EARLY DIAGNOSIS OF BREAKDOWN THROUGH TRANSFER LEARNING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

By
Seren Özbek
May 2019

EARLY DIAGNOSIS OF BREAKDOWN THROUGH TRANSFER LEARNING
By Seren Özbek
May 2019

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

H. Altay Güvenir(Advisor)

Hamdi Dibeklioğlu(Co-Advisor)

Ramazan Gökberk Cinbiş

Yiğit Karpat

Approved for the Graduate School of Engineering and Science:

Ezhan Karaşan
Director of the Graduate School

ABSTRACT

EARLY DIAGNOSIS OF BREAKDOWN THROUGH TRANSFER LEARNING

Seren Özbek
M.S. in Computer Engineering
Advisor: H. Altay Güvenir
Co-Advisor: Hamdi Dibeklioğlu
May 2019

Breakdown prediction of equipment is an essential task considering the management of resources and maintenance operations. Early diagnosis systems allow creating alerts on time for taking precautions on production. A significant challenge for diagnosis is to have an insufficient size of data, yet, transfer learning approaches can alleviate such an issue when there is a constrained supply of training data. We intend to improve the reliability of breakdown prediction when there is a limited quantity of training data. We recommend similarity correlation on Remaining Useful Life of these equipment. To do this, we offer learning a common feature space between the target and the source equipment, where we acquire prior knowledge from the source that has different measurements than the target. Within the learned joint feature matrices, we train our model on the vast amount of data of different equipment and finetune it using the data of our target equipment. In this way, we aim to obtain an accurate and reliable model for early breakdown prediction.

Keywords: Transfer Learning, Predictive Maintenance, Fault Diagnosis, Deep Learning, LSTM, Canonical Correlation Analysis.

ÖZET

TRANSFER ÖĞRENİMİ İLE KESTİRİMCİ BAKIM

Seren Özbek
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Danışmanı: H. Altay Güvenir
İkinci Tez Danışmanı: Hamdi Dibeklioğlu
Mayıs 2019

Kestirimci Bakım, arıza kaynaklı sistem kesintilerini en aza indirgeyerek bakım maliyetlerinin azaltılmasını amaçlamaktadır. Erken tanı sistemleri, arızalar konusunda önlem almak için zamanında alarm oluşturulmasına olanak sağlamaktadır. Arıza teşhisi için önemli bir zorluk, yetersiz veri örneğine sahip olmaktır, ancak Transfer Öğrenimi yaklaşımları kısıtlı bir eğitim verisi olması sorununu hafifletebilmektedir. Bu çalışmada, hedeflenen ekipman ve kaynak ekipman arasında ilişki kurularak, kaynak ekipman üzerinde öğrenilen bozulma bilgileri hedef ekipmana transfer öğrenimi ile aktarılmaktadır. Aktarım yapılabilmesi için ekipmanlar arasında ortak benzerlik kümesi oluşturulması gerekmektedir. Bu küme, ekipmanların Kalan Yararlı Ömür niteliği üzerinden elde edilmektedir. Ortak benzerlik kümesinde; hedef ekipmandan farklı ölçümlere sahip olan bir kaynaktan bilgi aktarılmaktadır. Öğrenilen ortak nitelik kümelerinde, model farklı ekipmanların geniş miktarda verisine göre eğitmektedir ve öğrenilen bilgi, hedef ekipmanın arıza teşhisi için kullanılmaktadır. Çalışmada, kısıtlı veri olması durumunda erken arıza tahmini için transfer öğrenme ile güvenilir bir model elde edilmesi amaçlanmaktadır.

Anahtar sözcükler: Transfer Öğrenmesi, Kestirimci Bakım, Arıza Tanısı, Derin Öğrenme, LSTM, Kanonik Korelasyon Analizi.

Acknowledgement

First, I would like to declare my honest gratitude to my advisor Prof. H. Altay Güvenir and my co-advisor Dr. Hamdi Dibeklioğlu for providing me with their immense knowledge, continuous guide, valuable time and understanding during my M.Sc. studies. Their guidance and wisdom played a vital role while developing my research and completing the thesis. This thesis would not have been conceivable without their valuable contributions.

I would also like to thank the rest of my thesis committee, professor Gökberk Cinbiş and professor Yiğit Karpat for their great remarks and assistance. I can not achieve this thesis without declaring my gratitude to our department's secretary, Mrs. Ebru Ateş and our graduate school secretary, Mrs. Damla Ercan Atay for their kind helps.

I would like to thank my dear friends Sanem Elbaşı, Onur Koçak, Mehmet Saygın Seyfioğlu, Batuhan Bardak, Begüm Çıtamak and Merve Nur Yılmaz for their support and all the great memories.

Additionally, I have to thank my family for their love and support throughout my life. I would like to thank my parents Sibel - Fatih Güldamlasıoğlu, lovely twin sister Selin Güldamlasıoğlu, grandparents Münevver - İsa Dinç and Nilhan - Latif - Murat Özbek, for supporting me spiritually throughout writing this thesis and my life in general.

Most importantly, none of these could have occurred without the best husband in the world, Mesut Özbek, who gave his support and love throughout these years. This thesis reaches as evidence of his pure love and encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Predictive Maintenance arises to replace the Scheduled Maintenance that is the standard way to handle maintenance operations. Scheduled maintenance has a series of maintenance procedures defined at the design stage. To prevent breakdowns, scheduled maintenance manages diagnosis operations in a specified number of periods. Nowadays, scheduled maintenance goes out of date since this kind of service cannot offer value. Scheduled maintenance makes a piece of equipment scrap when an industry applies a certain number of maintenance operation. Additionally, scheduled maintenance requires operations to stop when an unexpected breakdown occurs independently from the planned maintenance schedule. Therefore, scraps and unexpected breakdowns make the scheduled maintenance expensive for the equipment partners considering logistic cost, inventory cost, and labor cost. Moreover, due to the unexpected downtimes, customers may not get the requested services.

Predictive maintenance aims to mitigate the undesired consequences of scheduled maintenance. Predictive maintenance proposes an optimal investigation and replacement decision considering the failure cost before a breakdown happens. This approach continuously observes the equipment behavior and makes the maintenance decision by collecting data from the equipment. When there is a possibility of breakdown, above a predefined threshold, a predictive maintenance system

raises an alert. Thanks to generated alerts and continuous tracking of a piece of equipment, the early diagnosis of a breakdown work well in advance [1].

Although there are several predictive maintenance approaches, most of the predictive maintenance solutions are not adaptable to many facilities, mostly due to the limited access to data [2]. Since industrial companies may not be willing to share their data reservoir, predictive models are not able to afford significant decision supports in case of detecting faults.

Machine learning algorithms have been used to learn predictive models from existing data. Models learned by machine learning algorithms concentrate on adjusting the information gained from a labeled source space to unlabeled target space. Many learning strategies work well under when there is a good number of training instances. Even though machine learning demands an adequate amount of data, it is possible to cope with a data quantity problem. New data can be enriched by consuming lots of human labor and time to interpret and label large amounts of training data. However, regular hand-operated labeling is costly, unreasonable, and impractical. Therefore, insufficient amount of labeled data brings out Transfer Learning studies.

Transfer Learning is considered as a powerful technique, where the model learned with one dataset can be reused to learn a model in a different but related domain. The intuitive principle behind this methodology is to receive suitable feature representation for similar feature spaces [3]. Transfer Learning minimizes the distinctions between diverse domains to minimize the cross-space prediction error. Transfer Learning utilizes knowledge to transfer across domains. A common representation is constructed between comparable domains. With the help of this common representation, the source and the target space turn out to be progressively related and comparable.

In the literature, transfer learning is mostly applied to computer vision and classification tasks. Motivated by the achievement of the transfer learning in such jobs, in this study, we aim to construct a bridge between different data spaces for

the early diagnosis problem. When a neural network is in deficiency for predicting the failure status of industrial equipment due to the insufficient size of data, we infer knowledge from a piece of different equipment; therefore, our network gains more information for solving the diagnosis problem. To acquire the knowledge from the different equipment having diverse measurements/ calibrations, we create a common representation on both equipment. By learning the common representation, we carry the knowledge from the source to the target equipment on the joint space. Starting from this point, we assume that the source equipment has a larger data size than the target equipment.

1.1 Objective of the Thesis

This thesis investigates how transfer learning improves predictive model success on the correlated feature space when there is a deficient amount of data for predicting the breakdown of an equipment. We believe that our approaches can help industries in making effective decisions on early diagnosis of the failure. In this manner, industries can maintain their operating duties proficiently while reducing their maintenance cost.

The key contributions of this study can be listed as follows:

- For the first time in the literature, we design a system for accurate prediction of the health status of equipment in case of insufficient training data, through transfer learning.
- We show that transfer learning can be applied between feature spaces with different measurements, learning a common subspace, for early diagnosis of breakdown.
- We evaluate our system on five different datasets using four different common representation learning methods. The results demonstrate that the accuracy of predicting health status of pieces of equipment can be significantly improved, employing transfer learning on jointly projected datasets.

1.2 Organization of the Thesis

The rest of the paper is structured as follows:

- Chapter 1 - Introduction

  This chapter is presented here.

- Chapter 2 - Related Work

  This chapter presents the related work mostly considering transfer learning studies with predictive maintenance objective. We define how transfer learning becomes attractive and why it is applicable on predictive maintenance.

- Chapter 3 - Methodology

  This chapter presents the proposed methodologies for the best results for breakdown classification. The proposed methodologies are for obtaining labels (including feature normalization, remaining useful life, health status label, the source and the target space matching) learning the common representation between two different domains (including the other representation methods), modeling and transfer learning.

- Chapter 4 - Experiments

  This chapter presents experimental results. It includes the sections for database (Turbofan Engine dataset, Wind Turbine dataset, Milling Machine Cutter dataset, Hard Drive dataset, Semiconductor dataset), experiments, settings, effects of the different parameters and comparison to other methods.

- Chapter 5 - Conclusion

  This chapter presents the overall evaluation and offers some future study in order to extend the research.

Chapter 2

Related Work

Transfer Learning was sprung up during the Neural Information Processing Systems 1995 Workshop with this definition: Knowledge Consolidation and Transfer in Inductive Systems is believed to have offered the underlying inspiration to investigate in. From that point onward, terms, for example, Learning to Learn, Knowledge Consolidation, and Inductive Transfer have been utilized alternatively with transfer learning. In the celebrated book, Deep Learning [4] defines transfer learning is as per the following: "Situation where what has been learned in one setting is exploited to improve generalization in another context."

As a formal definition, the essential motivation behind transfer learning is to shift prior knowledge from one to another domain. Transfer Learning grows more attractive when there is a limited amount of training data due to the several reasons such as data being limited, data being high-priced to gather and label or data being out of reach.

Most neural networks which aim to solve complicated issues expect lots amount of data and getting the immense size of labeled data can be extremely troublesome, considering the time and effort it takes to label data. A candid example would be the ImageNet dataset, which has a vast number of pictures relating to various classes, through years of hard work beginning at Stanford [5]. However, getting

such a dataset for each specialty is difficult. Additionally, most profound learning models are extremely specific to a unique domain or even a particular task. These objectives structure why studies attend more consideration for transfer learning.

Transfer learning yields to store knowledge gained while solving a learning problem and apply this prior knowledge to different but related problems. Thanks to the gained knowledge, transfer learning avoids expensive data labeling effort by improving the performance of learning [3]. There are many machine learning applications that transfer learning has been effectively applied to including sentiment text classification [6], image classification [7], human activity classification [8] software defect classification and voice processing [9].

Even though transfer learning is mostly available for computer vision, we focus on its application on time series data, and there is yet a lot to be examined in developing deep neural networks for time series datasets [10]. For detecting anomalies in time series and grabbing the breakdowns, Dynamic Time Warping (DTW) classifier was employed to execute with the aim of transfer learning [11]. DTW algorithm maps the relation between two-time series, which may vary in time or momentum [12]. Next, a transfer learning approach for an auto-encoder was applied to predict the power of wind speed in a farm [13]. The authors assigned knowledge from the past wind farm to a new one through training a model on the historical wind speed data of an old farm and tuning it using the data of the new farm.

Based on the previous studies, we decide to investigate transfer learning for solving breakdown prediction. Several Deep Learning models have been already applied to fault diagnosis until now [14]. Long-Short Term Memory (LSTM) networks were proposed to classify failures on equipment based on Remaining Useful Life (RUL) label [15]. Bi-directional Long-Short Term Memory (BLSTM) was designed to obtain the bidirectional long-range dependences of features and intended to increase the prediction of RUL[16]. Additionally, several researchers conducted asset health classification with deep neural networks for predicting failure and non-failure conditions. The previous study proposed by Microsoft Azure classified the asset health status with 97 % accuracy in 2017 with LSTM

networks [17]. Another study combined LSTM networks with survival analysis for pointing out the breakdowns on assets [18]. Researchers proposed an Extreme learning machine using quantum-behaved particle swarm optimization (Q-ELM) for turbine fan engine fault diagnosis with 93 % accuracy [19]. [20] used Support Vector Machines (SVM) to model faults while employing a multilayer perceptron (MLP) to estimate the number of the faults with 96.8 % accuracy. Real-Time Adaptive Performance Model (RTAPM) Kalman filter is implemented to estimate engine dynamic states [21].

Perhaps, the recent work [22] is the most related to our study in terms of applying transfer learning for early diagnosis of breakdown. The authors have transferred gained knowledge between different but similar working conditions of a turbo-fan engine to predict the remaining useful life (RUL). Our study distinguishes from this study since we propose transfer learning to classify breakdown by shifting knowledge between different equipment that is more or less related, instead of moving knowledge between different conditions of the same asset. To move knowledge from one material to target material, we utilize the study [23] that they minimized the differences between distributions with Canonical Correlation Analysis (CCA) where CCA learns the shared space from different domains. This study transferred the knowledge on the cross-domain area as we aim to do for breakdown classification.

Chapter 3

Methodology

We propose an architecture for early breakdown prediction on sequential data spaces with transfer learning that works on common represented feature space. Our goal is to enhance the reliability of breakdown prediction when there is an insufficient amount of training data.

First, we obtain the labels on our source and target feature spaces, where we aim to shift the knowledge from the source to the target equipment. We use similarity correlation on the obtained label (remaining useful life). Next, we match both datasets based on the remaining useful life in decreasing order. Second, we learn the joint representation of the associated source and target feature vector. Third, we offer LSTM architecture, where we pretrain the source network. Last, we finetune the pretrained network to the target vector, to shift the prior knowledge. Figure 3.1 shows the flow that we propose in this study.

As shown in Figure 3.2, first we learn the common representation between the source and target equipment where source equipment has a large data size of other equipment. We propose to learn a common feature space between the data of our target equipment and that of another equipment that includes different/uncalibrated measurements, using a similarity correlation on the remaining useful life (RUL) of this equipment. Then, in the learned common feature space,

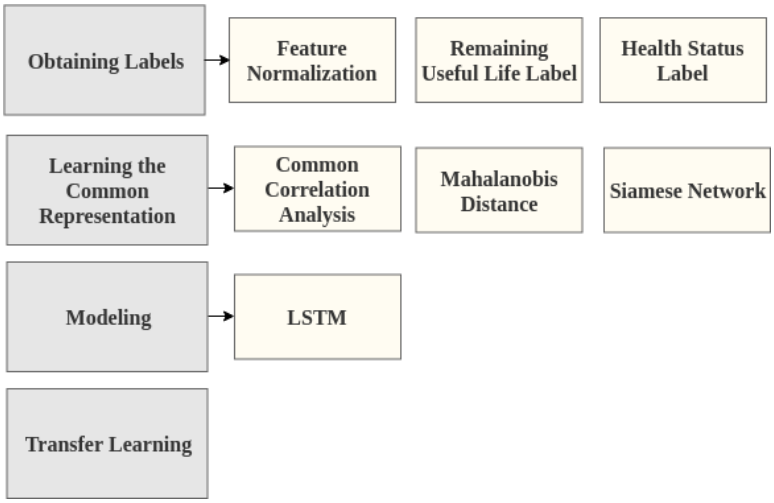

Figure 3.1: Proposed stream

we train our model on the more significant amount of data of other equipment and finetune it using the data of our target equipment. Therefore, we learn earlier knowledge from the source space and adjust the gained knowledge on the target equipment to predict failure. We evaluate the proposed model using a different pair of source and target datasets to enhance our target task.

3.1 Obtaining Labels

In order to shift the knowledge to the target equipment from other equipment that different and plentiful size of features, we intend to acquire a joint feature space. To learn a common feature space, we require a shared feature on the source and the target equipment. Therefore, we adjust the data vectors to obtain the common labels on feature vectors.

First, we normalize our feature vectors, second, we generate the common labels (remaining useful life and health status). As the last, we match a pair of feature vectors based on the remaining useful life label.

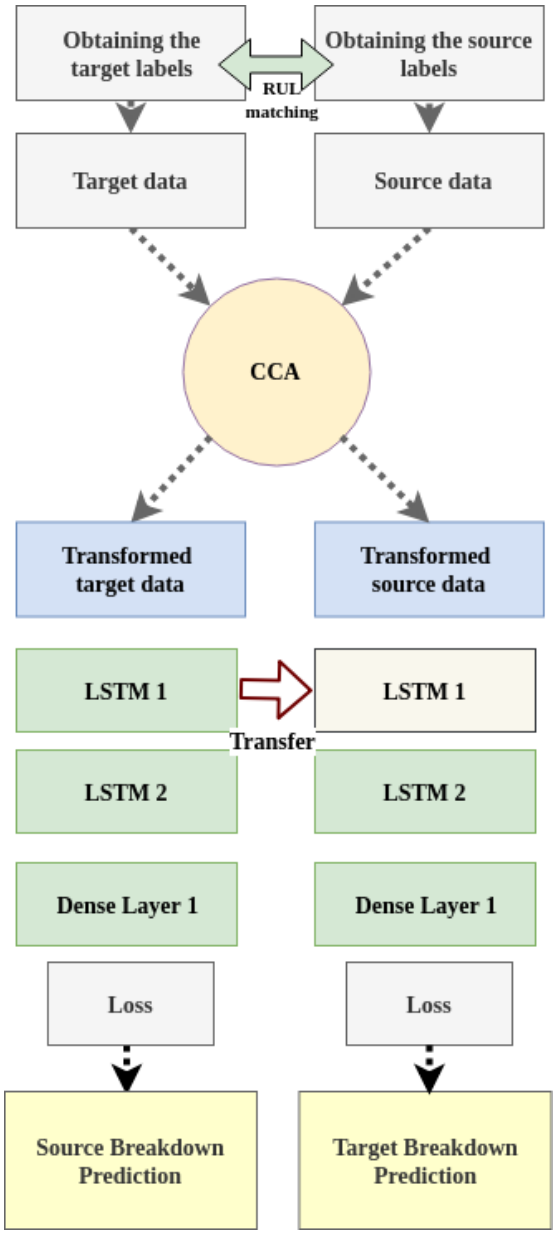

Figure 3.2: System overview

3.1.1 Feature Normalization

As the first step, we normalize our vectors belonging to the data of our target equipment and that of another equipment considering the minimum and maximum scale (3.1). We scale the features with Min-Max Scaler that subtracts the minimum value in the feature vector and divides by the range. The range is the difference between the global maximum and global minimum. Min-Max Scaler maps feature values into the range between zero and one without transforming the raw data remarkably [24].

The formula given below (3.1) represents the feature scaling, where X is the raw feature and $X^{'}$ is the scaled feature based on global maximum and global minimum, these are X_{max} and X_{min}.

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{3.1}$$

3.1.2 Remaining Useful Life Label

In the second step, we obtain the Remaining Useful Life (RUL) label that indicates the time left until breakdown. Since we have to get a common representation with similarity correlation on RUL of the source and the target equipment, we need to calculate the RUL value for each sample feature vector in both data vectors as suggested by [25]:

$$RUL = Time\ to\ Failure\ -\ Current\ Age \tag{3.2}$$

Figure 3.3 depicts how RUL value decreases as the equipment operates in a time cycle. For each equipment unit and each time cycle, we calculate the exact RUL values. We assume that when the equipment unit has 200 cycles (time), the remaining useful life for this unit is equal to 200. As the equipment operates, this value decreases and converges to zero.

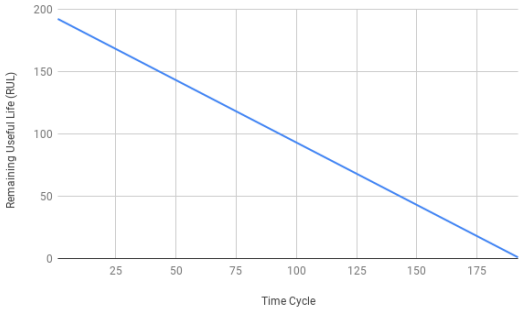

Figure 3.3: Remaining useful life vs time cycle

When we proportion RUL value to time to failure as below formula, we get a label of RUL percentages that depicts the remaining life in portion. The rate of RUL percentage changes between zero and one continuously. This value converges to zero as the equipment starts to fail. We assume that the equipment deteriorates when the RUL percentage is lower than 0.15.

$$RUL(percentage) = \frac{RUL}{Time\ to\ Failure} \tag{3.3}$$

3.1.3 Health Status Label

Additional to RUL, we generate binary labels for the source and target vectors. Binary labels represent the equipment status, whether it runs or failures. The motivation behind this labeling is that we predict the breakdown of equipment based on the binary labels.

We pretend that the equipment deteriorates when RUL is lower than 0.15. At this deterioration range, we assign the equipment health class as a failure and equipment status label as 1. Otherwise, we assign the status label as 0 as we depict in Table 3.1.

RUL Value	Engine Health Class	Engine Status Label
$0<=RUL_i<=0.15$	Failure	1
$0.15<RUL_i<=1.0$	Run	0

Table 3.1: Engine health status label

Table 3.2 points out the our generated features (RUL, RUL percentages and status label) with a data example.

Unit #	Cycle #	RUL	RUL %	Status
1	1	192	0.994	0
1	2	191	0.989	0
1	3	190	0.984	0
1	...	...	...	...
1	191	2	0.005	1
1	192	1	0.005	1
2	1	287	0.996	0
2	2	286	0.993	0
2	...	...	...	...
2	287	1	0.003	1

Table 3.2: Data sample of the generated labels

3.1.4 Source and Target Space Matching

As the last process, for each of the RUL value between zero and one, we match our source and target spaces one-to-one in decreasing order. During sample matching, Euclidean distance is used to compare RUL values. In this way, we ensure that the correlation between the measurements/features in the source and target datasets are computed correctly (based on RUL).

Once we match two vectors on the RUL percentages, later, we propose to learn a common feature space between our target equipment and the source equipment using a similarity correlation on RUL of this equipment. We pretend to gain

prior knowledge based on associated RUL percentages, and we assume to transfer knowledge by using this label.

3.2 Learning the Common Representation

To transfer the knowledge between different feature vectors belonging to our source and target equipment, we may reduce the difference across domains. When we learn the common representation between feature vectors, we get the correlated feature space in which we shift the knowledge to our target equipment from another equipment.

To obtain a common representation for associating data spaces, we employ Canonical Correlation Analysis (CCA). In this process, the data matrices (feature vectors as rows) of source and target datasets are transformed, by linear projection, onto a new space where the columns are maximally correlated. CCA provides a common representation of two different feature spaces by maximizing the correlation, and derived matrices are applicable for solving cross domain classification problems. Figure 3.4 shows how CCA enables us to represent the feature vectors.

Suppose $x \in \mathbb{R}^p$ and $y \in \mathbb{R}^q$ are two column vectors of random variables. $x' = W_x^T x$ and $y' = W_y^T y$ are the transformed vectors to the joint embedded space, where $x', y' \in \mathbb{R}^{\min(p,q)}$. $W_x \in \mathbb{R}^{p \times \min(p,q)}$ and $W_y \in \mathbb{R}^{q \times \min(p,q)}$ maximize the similarity coefficient between these variables with linear combinations since CCA aims to find a joint embedded space with optimal linear transformation between x and y as follows:

$$\rho = \max_{W_x, W_y} \frac{W_x^T C_{xy} W_y}{\sqrt{W_x^T C_{xx} W_x \cdot W_y^T C_{yy} W_y}}, \tag{3.4}$$

where, C_{xx} and C_{yy} are the within set covariance matrices, and C_{xy} is the between-sets covariance matrix [26].

Notice that solution of Eqn. 3.4 would not be influenced by rescaling W_x and W_y unitedly or separately. The optimization of ρ equals to maximizing the number subject to below expression.

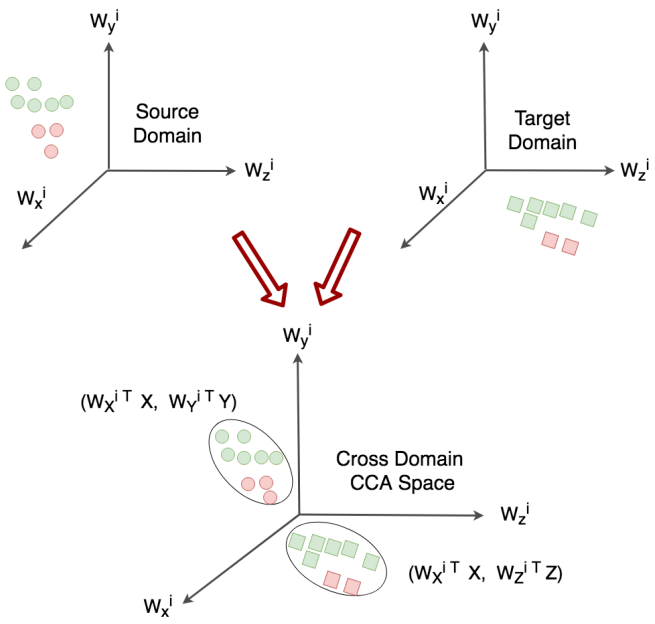

Figure 3.4: Learning the common representation with CCA

$$\begin{aligned} W_x^T\ C_{xx} W_x &= 1 \\ W_y^T\ C_{yy}\ W_y &= 1 \end{aligned} \tag{3.5}$$

The canonical relationship between between x and y can be found with Eigenvalue equations with Langrange multiplier.

$$\begin{aligned} \mathrm{C}_{xx} C_{yy}^{-1} C_{yx} W_x &= \lambda^2 C_{xx} W_x \\ \mathrm{C}_{yx} C_{xx}^{-1} C_{xy} W_y &= \lambda^2 C_{yy} W_y \end{aligned} \tag{3.6}$$

As Figure 3.4 shows, CCA generates common feature subspaces with the optimal linear transformation between different data spaces that are x and y. Therefore, we learn the joint representation of varying feature spaces, x' and y'. This derived representation enables us to transfer prior knowledge from the source space to target space at maximally correlated matrices. We also test the label of the target testing data in this commonly derived subspace.

As a limitation, CCA restricts the joint matrices to be orthogonal, therefore,

narrows the search space for the optimal solution [27]. When we have the common represented feature space, CCA is bounded to the least dimensionality of x and y; therefore, we get the maximum number of canonical correlations as the smallest dimension. For instance, if the dimension of the source and the target space is 9 and 4, we get the maximum number of canonical correlations as 4 [28].

Since CCA defines the common representation of matrices obtained from the source and the target vectors by maximizing the similarity correlation, we utilize maximally represented matrices on cross-domain transfer learning.

3.2.1 Other Methods to Learn Common Representation

- Different from CCA, we implement Mahalanobis Distance Metric Learning, which maximizes the distance between the classes (faulty and healthy states) and minimizes the distance within each class while finding a common representation space between two different datasets. Through reducing the corresponding entropy between two distributions on the source and target datasets, the learned representation gather the same class on the source close together, while pushing away the differently labeled samples [29].

  Mahalanobis distance between x_i and x_j is calculated as $d_A\left(x_i,\ x_j\right)$ where the Equation 3.7 depicts which $A \in \mathbb{R}^{\,d\times d}$ is positively semidefinite where the task of Mahalanobis Distance Metric Learning is to learn a shared distance A across domains D_s and D_t under which distributions of $P_s(D_s)$ and $P_t(D_t)$ is explicitly reduced [30].

$$\begin{aligned} d_A\left(x_i,\ x_j\right) &= \left(x_i\ -\ x_j\right)^T A\left(x_i\ -\ x_j\right) \\ D_s &= \{(x_1^s, y_1^s), ..., (x_n^s, y_n^s)\} \\ D_t &= \{x_1^t,\ ..., x_m^t\} \end{aligned} \tag{3.7}$$

  CCA and Mahalanobis Distance Metric Learning works similarly to the source and target spaces as where Figure 3.4 depicts.

- Siamese Network based approaches find a common representation between two datasets through a Siamese Network with shared weights. Siamese network has two identical networks with different inputs [31]. Each subnetwork of the Siamese structure is a two layered LSTM where the first and second layers have 100 and 50 units, respectively. The Siamese network is trained to minimize the Euclidean or Cosine distance between data pairs (matched samples based on RUL value) in the learned representation space.

3.3 Modeling

To classify the health status of the equipment, we build a transfer learning architecture on jointly represented spaces with Long-Short Term Memory (LSTM) cells.

LSTMs were presented in 1997 and researchers utilized LSTMs effectively in several products, for example, Google Allo, Google Translate, Amazon Alexa and Apple Siri [32]. LSTM overwhelms the boundaries of standard Recurrent Neural Networks (RNN). Rather than storing extended term dependency knowledge, LSTM is designed to use storage elements to pass information from past outputs to current outputs in a selective manner. As previous studies offer, LSTM networks are more efficient than other architectures considering sequential data spaces [33].

LSTMs offer three more layers to Vanilla RNNs that has a single neural network layer such as Tanh [34]. These three signals are current input (x_t), nonlinear function activated by weighted sum of input (σ) and previous hidden state (h_t-1). Depending on these signals, LSTM utilizes these three gates for processing the knowledge. These gates are the forget gate f_t, input gate i_t, and output gate o_t, and can be defined as the Equation 3.8. Forget gate is responsible for forgetting the information retrieved from previous state. Input gate decides whether to update cell with current input or not. Output gate decides whether to pass on hidden state h_t for the next iteration or not [35].

$$
\begin{aligned}
f_t &= \sigma(W_f x_t + H_f h_{t-1} + b_f), \\
i_t &= \sigma(W_i x_t + H_i h_{t-1} + b_i), \\
o_t &= \sigma(W_o x_t + H_o h_{t-1} + b_o),
\end{aligned} \tag{3.8}
$$

As Figure 3.5 points out the inside structure of LSTM cells, we represent the layers with yellow rectangles. Each line shifts the entire vector of values among nodes and layers. LSTM can transform the information originating from the straight line in the top of the cell. Red circles handle the pointwise processes on the vectors such as multiplication and summation.

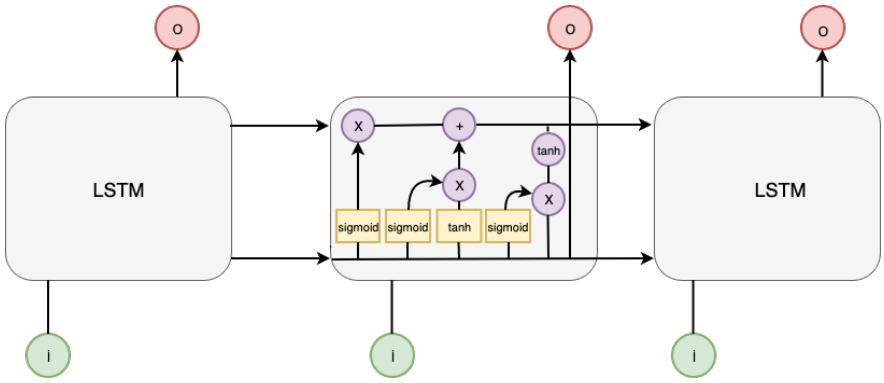

Figure 3.5: Simple LSTM units

In the beginning, LSTM cells remove outdated information with the multiplication (X) operation. Then, three Sigmoid neural network layers decide the amount of data to be discarded or processed by setting the output between zero and one. When information passes to a cell, LSTM determines what current information to store in the cell state. Tanh layer creates the vector of new values which is multiplied by the output of the Sigmoid layer. Summation (+) process adds revealed new vector to the cell state. In the end, LSTM cell state produces the final output. All these operations are implemented by squashing the values between -1 and 1 and utilizing the Tanh operation pursued by a Sigmoid layer that decides which benefits to output [36].

Once we find the maximally correlated representation space between our source and target feature vectors, we can use the transformed features in our learning

model on LSTM networks. We aim to transfer knowledge from pretrained network to target in the common feature space. The source (pretraining) and target (finetuning) models are designed using the same LSTM architecture that has 2 layers. For each of the LSTM layers dropout is used. There is final dense layer with a single unit for the binary classification task of health status of devices (faulty/healthy). Sigmoid function is used for activations. To train the model, the binary cross-entropy loss is minimized. Hyperparameters of our architecture is optimized using 10-fold cross validation. Considered values for the hyperparameters of the network are given in Table 3.3.

Hyperparameter	Considered values
Number of units for the first LSTM layer	{100, 128, 256}
Number of units for the second LSTM layer	{50, 64, 128}
Dropout rate for the first LSTM layer	{0.2, 0.4, 0.6, 0.8}
Dropout rate for the second LSTM layer	{0.2, 0.4, 0.6, 0.8}

Table 3.3: List of the considered hyperparameters

As Figure 3.2 depicts, once we pretrain our model for the source task, we freeze the first layer of the network, and we finetune the pretrained model.

3.4 Transfer Learning

To transfer prior knowledge from the source equipment to target equipment we benefit from transfer learning, which we engage this on commonly represented feature matrices. We learn the joint representation for our source and target space, where the source space is as the relatively large dataset that we pretrain our model and the target is the dataset that we use to finetune the pretrained model. With the commonly represented matrices, we favor the target equipment to predict the failure with the data of another equipment having the different measurements/calibrations.

Instead of starting to learning process from scratch, transfer learning enables the

model to learn from already processed previous models on different tasks. Therefore, this simplified approach leverages the past learnings, inferences the previous patterns and prevents to re-learning process for varying a different problem. Industries can transfer the one overtaken knowledge to various tasks, and acquired knowledge can match different domains just by adjusting the parameters.

This methodology copes with the fragility when there are unknown or unsatisfying issues in industries. For instance, when we are not able to estimate the equipment performance in a different environment, transferring the obtained prior knowledge to this unestimated environment makes the equipment more powerful. Therefore, Transfer Learning enables equipment to fit any situation by estimating their behavior with enriched prior information.

We aim to transfer the knowledge on the learned common feature space using a similarity correlation based on the remaining useful life (RUL) of this equipment. Transfer learning by commonly associated matrices enables us to shift the knowledge earned from a pretrained model and the knowledge is reusable when training the model on another dataset [37].

Transfer learning draws its strength from the pretrained models. A pretrained model is a model that is trained on an extensive benchmark dataset to take care of an issue like the one that we need to solve. We pretrain the predictive model on the commonly represented source dataset. To utilize the pretrained model, we start by finetuning the base model that exposes the prior knowledge. Refining the original pretrained model can be implemented with in two ways: (1) by training the entire model or (2) by training some layers and freezing the other ones.

Training the entire model is high-priced since the original model re-runs from the scratch. As the finetuning strategy demands a lot of computational capability, we prefer the other option for this research, that is training some layers and freezing the others. Technically, when we have a small dataset and a large number of parameters, we have to freeze more layers on the primary pretrained model. Therefore, more frozen layers solve the over-fitting problem. Contrarily, when we have a large dataset and the small number of parameters, we can use exceeding

layers from the pretrained model since over-fitting is not an argument for the new task. Briefly, we convey less frozen layers on the pretrained network, since we have a large amount of source dataset in our problem. We freeze the first one layer which we do not want to train and shift the parameters to the target network. Figure 3.6 presents an overview of the transfer learning.

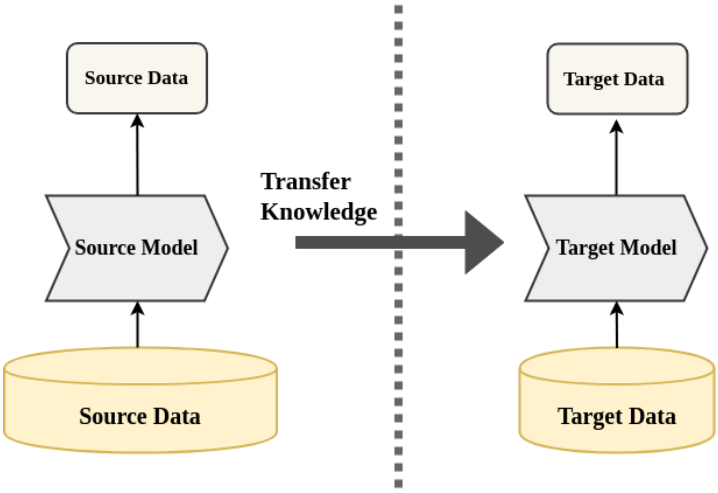

Figure 3.6: Transfer learning overview

As in [38], the parameters for transfer learning can be represented as:

$$\begin{aligned} D_s &= \{X_s, T_s\}, \\ D_t &= \{X_t, T_t\}, \end{aligned} \tag{3.9}$$

where, D_s, X_s, T_s, D_t, X_t, and T_t denote the source domain, source samples, source labels, target domain, target samples and target labels, respectively. Note that both domains D_s and D_t are assumed to be related but not same/similar.

The relationship between the model weights of the source domain and the target domain can be given as:

$$\begin{aligned} W_s &= W_0 + W_1, \\ W_t &= W_0 + W_2, \\ Y_s &= f_s(X_s + W_s), \\ Y_t &= f_t(X_t + W_t), \end{aligned} \tag{3.10}$$

where, W_s and W_t show the parameters in the source and target problems, respectively. W_o is the common parameter where W_1 and W_2 are the different ones

that play role for the transfer learning. Real outputs of the models are Y_s and Y_t which depends on the learning models represented with f_s and f_t mapping the sample inputs to the related task labels. Parameters are transferred from W_s to W_t by making use of the common parts W_o and fine tuning the different parts called W_1 and W_2. It is essential to remark that, in this research, we project our source and target data onto a shared feature space and then apply transfer learning.

We follow the below steps to engage transfer learning, and we intend to leverage information from the pretrained network to solve a different task:

- We represent the source and the target vectors on commonly associated space.
- We pretrain our neural network on the source matrix derived from the common space.
- We finetune the pretrained network by freezing some layers.
- We use the trained weights to initialize a new neural network.

Chapter 4

Experiments

Our method aims to classify the health status of equipment, even with a limited size of training data. In this way, we enable any industrial system to have early breakdown alert on time independently from data size. To assess the reliability of our proposed method, we handle a series of experiments using multiple (source/target) pairs of datasets.

During the experiments, we evaluate how transfer learning increases the accuracy of predicting health status of equipment by exploiting prior knowledge learned in terms of a common representation from a relatively larger dataset. We assess the effect of using varying size of target (training) data and different embedding representations (for learning a common subspace between target and source). Furthermore, we evaluate how similarity level of source and target devices would influence the accuracy.

We conduct our experiments using five different datasets so as to obtain five different source/target pairs. We evaluate our experimental cases with the Area Under the Curve (AUC) value in percentages. During the experiments, we aim to have higher score confronting with the primary task; that is classifying the equipment condition into failure or non-failure classes without the favor of transfer learning by commonly represented feature matrices.

In the rest of this section, details and results of our investigations will be provided.

4.1 Datasets

In each experiment, we use a pair of datasets (D_s and D_t), where D_s and D_t denote the transformed (onto the common feature space) data of the source and target equipment.

As shown in Table 4.1, in our experiments, we employ five different source/target pairs as wind turbine - turbofan engine, milling cutter - turbofan engine, hard drive - semiconductor, hard drive - turbofan engine, semiconductor - turbofan engine.

Source Data	Target Data
Wind Turbine	Turbofan Engine
Milling Machine Cutter	Turbofan Engine
Semiconductor	Turbofan Engine
Hard Drive	Turbofan Engine
Hard Drive	Semiconductor

Table 4.1: Employed source/target data pairs in the experiments

Table 4.1 depicts the information flow between the source and target spaces. We have the source datasets from the wind turbine, milling machine cutter, and hard drive, where we select the turbofan engine and semiconductor as our target equipment. We assume that the source vectors have a full size of data than the targets and source equipment have different measurements and calibrations than the target vectors. Those vectors are appropriate for the breakdown classification task. We project these vectors on common representation matrices, and we shift the prior information from the source to target vector through the derived matrices.

4.1.1 Turbofan Engine Dataset

The Turbofan Engine Degradation dataset (Turbofan Engine) [39] is available by NASA Prognostics Center of Excellence repository to public. It is open to research concerning engine health status classification, and early breakdown alert.

The dataset has 20,631 training samples, where a 29-dimensional feature vector represents each sample. Three of these features represent the operational settings; 21 of them are the sensor measurements, and the remaining ones include unit number (engine) and cycle number. As described in the methodology, we have also obtain three labels such that Remaining Useful Life (RUL), RUL percentages, and binary failure labels. Details of the turbofan engine degradation dataset are given in Table 4.3.

Turbofan engine data analysis	
# of training samples	100
# of test samples	100
Minimum life span (cycles)	128
Maximum life span (cycles)	362
Average life span (cycles)	206

Table 4.2: Statistical numbers of the Turbofan Engine

For the train and test set, we have a hundred different units (engines) that record the cycle of the engine. Cycles represent the life span and decreases during the engine continues its operation. Table 4.2 points out the maximum life span and minimum life span reported at units. The maximum number of cycles equals 362, and the minimum number of cycles equals 128 where the average life span is 206. We provide the features on Table 4.3 that includes the original features and the obtained labels.

Turbofan engine dataset has several multivariate time series separated into training and test subsets. Each time series outlines the different engine that starts with particular degrees of initial corrosion. At the starting of each time series, each motor starts its operations ordinarily, and a fault occurs at some point as the engine continues the service. Corresponding dataset feed from several sensors

Engine Settings	
Settings No	**Description**
1	Unit number
2	Cycle
Operational Settings	
Settings No	**Description**
1	Altitude
2	Mach number
3	Throttle resolver angle
Sensor Measurements	
Sensor No	**Description**
1	Total temperature at fan inlet (R)
2	Total temperature at LPC outlet (R)
3	Total temperature at HPC outlet (R)
4	Total temperature at LPT outlet (R)
5	Pressure at fan inlet (psia)
6	Total pressure in bypass-duct (psia)
7	Total pressure at HPC outlet(psia)
8	Physical fan speed (rpm)
9	Physical core speed (rpm)
10	Engine pressure ratio (P50/P2)
11	Engine pressure ratio
12	Ratio of fuel flow to Ps30 (pps/psi)
13	Corrected fan speed (rpm)
14	Corrected core speed (rpm)
15	Bypass Ratio
16	Burner fuel-air ratio
17	Bleed Enthalpy
18	Demanded fan speed (rpm)
19	Demanded corrected fan speed (rpm)
20	HPT coolant bleed (lbm/s)
21	LPT coolant bleed (lbm/s)
Obtained Labels	
Label No	**Description**
1	Remaining useful life (RUL)
2	Remaining useful life (RUL) - percentages
3	Health status label

Table 4.3: Training and test features of the turbofan engine

that collect features as temperature, engine pressure, fuel and coolant blend data from the turbofan engine.

We analyze the feature distribution where some features have a normal distribution as shown in Figure 4.1. There is that there is a high variation between units considering the maximum number of cycles. As expected, the test set has shorter the number of cycles than a train set as we provide in Figure 4.2.

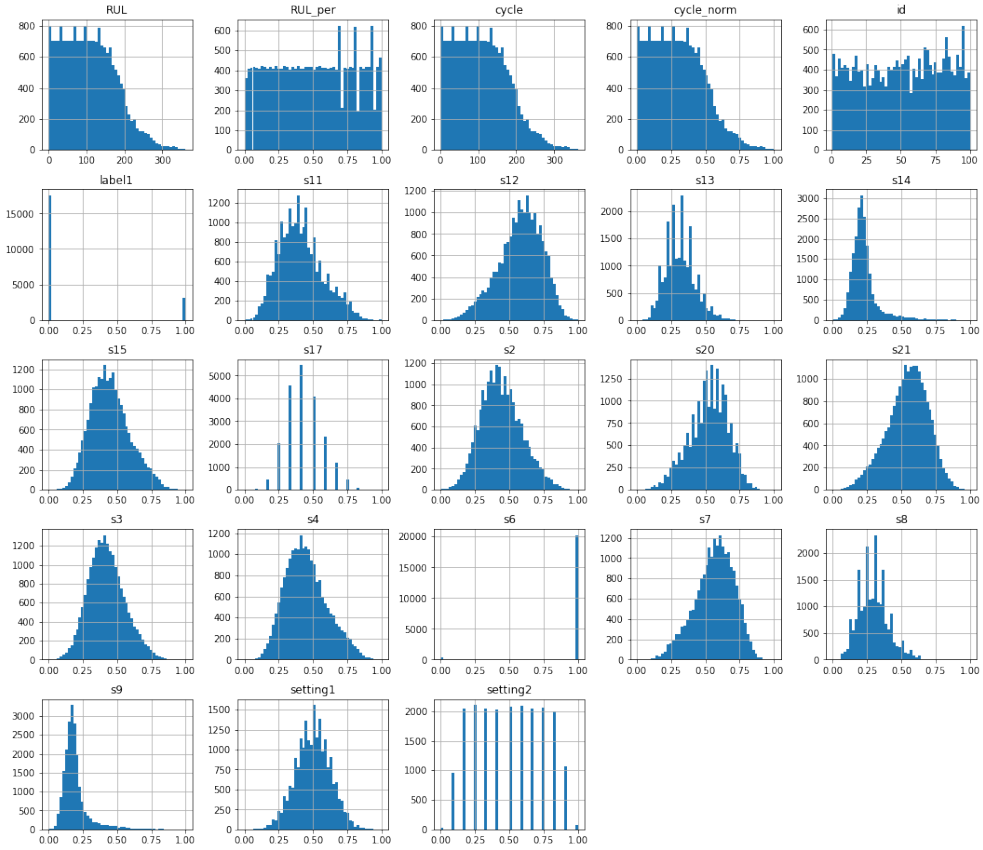

Figure 4.1: Turbofan engine feature distribution

4.1.2 Wind Turbine Dataset

Wind turbine dataset has been collected from 3 MW direct-drive turbines which generate power to a biomedical devices plant. The measures has been collected for 11 months, thus, it has a sequential series of wind turbine operations produced on SCADA (Supervisory Control and Data Acquisition) system which observes the

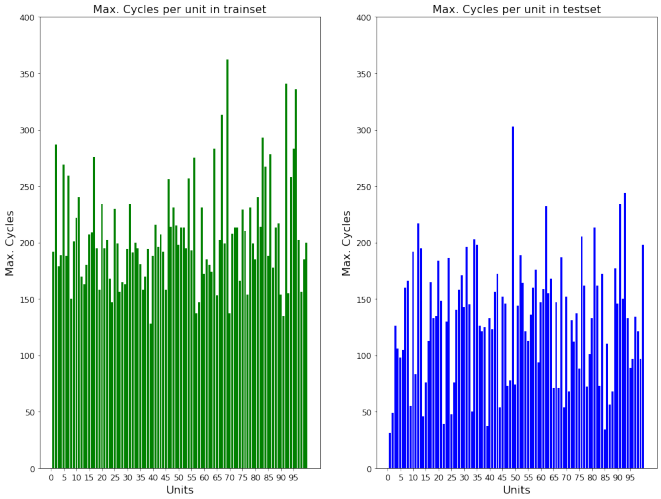

Figure 4.2: Max cycles per unit in train and test set

wind turbine continuously [40]. This dataset stores the data of the Wind Turbine components such as real and reactive energy, voltages, heats with 10-minutes time intervals.

The dataset includes 29 features and 39,210 training and 9,878 test samples mostly related to bearing temperature, angle, rotation, and power. We normalize the features because of some features, e.g., energy output extends from zero to thousands, whereas heat varies from zero to tens

According to generated operational data, the dataset has status data to point out the labels related to faults, warnings and turbine status, as Table 4.4 points out. Turbine status displays the state of the turbine when it is delivering conventional energy during the wind speed is below than required or when the storm mode is on. The dataset represents the turbine status with a message included inside the status data. When the state of the turbine changes, status message is generated with a time-stamp.

Status messages show that wind turbine faults occur due to feeding fault, generator heating fault, compressor bleed band failure and excitation fault. Feeding fault occurs with a defect inside the power feeder cables. Generator excitation system includes excitation failure, and heating fault indicates to the overheating.

According to status data, data has a label as "fault" and "no-fault." If a status message includes fault, corresponding time-band generate a "fault" label. For instance, if an excitation fault occurs between 11:39-13:42, the label of this time band set as a "fault." According to a fault and no-fault labels, we obtain the labels as described in Section 3.1.

Timestamp	Main Status	Sub Status	Status Text
13/07/2014 13:06:23	0	0	Turbine in operation
14/07/2014 18:12:02	62	3	Feeding fault
14/07/2014 18:12:19	80	21	Excitation error: Overvoltage
14/07/2014 18:22:07	0	1	Turbine starting
14/07/2014 18:22:28	0	0	Turbine in operation

Table 4.4: Status data of the Wind Turbine

4.1.3 Milling Machine Cutter Dataset

BEST lab at UC Berkeley provides experiments on a milling machine for various speeds, feeds, and deepness of cut. The dataset represents high-speed CNC milling machine cutters with 29 features and 45,745 data points [41].

Milling operation removes metal by pivoting with a cutter that has single or multiple cutting edges. A milling cutter shapes the bowed or even surfaces with an excellent finish. A milling machine also feasible for drilling, slotting, making a round form and material cutting by having fitting attachments. The cutting rates in high-speed milling are usually as high as 10,000 to 50,000 RPM, and the feeds are additionally very high, which deliver items with surface quality and high capability. However, this compelling operation generates cutting deterioration [42]. As long as milling cutters serve the operation, deterioration increases and industries require an early diagnosis for a milling machine to continue their service.

4.1.4 Hard Drive Dataset

Hard-drives are fundamental components for information storage. Hard drive diagnosis is crucial in case of danger of information disaster. Sensors notice a hard-drive breakdown, and they generate alerts. These alerts show that there is some problem with the drive, and users have to replace the hard drive soon. We employ the hard drive dataset introduced by [43]. The dataset has 3843 samples with 10 features including quantities such as time, serial number, type, capacity bytes, and failure. It is accessible from the Center for Magnetic Recording Research (CMRR) University of California, San Diego.

4.1.5 Semiconductor Dataset

Semiconductor production has a complicated process with a hundred steps. The fundamental processes in semiconductor production are as the following: Making of silicon wafers from raw silicon material, fabrication of combined circuits onto the new bare silicon wafers, adjustment by setting the combined circuit inside a package to form a ready-to-use output, and testing of the complete products [44]. Sensors monitor all of these processes and generate data to track breakdowns. In our experiment, we employ a semiconductor dataset from [44]. It has 1567 samples with 591 features. Since several features are assumed to be redundant and correlated [44], we reduce its dimensionality to 10 using Principal Component Analysis .

Number of training and tests examples in each of the aforementioned datasets are reported in Table 4.5.

	# of Samples	
Dataset	Training	Test
Turbofan Engine	20,631	13,096
Wind Turbine	39,210	9,878
Milling Cutter	45,725	29,821
Hard Drive	3,843	1,647
Semiconductor	1,158	374

Table 4.5: Number of training and tests samples in datasets

4.2 Settings

Hyperparameters of our model are optimized using 10-fold cross-validation. Based on the minimum validation error: (1) the number of units of first and second LSTM layers have been set to 128 and 64, respectively; (2) dropout rates for both layers have been determined as 0.2. Adam optimizer is used and the maximum number of epochs is set to 50, employing early stopping. Considered values for minibatch size are {10, 50, 100, 150, 200}. The offered settings utilize the transfer learning (with transformed features) to perform well.

4.3 Influence of Transfer Learning

In the first experiment, we assess the benefit of exploiting transfer learning for early diagnosis of breakdown. To this end, we randomly choose 500 training samples from the turbofan engine dataset to train our model without using transfer learning. In other words, our model is solely trained on those 500 samples. Furthermore, we train two different models using our transfer learning framework, where the wind turbine and the milling cutter datasets are used as pretraining source. For a fair comparison, we set the number of finetuning samples (turbofan engine) to 500 for each of these models.

As reported in Table 4.6, transfer learning drastically enhance the reliability of detecting faulty states of commercial equipement. While a correct classification

Pre-training Data	Classification AUC
n/a	0.54
Wind Turbine	0.75
Milling Cutter	0.72

Table 4.6: Influence of enabling transfer learning for classifying faulty and healthy states

rate (AUC) of 54% can be achieved with solely learning from 500 samples, exploiting knowledge obtained from the milling machine cutter and the wind turbine datasets increases this rate by 33% and 39%, respectively.

4.4 Influence of the Size of Training Data

In order to explore how much improvement can be achieved through transfer learning while the amount of training (finetuning) data changes, we train several models where the milling machine cutter and the wind turbine datasets are used as pretraining data.

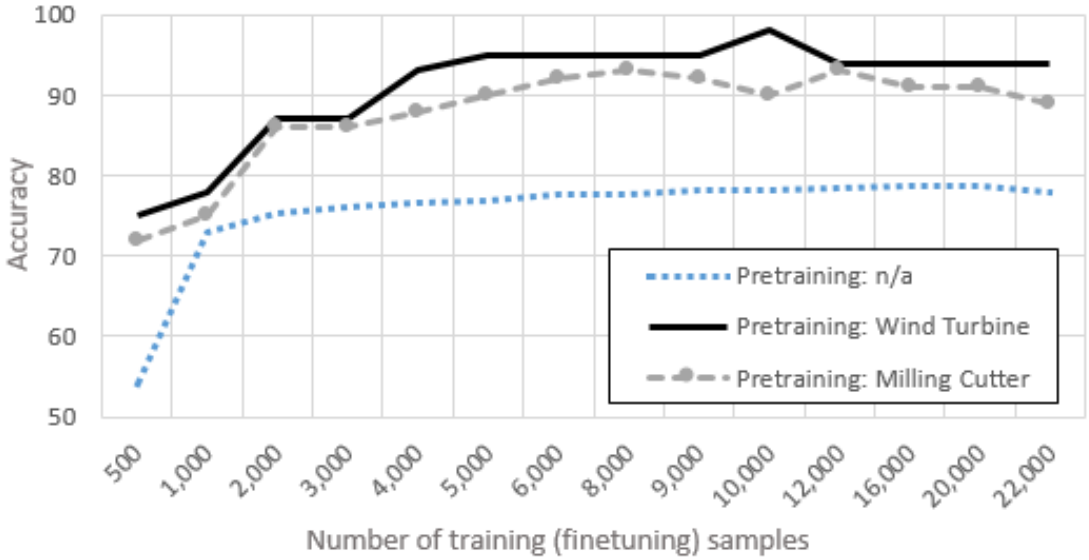

Figure 4.3: Effect of transfer learning with different amount of training data

As shown in Figure 4.3, while the contribution of transfer learning decreases when we increase the number of training (finetuning) samples of the target task, still there is a significant improvement compared to results obtained without transfer learning.

4.5 Influence of Domain Similarity

To evaluate the impact of domain similarity in transfer learning (pretraining), we train different models for distinguishing between faulty and healthy states of turbofan engines employing knowledge transfer from the wind turbine, milling machine cutter, hard drive, and semiconductor datasets. For a detailed analysis, we also use different number of finetuning samples (turbofan engine).

# of Finetuning Samples (Turbofan Engine)			
Pre-training Data	500	10,000	20,000
Wind Turbine	0.75	0.98	0.94
Milling Cutter	0.72	0.90	0.91
Hard Drive	0.63	0.75	0.75
Semiconductor	0.59	0.70	0.70
n/a	0.54	0.78	0.79

Table 4.7: AUC rates for transferring knowledge from different domains using different number of finetuning samples

As Table 4.7 shows, using all four different datasets increase the correct classification rate (AUC) in case of very limited number of training samples. On the other hand, while similar domains such as using the wind turbine and the milling cutter knowledge in the modeling of the turbofan engine data works well, knowledge obtained from less-similar or unrelated domains such as hard drive and semiconductor, would even decrease the classification accuracy once we have sufficient amount of training samples.

When we examine the operating principle of the wind turbine and turbofan engine, wind turbine working principle is similar to turbofan engines where both avail from the air.

Wind turbine operates on the rotor that two or three propeller-like edges around it. The rotor connects to the inner shaft, which turns a generator to make energy. Wind turbines are fixed on a tower to catch the most energy. At 100 feet (30 meters), they can utilize fleeter and less turbulent breeze. Wind turbines can be reserved to create power for a single place or building or public power grids.

Turbofan engines have a working system that takes air into the front of the motor using a fan. From there, the engine compresses the air, combines fuel with it, burns the fuel/air mixture, and fires it out the back of the engine, creating force.

4.6 Influence of Using Different Methods for Learning a Common Representation

To evaluate the reliability of CCA for learning a common representation between two different set of features/measures, we implement three competitor methods by modifying our framework replacing the CCA module with three different approaches, namely: Mahalonobis Distance, Siamese Network based Euclidean Distance, Siamese Network based Cosine Distance (Table 4.8).

Common representation learning methods
1- Canonical Correlation Analysis
2- Mahalonobis Distance
3- Siamese Network, Cosine Distance
4- Siamese Network, Euclidean Distance

Table 4.8: Methods to learn common representation

To evaluate the influence of methods that learn the joint representations, we represent the pair of datasets (the source and the target) on the common space and we train several models where we use our source datasets as pretraining and our target dataset as the finetuning. We compare how common representation embeddings behave on different data pairs respectively.

We compare the different methods to learn a common representation with different pretraining datasets (wind turbine, milling machine cutter, hard drive and semiconductor) on the turbofan engine. We aim to understand which common representation outperforms to results obtained without transfer learning considering different pretrain datasets and finetuning samples.

4.6.1 Wind Turbine and Turbofan Engine

First, we analyze which method that learns the common representation performs well for transforming the wind turbine and the turbofan engine to commonly represented space. To analyze this, we pretrain the network with the transformed subspace (wind turbine); then, we finetune the samples (turbofan engine).

As Figure 4.4 shows, CCA is well ahead among the common representations, by transforming the feature vectors on similarity based common space on which we predict the breakdown of the turbofan engine (AUC). We compare the embedding methods for each finetuning samples (turbofan engine).

CCA improves the correct classification rate (AUC) by 39%, 26%, 20%, (absolute) on average using finetuning samples 500, 10,000, 20,000, respectively. Other methods increase the AUC relatively little, where Siamese Network with Cosine distance enhances AUC by 31%, 2%, 4%, Siamese Network with Euclidean distance increases AUC by 20%, 0%, 2%, and Mahalanobis distance decreases AUC at some samples by 2%, -4%, -1%, respectively. Table 4.9 shows the comparison with the correct classification rates for using different methods to learn common representation with different number of finetuning samples.

Since the working principle of the wind turbine and the turbofan engine is similar/relevant, CCA outperforms the other methods. The reason behind this result is that CCA projects the vectors onto subspace with the connectivity projection matrix. This representation reveals the most positive associations with the related pretraining and finetuning datasets.

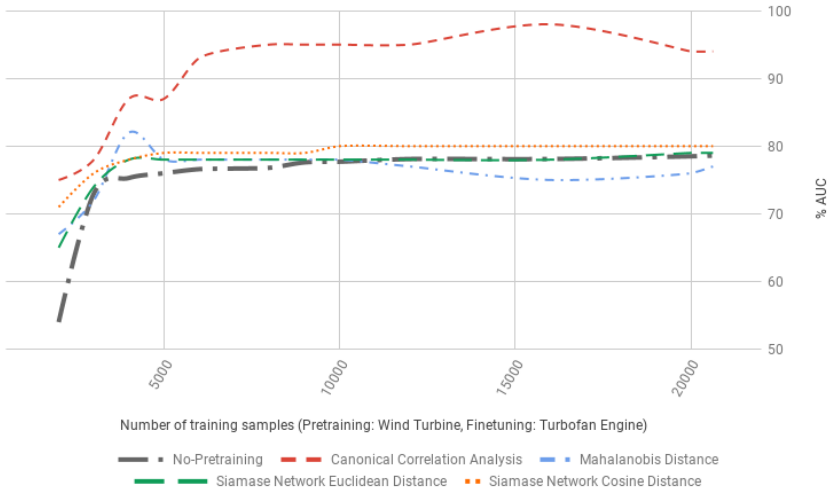

Figure 4.4: The impact of different common representation learnings on Transfer Learning

	Pretraining: Wind Turbine		
	# of Finetuning Samples		
Methods	500	10,000	20,000
CCA	0.75	0.98	0.94
Mahalonobis Distance	0.67	0.75	0.77
Siamese Network, Euclidean Distance	0.65	0.78	0.79
Siamese Network, Cosine Distance	0.71	0.80	0.81
n/a	0.54	0.78	0.78

Table 4.9: AUC rates for using different methods for learning a common representation using different number of finetuning samples

4.6.2 Milling Machine Cutter and Turbofan Engine

Next, we change our source dataset to analyze different relation. We transform the milling machine cutter and the turbofan engine dataset with the proposed methods. We aim to analyze how different representation methods perform on the pretraining dataset (milling machine cutter) and finetuning dataset (turbofan engine) to improve the breakdown classification. As Figure 4.5 shows, we evaluate the changes on the AUC considering the data samples of the finetuning as 500, 10,000, 20,000, relatively.

CCA improves the AUC by 33%, 15%, 16% (absolute) on average using finetuning samples; 500, 10,000, 20,000. Siamese Network with Cosine distance increases AUC very few as 3%, 0%, 0%. Siamese Network with Euclidean distance increases AUC by 2%, 1%, 1%. Mahalanobis distance changes the AUC as 5%, -4%, -4%.

Except for CCA, the other methods that learn the common representation has a slight impact on the breakdown classification, where CCA overperforms the other representation learning methods with high impact (Table 4.10).

Even though the working principle of the milling machine cutter and the turbofan engine is not similar/relevant, milling machine cutter proposes an huge amount of pretraining data to finetune. Therefore, the methods - especially CCA, learn the common representation well, in particular for the limited supply of pretraining dataset such as 500 samples.

	Pretraining: Milling Cutter		
	# of Finetuning Samples		
Methods	500	10,000	20,000
CCA	0.72	0.90	0.90
Mahalonobis Distance	0.57	0.77	0.75
Siamese Network, Euclidean Distance	0.57	0.79	0.79
Siamese Network, Cosine Distance	0.56	0.77	0.78
n/a	0.54	0.78	0.78

Table 4.10: AUC rates for using different methods for learning a common representation using different number of finetuning samples

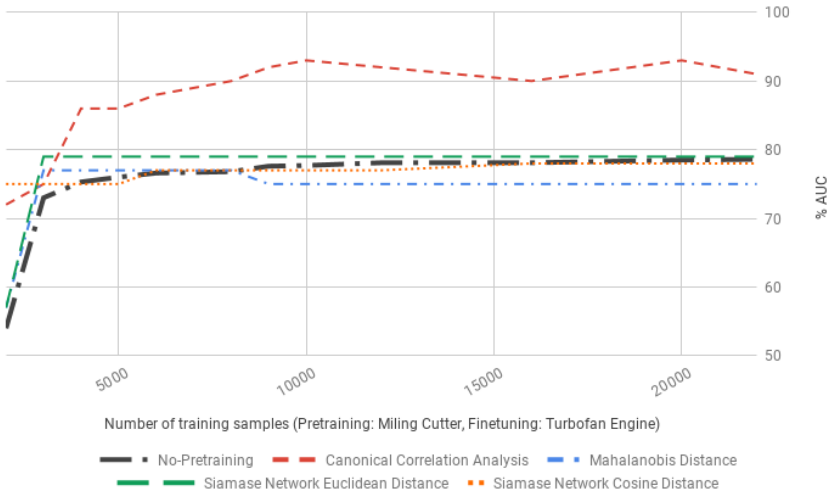

Figure 4.5: The impact of different common representation learnings on transfer learning

4.6.3 Hard Drive and Turbofan Engine

Next, we replace the pretraining data to hard drive dataset that is less similar to the turbofan engine. We aim to show how different common representation methods perform for the less similar/related pretraining dataset (hard drive) for improving the breakdown correct classification rate on the finetuning dataset (turbofan engine).

As Figure 4.6 and Table 4.11 shows, CCA changes the AUC little by 16%, -3%, -3% (absolute) on average using finetuning samples relatively. Siamese Network with Cosine distance impacts the AUC almost negatively as 2%, -14%, -9%, Siamese Network with Euclidean distance changes AUC with 7%, -12%, -9%. Mahalanobis distance impacts the AUC by 11%, -5%, -4%.

Since the pair of pretraining and finetuning datasets are not relevant, and the hard drive feature vector does not consist huge amount of data, common representation embeddings are not supporting the transfer learning well.

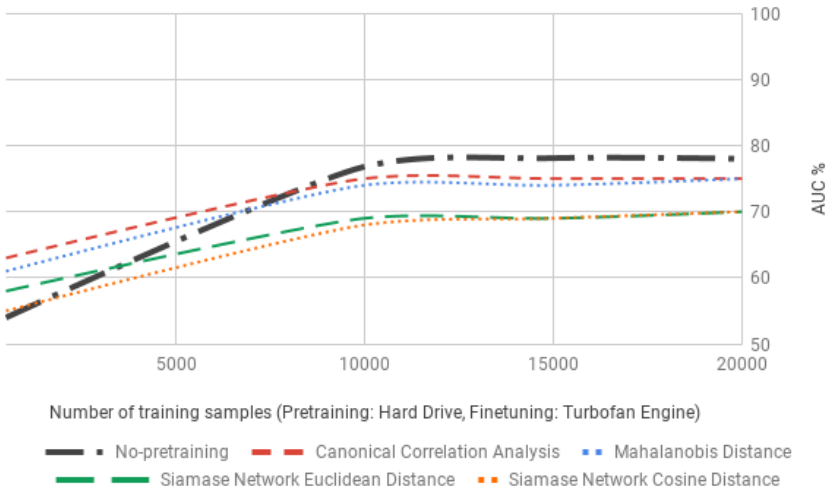

Figure 4.6: The impact of different common representation learnings on transfer learning

	Pretraining: Hard Drive		
	# of Finetuning Samples		
Methods	500	10,000	20,000
CCA	0.63	0.75	0.75
Mahalonobis Distance	0.61	0.74	0.75
Siamese Network, Euclidean Distance	0.58	0.69	0.70
Siamese Network, Cosine Distance	0.55	0.68	0.70
n/a	0.54	0.78	0.78

Table 4.11: AUC rates for using different methods for learning a common representation using different number of finetuning samples

4.6.4 Semiconductor and Turbofan Engine

As the last case on the finetuning data (turbofan engine), we replace the pretraining data to semiconductor. Again, the goal is to analyze how different methods behave when we have a less related and less sized pretraining dataset.

As Figure 4.7 points out, CCA changes the AUC little by 9%, -8%, -8% (absolute) on average using finetuning samples relatively. Siamese Network with Cosine distance impacts the AUC negatively as -4%, -14%, -9%, Siamese Network with Euclidean distance decreases AUC with -2%, -14%, -9%. Mahalanobis distance changes the AUC negatively with -7%, -11%, -11% (Table 4.12).

When the pair of the pretraining and finetuning datasets are not similar, and the pretraining feature vector does not consist huge amount of data, common representation methods are not supporting the transfer learning as expected.

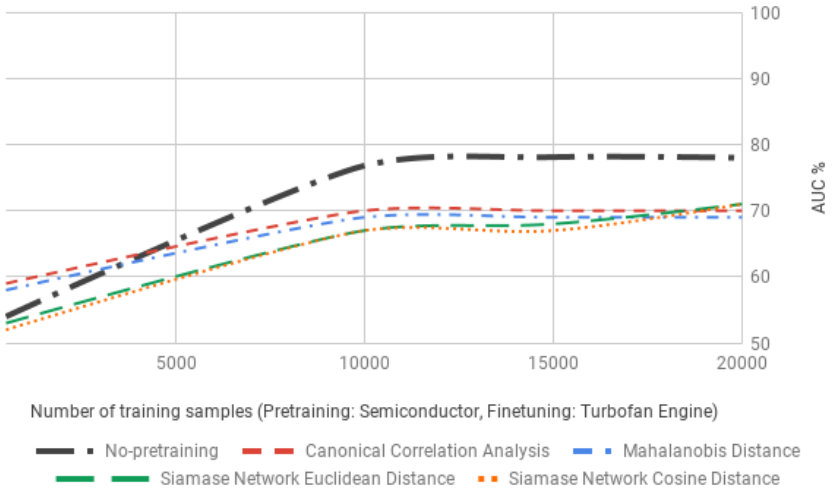

Figure 4.7: The impact of different common representation learnings on transfer learning

Briefly, when we have similar pair of pretraining (wind turbine and milling machine Cutter) and finetuning (turbofan tngine) datasets, as shown in Table 4.9 and Table 4.10, CCA outperforms all the candidate methods significantly. Over

	Pretraining: Semiconductor		
	# of Finetuning Samples		
Methods	500	10,000	20,000
CCA	0.59	0.70	0.70
Mahalonobis Distance	0.58	0.69	0.69
Siamese Network, Euclidean Distance	0.53	0.67	0.71
Siamese Network, Cosine Distance	0.52	0.67	0.71
n/a	0.54	0.78	0.78

Table 4.12: AUC rates for using different methods for learning a common representation using different number of finetuning samples

the best competitor (Siamese Network based Cosine Distance Method), CCA improves the AUC by 10%, 15%, and 13% (absolute) on average using 500, 10,000, and 20,000 finetuning samples, respectively.

The reason behind the success of CCA may be defined by the certainty that CCA projects the target and source data on to subspace with the connectivity projection matrix. Thus, CCA reveals the most positive correlation between both source and target spaces [28].

4.7 Influence of the Optimizer on Canonical Correlation Analysis

To evaluate the influence of the optimizing parameters of the methods that learn common representation, we engage Canonical Correlation Analysis, where this representation outperforms the other competitors.

CCA has a regularization parameter: number of components to keep. This parameter decides the number of components to preserve while projecting the subspaces. Empirically, we evaluate this regularization parameter for finding the best-related subspaces between the source and the target sets. We aim to choose the suitable regularization parameter that maximizes the CCA score. The best possible CCA score is 1, and the decline in this score shows that the pair of

pretraining and finetuning datasets are not similar/related.

When we decide the number of components to keep, CCA is bounded to the smallest dimensionality of the pretraining and the finetuning datasets. CCA assumes the smallest dimension as the maximum number of the component to keep [28].

We select the smallest dimension for the pretraining (wind turbine) and the finetuning (turbofan engine) datasets. We choose the number of preserved components as 26, which is the raw feature dimension (without obtained labels) belonging to the turbofan engine.

We experience the number of components between 2 and 26. When the number of components increases, the CCA performs a better correct classification rate - AUC as we describe in Table 4.13.

# of Components	CCA Score	Transfer Learning AUC
26	-0.14	0.94
10	-0.18	0.87
15	-0.20	0.86
10	-0.21	0.77
5	-2.98	0.77
2	-3.01	0.67

Table 4.13: Comparison to number of components on CCA

4.8 Canonical Correlation Analysis for Normalization

To analyze how CCA influences on transfer learning effectiveness by itself as normalization, we engage CCA on the finetuning dataset (turbofan engine) without the pretraining on the source space.

Figure 4.8 points out the impacts of CCA on the transfer learning and the impact line is between the baseline and the transfer learning as expected. Since the impact trend is between the baseline and our best line (proposed), transfer learning has more importance for having high confidence in prediction than the methods that learn the common representation. Table 4.14 provides this analysis for each finetuning samples.

	Pretraining: Wind Turbine		
Data samples of Turbofan Engine	No-pretraining	Pretraining	Pretraining Normalization
500	0.54	0.75	0.68
1000	0.73	0.78	0.74
2000	0.75	0.87	0.79
3000	0.76	0.87	0.79
4000	0.76	0.93	0.80
5000	0.76	0.95	0.82
6000	0.77	0.95	0.82
8000	0.77	0.95	0.82
9000	0.78	0.95	0.83
10000	0.78	0.98	0.84
12000	0.78	0.94	0.84
16000	0.78	0.94	0.84
20000	0.78	0.94	0.84
20631	0.78	0.94	0.84

Table 4.14: Detailed comparison on the turbofan engine dataset

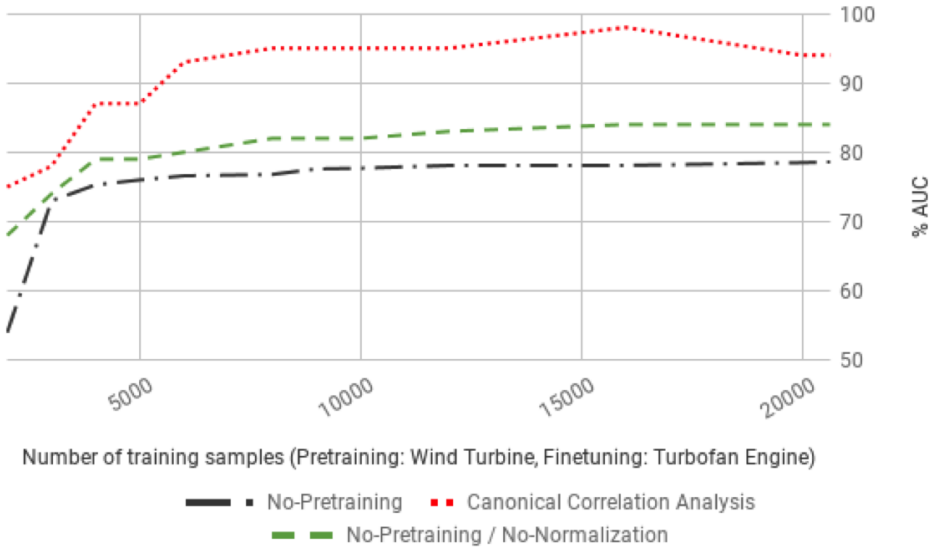

Figure 4.8: CCA for normalization

4.9 Analysis of Transformation Coefficients of Canonical Correlation Analysis

To understand how CCA associates the features of the source (pretraining) and the target (finetuning) datasets, we analyze the transformation coefficients (transformation matrix) of CCA. CCA generates two transformation matrices; one of them is for the source (pretraining) and the other one is for the target (finetuning) dataset.

The transformation matrices represent the linear transformations of the source and the target datasets. These matrices would be used for selecting the features by ranking the absolute values of the weights.

In this experiment, for each dimension of the learned common space, we select the most important three features of the source and the target datasets by ranking the corresponding weights in the transformation matrices. Table 4.15 reports the top three ranking features. Here, we expect to see a relation between conceptually associated features in two different datasets.

Features #	Wind Turbine			Turbofan Engine		
	Ranking1	Ranking 2	Ranking 3	Ranking 1	Ranking 2	Ranking 3
1	wind speed	blade angle	spinner temp	fan temp	LPT temp	HPC pressure
2	front bearing temp	rotation	transformer temp	LPC temp	engine pressure	fan speed
3	rotor temp 1	stator temp 1	rotation	HPC pressure	fan speed	core speed
4	tower temp	front bearing temp	power	engine pressure	HPT coolant	fan pressure
5	transformer temp	stator temp 2	rotor temp 1	LPT temp	HPT coolant	LPC temp
6	rear bearing temp	stator temp 1	nacelle temp	core speed	HPC pressure	fan speed
7	fan inv. cabin. temp	rotor temp 1	cabinet blade B	LPC temp	LPT coolant	core speed
8	power	rotation	ambient temp	LPC temp	core speed	engine pressure
9	rotor temp 2	transformer temp	rear bearing temp	engine pressure	LPC temp	HPC pressure
10	spinner temp	wind speed	power	LPC temp	core speed	fan pressure
11	rotor temp 2	ambient temp	stator temp 2	LPT temp	fan pressure	fan temp
12	rear bearing temp	rotor temp 2	spinner temp	fan speed	engine pressure	HPC temp
13	rear bearing temp	ambient temp 1	cabinet blade B	LPC temp	HPT coolant	HPC temp
14	front bearing temp	wind speed	spinner temp	HPC pressure	LPT coolant	bypass ratio
15	front bearing temp	power	cabinet blade A	fan speed	HPC pressure	engine pressure
16	transformer temp	fan inv. cabin. temp	tower temp	engine pressure	HPC temp	LPC temp
17	rotation	rotor temp 1	front bearing temp	HPT coolant	core speed	fan temp
18	front bearing temp	stator temp 2	power	LPT temp	LPT coolant	fan pressure
19	front bearing temp	wind speed	rotor temp 1	LPC temp	engine pressure	fan pressure
20	spinner temp	fan inv. cabin. temp	tower temp	LPC temp	core speed	LPT coolant
21	rear bearing temp	cabinet blade A	spinner temp	fan speed	HPC pressure	core speed
22	rear bearing temp	tower temp	power	fan temp	LPT temp	HPT coolant
23	ambient temp 1	wind speed	power	fan pressure	LPT coolant	fan speed
24	rotation	rotor temp 2	rear bearing temp	fan speed	fan temp	LPC temp
25	rectifier cabinet temp	power	spinner temp	fan speed	engine pressure	LPC temp
26	rear bearing temp	stator temp 1	stator temp 2	LPC temp	core speed	HPC temp

Table 4.15: Features with higher coefficients on the transformation matrices of CCA

As expected, temperature related features on the source and the target datasets are clearly associated in the common representation space. More specifically, for 20 of 26 dimensions of the common space, temperature related features of both datasets are in the top three most weighted features.

Furthermore, we also analyze the most important features in our experiments. For the source dataset (wind turbine), rear bearing temperature has the highest weight on the 6 of 26 features considering the first ranking. For the top three rankings, rear bearing temperature has the highest weight on the 7 of 26 features. For the target dataset (turbofan engine), LPC temperature has the highest weight in the 8 of 26 features. Our results suggest that rear bearing temperature (wind turbine) is conceptually the most associated feature with HPC temperature (turbofan engine) since these features are observed to highly contribute in 6 of 26 dimensions (of the common space).

4.10 Influence of Remaining Useful Life in Matching Quality

In this work, the source and the target datasets are matched based on a common label, namely, the Remaining Useful Life (RUL). RUL provides promising results for predicting the failures on the target equipment by transfer learning with a common representation.

To assess the effectiveness of RUL in terms of matching quality, we compare RUL based matching with two other matching approaches. First, we match the source and the target datasets based on the classes that indicate the health status of the equipment (healthy/faulty). Once we associate both datasets on the health status, we randomly match data samples within the same classes. Second, we randomly match the source and the target datasets without considering health status or RUL. In these experiments, 20,000 samples are used for finetuning.

Table 4.16 shows the AUC values achieved by different matching approaches. As

Finetuning Dataset: Turbofan Engine			
Pre-training Data	RUL	Class	Random
Wind Turbine	0.94	0.90	0.82
Milling Cutter	0.91	0.87	0.81
Hard Drive	0.75	0.68	0.58
Semiconductor	0.70	0.64	0.57

Table 4.16: AUC rates for transferring knowledge from different domains with different matchings

expected, class based matching and random matching performs worse than RUL matching, suggesting the informativeness of RUL values.

4.11 Application to a Different Domain

Until this study, we studied on the engine related domain. To assess how the methods that learn common representation engage on a different domain, we select the different finetuning (semiconductor) with the pretraining (hard drive). These pair of datasets has a common domain: the electronics circuitry.

As Table 4.17 and Figure 4.9 points out, CCA changes the AUC little by 4%, 1%, -1% (absolute) on average using finetuning samples 500, 1,000 and 1,200 relatively. Siamese Network with Cosine distance impacts the AUC negatively as -12%, -4%, -5%, Siamese Network with Euclidean distance decreases AUC with -2%, -4%, -2%. Mahalanobis distance changes the AUC by 3%, 5%, -1% (Table 4.12).

Even though the pair of the pretraining and finetuning datasets are from the similar domain, the pretraining feature vector does not consist huge amount of data. Therefore, the methods that learn the common representation are not supporting the transfer learning as expected.

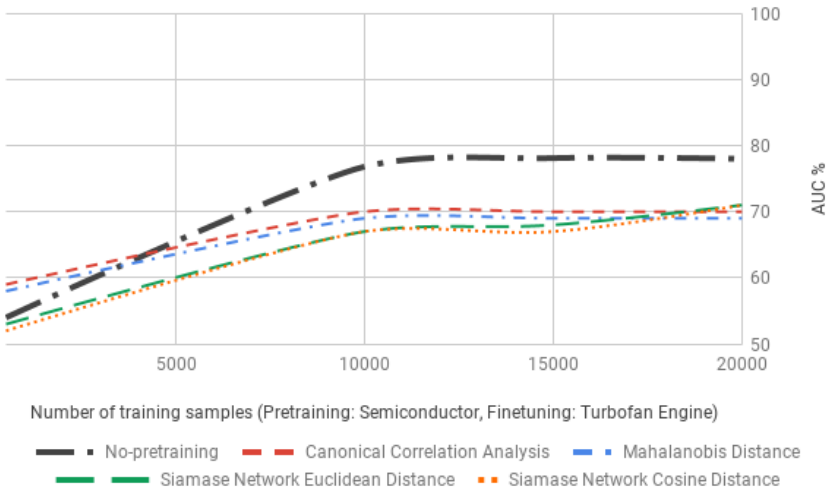

Figure 4.9: The impact of different common representation learnings on transfer learning

	Pretraining: Hard Drive Finetuning: Semiconductor		
	# of Finetuning Samples		
Methods	500	1,000	1,200
CCA	0.93	0.93	0.92
Mahalonobis Distance	0.92	0.93	0.92
Siamese Network, Euclidean Distance	0.87	0.90	0.91
Siamese Network, Cosine Distance	0.78	0.88	0.88
n/a	0.89	0.92	0.93

Table 4.17: AUC rates for using different methods for learning a common representation using different number of finetuning samples

4.12 Comparison to Other Methods

To evaluate the overall performance of our recommended method for early diagnosis of breakdown, we compare it to three state-of-the-art methods, namely [19], [20], and [21] which report results on the turbofan engine dataset. Notice that the reported accuracy and AUC of our method is obtained with a pre-training on the wind turbine dataset, and finetuning the model using all the training samples of the turbofan engine dataset.

[19] propose an Extreme learning machine using quantum-behaved particle swarm optimization (Q-ELM) for turbine fan engine fault diagnosis. [20] use Support Vector Machines (SVM) to model faults while employing a multilayer perceptron (MLP) to evaluate the quantity of the faults. In [21], Real-Time Adaptive Performance Model (RTAPM) Kalman filter is performed to evaluate dynamic engine states.

Method	AUC	Accuracy
Proposed Method	0.94	0.98
[19]	-	0.93
[20]	-	0.97
[21]	0.81	-

Table 4.18: Comparison to other methods on the turbofan engine dataset

As shown in Table 4.18, our proposed method reaches an accuracy of 98%, while that of the best performing competitor [20] is 96.8%. [19] and [21] perform significantly worse than the proposed method.

Chapter 5

Conclusion

A significant challenge for predicting a breakdown using a classification model built by a machine learning algorithm, is the lack of a sufficient amount of training data. To cope with this issue, we exploit prior knowledge obtained from diverse but approximately related tasks for the breakdown classification. To this end, a common representation space is learned using CCA, and in this learned representation space, transfer learning is applied.

We have assessed the reliability of our method through extensive experiments. Our experimental results show that transfer learning from another database with different/uncalibrated measures, can be effectively applied to the task of breakdown detection by learning a common representation space. Therefore, the data insufficiency problem can be handled. In order for transfer learning to be successful, the source and the target dataset must have at least one common feature.

We have showed that our proposed architecture improves the state of the art in the area. For example, In terms of AUC, the model obtained by transfer learning achieved 94%, where as the Kalman Filter model proposed by [21] reached 81% only. The model constructed by transfer learning approach for the turbofan engine breakdown classification task can achieve an accuracy of 98%, while the next best performing competitor offering the Support Vector Machines (SVM)

with a multilayer perceptron (MLP) [20] attained 96.8%.

We evaluated the proposed method with different data pairs that have different or similar features, our study can be applied to other breakdown prediction problems such as electronics circuitry, transportation vehicles, production lines, as long as large amount of training data from the source domain is available. As a future work, we will spread our recommended method for regression problems such as the remaining useful life prediction problem.

Bibliography

[1] Gian Antonio Susto, Andrea Schirru, Simone Pampuri, Seán McLoone, and Alessandro Beghi. Machine learning for predictive maintenance: A multiple classifier approach. *IEEE Transactions on Industrial Informatics*, 11(3):812–820, 2015.

[2] Kåre Hartlapp Lærum. A study of machine learning for predictive maintenance-a topic and programming guidance. Master's thesis, NTNU, 2018.

[3] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[5] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[6] ID Nagtegaal and JHJM Van Krieken. The role of pathologists in the quality control of diagnosis and treatment of rectal cancer—an overview. *European Journal of Cancer*, 38(7):964–972, 2002.

[7] Lixin Duan, Dong Xu, and Ivor Tsang. Learning with augmented features for heterogeneous domain adaptation. *arXiv preprint arXiv:1206.4660*, 2012.

[8] Yin Zhu, Yuqiang Chen, Zhongqi Lu, Sinno Jialin Pan, Gui-Rong Xue, Yong Yu, and Qiang Yang. Heterogeneous transfer learning for image classification. In *AAAI*, 2011.

[9] Leandro L Minku and Xin Yao. How to make best use of cross-company data in software effort estimation? In *Proceedings of the 36th International Conference on Software Engineering*, pages 446–456. ACM, 2014.

[10] John Cristian Borges Gamboa. Deep learning for time-series analysis. *arXiv preprint arXiv:1701.01887*, 2017.

[11] Vincent Vercruyssen, Wannes Meert, and Jesse Davis. Transfer learning for time series anomaly detection. In *IAL@ PKDD/ECML*, pages 27–36, 2017.

[12] Lindasalwa Muda, Mumtaj Begam, and Irraivan Elamvazuthi. Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques. *arXiv preprint arXiv:1003.4083*, 2010.

[13] Qinghua Hu, Rujia Zhang, and Yucan Zhou. Transfer learning for short-term wind speed prediction with deep neural networks. *Renewable Energy*, 85:83–95, 2016.

[14] Yong Yao, Honglei Wang, Shaobo Li, Zhonghao Liu, Gui Gui, Yabo Dan, and Jianjun Hu. End-to-end convolutional neural network model for gear fault diagnosis based on sound signals. *Applied Sciences*, 8(9):1584, 2018.

[15] Preethi Anantharaman, Mu Qiao, and Divyesh Jadav. Large scale predictive analytics for hard disk remaining useful life estimation. In *2018 IEEE International Congress on Big Data (BigData Congress)*, pages 251–254. IEEE, 2018.

[16] Ya Song, Guo Shi, Leyi Chen, Xinpei Huang, and Tangbin Xia. Remaining useful life prediction of turbofan engine using hybrid model based on autoencoder and bidirectional long short-term memory. *Journal of Shanghai Jiaotong University (Science)*, 23(1):85–94, 2018.

[17] Microsoft Azure. Deep learning for predictive maintenance with Long Short Term Memory Networks, year = 2017, url = https://azure.microsoft.com/tr-tr/blog/deep-learning-for-predictive-maintenance/, urldate = 2017-06-21.

[18] Linxia Liao and Hyung-il Ahn. Combining deep learning and survival analysis for asset health management. *International Journal of Prognostics and Health Management*, 2016.

[19] Xinyi Yang, Shan Pang, Wei Shen, Xuesen Lin, Keyi Jiang, and Yonghua Wang. Aero engine fault diagnosis using an optimized extreme learning machine. *International Journal of Aerospace Engineering*, 2016, 2016.

[20] Amare Desalegn Fentaye, Syed Ihtsham Ul-Haq Gilani, Aklilu Tesfamichael Baheta, and Yi-Guang Li. Performance-based fault diagnosis of a gas turbine engine using an integrated support vector machine and artificial neural network method. *Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy*, page 0957650918812510, 2018.

[21] Jeffrey Armstrong and Donald Simon. Implementation of an integrated on-board aircraft engine diagnostic architecture. In *47th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit*, page 5859, 2011.

[22] Ansi Zhang, Honglei Wang, Shaobo Li, Yuxin Cui, Zhonghao Liu, Guanci Yang, and Jianjun Hu. Transfer learning with deep recurrent neural networks for remaining useful life estimation. *Applied Sciences*, 8(12):2416, 2018.

[23] Bo Zhang and Zhong-Zhi Shi. Classification of big velocity data via cross-domain canonical correlation analysis. In *2013 IEEE International Conference on Big Data*, pages 493–498. IEEE, 2013.

[24] MARCO MICHAEL Rigamonti, Piero Baraldi, Enrico Zio, et al. Echo state network for the remaining useful life prediction of a turbofan engine. In *annual conference of the prognostics and health management society 2015*, pages 255–270, 2016.

[25] G Sateesh Babu, Xiao-Li Li, and Sundaram Suresh. Meta-cognitive regression neural network for function approximation: Application to remaining useful life estimation. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 4803–4810. IEEE, 2016.

[26] Shi Zhong Zhang Bo, Zhao Xiao Zhi, et al. A transfer learning based on canonical correlation analysis across different domains. *Chinese Journal of Computers*, 38(7):1326, 2015.

[27] Harold Hotelling. Relation between two sets of variates. *Biometrica*, 1936.

[28] Magnus Borga. Canonical correlation: a tutorial. *On line tutorial http://people. imt. liu. se/magnus/cca*, 4(5), 2001.

[29] Yonghui Xu, Sinno Jialin Pan, Hui Xiong, Qingyao Wu, Ronghua Luo, Huaqing Min, and Hengjie Song. A unified framework for metric transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 29(6):1158–1171, 2017.

[30] Hao Wang, Wei Wang, Chen Zhang, and Fanjiang Xu. Cross-domain metric learning based on information theory. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[31] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *Advances in neural information processing systems*, pages 737–744, 1994.

[32] Young-Bum Kim, Dongchan Kim, Anjishnu Kumar, and Ruhi Sarikaya. Efficient large-scale neural domain classification with personalized attention. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2214–2224, 2018.

[33] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[34] Yuting Wu, Mei Yuan, Shaopeng Dong, Li Lin, and Yingqi Liu. Remaining useful life estimation of engineered systems using vanilla lstm neural networks. *Neurocomputing*, 275:167–179, 2018.

[35] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. 1999.

[36] Christopher Olah. Understanding lstm networks. 2015.

[37] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[38] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big Data*, 3(1):9, 2016.

[39] A Saxena and K Goebel. Turbofan engine degradation simulation data set. *NASA Ames Prognostics Data Repository*, 2008.

[40] Kevin Leahy, R Lily Hu, Ioannis C Konstantakopoulos, Costas J Spanos, and Alice M Agogino. Diagnosing wind turbine faults using machine learning techniques applied to operational data. In *2016 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pages 1–8. IEEE, 2016.

[41] A Agogino and K Goebel. Mill data set. best lab, uc berkeley. nasa ames prognostics data repository, 2007.

[42] Amit Kumar Jain and Bhupesh Kumar Lad. Data driven models for prognostics of high speed milling cutters. *International Journal of Performability Engineering*, 12(1):3–12, 2016.

[43] Chang Xu, Gang Wang, Xiaoguang Liu, Dongdong Guo, and Tie-Yan Liu. Health status assessment and failure prediction for hard drives with recurrent neural networks. *IEEE Transactions on Computers*, 65(11):3502–3508, 2016.

[44] Sathyan Munirathinam and Balakrishnan Ramadoss. Predictive models for equipment fault detection in the semiconductor manufacturing process. *IACSIT International Journal of Engineering and Technology*, 8(4):273–285, 2016.