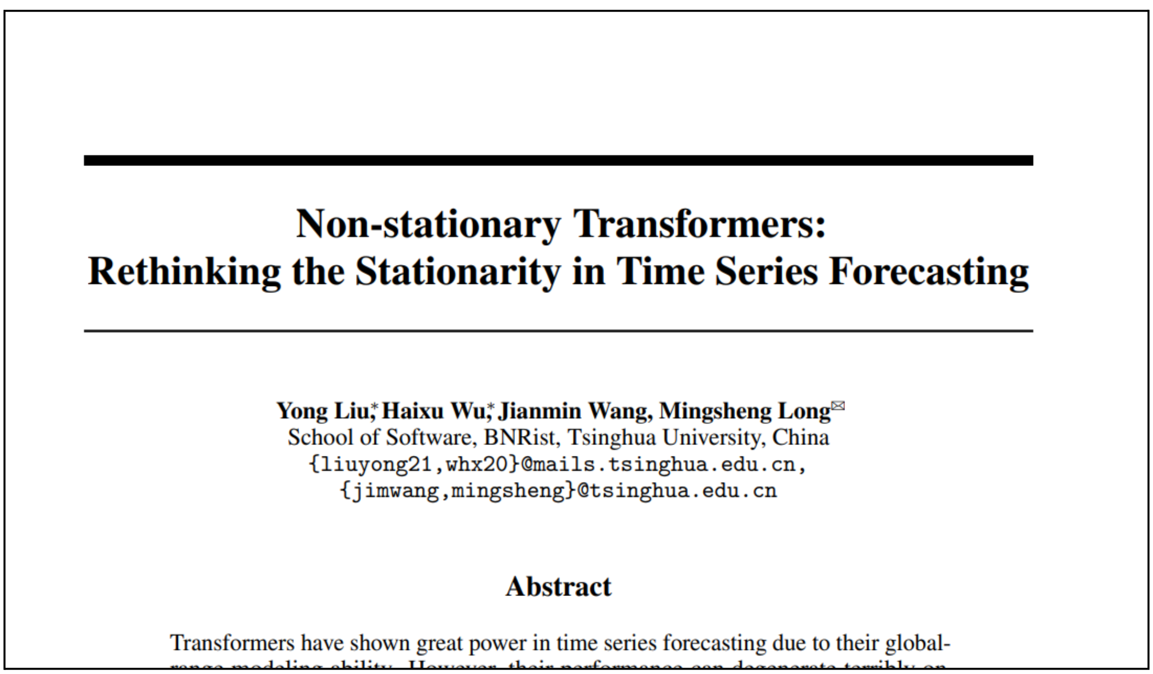
> Published as a conference paper at ICLR 2023
>
> KOOPMAN NEURAL FORECASTER FOR TIME SERIES WITH TEMPORAL DISTRIBUTION SHIFTS
>
> Rui Wang* UC San Diego · Yihe Dong Google Cloud AI · Sercan Ö. Arik Google Cloud AI · Rose Yu UC San Diego
>
> ABSTRACT
>
> Temporal distributional shifts, with underlying dynamics changing over time, frequently occur in real-world time series, and pose a fundamental challenge for deep neural networks (DNNs). In this paper, we propose a novel deep sequence model based on the Koopman theory for time series forecasting: Koopman Neural

Summary: A deep neural network was proposed based on the Koopman Theory of Time Series forecasting called Koopman Neural Forecaster (KNF) that leverages DNNs to learn the linear Koopman space. KNF imposes biases for improved robustness against distributional shifts, employing both a global operator to learn shared characteristics, and a local operator to capture changing dynamics, as well as a specially designed feedback loop to continuously update the learned operators over time for rapidly varying behaviors. It gave SOTA performances on the tested datasets (M4, Cryptocurrency, NBA players) which tend to suffer from distribution shifts.




**Paper-2** Non-stationary Transformers: Rethinking the Stationarity in Time Series Forecasting

Link: [(PDF) Non-stationary Transformers: Rethinking the Stationarity in Time Series Forecasting (researchgate.net)]



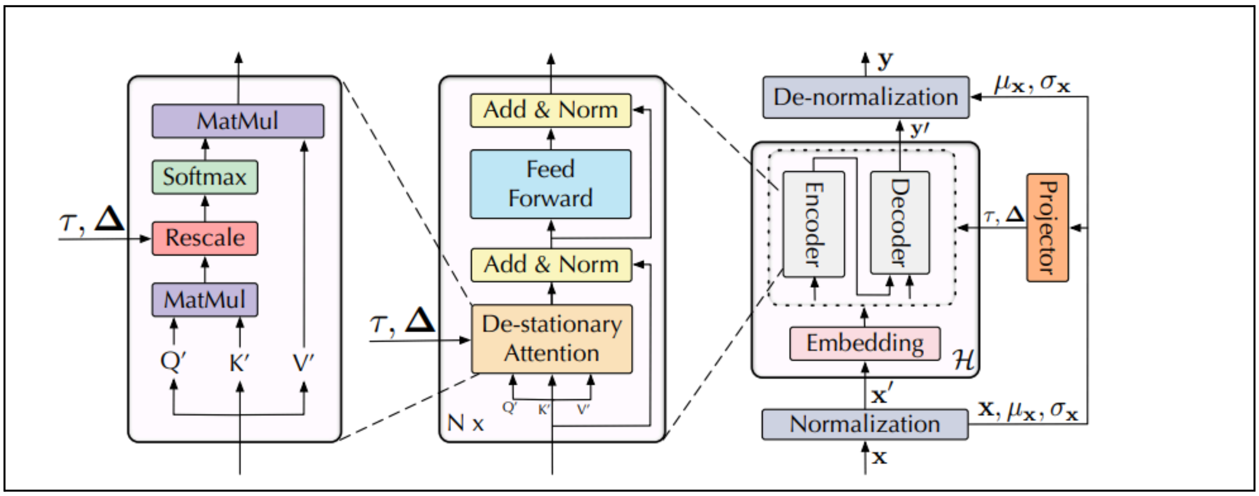
# Non-stationary Transformers: Rethinking the Stationarity in Time Series Forecasting

**Yong Liu*, Haixu Wu*, Jianmin Wang, Mingsheng Long**[✉]
School of Software, BNRist, Tsinghua University, China
{liuyong21,whx20}@mails.tsinghua.edu.cn,
{jimwang,mingsheng}@tsinghua.edu.cn

## Abstract

Transformers have shown great power in time series forecasting due to their global-

Summary: The paper argues that stationarization of non-stationary data deprives important information from the data. Hence non-stationary Transformers are deduced as a generic framework with two interdependent modules: Series Stationarization and De-stationary Attention series stationarization unifies the statistics of each input and converts the output with restored statistics for better predictability and for addressing over-stationarization, De-stationary Attention is devised to recover the intrinsic non-stationary information into temporal dependencies by approximating distinguishable attentions learned from un-stationarized series.



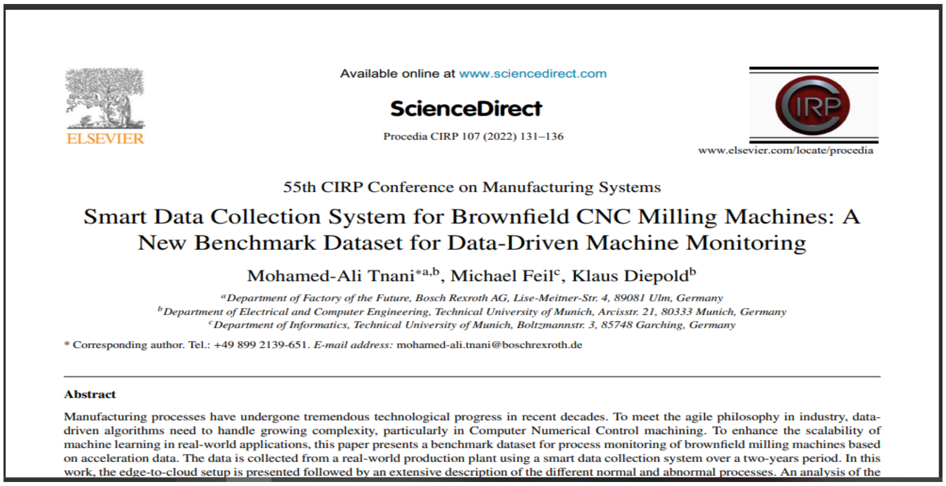

Conclusion and Future Work Direction

Sir instructed that these methods used in the papers can prove to be useful for non-stationary data, not from a forecasting point of view but other developments such as in the field of classifications, etc. The idea of extracting global and local Koopman features can be paired with the idea of non-stationary transformers to gain more insights into the data and can be a direction in which there is scope for further exploration.

**Week 7**---------------------------------------------

➔ Worked on Machine Monitoring

- [ ] Started Working on the research paper provided by Teja Sir



Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 107 (2022) 131–136

www.elsevier.com/locate/procedia

55th CIRP Conference on Manufacturing Systems

Smart Data Collection System for Brownfield CNC Milling Machines: A New Benchmark Dataset for Data-Driven Machine Monitoring

Mohamed-Ali Tnani[*a,b], Michael Feil[c], Klaus Diepold[b]

[a]Department of Factory of the Future, Bosch Rexroth AG, Lise-Meitner-Str. 4, 89081 Ulm, Germany
[b]Department of Electrical and Computer Engineering, Technical University of Munich, Arcisstr. 21, 80333 Munich, Germany
[c]Department of Informatics, Technical University of Munich, Boltzmannstr. 3, 85748 Garching, Germany

* Corresponding author. Tel.: +49 899 2139-651. E-mail address: mohamed-ali.tnani@boschrexroth.de

**Abstract**

Manufacturing processes have undergone tremendous technological progress in recent decades. To meet the agile philosophy in industry, data-driven algorithms need to handle growing complexity, particularly in Computer Numerical Control machining. To enhance the scalability of machine learning in real-world applications, this paper presents a benchmark dataset for process monitoring of brownfield milling machines based on acceleration data. The data is collected from a real-world production plant using a smart data collection system over a two-years period. In this work, the edge-to-cloud setup is presented followed by an extensive description of the different normal and abnormal processes. An analysis of the

- [ ] Paper link: [https://www.sciencedirect.com/science/article/pii/S2212827122002384](https://www.sciencedirect.com/science/article/pii/S2212827122002384)
- [ ] Edge-to-Cloud Stack architecture is implemented to get the vibrational data of three different Brownfield CNC machines in which 1 different type of tool operation was operated for 6 different time frames each lasting for 6 months. In this, both frequency domain and time domain series are studied and features of the machines are mapped to 2D features using PCA and then the machine was classified as normal vibrating and anomaly vibrating.
- [ ] The Accelerometer sensor was attached to the rear end of the tool to obtain the acceleration of the machine in the X, Y, and Z axis.

*2.2.1. Edge stack*

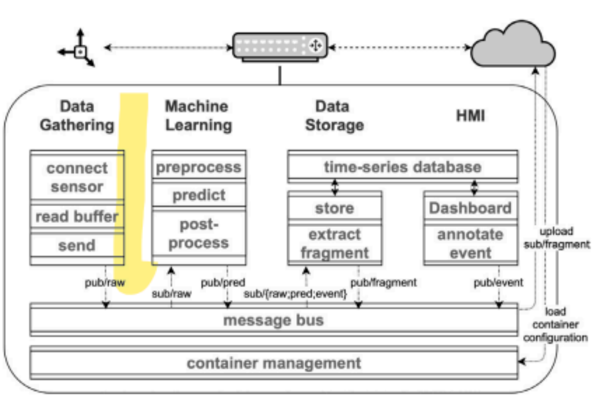
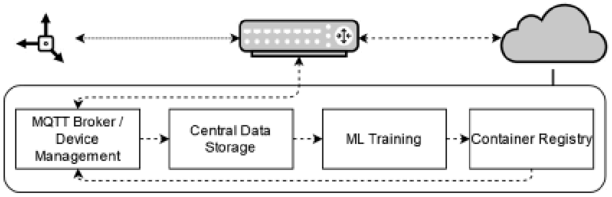



**Fig. 2:** Concept and interaction of containers in the edge stack.

required in the cloud. Annotated vibration fragments from edge

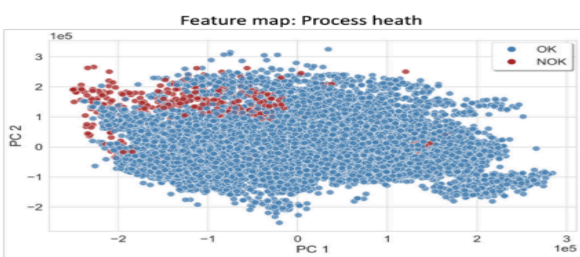



- [ ] **Fig. 3:** Services and building blocks in cloud stack.

**Table 1** Tools operations collected from M01, M02 and M03.

| Tool operation | Description | speed [$Hz$] | feed [$mm\ s^{-1}$] | duration [$s$] |
|---|---|---|---|---|
| **OP00** | Step Drill | 250 | ≈ 100 | ≈ 132 |
| **OP01** | Step Drill | 250 | ≈ 100 | ≈ 29 |
| **OP02** | Drill | 200 | ≈ 50 | ≈ 42 |
| **OP03** | Step Drill | 250 | ≈ 330 | ≈ 77 |
| **OP04** | Step Drill | 250 | ≈ 100 | ≈ 64 |
| **OP05** | Step Drill | 200 | ≈ 50 | ≈ 18 |
| **OP06** | Step Drill | 250 | ≈ 50 | ≈ 91 |
| **OP07** | Step Drill | 200 | ≈ 50 | ≈ 24 |
| **OP08** | Step Drill | 250 | ≈ 50 | ≈ 37 |
| **OP09** | Straight Flute | 250 | ≈ 50 | ≈ 102 |
| **OP10** | Step Drill | 250 | ≈ 50 | ≈ 45 |
| **OP11** | Step Drill | 250 | ≈ 50 | ≈ 59 |
| **OP12** | Step Drill | 250 | ≈ 50 | ≈ 46 |
| **OP13** | T-Slot Cutter | 75 | ≈ 25 | ≈ 32 |
| **OP14** | Step Drill | 250 | ≈ 100 | ≈ 34 |

Figure 7.




- [ ] **Fig. 7:** Feature maps of the complete dataset reduced into 2D using principle component analysis [17].

- [ ] Afterward, studied if the machine that was labeled “ok” before get classified as “not ok” in a different time frame later.
- [ ] The sampling rate is chosen as the low integer multiple of spindle speed.
- [ ] Dataset link: [GitHub - boschresearch/CNC_Machining: data set for process monitoring on CNC machines]()

- Tried getting the dataset used in the research paper but it was not available publicly.
- Moved on to find various ways to represent vibration signals in vibration images which can be used to detect faults in rotating machines
- This can be done using time domain, frequency domain, and time-frequency domain vibration signals.
- In time domain representation, the most accurate and efficient representation of time series-based signal data to images is found to be a 2-D grayscale image from rectified vibration signal as VIR technique along with kNN, ANN, Bayes net as feature learning and classification method, and Multiple channels based vibration images fusion as VIR technique along with SVM as feature learning and classification method.

| | | | | | |
|---|---|---|---|---|---|
| [48] | Grayscale image + LBP | RF, *k*-NN, naive Bayes, Bayes net, ANN | Bearing | $f$ = 24 kHz<br>classes = 3 | 100 |
| [49] | Grayscale image + DNS | SVM | Motor faults | classes = 88 | 100 |

- In frequency domain representation, the most accurate and efficient techniques are-

| Ref | VIR Technique | ML Technique | RM Component | Dataset | Best Test Accuracies (%) |
|---|---|---|---|---|---|
| [69] | The FFT spectrum image | Minimum distance criterion based on the Eigen images | Bearing | classes = 4<br>loads = 4<br>CWRU BDC Link<br>https://engineering.case.edu/bearingdatacenter/download-data-file (accessed on 18 November 2022) | 100 |
| [73] | The FFT spectrum image based on segmented time series signal | ANN | Bearing | $f$ = 12 kHz<br>classes = 4<br>CWRU BDC Link<br>https://engineering.case.edu/bearingdatacenter/download-data-file<br>(accessed on 18 November 2022) | 100 |

- In the time-frequency domain, the most efficient techniques are-

| Ref | VIR Technique | ML Technique | RM Component | Dataset | Best Test Accuracies (%) |
|---|---|---|---|---|---|
| [84] | The STFT spectrogram image | CNN using the scaled exponential linear unit (SELU) and hierarchical regularization | Bearing | $f$ = 12 kHz<br>classes = 10<br>CWRU BDC Link<br>https://engineering.case.edu/bearingdatacenter/download-data-file<br>(accessed on 18 November 2022)<br>and<br>$f$ = 12.8 kHz<br>Classes = 4<br>Yanshan University, Qinhuangdao, Hebei 066004, P. R. China. Bearings dataset collected from a mechanical vibration simulator | 100<br>97.81 |
| [88] | The STFT spectrogram image | 2DCNN | Bearing and Tool wear | $f$ = 12 kHz<br>classes = 4<br>CWRU BDC Link<br>https://engineering.case.edu/bearingdatacenter/download-data-file<br>(accessed on 18 November 2022)<br>and<br>$f$ = 100 kHz<br>classes = 2 | 100<br>100 |
| [91] | The Grad-CAM activation maps for STFT-based images | CNN<br>NN<br>ANFIS | Bearing | $f$ = 12 kHz<br>classes = 4<br>CWRU BDC Link<br>https://engineering.case.edu/bearingdatacenter/download-data-file<br>(accessed on 18 November 2022) | 100<br>100<br>96.9 |

- Read and understand the workings of all these techniques and the mathematics involved in them.
- More research can be done in using different classification and feature learning methods with the improvement of AI. Models like VGG, MobileNet, EfficientNet, ViT, etc. can be tried with the above-mentioned techniques and they might give better results than the existing results.
- Research Paper: https://www.mdpi.com/2075-1702/10/12/1113
- Learned about Inter-Domain Style Adaptation and Intra domain gradual self-training.

## Week 8-----------------------------------------------

Started Working on two datasets: Mendely and Paderborn Motor Vibration dataset.

### Mendeley:

- Converted Current data→ Image data Using MFCC and GAF → Classification using CNN → Not a satisfactory result. (56% Accuracy).
- Converted Vibration data → Image Data using MFCC → Classification using CNN → 100% Accuracy Achieved.
- Architecture and Accuracy For Mendely Dataset:

```
model = keras.Sequential()
model.add(layers.Conv2D(12, (3, 3), activation='relu', input_shape=(13,126, 1)))

model.add(layers.Conv2D(24, (3, 3), activation='relu'))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.Flatten())
model.add(layers.Dense(26, activation='relu'))
model.add(layers.Dense(3, activation='softmax'))
```

```
model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

```
In [48]: # Evaluate the model on the test set
test_loss, test_acc = model.evaluate(X_test, y_test)
print('Test accuracy:', test_acc)

5/5 [==============================] - 0s 24ms/step - loss: 3.6680e-09 - accuracy: 1.0000
Test accuracy: 1.0
```

### Paderborn:

- Similarly applied this model to the Paderborn dataset:
    - Paderborn Dataset has 4 different working Conditions leading to 4 different vibrational signatures.

**Table 6**
Four working conditions of PU dataset.

| Setting name | Rotating speed (rpm) | Load torque (Nm) | Radial force (N) |
|---|---|---|---|
| N15_M07_F10 | 1500 | 0.7 | 1000 |
| N09_M07_F10 | 900 | 0.7 | 1000 |
| N15_M01_F10 | 1500 | 0.1 | 1000 |
| N15_M07_F04 | 1500 | 0.7 | 400 |

    - For my testing purposes, I started working on two different Load conditions
        - A = N15_M07_F10 Working Condition

    - B = N15_M01_F10
  - For *Load Condition A*, I got a maximum classification accuracy of 97.91%:

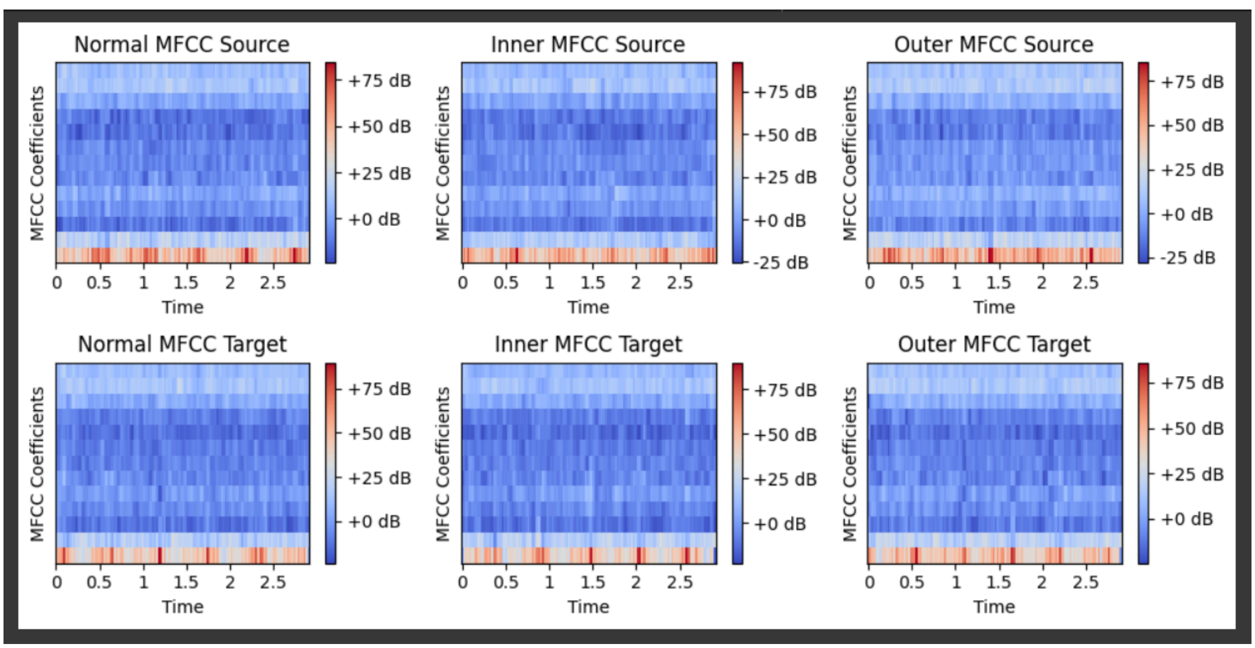
```
model = keras.Sequential()
model.add(layers.Conv2D(12, (3, 3), activation='relu', input_shape=(13,126, 1
model.add(layers.Conv2D(24, (3, 3), activation='relu'))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.Flatten())
model.add(layers.Dense(20, activation='relu'))
model.add(layers.Dense(3, activation='softmax'))
```

```
3/3 [==============================] - 0s 40ms/step - loss: 0.1251 - accurac
y: 0.9792
Test accuracy: 0.9791666865348816
```

**Domain Adaptation:**

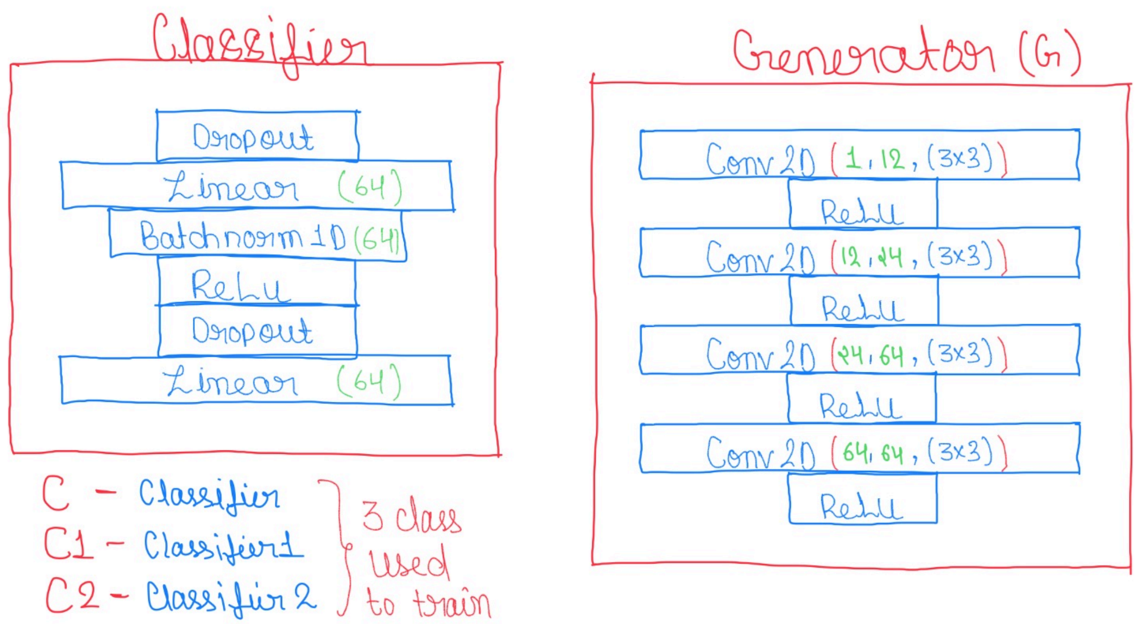
As I got 100% and 97.91% accuracy on individual datasets, I tried to modify CDADA architecture and tried to predict them in an unsupervised way.

**Source Domain:** A → N15_M07_F10, **Target Domain:** B → N15_M01_F10 of Paderborn

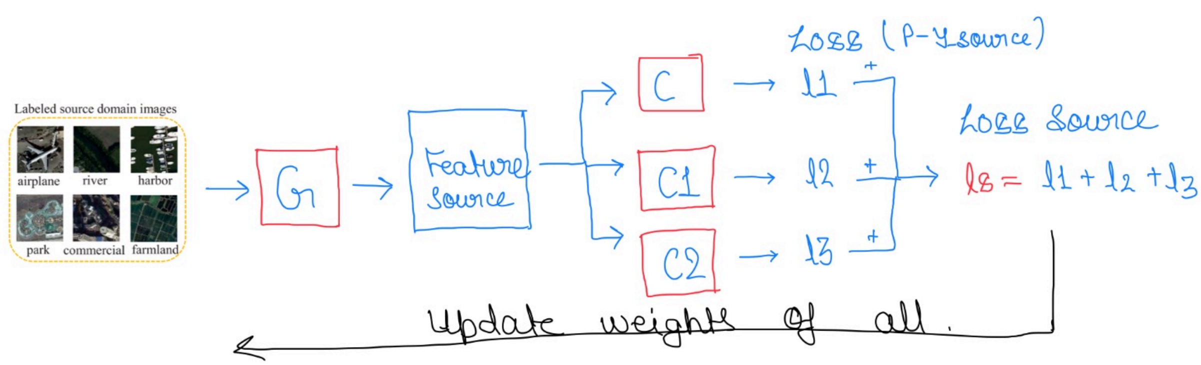


MFCC images of Normal bearing and Inner and Outer bearing fault of two working conditions of the Paderborn dataset.

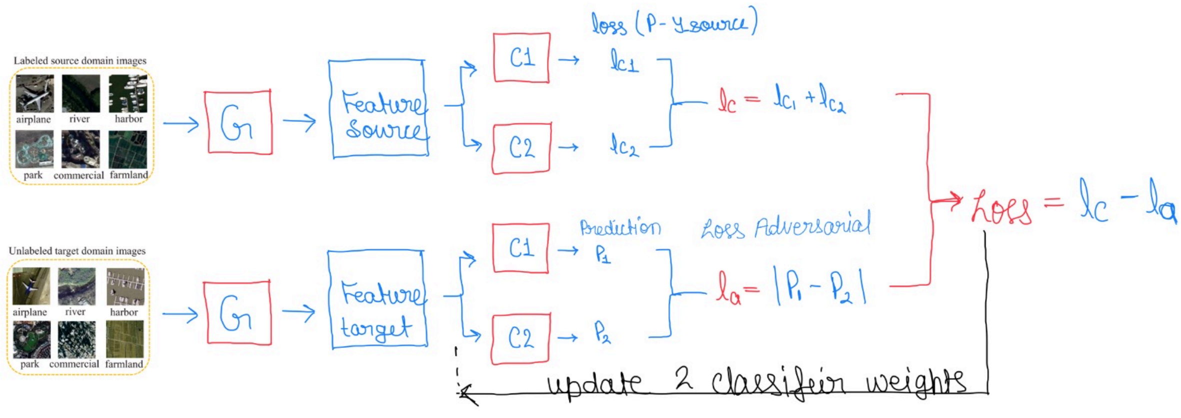
- Original CDADA architecture used pre-trained VGGnet for image feature generation and VGG classifier for Classification as it tried to adapt the domain for Natural image classification.
- I, on the other hand, had a very small size image and very few features, so I replaced Pre-Trained VGGnet with my own feature extractor model, which individually gave 100% and 97.91% accuracy.
- For classification, I created a shallower model than the VGG Classifier, and both of these replacements worked better than the original CDADA model on my task.




**Step 1: Training of C,C1,C2,G on Labeled Data**

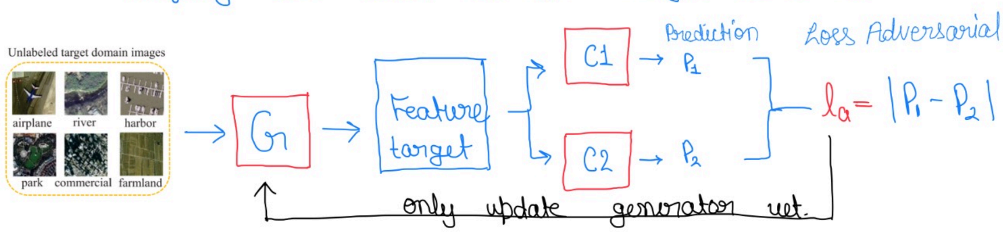


## Step 2: Keeping Generator weight constant, training classifier for Adversarial task




## Step 3: Keeping Classifier wt constant, Train Generator for Adversarial task.

with this updated weight now train only generator for few times keeping all other classifier weight constant.




- First, I tried Domain Adaptation for different Load load conditions of the Paderborn dataset.

  **Labeled data** A → N15_M07_F10 Working Condition Paderborn data

  **Unlabeled data** B → N15_M01_F10 Working Condition Paderborn data

```
[431] # Call the function // lr = 0.02  // 64 10 epoch without alignment 20 updata // From paderborn_s to paderborn 20updata smaller classifier 64
train_model(Generator, Classifier, Classifier1, Classifier2, src_data_loader, tgt_data_loader, max_epoch, opt_generator, opt_classifier, opt_classifier1, opt_classifier2, updata)

Train Epoch:1 Adversarial Loss: 0.083906
Source: Mendeley to Target: Paderborn  Accuracy: 52/80 (65.00%) Max Accuracy: 52/80 (65.00%)
Train Epoch:2 Adversarial Loss: 0.111220
Source: Mendeley to Target: Paderborn  Accuracy: 36/80 (45.00%) Max Accuracy: 36/80 (45.00%)
Train Epoch:3 Adversarial Loss: 0.046237
Source: Mendeley to Target: Paderborn  Accuracy: 46/80 (57.50%) Max Accuracy: 46/80 (57.50%)
Train Epoch:4 Adversarial Loss: 0.054968
Source: Mendeley to Target: Paderborn  Accuracy: 62/80 (77.50%) Max Accuracy: 62/80 (77.50%)
Train Epoch:5 Adversarial Loss: 0.069740
Source: Mendeley to Target: Paderborn  Accuracy: 62/80 (77.50%) Max Accuracy: 62/80 (77.50%)
Train Epoch:6 Adversarial Loss: 0.041698
Source: Mendeley to Target: Paderborn  Accuracy: 67/80 (83.75%) Max Accuracy: 67/80 (83.75%)
Train Epoch:7 Adversarial Loss: 0.051091
Source: Mendeley to Target: Paderborn  Accuracy: 66/80 (82.50%) Max Accuracy: 66/80 (82.50%)
Train Epoch:8 Adversarial Loss: 0.042893
Source: Mendeley to Target: Paderborn  Accuracy: 76/80 (95.00%) Max Accuracy: 76/80 (95.00%)
Train Epoch:9 Adversarial Loss: 0.033663
Source: Mendeley to Target: Paderborn  Accuracy: 63/80 (78.75%) Max Accuracy: 63/80 (78.75%)
Train Epoch:10 Adversarial Loss: 0.047149
Source: Mendeley to Target: Paderborn  Accuracy: 77/80 (96.25%) Max Accuracy: 77/80 (96.25%)
```

- Got a maximum accuracy of 96.25% max and >85% in multiple runs in an unsupervised way.
- Supervised accuracy on this data is 97.91% using CNN architecture.

**Domain Adaptation from Mandeley to Paderborn:**

Using Mendeley data as labeled data and Paderborn as unlabeled data, I wasn't able to achieve an accuracy of more than 48.75%.

```
# Call the function // lr = 0.02  // normalised // 64 32-batch 10 epoch  // mendely to paderborn 20updata smaller classifier 64
train_model(Generator, Classifier, Classifier1, Classifier2, src_m_data_loader, tgt_data_loader, max_epoch, opt_generator, opt_classifier, opt_classifier1, opt_classifier2, updata)

Train Epoch:1 Adversarial Loss: 0.041039
Source: Mendeley to Target: Paderborn  Accuracy: 29/80 (36.25%) Max Accuracy: 29/80 (36.25%)
Train Epoch:2 Adversarial Loss: 0.005958
Source: Mendeley to Target: Paderborn  Accuracy: 20/80 (25.00%) Max Accuracy: 20/80 (25.00%)
Train Epoch:3 Adversarial Loss: 0.004609
Source: Mendeley to Target: Paderborn  Accuracy: 29/80 (36.25%) Max Accuracy: 29/80 (36.25%)
Train Epoch:4 Adversarial Loss: 0.016029
Source: Mendeley to Target: Paderborn  Accuracy: 47/80 (58.75%) Max Accuracy: 47/80 (58.75%)
Train Epoch:5 Adversarial Loss: 0.005515
Source: Mendeley to Target: Paderborn  Accuracy: 39/80 (48.75%) Max Accuracy: 39/80 (48.75%)
Train Epoch:6 Adversarial Loss: 0.003395
Source: Mendeley to Target: Paderborn  Accuracy: 39/80 (48.75%) Max Accuracy: 39/80 (48.75%)
Train Epoch:7 Adversarial Loss: 0.002433
Source: Mendeley to Target: Paderborn  Accuracy: 39/80 (48.75%) Max Accuracy: 39/80 (48.75%)
Train Epoch:8 Adversarial Loss: 0.007676
Source: Mendeley to Target: Paderborn  Accuracy: 39/80 (48.75%) Max Accuracy: 39/80 (48.75%)
Train Epoch:9 Adversarial Loss: 0.001547
Source: Mendeley to Target: Paderborn  Accuracy: 39/80 (48.75%) Max Accuracy: 39/80 (48.75%)
Train Epoch:10 Adversarial Loss: 0.005450
Source: Mendeley to Target: Paderborn  Accuracy: 39/80 (48.75%) Max Accuracy: 39/80 (48.75%)
```

**Currently Working on:**

- Scope of improvement in better Image Representation Techniques.
- Noise reduction methods in vibrational data.
- Methods to maximize overlapping feature extraction for better domain adaptation.