Faculty of Engineering and Information Technology
University of Technology Sydney

Big Data Analytics for Condition Based Monitoring and Maintenance

A thesis submitted in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

by

Zhibin Li

December 2020

CERTIFICATE OF ORIGINAL AUTHORSHIP

I, Zhibin Li declare that this thesis, is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the School of Electrical and Data Engineering, Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

Signature of Candidate

Production Note:
Signature removed prior to publication.

Acknowledgments

Foremost, I would like to express my sincere gratitude to my principal supervisor, Associate Prof. Jian Zhang, for his professional guidance, persistent help and continuous support throughout my PhD study and research.

I also deeply appreciate my co-supervisor Associate Prof. Qiang Wu, for his help and guidance in both research and living in Sydney.

Besides, I offer my regards and blessings to all of my co-workers at the lab: Yongshun Gong, Lu Zhang, Yazhou Yao, Xiaoshui Huang, Muming Zhao, Junjie Zhang, Huaxi Huang, Anan Du, Lingxiang Yao, Guofeng Mei and Litao Yu. I really enjoyed the time we spent together. I would also like to express my gratitude and appreciation to Lu Zhang for her help in proofreading this thesis, and her encouragement and support during my PhD journey.

Finally and most essentially, I would like to thank my parents. Without their encouragement, finishing this dissertation would be impossible; without them, nothing would have any value.

Zhibin Li
August 2020 @ UTS

Contents

List of Figures

List of Tables

List of Publications

Papers Published

- **Zhibin Li**, Jian Zhang, Yongshun Gong, Yazhou Yao, and Qiang Wu. *Field-wise Learning for Multi-field Categorical Data*, Advances in Neural Information Processing Systems, 2020.

- **Zhibin Li**, Jian Zhang, Qiang Wu, Yongshun Gong, Jinfeng Yi, and Christina Kirsch. *Sample adaptive multiple kernel learning for failure prediction of railway points*, In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2848–2856, 2019.

- **Zhibin Li**, Jian Zhang, Qiang Wu and Christina Kirsch. *Field-regularized Factorization Machines for Mining the Maintenance Logs of Equipment*, The 31st Australasian Joint Conference on Artificial Intelligence, 2018.

- Yongshun Gong, **Zhibin Li**, Jian Zhang, Wei Liu and Yu Zheng. *Online Spatio-temporal Crowd Flow Distribution Prediction for Complex Metro System*, IEEE Transactions on Knowledge and Data Engineering, 2020.

- Yongshun Gong, **Zhibin Li**, Jian Zhang, Wei Liu, and Jinfeng Yi. *Potential passenger flow prediction: A novel study for urban transportation development*. In Proceedings of the 34th AAAI Conference on Artificial Intelligence, pages 4020–4027, 2020.

- Yongshun Gong, **Zhibin Li**, Jian Zhang, Wei Liu, Yu Zheng, and Christina Kirsch. *Network-wide crowd flow prediction of Sydney trains via customized online non-negative matrix factorization.* In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pages 1243–1252, 2018.

- Yongshun Gong, **Zhibin Li**, Jian Zhang, Wei Liu, Bei Chen and Xiangjun Dong. *A Spatial Missing Value Imputation Method for Multiview Urban Statistical Data.* In Proceedings of the 29th International Joint Conference on Artificial Intelligence, pages 1310-1316, 2020.

- Lu Zhang, Jian Zhang, **Zhibin Li**, and Jingsong Xu. *Towards Better Graph Representation: Two-branch Collaborative Graph Neural Networks for Multimodal Marketing Intention Detection.* IEEE International Conference on Multimedia & Expo. 2020.

Abstract

Condition-based Maintenance (CBM) will significantly achieve the cost-saving while monitoring the related infrastructure through the most accurate maintenance scheduling. It also increases the reliability of monitored equipment. For example, in the field of rail transport, it helps ensure trains run on time and plays a critical role in the safety of railway operation. A key prerequisite for CBM is accurate fault prediction, which can be achieved through predictive machine learning models. Although artificial intelligence and machine learning have become successes in many applications, their potentials in CBM have not been fully recognised. The growing scale and modality of railway data bring opportunities as well as challenges to machine learning models. In this thesis, three key challenges were abstracted with regard to data analytics using machine learning technics for fault prediction, resulting from the sparse high-dimensional data, the incomplete data, and the multi-source data. Then the three challenges were studied from an algorithmic point of view.

The sparse high-dimensional data commonly exist in maintenance logs, in a format of categorical variables. Normally, a sophisticated feature engineering process is required to extract the complex feature-interactions, while the high dimensionality, sparseness, and the lack of reliable domain knowledge make this process quite ad-hoc and subject to strong personal opinion/experience of each individual engineer. This thesis proposed field-regularised factorisation machines to learn the complex feature-interactions automatically from such data and evaluated the proposed method with main-

tenance logs of railway points in a railway network. Another challenge comes with the fact that real-world data are usually incomplete due to various reasons, e.g., faults in the database, operational errors or transmission faults. To address these issues, this thesis proposed a missingness-pattern-adaptive model, which adaptively adjusts the predictive function for incomplete data. Some theoretical evidence was provided to support the correctness of our model. This model was tested with several public datasets with internal missing values. Generally, the predictive task for CBM can involve data from multiple sources, such as weather conditions, sensors, and maintenance logs. For the multi-source data, this thesis proposed a sample-adaptive multiple-kernel learning algorithm to facilitate the fusion of data for the predictive task. To verify the effectiveness of this method, experiments were conducted on real-life data generated by a large-scale railway network.

Chapter 1

Introduction

1.1 Background

Condition-based maintenance (CBM) is a maintenance strategy that through **monitoring** the actual condition of the asset to decide what maintenance needs to be done. The goal of condition-based maintenance is to spot upcoming equipment failures, so maintenance can be pro-actively scheduled when it is needed [1]. Traditionally, maintenance actions are based on a fixed time interval or raised after equipment failures. Instead, it would be of great value if we could predict failures and take action beforehand, minimising any negative effects. As mentioned in (Núñez, Hendriks, Li, De Schutter & Dollevoet 2014), currently, a huge amount of railway track condition-monitoring data is being collected from different sources in different countries. However, the data are not yet fully used because of the lack of suitable techniques to extract relevant events and crucial historical information. Valuable information is hidden behind a huge amount of data from different sources.

Data analysis for CBM will mainly be based on condition-monitoring data collected in two ways:

- **On-site inspection**: In this way, data are collected by field engi-

[1] https://www.fiixsoftware.com/condition-based-maintenance/

neers. They record the status of the equipment through on-site test and inspection. Portable instruments can also be utilised to test the equipment.

- **Sensors reading**: Some equipment or its nearby environments are equipped with sensors. Readings of sensors provide valuable data that can reflect the real-time status of the equipment.

Data collected through on-site inspection include maintenance logs, where the maintenance are recorded in detail. They can be of great value in fault prediction. Predictions are mostly provided by domain experts with hand-craft features and thus subject to strong personal experience. Typically, maintenance logs of equipment contain formatted maintenance records, including maintenance type, components, finished time, etc. These data often carry information about equipment status with timestamps. A piece of equipment can consist of many components, and failures can be a result of their interactions. Domain knowledge regarding such interactions might be limited and vary over equipment types. These factors make it difficult to hand-craft effective features, even for those most experienced experts. Therefore, automatically extracting critical features from data, although it is significantly challenging, is imperative and will address the lack of sufficient reliable prior knowledge.

Sensors reading is another way to gather information from equipment. Real-time statistics of equipment, such as voltage, current and temperature are precious data for accurate fault prediction. However, installation of sensors incurs costly labour and material expenses, as well as the possibility of sensor malfunction. Adding sensors for in-service equipment would also induce disruption to a related system. This is especially unacceptable for equipment in a large and busy network. Thus, the prediction with sensors reading can be expensive, or sometimes infeasible. We can not expect every piece of equipment is equipped with sensors, but normally the critical components of an infrastructure system will be installed with sensors.

Big data. With the growing scale of infrastructure systems, as well as the development of condition monitoring technologies, the data available for CBM have become unprecedentedly large. For example, in Sydney Trains, the operation logs for a single type of equipment can include tens of millions of entries. Zhai, Ong & Tsang summarised the challenges as 5Vs for big data analytics. For the condition monitoring data analytics, they can be:

- **Volume**: For monitoring an infrastructure system, e.g. a railway network, 100 terabyte data can be generated per day from only one data source, because of the high sampling rate of sensors. (Núñez et al. 2014).

- **Velocity**: For modern condition monitoring, daily or weekly data acquisition is necessary, while the large volume of data requires computational intelligence for timely and effective processing of the available data.

- **Variety**: The condition monitoring data can be collected from multiple sources with different data-collecting systems, leading to a variety of feature representations.

- **Veracity**: Missing attributes or incomplete data are pervasive in condition monitoring data. The data can also be collected with diverse quality for multiple sources. In many cases, the data can be noisy.

- **Value**: Refers to the benefits that can be gained from analysis on condition monitoring data. Reducing cost and increasing system reliability are the two most important targets of data analytics.

Given the importance of CBM and available data, in this thesis, we mainly study the fault prediction for CBM with machine learning models. The thesis developed several machine learning models focusing on the characteristics of condition-based monitoring data. Specifically, some case studies were conducted on railway points, which are a kind of mechanical installations allowing railway trains to be guided from one track to another.

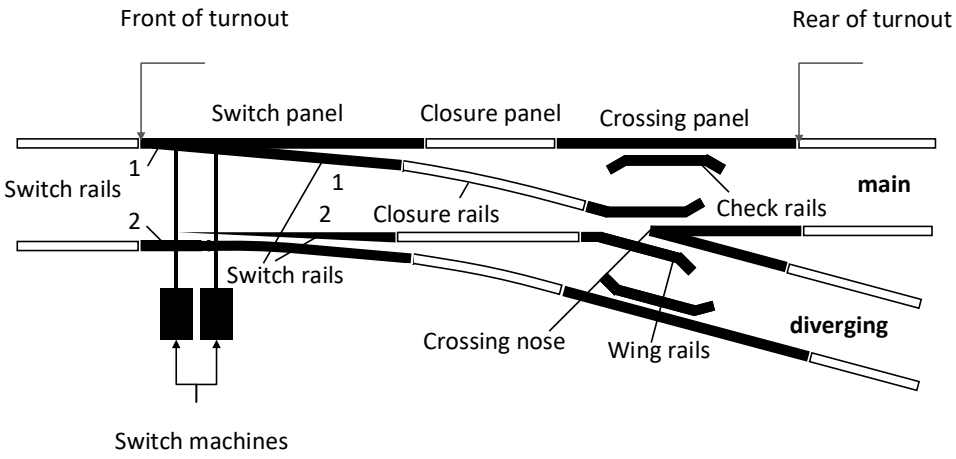


Figure 1.1: Illustration of a type of railway points.

Railway points are among the key components of the railway infrastructure, having a great impact on the reliability and punctuality of rail transport. Figure 1.1 illustrates a type of railway points. Points failures constituted the second largest number of Incident Information Management System entries in Sydney Trains, indicating that they caused second-most train delays. Sydney Trains stores some large-scale datasets including maintenance and operation logs. Those data are accumulating with a growing speed, because of the expanding railway network and installation of more and more field sensors. The big data bring an opportunity to apply Artificial Intelligence for safer and more economical railway operation yet have not been adequately exploited. Thesis, therefore, chose this vulnerable equipment for our case studies and explored the feasibility of predicting their failures with machine learning models.

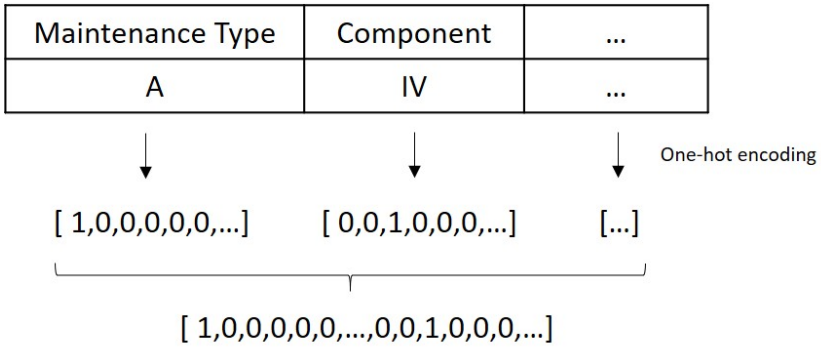


Figure 1.2: An example of sparse high-dimensional data generated from a piece of maintenance log.

1.2 Research Challenges

In this thesis, we focused on three research challenges regarding failure prediction for CBM, resulting from the sparse high-dimensional data, the incomplete data, and the multi-source data.

1.2.1 Sparse High-dimensional Data

The sparse high-dimensional data commonly exist in maintenance logs, in the format of categorical variables, as illustrated in Figure 1.2. The sparseness and high-dimensionality of data bring difficulties to feature engineering. In a large-scale infrastructure system, condition monitoring data can be collected from the equipment of different types located in a wide range, so domain knowledge regarding such data is quite limited. This makes it almost impossible to manually extract effective features for every set of equipment. On the other hand, the sparseness and high-dimensionality also hinder the learning of machine learning models. Therefore, designing effective models to automatically extract the effective features from sparse high-dimensional data is an imperative while challenging task for failure prediction.

1.2.2 Incomplete Data

Real-world data are usually incomplete due to various reasons, e.g., faults in the database, operational errors or transmission faults. Sometimes the data formats are not strictly defined, and this could also result in missing entries in a systematic way. For example, when inspecting a set of equipment, the field engineers may find some defects that are not properly defined in the data system, causing some "Not Applicable" entries which are actually missing entries. Condition monitoring data on a network-wide level further intensify the incomplete data problem, because for some sets of equipment only a subset of features is available. This can happen when sensors are only installed on critical equipment. If we collectively train a model using all the data, then the unavailable feature sets for them would be treated like missing. Clearly, developing a model that can directly handle incomplete data and identify the hidden information behind missingness patterns will be a challenge while it is necessary for failure prediction.

1.2.3 Multi-source Data

Data available for CBM are usually from multiple sources. For example, maintenance logs and real-time operation logs can be generated from two different sources. Even more data sources can join the failure prediction task, as illustrated in Figure 1.3. Multi-source data bring benefits as well as challenges for failure prediction. The multiple data channels can be complementary to each other so that they can enhance the performance of related models, but leveraging such complementary information requires designing novel models with CBM data. The multi-source data can be noisy, and in most cases, data sources are not of equal importance for failure prediction. The importance of each data source can vary with different sets of equipment. Those data are often produced in heterogeneous formats with varying frequencies. Besides, the incomplete data problem is more complicated considering multiple data channels. Devising an optimal data fusion scheme is

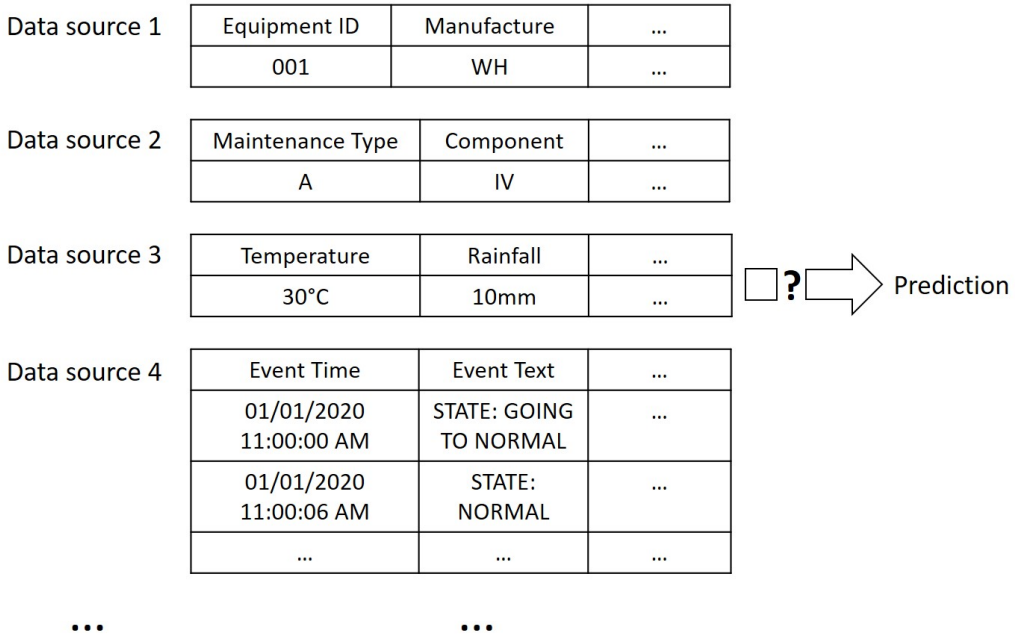


Figure 1.3: An example of multi-source data for failure prediction of railway points.

challenging but can greatly facilitate the CBM-related tasks.

1.2.4 Summary

In short, our three research challenges will be 1) how to design effective models for sparse high-dimensional condition monitoring data, 2) how to design a model that can directly handle incomplete condition monitoring data for fault prediction, and 3) how to design a model to combine condition monitoring data from multiple sources. The three challenges are not separate, and they can interplay with each other.

1.3 Research Contributions

Focused on above-mentioned research issues, this thesis developed several practical machine learning models related to failure prediction for CBM. In

detail, we highlight the main contribution of this work as follows:

- The field-regularised factorisation machines for sparse high-dimensional data were proposed. The method was able to leverage the field information to help the learning of feature-interactions for failure prediction. Effective feature-interactions were automatically learned from data without the complicated feature engineering process. Experiments were conducted based on real-world maintenance data collected from a large-scale railway network. The experiment results showed that our model outperforms other competitive baselines. (Chapter 3)

- A missingness-pattern-adaptive model for incomplete data were proposed. The model was allowed to be adaptively adjusted for different missingness patterns, and it could be trained even when all data were incomplete. It was also extended to the non-linear case with neural networks as the backbone. An efficient optimisation algorithm was designed for the linear model based on the restart subgradient method and proved its rate of convergence. This thesis also theoretically proved the generalisation error of this model. Experiments were conducted on several public datasets to validate the advantages of this model over the state-of-the-arts. (Chapter 4)

- A sample-adaptive multiple-kernel learning algorithm for learning with multi-source data was designed. It enabled us to find an optimal combination scheme for data from multiple sources. In the meantime, the model was also allowed to be equipment-specific, so that it could be flexibly adjusted to fit each piece of equipment. The model was tested with condition monitoring data collected from a large-scale railway network. The data used included maintenance logs, movement logs, weather and equipment details. Experiment results confirmed the superiority of multi-source data over the single source and also demonstrated that this method was better on two considered evaluation metrics compared to some state-of-the-arts. (Chapter 5)

- Detailed empirical studies were performed on failure prediction of railway points based on a large-scale railway network. Some practical experience was provided in the data pre-processing and feature extraction steps. Several existing machine learning models were also explored for this task with extensive experiments (Chapter 3 and Chapter 5), which exhibited the potential of using machine learning models in CBM-related tasks. Results showed that those predictive models could provide a baseline level of recommendations for maintenance work. The research also indicates that there is a great potential for industrial players to apply machine learning models for cost reduction and reliability enhancement with condition monitoring data.

1.4 Thesis Structure

The rest of this thesis is organised as follows:

We first reviewed the related work for CBM and associated research issues in Chapter 2. Tasks related to CBM were categorised into deterioration modelling, maintenance strategy optimisation and failure prediction. We specifically reviewed some models related to our work.

Chapter 3 introduced the field-regularised factorisation machine model for mining the maintenance logs. Two regularisation terms were introduced to facilitate the learning of Factorisation Machine model, based on Euclidean distance and Cosine distance, respectively. Its performance was tested with maintenance logs of railway points.

The missingness-pattern-adaptive model for incomplete data was presented in Chapter 4, together with its generalisation error bound. We designed its linear version based on max-margin classifier and non-linear version based on neural networks. Experiments were conducted on public datasets to demonstrate the effectiveness of the proposed model.

We formulated the sample-adaptive multiple-kernel learning in Chapter 5. We first proposed a missingness-pattern-adaptive multiple-kernel learning al-

gorithm to adaptively make predictions when samples had absent data channels. It was further refined with sample-adaptive multiple-kernel learning algorithm which was able to distinguish between different sets of equipment for better predictions. Corresponding models were validated with multi-source CBM data collected from a large-scale railway network.

We concluded the thesis with possible future directions in Chapter 6.

Chapter 2

Literature Review

In this chapter, we introduce some related research for CBM, including deterioration modelling, maintenance strategy optimisation and failure prediction. Different from existing research on CBM, which focus more on each individual case, our study will be focused on designing general machine learning models for CBM task. Specifically, this thesis will put emphasis on the failure prediction task and revisit several classes of machine learning models related to our research issues.

2.1 Deterioration Modelling

Deterioration modelling models the long-term deterioration process of equipment. The learned deterioration model can serve as a long-term baseline for CBM. With the estimated deterioration process, maintenance work could be arranged in a most cost-effective way and make related equipment more reliable. Statistical models are prevailing in this field. Different from physical methods, statistical method models the regularities existing behind deterioration processes without clarifying the deterioration mechanism. Therefore, statistical models are useful for modelling the average deterioration at a macroscopic level (Kobayashi, Kaito & Lethanh 2012).

He, Li, Bhattacharjya, Parikh & Hampapur built a track deterioration model to capture the degradation process of different types of geo-defects, and a survival model to assess the dynamic derailment risk as a function of track defect and traffic condition. These models were used to plan track rectification activities with two different objectives: a cost-based formulation and a risk-based formulation. Ohadi & Micic identified that Gamma process represented a very simple and effective method to establish consistent deterioration models for structures that were subject to inspection. They conducted numerical experiments on a circular bar element belonged to a part of bridge deck reinforcement.

Most of the research considered the deterioration process as a multi-state degradation process with discrete-state space. Podofillini, Zio & Vatn proposed a non-homogeneous Markov model for determining the failure probability of a rail section under periodic inspection, where maintenance procedures were also included in the model. Tsuda, Kaito, Aoki & Kobayashi modelled the bridge deterioration process as a Markov chain model, with the transition probabilities described by the exponential hazard models. They first estimated the Markov transition probabilities for the condition states of each bridge component and then using them for estimating the Markov transition probabilities describing the average deterioration process of the bridge a whole. Kobayashi et al. further improved this model with Bayesian estimation method to improve the estimation of Markov transition probability by Markov chain Monte Carlo method using Gibbs sampling. When monitoring data were insufficient at the early stage, prior knowledge provided by domain experts could be incorporated into this model. With data accumulating, the model would be constantly updated. The empirical study was conducted on the reinforced concrete slabs of bridges.

Noticing that in most of the multi-state models, the transitions between states follow an identical type, so that the aging and deterioration of device over time in each state are ignored. In reality, due to technical problems, directly observing the actual health condition of the equipment may not be

possible. In order to address these issues, Moghaddass, Zuo & Zhao proposed to use the non-homogeneous continuous-time hidden semi-Markov process for health monitoring of equipment. Moghaddass & Zuo applied this method to address the practical challenges of online diagnostics and prognostics of mechanical systems under unobservable degradation. The effectiveness of this method was verified with health monitoring data of turbofan engines.

Estimating the remaining useful life of a system is also an important task related to deterioration modelling. Rama & Andrews identified the two-parameter Weibull distribution as the most appropriate distribution to model lifetimes of switches and crossings (S&C) components used in railway. They described the derivation of lifetime distributions of individual S&C components based on field data collected. The method can be used to predict the expected number of maintenance activities, and associated costs for S&C units over any specified period of time. Le Son, Fouladirad & Barros proposed to use the noisy gamma process for estimating the remaining useful life of a system, under the assumption that the observed degradation data were possibly contaminated with Gaussian noise. By using the Gibbs sampling technique, the hidden degradation states were approximated. Then the system's remaining useful lifetime distribution was estimated with the noisy observation. The case study was conducted on the aircraft engine data introduced in (Saxena, Goebel, Simon & Eklund 2008).

2.2 Maintenance Strategy Optimisation

Life-cycle cost minimisation is the main target of maintenance strategy optimisation, with an objective to minimise the system's life cycle cost per unit time that includes maintenance cost, failure loss, and the cost of system unavailability. Maintenance strategy optimisation will usually require and largely depend on prior knowledge regarding the deterioration process of the equipment.

Most of the related approaches designed the maintenance strategy fol-

lowing the cost-risk balance rule. They searched for a balanced strategy which guarantees the lowest cost while keeps the failure probability of equipment below a pre-defined threshold. Mathew & Isaac developed an optimised maintenance strategy for the rural road network of Kerala state. They formulated the problem as a bi-objective deterministic optimisation problem which simultaneously required minimisation of total maintenance cost and maximisation of performance of the road network. A genetic algorithm was utilised to solve this optimisation problem. Podofillini, Zio & Vatn introduced a multi-objective optimisation problem to optimise inspection and maintenance procedures with respect to both economical and safety-related aspects. More precisely, the objective functions were to search for solutions characterised by low expenditures and low derailment probability. Their work was conducted based on a railway system.

In (Zhang & Gao 2012), the life-cycle cost of an infrastructure system was minimised under the constraint of the cumulative failure rate. This model was applied to the maintenance of bridge decks, where Weibull distribution was used to model the deterioration process of the bridge deck. Caetano & Teixeira considered a combination of maintenance work that integrates ballast, rail and sleeper degradation models in a mixed-integer linear programming model. They assumed that integrated work would be cheaper than performing them separately. Weibull distribution was also adopted for modelling the probability distribution that describes the failure occurrence. Shafiee, Patriksson & Chukova firstly combined age and usage thresholds in the optimisation of maintenance strategy. An optimal bi-variate (age-usage) maintenance strategy for railway tracks was proposed such that the average long-run maintenance cost per unit time was minimised. They performed a case study on a small part of the Swedish railroad. Park, Jung & Yum considered the situation where each preventive maintenance relieved stress temporarily and hence slowed the rate of system degradation, while the hazard rate of the system remained monotonically increasing. The minimal repair cost also varied with time. Their model was able to get the correspond-

ing optimal number and period for the periodic preventive maintenance that minimised the expected cost rate per unit time over an infinite time span.

Maintenance operations planning of railway infrastructures was formulated as model predictive control problems in (Su, Núñez, Jamshidi, Baldi, Li, Dollevoet & De Schutter 2015, Su, Núñez, Baldi & De Schutter 2016). They sought for a balanced plan between rail quality and maintenance cost. Su et al. developed a decision-making method for optimal planning of railway maintenance operations using hybrid model predictive control. The hybrid characteristics arose from the three possible control actions: performing no maintenance, performing corrective maintenance, or doing a replacement. They used a linear dynamic model to describe the evolution of the health condition of a railway track segment. Based on this method, Su et al. further developed A multi-level decision-making approach for condition-based maintenance of rail. The high-level decision-making problem was to produce the optimal long-term coarse-grained maintenance plan for the entire track, and the low-level problem was to produce a fine-grained efficient schedule to execute the actions suggested by the high-level controller. In summary, the factors considered in maintenance strategy optimisation task vary case by case, while all the methods highly rely on accurate modelling of failure probability of equipment.

2.3 Failure Prediction

Failure prediction is the key step for CBM. Different from deterioration modelling, failure prediction will mostly focus on the short-term behaviour of equipment and is more sensitive to accidental events.

Wang, Xu, Wang & Zou focused on the prediction of weather-related failure of railway points. Their target was to predict the total number of failures in a large railway system. They used a modified AdaBoost.RT algorithm (Kankanala, Das & Pahwa 2013) with support vector machines as the weak learner to predict the number of turnout failures in a weekly manner.

The proposed approach can help railway corporations make a better plan of maintenance activities and has the potential to reduce spare inventories as well as repair/maintenance cost. A weakness of their method is that it could not determine the source of failures. In other words, only the total number of failures are predicted rather than the exact location of every failure.

Axle box acceleration (ABA) measurements were used in (Núñez et al. 2014) to detect surface defects (squats) and corrugation of railway track. The energy values measured using the ABA system were analysed to reveal the defects. Molodova, Li, Nunez & Dollevoet also utilised ABA measurements, but they analysed these measurements in the frequency domain.

Yilboga, Eker, Güçlü & Camci proposed to predict failure of railway turnouts with Time-delay neural networks. Force signals from an electro-mechanical railway turnout system were collected as input. In order to predict failure of point machines, current signal and movement duration of point machines were approximated by harmonic regression and vector auto-regressive moving-average model respectively in (García, Pedregal & Roberts 2010). Switch position sensors were used to record the duration of points machine movement. García Márquez, Roberts & Tobias reviewed some fault detection and diagnosis methods for railway points with data collected from line-side equipment and lab-based test rigs.

2.3.1 Failure Prediction with Sparse High-dimensional Data

Log files of equipment mainly consist of categorical variables. For example, the categorical variables in the event log include pre-defined events. Log files can be generated by sensors, software applications and maintenance records, reflecting the condition of associated equipment. Depending on the number of possible values of categorical variables, the generated feature vectors can be very high-dimensional and sparse. We will mainly review some work focused on the log files.

Liang, Zhang, Xiong & Sahoo proposed a customised nearest neighbour approach to predict the failure events of an IBM supercomputer. Their method requires strong domain knowledge to extract some critical features from the event logs. They firstly eliminated the redundancy of the event logs using the adaptive semantic filter proposed in (Liang, Zhang, Xiong & Sahoo 2007*a*). Then they identified several raw features from those events for subsequent finer feature design. A rule-based classifier, support vector machines and a traditional nearest neighbour method were compared to the proposed method. Their method can achieve an F measure of 70% and 50% for a 12-hour and 6-hour prediction window size. Salfner & Tschirpke also emphasised the importance of data preparation with prior knowledge. They provided three algorithms: (a) assignment of error IDs to error messages based on Levenshtein's edit distance, (b) a clustering approach to group similar error sequences, and (c) a statistical noise filtering algorithm for accurate error-based online failure prediction of a commercial telecommunication system. Zheng, Lan, Park & Geist proposed a system log pre-processing method consisted of three integrated steps: (1) event categorisation; (2) event filtering; (3) causality-related filtering. Their approach was applied to failure prediction of supercomputer systems. Wang, Li, Han, Sarkar & Zhou presented a general classification-based failure prediction method which can leverage system messages, error events, or log files for failure prediction. They systematically defined four categories of features then used feature selection to identify the most important features for model construction.

Sipos, Fradkin, Moerchen & Wang presented a data-driven approach based on multiple-instance learning for predicting equipment failures by mining equipment event logs. They conducted experiments on real-life datasets with billions of log messages from two large fleets of medical equipment. The proposed method has been deployed by a major medical device provider to monitor thousands of medical scanners. Fronza, Sillitti, Succi, Terho & Vlasenko used log files to predict failures of software systems. Random Indexing was applied to represent sequences, where each operation was char-

acterised in terms of its context. Weighted support vector machines were applied to deal with imbalanced datasets and to improve the true positive rate. Their method was verified with log files collected during three months of work in a large European manufacturing company.

Most of the related approaches relied on hand-craft features and were either restricted to linear model or low-dimensional features. How to accurately predict failures with least prior knowledge and high-dimensional features is an open problem.

2.3.2 Failure Prediction with Incomplete Data

Missing data are a pervasive problem in CBM related tasks. Twala proposed a probabilistic approach for the classification of incomplete data, which was then used to predict robot execution failure. The main limitation of their work is that they required complete training data, whereas in practice it is common that the training data are incomplete as well. Dhlamini, Nelwamondo & Marwala used neural networks with particle swarm optimisation (PSO) and genetic algorithms (GA) to compensate for missing data in classifying high voltage bushings. Specifically, the autoencoder was used to alleviate the influence of missing data.

Suh, Woodbridge, Lan, Bui, Evangelista & Sarrafzadeh exploit several machine learning models to impute congestive heart failure (CHF) data for patient monitoring systems. Although their research was based on human beings, it was similar to research on infrastructure monitoring systems. The proposed projection adjustment by contribution estimation regression (PACE) method enhanced the accuracy of the CHF missing data on predicting and imputing non-binomial data. For binomial data, they adopted Bayesian methods and voting feature interval algorithms.

Li, Khoo & Tor proposed a two-stage data mining technique for condition-based fault diagnosis. They firstly imputed the missing variables and then extracted associated rules for fault prediction. The missing entries were imputed with all possible observations, and this generates many artificial

samples. A rough set approach was used to extract a minimal diagnostic rule set for condition-based fault diagnosis. A prototype condition-based fault diagnosis system was then built upon this set. The capability of the system was verified through a case study on a centrifugal pump system for a refinery.

It is surprising that research regarding incomplete CBM data is not as pervasive as the problem itself. Intuitively, the absent data can be filled with mean or zero and fed to downstream models as usual, while this model-agnostic strategy can hinder the performance of the model. Most of the practical solutions on failure prediction with incomplete CBM data depends on sophisticated imputation steps, so they are restricted to related applications only and lack of theoretical support.

2.3.3 Failure Prediction with Multi-source Data

Gathering available data from multiple sources enriches our knowledge on the working status of related equipment, but combining information from multiple sources efficiently and effectively is a challenging task.

More often than not, multi-source data can be incomplete and noisy. Undoubtedly, optimising the data fusion scheme will improve the accuracy of failure prediction. Several papers have put emphasis on combining multi-source data (Li, Huang, Li, Zhou & Mi 2018, Ortiz, Babbar, Syrmos, Clark, Vian & Arita 2008, Guo, Sun, Li & Tang 2019, Kabir, Demissie, Sadiq & Tesfamariam 2015). These papers highlighted the value of multi-source data through many different applications.

Li, Huang, Li, Zhou & Mi incorporated two information sources, simulation and lifetime data, to predict the reliability of a turbine blade. A linear fusion scheme was adopted to extract and integrate information from multiple sources. In Guo et al. the weights of different data sources using for multi-source reliability data were decomposed into subjective weight, objective weight, and comprehensive weight, which were based on the experts' judgement. Peng, Li, Li, Jiang & Zhang proposed a systematic method for

evaluating the slope safety utilising multi-source monitoring information. A Bayesian network was constructed to fusion data from multiple sources. A Bayesian belief network based data fusion model was developed in (Kabir et al. 2015) for the failure prediction of water mains. Most of the research was limited to simple cases like each data source contains only one feature, or only several similar data sources were presented.

Methods involving multi-source data are sometimes termed multi-view learning (Zhao, Xie, Xu & Sun 2017). Related works on this topic including multi-view subspace learning (White, Zhang, Schuurmans & Yu 2012, Xu, Tao & Xu 2015*b*), incomplete multi-view learning (Xu et al. 2015*b*, Liu, Zhu, Li, Tang, Zhu, Yin & Gao 2019), and multiple kernel learning (Gönen & Alpaydın 2011). Compared to their popularity in the machine learning community, their practical usage with condition monitoring data hasn't been fully investigated.

2.4 Related Models

Other than designing case-specific models, our study was focused more on designing general machine learning models for CBM tasks. Therefore, we revisit several classes of machine learning models, based on which our models were developed.

2.4.1 Factorisation Machines

The most common sparse high-dimensional data are in the format of categorical variables. Categorical variables are usually converted to binary features through one-hot encoding. The data after one-hot encoding might be very sparse and of very high dimension. For example, there can be over ten million features in datasets of Kaggle challenge on click-through rate prediction. Factorisation Machine (FM) model is specifically designed for such sparse data. It is widely used in Click-through rate prediction and recommendation systems. FMs combine the advantages of support vector machines with

factorisation models (Rendle 2010). In contrast to support vector machines, FMs factorise all interactions between variables into products of two low-rank matrices. In this way, they are able to learn interactions which even do not appear in the training data.

Many variants of FMs have been proposed and achieved promising performance. Locally Linear Factorisation Machines (Liu, Zhang, Zhao, Zhou & Sun 2017) adopted locally linear coding scheme and jointly optimised the FM model with anchor points. They were capable of learning complex non-linear data by exploring the local coding technique. Wang, Zhou, Fei, Chang & Liu proposed Contextual and Position-Aware Factorisation Machines targeting at sentiment analysis of the text. Inspired by neural skip-gram model, Contextual and Position-Aware Factorisation Machines limited interactions to a range of words. In addition, latent vectors were learned based on the relative position of words. This means that there would be several independent latent vectors for one word. The optimisation problems associated with FMs are non-convex in its original form. Yamada, Lian, Goyal, Chen, Wimalawarne, Khan, Kaski, Mamitsuka & Chang reformulated the optimisation problem of FMs as a semi-definite programming problem. By introducing the nuclear norm in FMs, their loss function of FMs became convex.

Above-mentioned models focus less on the inherent properties of data carried by field information. Implementation of FMs for categorical variables requires that data being one-hot encoded. Then FMs treat each feature equally disregard their field information. Field-aware factorisation machines (FFMs) (Juan, Zhuang, Chin & Lin 2016) considered the field structure of data and learned pairwise interactions with regard to each pair of fields. They were more complex than FMs in terms of the number of parameters and computational complexity. Field-weighted Factorisation Machines (Pan, Xu, Ruiz, Zhao, Pan, Sun & Lu 2018) added additional coefficients to depict the interactions of fields, and reduced the number of model parameters compared to FFMs. These models improved the performance of FMs by considering the field information at the price of adding the model complexity.

As an important hyper-parameter in FM, the embedding dimension was usually the same for all features. Different from many FM-based approaches, which used a fixed embedding dimension for all features, Rank-Aware FM model (Chen, Zheng, Wang, Ma & Huang 2019) adopted different embedding dimensions for different pairwise interactions. The embedding dimensions were decided by frequencies of occurrences of corresponding feature interactions. The model achieved better performance on real-world datasets and could be stored and trained as efficiently as FM.

Although the most discussed FM model is in its second-order form, FM model was extended to higher-order form in the original work of (Rendle 2010). Blondel, Fujino, Ueda & Ishihata provided an efficient optimisation algorithm for the learning of Higher-order Factorisation Machines. They developed dynamic programming algorithms for evaluating the related ANOVA kernel and computing its gradient. Cao, Zhou, Li & Yu formulated the High-order Factorisation Machines as a CANDECOMP/PARAFAC factorisation (Kolda & Bader 2009) problem which could be efficiently optimised and stored.

FMs have also been incorporated into many deep neural networks to capture high-order feature interactions. Neural Factorisation Machines (He & Chua 2017) took in the advantages of deep neural networks to modelling higher-order feature interactions. They firstly embedded feature vectors similar to an FM module and then fed the embedding vectors into a bi-interaction layer that converted a set of embedding vectors to one vector. Next, the vector was feed into a multilayer perceptron to produce the final prediction score. DeepFM (Guo, Tang, Ye, Li & He 2017) was another example of combining an FM model with deep neural networks. It learned an FM model and a deep neural network jointly with shared embedding vectors. Unlike the bi-interaction layer in Neural Factorisation Machines, DeepFM concatenated the embedding vectors as the input of deep neural network module. Xiao, Ye, He, Zhang, Wu & Chua introduced the attention mechanism into FMs by adding an attention-based pooling layer. They focused on regression

tasks and got an 8.6% relative improvement on their experimental datasets. Weiyu, Yanyan & Linpeng has explored the feasibility of learning arbitrary-order cross features adaptively from data. They adopted a logarithmic transformation layer to convert the power of each feature in a feature combination into the coefficient to be learned. The number of cross features to be learned was fixed in advance as a hyper-parameter so that the model complexity was under controlled. Field information can also be used to help promote the performance of deep variants of the FM model, as shown in (Zhang, Shen, Huang, Li & Pan 2019, Lu, Yu, Chang, Wang, Li & Yuan 2020, Yu, Wang & Yuan 2019).

2.4.2 Learning with Incomplete Data

Learning from incomplete data is of great practical and theoretical interest. Commonly, we are faced with incomplete data in many applications. For example, in medical analysis, measurements on some subjects may be lost due to the lack of patient's compliance or unaffordable examination fees. In traffic prediction problems, some segments of a road network may contain no data collectors. For CBM, failure of a sensor will cause the absence of some records for a set of equipment. Even though all data are successfully collected without corruption, in a large-scale infrastructure network, we are not guaranteed that every set of equipment would be equipped with same sensors or keep the same format of maintenance logs. In this situation, some data could be considered absent when training a universal model.

A typical strategy is to fill the missing attributes in advance and then feed the data into traditional machine learning models. This strategy aims to recover the optimal model under complete data setting, with the prerequisite that missing entries are imputed appropriately. Filling missing attributes with zeros or means is a simple yet sometimes efficient method, but without considering the specific structure of data, such methods can be inferior. Another way is to complete the incomplete instances with data from their neighbours. For example, K-nearest-neighbours could be utilised to estimate

the missing values as in (Batista & Monard 2002). MICE (Multivariate imputation by chained equations) (Buuren & Groothuis-Oudshoorn 2010) was an iterative method dealing with missing data under the assumption of missing at random. It estimated each missing feature with regression models on other features, so it could be of high computation complexity when dealing with high-dimensional data. Probabilistic generative models such as Gaussian mixture model (Ouyang, Welsh & Georgopoulos 2004) used expectation maximisation algorithm to find the most probable completion. A limitation of imputation strategy is that errors in imputation stage may propagate to the following model learning stage. An intuitive way was to delete incomplete instances and make some assumptions on missingness patterns in training (Dekel, Shamir & Xiao 2010). Method for tuning the decision function for incomplete test data was proposed in (Xia, Zhang, Cai, Li, Pan, Yan & Ning 2017). These methods require complete data for training. This limits the application of such methods when most of the instances are incomplete, which is fairly common for condition monitoring data.

Some methods process the missing data in a task-specific way. Ghahramani & Jordan proposed to use the EM algorithm to learn from incomplete data for a classifier. Similarly, Williams, Liao, Xue & Carin proposed a classification model that dealt with missing data by performing analytic integration with an estimated conditional density function. Chechik et al. avoided the imputation procedure by introducing instance-specific margins for large margin classifiers. Goldberg, Recht, Xu, Nowak & Zhu connected the matrix completion task with classification task in a transductive way, whereas Hazan, Livni & Mansour argued that completion was neither necessary nor sufficient for classification. They proposed a kernel method for incomplete data based on observed features. Liu, Pan, Dezert & Martin used multiple imputations adaptively to improve the classification results. Apart from the methods mentioned above, many other works fall into this category (Shivaswamy, Bhattacharyya & Smola 2006, Dick, Haider & Scheffer 2008).

In addition to the above-mentioned methods, many neural networks can

be utilised to process data with missing attributes (Goodfellow, Mirza, Courville & Bengio 2013, Yoon, Jordon & Schaar 2018, Li, Jiang & Marlin 2019, Yang, Lu, Lin, Shechtman, Wang & Li 2017, Pathak, Krahenbuhl, Donahue, Darrell & Efros 2016), yet they required complete instances for learning the model. Recently, Śmieja, Struski, Tabor, Zieliński & Spurek proposed a model that could be trained without complete data. They replaced the typical neuron's response in the first hidden layer by its expected value when data were incomplete. The missing data density was depicted by a Gaussian mixture model and trained together with the neural network. One limitation of this model is that it requires an appropriate missing percentage of data so that the Gaussian mixture component could be fitted sufficiently. Otherwise, the model could perform poorly because of the under-fitting of Gaussian mixture component used in their model.

Most methods tend to use a universal model for all data, and thus ignore the inherent differences between data with different missingness patterns.

2.4.3 Multi-view Learning

As mentioned before, data related to CBM can be collected from multiple sources. This coincides with the fundamental assumption of multi-view learning, where data from each source are referred to as a particular view. Following (Xu, Tao & Xu 2013), we classify multi-view learning algorithms into three groups: 1) co-training, 2) subspace learning, and 3) multiple kernel learning.

Co-training (Blum & Mitchell 1998) is one of the earliest schemes for multi-view learning. It trains alternately to maximise the mutual agreement on two distinct views of the unlabelled data. Sindhwani, Niyogi & Belkin extended this idea to propose a co-regularisation framework for semi-supervise learning. Their algorithms naturally extended standard methods like support vector machines and regularised least squares for multi-view semi-supervised learning. Chen, Weinberger & Blitzer and Qin, Wang, Zhang & Fu used variants of co-training for the domain adaptation task. In (Chen, Weinberger &

Blitzer 2011), a single optimisation problem was formulated in each iteration of co-training to simultaneously learn a target predictor, a split of the feature space into views, and a subset of source and target features to include in the predictor. In (Qin, Wang, Zhang & Fu 2019), co-training was employed in a deep learning framework to bridge conditional distribution shift by assigning high-confident pseudo labels on target domain inferred from two distinct classifiers. Qiao, Shen, Zhang, Wang & Yuille explored a new way of implementing the co-training framework. They trained multiple deep neural networks to be the different views and exploits adversarial examples to encourage view difference. Closely related to our topic, Abdelgayed, Morsi & Sidhu applied the co-training algorithm to fault detection and classification tasks, to handle both labelled and unlabelled data. Co-training algorithms are based on the assumptions that the views are conditionally independent given the class label, and moreover, each view is sufficient for classification on its own. These assumptions are often too strong for CBM data, as multiple data channels may interact with each other, and data from only one source may lack essential information for failure prediction.

Subspace learning aims at learning a shared representation from multi-view data. Given the possible high dimensionality of multi-view data, the resulting latent subspace is usually a lower-dimensional space. Thus, it provides a way for dimensionality reduction and denoising, which helps subsequent tasks like classification, clustering and regression. Canonical correlation analysis (Hotelling 1992) and kernel canonical correlation analysis (Lai & Fyfe 2000) maximises the mutual information between the projections of two views in the lower dimension, and they are straightforward to be extended to multiple views (Hardoon, Szedmak & Shawe-Taylor 2004). Based on the idea of canonical correlation analysis, White et al. proposed a convex version of subspace learning by adding a regularisation term on matrix block norm. An efficient training procedure was introduced for the associated optimisation problem. By assuming view insufficiency that each view only captured partial information but all the views together carried redundant information

about the latent intact representation, Xu, Tao & Xu proposed the multi-view intact space learning. Lin, Wang, Meng & Zhao further incorporated unit intact space assumption to reduce the regularisation parameters and accelerated the learning process. In the basic formulation of subspace learning, each view is treated equally. For example, it does not distinguish between numerical or categorical data. Considering the practical case of CBM data, where multiple views can be of heterogeneous format, this should be carefully taken into account. More importantly, the learning process is conducted in typically an unsupervised manner. Incorporating the label information for a specific learning task will require redesigning of the learning framework.

Multiple kernel learning (MKL) (Lanckriet, Cristianini, Bartlett, Ghaoui & Jordan 2004) was originally designed to find an optimal kernel function by combining multiple kernel functions and to maximise a generalised performance measure. It has been widely used in various regression and classification tasks (Althloothi, Mahoor, Zhang & Voyles 2014, Bucak, Jin & Jain 2013, Liu, Zhou, Shen & Yin 2013, Yang, Tian, Duan, Huang & Gao 2012, Yeh, Huang & Lee 2011). MKL has been naturally applied to data with multiple views, where each view is associated with one or more kernel functions. Similar to deep neural networks, functions defined in reproducing kernel Hilbert space (RKHS) can model a highly non-linear relationship. Multiple kernel learning further takes the advantages of such functions by combining them wisely. Compared to deep neural networks, MKL enjoys better interpretability while requires less training data, which is more in line with the fundamental requirements of CBM, where more interpretable results can provide more comprehensive guidance for maintenance work.

Many variants of the MKL have been proposed to improve the accuracy of MKL algorithms. A natural extension is to change the L_1-norm constraint for kernel weights to L_p-norm as in (Kloft, Brefeld, Laskov, Müller, Zien & Sonnenburg 2009). Algorithms in (Kloft, Brefeld, Sonnenburg & Zien 2011) further simplified the optimisation procedure by adopting a closed-form solution for kernel weights. In (Liu, Wang, Zhang & Yin 2014), a binary

vector was introduced for every sample to switched on/off base kernels. The optimisation problem was an integer linear programming problem. The work in (Gönen & Alpaydin 2008) put forward a localised MKL algorithm. They utilised a gating model for selecting the appropriate kernel function locally. A convex variant was presented in (Lei, Binder & Kloft 2016) and corresponding generalisation error bounds were provided.

Another branch of studies focuses on improving the efficiency and scalability of MKL. In (Sonnenburg, Rätsch, Schäfer & Schölkopf 2006), they worked on a special scenario that when feature maps were sparse and can be explicitly computed. Combined with chunking optimisation, they were able to deal with large volumes of data. The work in (Rakotomamonjy & Chanda 2014) improved the scalability of MKL through Nystrom methods to approximate the kernel matrices and used proximal gradient algorithm in optimisation. Some methods were also developed for the situation when the number of kernels to be combined was very large (Afkanpour, György, Szepesvári & Bowling 2013). Besides, many online methods for MKL were proposed recently (Shen, Chen & Giannakis 2018, Shen & Chen 2018, Li, Gu, Ao, Wang & Ling 2017, Sahoo, Hoi & Li 2014, Hoi, Jin, Zhao & Yang 2013), and the related mistake bounds have been investigated in (Jin, Hoi & Yang 2010). Random feature approximation (Rahimi & Recht 2008) is popular among online MKL algorithms. It approximates some implicit feature map, like the feature map related to Gaussian kernel, with explicit functions.

Most of the research on multiple kernel learning was based on the prerequisite that all kernels were complete, whereas in most case, this is not satisfied. Liu, Wang, Yin, Dou & Zhang proposed an MKL algorithm to train a model with absent data channels. However, their model cannot be scaled up to large dataset due to the exponential computation complexity regarding the number of kernels, and it indeed treated different missing patterns equally in testing. This can be suboptimal for incomplete data.

For failure prediction related to CBM, we would require a new algorithm that can handle large datasets, while dealing with different missingness pat-

terns adaptively. The learning algorithm should also treat each group of samples adaptively so that we are able to capture the differences between different sets of equipment.

2.5 Summary

Deterioration modelling, maintenance strategy optimisation, and failure prediction are three key tasks for CBM, while our focus will be on the failure prediction task. Different from previous work that mostly being case-specific, this thesis will design more general methods for failure prediction, based on some of the related methods we revisited in this Chapter. As discussed, the real-world data related to CBM can be incomplete, sparse high-dimensional and multi-source. We need to design suitable models for such data. Importantly, the three characteristics can interplay with each other, so a model considering them all is also desirable.

Chapter 3

Field-regularised Factorisation Machines for Sparse High-dimensional Data

Sparse high-dimensional data are common in CBM related tasks. This data format brings difficulties to feature-engineering for predicting failures. Maintenance logs are a typical kind of sparse high-dimensional data. In this chapter, we introduced the Field-regularised Factorisation Machines for mining the maintenance logs of equipment, which leveraged the field information in maintenance logs to automatically learn effective cross-features from data for failure prediction. An empirical study was conducted with the maintenance logs of railway points to validate the method.

3.1 Introduction

As a part of the signal equipment, railway points control the routes of trains at railway junctions, having a great impact on the reliability and punctuality of rail transport. Existing research on failure prediction of points mainly relies on additional sensors' data (Yilboga, Eker, Güçlü & Camci 2010, Oyebande & Renfrew 2002, Guclu, Yilboga, Eker, Camci & Jennions 2010, Tao

& Zhao 2015, Camci, Eker, Başkan & Konur 2016, García Márquez, Roberts & Tobias 2010), e.g. voltages, currents and forces. Installation of sensors incurs costly labour and material expenses, as well as the possibility of sensor malfunction, which limits their implementation. Other research focuses on approximating the long-term degradation curve of equipment under certain maintenance strategy (Rama & Andrews 2013, Shafiee, Patriksson & Chukova 2016, Kobayashi et al. 2012, Tsuda, Kaito, Aoki & Kobayashi 2006, Le Son, Fouladirad & Barros 2016), rather than predicting failure of equipment in the near future.

Maintenance logs of equipment contain formatted maintenance records, including maintenance type, components, finished time, etc. They can be of great value in failure prediction. These data often carry information of equipment status with timestamps. Compared to data collected by sensors, maintenance records are usually ready to hand with a specified format. They mainly consist of categorical variables and could be very sparse after commonly performed one-hot encoding. Besides, railway points consist of many components, and failures can be viewed as a result of their interactions. Domain knowledge regarding such interactions might be very limited and depends on equipment types. In order to predict failures with maintenance logs, the model needs to learn the complex interactions from such sparse data.

Aiming at this challenging task, this chapter put forward Field-regularised Factorisation Machines (FrFMs) for failure prediction of railway points. Existing models either ignored the field information or only considered the inter-field information. They neglected the relationships among features inside each field, which is appropriately used in our models.

The contributions could be shown in two aspects. Firstly, to the best of our knowledge, it is the first time that maintenance logs are used to predict the failure of railway points. Secondly, this chapter proposes FrFMs which leverage field information and developed a method to solve the related optimisation problems. Experiments on two data sets show that this method

can achieve better performance compared to some state-of-the-art methods.

3.2 Preliminaries

A degree-2 polynomial mapping can often effectively capture the information of feature conjunctions (Chang, Hsieh, Chang, Ringgaard & Lin 2010). It learns a weight for each feature conjunction:

$$\phi_{Poly2}(W, \boldsymbol{x}) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{i,j} x_i x_j$$
$$W = (w_{i,j}) \in \mathbb{R}^{n \times n}, \boldsymbol{x} \in \mathbb{R}^n \tag{3.1}$$

where W is the learn-able weight matrix and $\boldsymbol{x}$ is the input feature vector of dimension n. $\phi_{Poly2}(W, \boldsymbol{x})$ outputs the score for related regression/classification tasks. Corresponding 2-way FMs can be written in following form:

$$\begin{aligned} \phi_{FM}(V, \boldsymbol{x}) &= \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle x_i x_j, \\ V &= [\boldsymbol{v}_1, ..., \boldsymbol{v}_n]^\top \in \mathbb{R}^{n \times k}, \boldsymbol{x} \in \mathbb{R}^n \end{aligned} \tag{3.2}$$

$\langle \cdot, \cdot \rangle$ stands for dot product of two vectors. $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ denote two row vectors of V with dimension k. $\boldsymbol{v}_i$ is referred to as **embedding vector** or **latent vector** for feature i. For simplicity of formulations, we omit linear terms and bias term following Juan et al., but we will include them in experiments.

Categorical data can be highly sparse after one-hot encoding. Some pairs of $x_i x_j$ might even not appear in any training instance. In this case, for polynomial mapping some $w_{i,j}$ are not able to be learned. By factorising the coefficient matrix W into VV^T, FMs are able to learn interactions for rare feature pairs. Each row vector $\boldsymbol{v}_i$ in V stands for the latent vector for feature x_i, to produce $w_{i,j}$ when multiplied by another vector x_j, so that $w_{i,j}$ can be implicitly learned as long as x_i and x_j appear in the dataset.

The complexity of straightforward computation of Eq (3.2) would cost $O(kn^2)$ as we need all pairwise interactions to be computed. It however

reduces to linear time $O(kn)$ with following reformulation (Rendle 2010),

$$
\begin{aligned}
&\sum_{i=1}^{n}\sum_{j=i+1}^{n}\langle \boldsymbol{v}_i, \boldsymbol{v}_j\rangle x_i x_j \\
=&\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\langle \boldsymbol{v}_i, \boldsymbol{v}_j\rangle x_i x_j - \frac{1}{2}\sum_{i=1}^{n}\langle \boldsymbol{v}_i, \boldsymbol{v}_i\rangle x_i x_i \\
=&\frac{1}{2}\left(\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{f=1}^{k} v_{i,f}v_{j,f}x_i x_j - \sum_{i=1}^{n}\sum_{f=1}^{k} v_{i,f}v_{i,f}x_i x_i\right) \\
=&\frac{1}{2}\sum_{f=1}^{k}\left(\left(\sum_{i=1}^{n} v_{i,f}x_i\right)\left(\sum_{j=1}^{n} v_{j,f}x_j\right) - \sum_{i=1}^{n} v_{i,f}^2 x_i^2\right) \\
=&\frac{1}{2}\sum_{f=1}^{k}\left(\left(\sum_{i=1}^{n} v_{i,f}x_i\right)^2 - \sum_{i=1}^{n} v_{i,f}^2 x_i^2\right)
\end{aligned}
$$

which is efficient and has been used in a wide range of applications like information retrieval (Qiang, Liang & Yang 2013), social network (Hong, Doumith & Davison 2013), and recommendation systems (Rendle, Gantner, Freudenthaler & Schmidt-Thieme 2011).

3.3 Field-regularised Factorisation Machines

In this section, we introduce the motivation for our method. We will formulate the two variants of our method following different distance metrics and derive corresponding optimisation algorithms.

3.3.1 Motivation

Table 3.1 presents some simple data constructed from maintenance records for failure prediction. "Maintenance Type" and "Component" are two different **fields**. A, B and C stand for different maintenance types that can probably be "Routine Inspection", "Corrective Maintenance" and so on. The field "Component" shows the maintenance was performed over which component. "1" and "-1" in column "Failure" stand for whether there was a

fault occurred after this maintenance and before next planned maintenance.

FMs will learn latent vectors for A, B, C, II and VI respectively. In engineering practice, we anticipate different effects with different maintenance behaviours. Each field can be regarded as a classification criterion for maintenance work, and corresponding features in that field are the class labels. We would prefer diverse latent vectors in the same field so that we could distinguish the effects caused by different maintenance work in this way. As a result, latent vectors for A, B and C should be diverse, as well as latent vectors for II and VI.

3.3.2 Methods

In this section, we introduce the FrFMs for binary classification. For simplicity of formulations, we omit linear terms and bias term following (Juan et al. 2016), but we will include them in experiments as they often improve the results. The loss function of FrFMs with logistic loss regarding one sample $(y, \boldsymbol{x})$ is:

$$\mathcal{L}(V) = \log(1 + \exp(-y\phi_{FM}(V, \boldsymbol{x}))) + \frac{\lambda_1}{2}\|V\|_F^2 + \frac{\lambda_2}{2}R(V) \tag{3.3}$$

$\phi_{FM}(V, \boldsymbol{x})$ is defined in Eq. (3.2), as we share the same prediction function with FMs. $\|\cdot\|_F$ is the Frobenius norm for matrices. $y \in \{-1, 1\}$ is the ground truth label for sample $\boldsymbol{x}$. The first term denotes the prediction loss compared to ground truth, and the second term forces the solution V sparse. $R(V)$ is a regulariser that measures the similarity of latent vectors in each field, and we prefer smaller similarity as discussed above. By introducing

Table 3.1: A sample of maintenance records with failures to be predicted.

Failure	Maintenance Type	Component
1	A	II
1	C	II
-1	B	VI

$R(V)$ into loss function, field information is included. λ_1, λ_2 are two non-negative parameters obtained by cross validation.

In order to capture the inherent properties come with fields of data, we construct a feature relation matrix A which will be included in $R(V)$:

$$A_{i,j} = \begin{cases} \frac{1}{N_{i,j}} & \text{if } x_i, x_j \text{ are in same field and } i \neq j, \\ 0 & \text{else.} \end{cases} \tag{3.4}$$

$N_{i,j}$ is the number of features in the field containing x_i and x_j. It is introduced to avoid deviation caused by different number of features in different fields. Each element in A stands for the relationship of two features. If they are in same field, then corresponding entries in A will be one divided by the number of features in this field. Otherwise they will be zeros.

Various metrics can be used to measure the similarity of latent vectors. In this work, we will present FrFM with Euclidean distance and cosine similarity.

FrFM-EUC

We refer to FrFM with Euclidean distance as FrFM-EUC. Euclidean distance is used to measure the similarity of two vectors in FrFM-EUC, and larger Euclidean distance indicates smaller similarity. Therefore, $R(V)$ has the following form:

$$R(V) = -\sum_{i=1}^{n}\sum_{j=i+1}^{n} A_{i,j}\|\boldsymbol{v}_i - \boldsymbol{v}_j\|_2^2 \tag{3.5}$$

$\|\cdot\|_2$ denotes l_2-norm for vectors. The loss function for FrFM-EUC is:

$$\mathcal{L}_{euc}(V) = \log(1+\exp(-y\phi_{FM}(V,\boldsymbol{x}))) + \frac{\lambda_1}{2}\|V\|_F^2 - \frac{\lambda_2}{2}\sum_{i=1}^{n}\sum_{j=i+1}^{n} A_{i,j}\|\boldsymbol{v}_i - \boldsymbol{v}_j\|_2^2 \tag{3.6}$$

FrFM-COS

FrFM-COS denotes FrFM with cosine similarity. $R(V)$ has the following form:

$$R(V) = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{i,j}\frac{\langle \boldsymbol{v}_i, \boldsymbol{v}_j\rangle}{\|\boldsymbol{v}_i\|_2\|\boldsymbol{v}_j\|_2} \tag{3.7}$$

Directly minimising Eq. (3.3) with Eq. (3.7) is complicated. Rewriting rows of V into products of their direction vectors and lengths leads to:

$$V = \begin{bmatrix} w_1\hat{\boldsymbol{v}}_1 \\ w_2\hat{\boldsymbol{v}}_2 \\ \vdots \\ w_n\hat{\boldsymbol{v}}_n \end{bmatrix} \in \mathbb{R}^{n\times k}, \quad \hat{\boldsymbol{v}}_i = \frac{\boldsymbol{v}_i}{\|\boldsymbol{v}_i\|_2}, \quad w_i = \|\boldsymbol{v}_i\|_2 \tag{3.8}$$

Then Eq. (3.7) could be rewritten into:

$$R(V) = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{i,j}\hat{\boldsymbol{v}}_i\hat{\boldsymbol{v}}_j^T = tr(\hat{V}^T A\hat{V}) \tag{3.9}$$

Substitute V with $\hat{V}$ and $\boldsymbol{w}$ in formulation of FMs:

$$\phi_{FM}(\hat{V}, \boldsymbol{w}, \boldsymbol{x}) = \sum_{i=1}^{n}\sum_{j=i+1}^{n} \langle w_i\hat{\boldsymbol{v}}_i, w_j\hat{\boldsymbol{v}}_j\rangle x_i x_j \tag{3.10}$$

and finally we get loss function for FrFM-COS:

$$\begin{aligned} \mathcal{L}_{cos}(\hat{V}, \boldsymbol{w}) = \log(1+\exp(-y\phi_{FM}(\hat{V}, \boldsymbol{w}, \boldsymbol{x}))) + \frac{\lambda_1}{2}\|\boldsymbol{w}\|_2^2 + \frac{\lambda_2}{2}tr(\hat{V}^T A\hat{V}) \\ s.t.\ \|\hat{\boldsymbol{v}}_i\|_2 = 1,\ \forall i = 1, 2, ..., n. \quad \boldsymbol{w} \in \mathbb{R}_+^{1\times n} \end{aligned} \tag{3.11}$$

3.3.3 Optimisation

Similar to FMs, our loss functions are non-convex. Gradient descent is used to find local minima of our loss functions. Stochastic Gradient Descent (SGD) is widely used in optimisation of FMs and its variants. It has shown its effectiveness. Mini-batch Gradient Descent also enjoys the advantages of SGD while it is more efficient. Thus we adopt Mini-batch Gradient Descent in optimisation. We apply AdaGrad (Duchi, Hazan & Singer 2011) to determine the learning rate in each iteration for it has shown great power in similar problems (Juan et al. 2016, Chin, Zhuang, Juan & Lin 2015). To lessen over-fitting, we utilise early-stop strategy in the training of FrFM-EUC and FrFM-COS. The best training epoch T will be decided based on a validation set.

FrFM-EUC

The gradient with regard to one sample $(y, \boldsymbol{x})$ is:

$$\begin{aligned}\frac{\partial \mathcal{L}_{euc}(V)}{\partial \boldsymbol{v}_i} =& \frac{-y}{1+\exp(y\phi_{FM}(V,\boldsymbol{x}))}(x_i \sum_{j=1}^{n} \boldsymbol{v}_j x_j - \boldsymbol{v}_i x_i^2) \\ &+ (x_i \neq 0)(\lambda_1 \boldsymbol{v}_i - \lambda_2 \sum_{j=1}^{n} A_{i,j}(\boldsymbol{v}_i - \boldsymbol{v}_j))\end{aligned} \tag{3.12}$$

$(x_i \neq 0)$ in Eq. (3.12) indicates that gradients would be zero if corresponding features are zero. This strategy has been used in FFMs and performs well. We can update model parameters with adaptive learning rate in iteration l:

$$G_{i,f}^{(l+1)} = G_{i,f}^{(l)} + (\frac{\partial \mathcal{L}_{euc}(V)}{\partial v_{i,f}}\Big|_{V=V^{(l)}})^2 \tag{3.13}$$

$$v_{i,f}^{(l+1)} = v_{i,f}^{(l)} - \frac{\eta}{\sqrt{G_{i,f}^{(l+1)} + \epsilon}} \circ \frac{\partial \mathcal{L}_{euc}(V)}{\partial v_{i,f}}\Big|_{V=V^{(l)}} \tag{3.14}$$

$\circ$ denotes element-wise multiplication of vectors. G stores the accumulated square gradient for AdaGrad and ϵ is s a smoothing term that avoids division by zero (we set it to 10^{-8} in experiments). The training process for FrFM-EUC is presented in Algorithm 3.1.

FrFM-COS

The gradient with regard to one sample $(y, \boldsymbol{x})$ is:

$$\begin{aligned}\frac{\partial \mathcal{L}_{cos}(\hat{V}, \boldsymbol{w})}{\partial \hat{\boldsymbol{v}}_i} =& \frac{-y}{1+\exp(y\phi_{FM}(\hat{V},\boldsymbol{w},\boldsymbol{x}))}(w_i x_i \sum_{j=1}^{n} \hat{\boldsymbol{v}}_j w_j x_j - \hat{\boldsymbol{v}}_i w_i^2 x_i^2) \\ &+ (x_i \neq 0)\lambda_2 \sum_{j=1}^{n} A_{i,j}\hat{\boldsymbol{v}}_j\end{aligned} \tag{3.15}$$

$$\begin{aligned}\frac{\partial \mathcal{L}_{cos}(\hat{V}, \boldsymbol{w})}{\partial \boldsymbol{w}} =& \frac{-y}{1+\exp(y\phi_{FM}(\hat{V},\boldsymbol{w},\boldsymbol{x}))}((\boldsymbol{w} \circ \boldsymbol{x})(\hat{V}\hat{V}^T - diag(\hat{V}\hat{V}^T))) \circ \boldsymbol{x} \\ &+ \lambda_1(\boldsymbol{x} \neq 0) \circ \boldsymbol{w}\end{aligned} \tag{3.16}$$

Algorithm 3.1 Training FrFM-EUC by Mini-batch Gradient Descent

INPUT: Data matrix $D \in \mathbb{R}^{M \times n}$ contains M samples, feature relation matrix A, latent dimension k, hyper-parameters λ_1, λ_2, learning rate η, batch size m, $G^{(0)} = \mathbf{0}$.

INITIALISE: Randomly initialise $V^{(0)} \in \mathbb{R}^{n \times k}$ with values sampled from a uniform distribution $[0, 1/\sqrt{k}]$. Calculate the number of batches $b = \lfloor \frac{M}{m} \rfloor$.

for *Epoch* $= 0$ to T **do**
  Shuffle the samples in D randomly.
  Split D into batches $X_1, X_2, ..., X_b \in \mathbb{R}^{m \times n}$.
  for $i \in \{1, 2, ..., b\}$ **do**
    Calculate the gradient of V by Eq. (3.12) for every sample in X_i and compute the average.
    Update accumulated square gradient G by Eq. (3.13).
    Update V by Eq. (3.14).
  end for
end for

($\boldsymbol{x} \neq 0$) is a binary row vector indicates non-zero indices of $\boldsymbol{x}$. Similarly, gradients would be zero if corresponding features are zero. With gradient in hand, we can train the model similar to Algorithm 3.1. Differences are that we need to project $\hat{V}$ and $\boldsymbol{w}$ into feasible sets in each iteration.

3.4 Experiments

The experiments were performed on maintenance logs of railway points from a large scale railway network. To make the results more persuasive, we also performed experiments on another public dataset related to phishing website detection, where the features are also categorical variables.

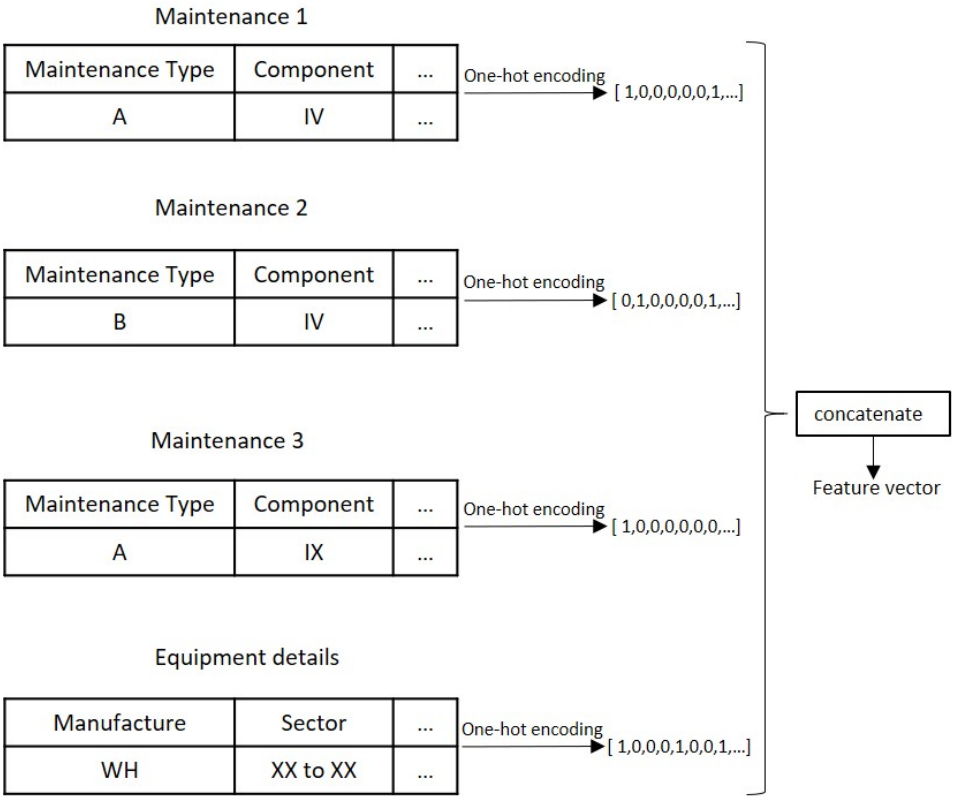


Figure 3.1: An example for constructing a feature vector from a sample in POINTS-3 dataset.

3.4.1 Data Set

POINTS-3 dataset was generated from the maintenance logs of Sydney Trains' railway points, plus corresponding equipment details. For numerical features, they were simply transformed into features "Zero" or "Non-Zero". As illustrated in Figure 3.1, for one piece of equipment, we selected three consecutive maintenance records: Maintenance 1, Maintenance 2 and Maintenance 3, and associated equipment details, to construct a feature vector. We labelled the sample depending on whether a failure occurred after Maintenance 3 and before maintenance 4, as shown in Figure 3.2. If there was a failure record, then this sample was labelled with "1", otherwise "-1".

Equipment details including equipment type, location and other features

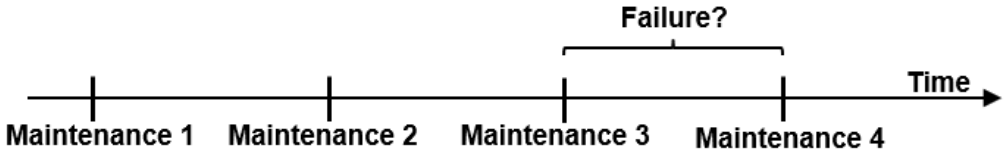


Figure 3.2: An example of labelling samples in POINTS-3 dataset.

were also concatenated to construct one data sample. We randomly split the data set into 60% training set, 20% validation set and 20% test set.

Phishing dataset contains important features that have been proven to be sound and effective in predicting phishing websites (Dheeru & Karra Taniskidou 2017). The data were randomly split into 64% for training set, 16% for validation set and 20% for test set.

Table 3.2 summarises the statistics of the datasets.

3.4.2 Baselines and Hyper-parameter Tuning

We compare the proposed method with three baselines.

LINEAR-LR denotes Logistic Regression with linear terms. It has been proven to be effective in classification tasks with sparse data and has been widely applied in failure prediction tasks thank to its interpretability and simplicity (Dong, Tung, Chen, Liao & Pan 2011, Robles-Velasco, Cortés, Muñuzuri & Onieva 2020). We implemented LINEAR-LR with Python library sklearn (Pedregosa, Varoquaux, Gramfort, Michel, Thirion, Grisel, Blondel, Prettenhofer, Weiss, Dubourg, Vanderplas, Passos, Cournapeau, Brucher, Perrot & Duchesnay 2011).

FM (Rendle 2010) is the implementation of Factorisation Machines de-

Table 3.2: Statistics of the datasets.

Data Set	# Instances	# Features	# Fields	# Positive Instances
POINTS-3	55784	2226	52	1701
Phishing	11055	68	30	6157

fined in (2). We also included linear terms and bias term.

FFM (Juan et al. 2016) is the implementation of Field-aware Factorisation Machines. We also included linear terms and bias term.

FrFM-EUC and **FrFM-COS** stand for our methods proposed in this Chapter.

Both FM and FFM were implemented by xLearn [1] with AdaGrad and SGD optimiser. All hyper-parameters were chosen based on the validation sets. The regularisation parameters were chosen from $\{10^{-6}, 10^{-5}, ..., 10^{6}\}$ for LINEAR-LR and $\{10^{-6}, 10^{-5}, ..., 10^{-1}\}$ for all other methods considering their performance and training time on the validation sets. Learning rates for AdaGrad were chosen from $\{0.02, 0.2\}$. Latent dimensions were chosen from $\{20, 40, ..., 100\}$ for FM and our method, and from $\{10, 20, ..., 50\}$ for FFM. The early-stop strategy was adopted for FM, FFM and our method to reduce over-fitting. The batch size was set to 64 in the training of FrFM-EUC and FrFM-COS.

3.4.3 Results and Metrics

Metrics

We use three performance metrics to evaluate the baselines and the proposed methods, named Logloss, AUROC and AUPRC.

Logloss is given by following equation.

$$\text{Logloss} = \frac{1}{M}\sum_{i=1}^{M} log(1 + \exp(-y_i\hat{y}_i)) \tag{3.17}$$

y_i and $\hat{y}_i$ are the label and model output for test sample i respectively. M is the total number of test instances.

AUROC and **AUPRC** stand for area under receiver operating characteristic curve and area under precision-recall curve respectively. AUROC is more suitable for imbalance datasets as reported in literature (Ozenne, Subtil

[1] https://github.com/aksnzhy/xlearn

Table 3.3: Comparison of LINEAR-LR, FM, FFM, FrFM-EUC and FrFM-COS. The best results are bold and the second-best are underlined

Method	POINTS-3			Phishing	
	AUROC	AUPRC ($recall > 0.1$)	Logloss	AUROC	Logloss
LINEAR-LR	0.7012	0.0641	0.1275	0.9886	0.1384
FM	0.6987	0.0622	0.1285	0.9911	0.1226
FFM	0.6974	0.0619	0.1291	0.9923	0.1134
FrFM-COS	<u>0.7090</u>	**0.0676**	<u>0.1271</u>	<u>0.9925</u>	<u>0.1120</u>
FrFM-EUC	**0.7108**	<u>0.0674</u>	**0.1270**	**0.9950**	**0.0919**

& Maucort-Boulch 2015, Saito & Rehmsmeier 2015, Davis & Goadrich 2006, Boyd, Eng & Page 2013).

Results

Table 3.3 shows the results on different data sets. The best results are bold, and the second best are underlined. We trained and tested these models five times on each data set and reported the average results. POINTS-3 dataset is an imbalanced data set with only 1701 positive samples out of 55784 samples, so AUPRC is more representative compared to AUROC. AUPRC were calculated from $recall > 0.1$. A very low recall (< 0.1) is meaningless because in that case, most of the failures will be ignored. Phishing data set is a balanced data set that won't show much difference between AUROC and AUPRC, so we only present the AUROC for it.

Experiment results show that the proposed methods perform best on these two datasets. Precision-recall curves with regard to POINTS-3 dataset for $recall > 0.1$ and $precision > 0.06$ are plotted in Figure 3.3. We drop the segments where recall is smaller than 0.1.

Figure 3.3 shows that FrFM-COS can also achieve the best F_1-score

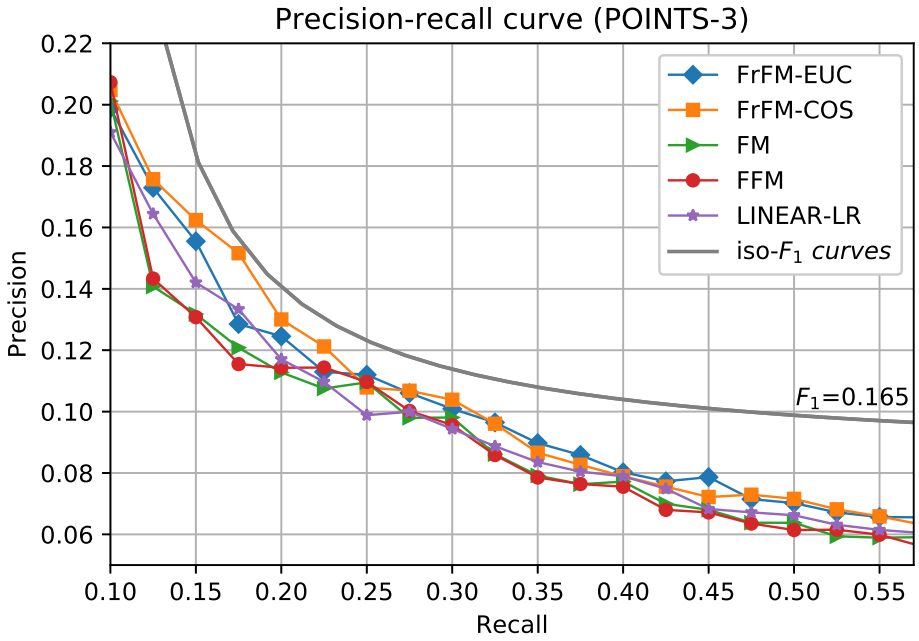


Figure 3.3: Precision-recall curves with regard to POINTS-3 dataset. We drop the segments where recall is smaller than 0.1.

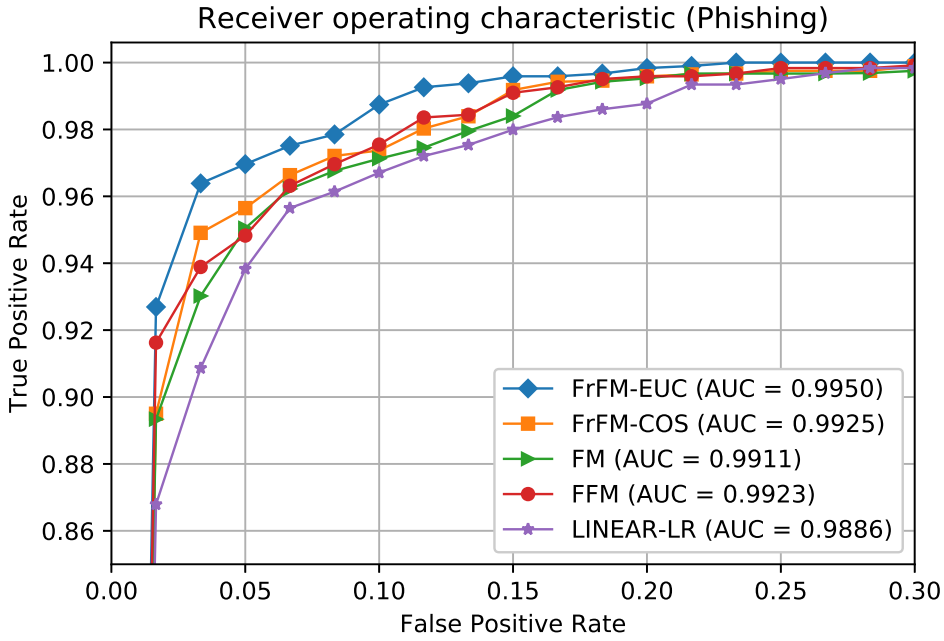


Figure 3.4: Receiver operating characteristic curves with regard to Phishing dataset.

(0.165) compared to other methods. By appropriately setting threshold value for the classifier got from FrFM-EUC, we can get an overall **Accuracy: 90.99%**, with **Precision: 11.02%** and **Recall: 27.65%**. This may not be a perfect prediction, but it is still acceptable considering that we didn't use any sensor data (e.g. current, voltage, force and so on). There are wrongly recorded data and failures that are caused by vandalism which makes some failures unpredictable. Outputs of the model could be used as references for maintenance plans.

Receiver operating characteristic curves with regard to Phishing dataset are plotted in Figure 3.4. Our method consistently outperformed other methods. Notice that this dataset contains important features that were carefully designed for the phishing website prediction task, so that it is similar to the maintenance logs where we can anticipate different features in the same field would work differently towards the learning objectives. Performance of

our method on this dataset again validates the effectiveness of the proposed method under such situation.

3.5 Conclusion

In this chapter, we proposed the Field-regularised Factorisation Machines for failure prediction of railway points. Field information is often ignored in many related methods. Especially for the inner-field relationships among features, there is limited work concerning them. The key components of FrFMs are the regularisation terms that incorporate field information in the training process. Two forms of FrFMs: FrFM-EUC and FrFM-COS are presented. Experiment results showed that the proposed method outperformed some state-of-the-art methods in predicting the failure of railway points. We also achieved a better result on a public dataset.

The predictions for points failure were not perfect but could be used as the reference for maintenance plans. More accurate predictions will involve data from other sources to enhance the model performance, as will be introduced in Chapter 5. Besides, the proposed method used only second-order feature-interactions, which limited its performance. However, the sparseness of the data often hinders the learning of higher-order feature-interactions and can cause the over-fitting problem. An important future research direction is to design appropriate learning algorithms for learning higher-order feature-interactions.

Chapter 4

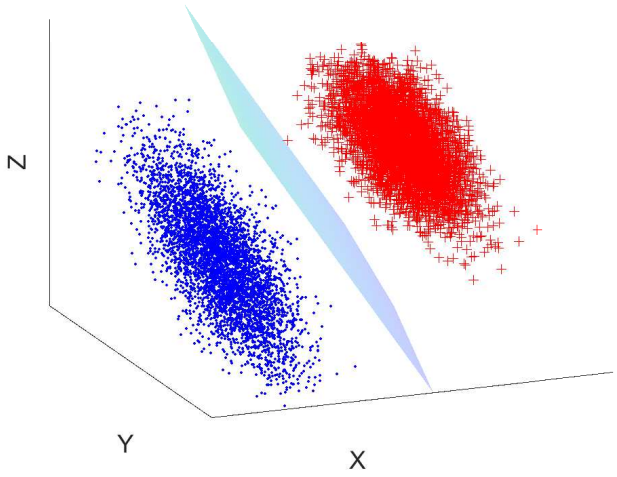
Missingness-pattern-adaptive Model for Incomplete Data

Due to various reasons, data in CBM-related tasks are often incomplete. In this chapter, we presented a general model for learning with incomplete data. We showed that there was a competition in learning with data of different missingness patterns, resulting in a suboptimal model for each pattern. This inspired us to develop a method that could be appropriately adjusted by missingness patterns, so that alleviated such competition between data. This method was solely based on the observable features, so it did not incur errors from imputation. In addition, a low-rank constraint was introduced to promote the generalisation ability of the proposed model. Analysis of the generalisation error justified this method theoretically. A subgradient method was proposed to optimise the linear model with a proven convergence rate. Experiments on different types of data showed that this method compared favourably with typical imputation strategies and other state-of-the-art methods for incomplete data. This idea was also combined with neural networks to show the effectiveness of the proposed method.

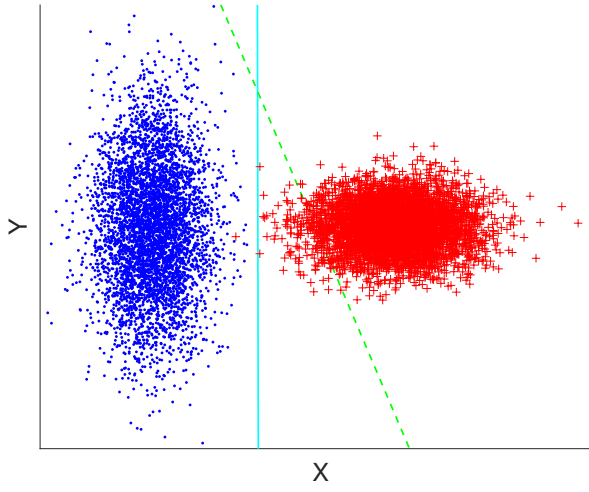
4.1 Introduction

As reviewed in Chapter 2, the main shortage of previous methods is that they tend to use a universal model for all data, and thus ignore the inherent differences between data with different missingness patterns. Missingness patterns are commonly used to indicate the locations of the missing entries. Samples may have varying subsets of observable features due to the inherent properties of the instances. Some of the features may not even be defined for some instances. Using the same model for these heterogeneous data limits the performance of the model, and imputation may lead to severe deviation. More importantly, the model could suffer from competition between data with different missingness patterns. We illustrate such a phenomenon in Figure 4.1. For two sets of data labelled as "." and "+", when we have complete features of an instance, the best decision plane for classification is shown in Figure 4.1a. However, if we use the available features (x, y) to classify a point when feature z is missing, then use the coefficients of the decision plane in Figure 4.1a regarding (x, y) is not optimal (shown as the dashed line in Figure 4.1b). The best separating line, in this case, is the solid line as shown in Figure 4.1b. These two patterns would compete against each other when training with incomplete data, leading to a suboptimal model for both cases. A straightforward way to minimise such influence is to learn different decision functions for each missingness pattern. However, for some missingness patterns, data can be insufficient for the training of the model, which causes difficulties in generalisation.

Contribution This chapter proposed an adaptive model that can apply associated decision functions to data with corresponding missingness patterns and does not require the imputation of missing data. The contribution is three-fold. First, for the first time, different models were learned for data with different missingness patterns, while improving the generalisation ability by a low-rank constraint. Second, the generalisation error bound and convergence property of this model were theoretically proven. Last, the idea



(a)



(b)

Figure 4.1: When all features (x, y, z) are observable, we have an optimal separating plane in 4.1a. When only (x, y) are observable, the best separating line is the solid line in 4.1b. The projection of optimal separating plane in 4.1b is the dashed line. If we train one model for both cases, we will probably end with a compromise of them and get an inferior result.

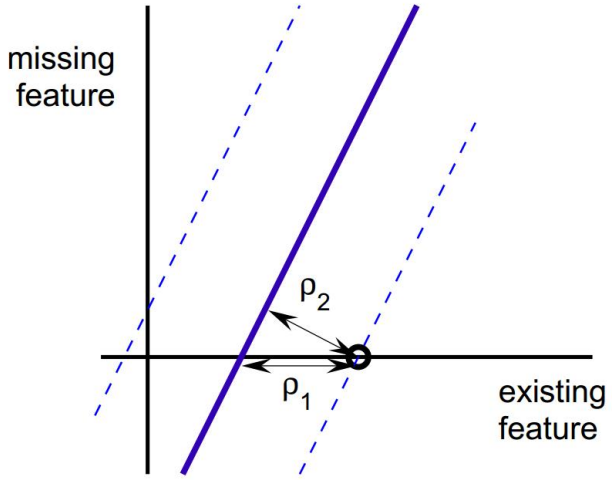


Figure 4.2: The margin of a sample that only has one feature (the x dimension) is measured both in the higher-dimensional space (ρ_2) and the lower one (ρ_1). The lower-dimensional margin is larger and therefore we overestimates the margin. (Chechik et al. 2008)

could be seamlessly incorporated into various neural networks with minimal modification of network architectures.

4.2 Preliminaries

Apart from the competition between data with different missingness patterns as shown in Figure 4.1, the margins under the incomplete data setting are also different from the complete case.

Maximising the margin under incomplete data setting has been investigated in (Chechik et al. 2008). Due to the absent features, the margins for each missingness pattern can vary. Considering the classification case, Figure 4.2 shows the change of margins with an incomplete sample.

In order to minimise the margins under incomplete data setting, considering a dataset of n labelled instances $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, the instance margin ρ_i

for i-th instance is defined as

$$\rho_i = \frac{y_i \mathbf{w}^{(i)} \cdot \mathbf{x}_i}{||\mathbf{w}^{(i)}||}, \tag{4.1}$$

where $\mathbf{w}^{(i)}$ is the vector obtained by taking the entries of classifier $\mathbf{w}$ that are relevant to known part of $\mathbf{x}_i$. $||\cdot||$ denotes the vector 2-norm. The decision function is given by $y_i = \mathbf{w}^{(i)} \cdot \mathbf{x}_i$.

we then arrive at a new optimisation problem for the incomplete data case

$$\max_{\mathbf{w}} \left(\min_i \frac{y_i \mathbf{w}^{(i)} \cdot \mathbf{x}_i}{\|\mathbf{w}^{(i)}\|} \right) \tag{4.2}$$

Extending above formulation to the non-separable case is however difficult for optimisation, so an alternative solution is to use the average norm define by $\sqrt{\frac{1}{n}\sum_{i=1}^{n} ||\mathbf{w}^{(i)}||^2}$ to approximate the sample-specific denominator $\left\|\mathbf{w}^{(i)}\right\|$, which leads to:

$$\max_{\mathbf{w}} \min_i \frac{y_i \mathbf{w}^{(i)} \cdot \mathbf{x}_i}{\|\mathbf{w}\|} \tag{4.3}$$

which is similar to the standard SVM optimisation. Introduce the threshold b, slack variables ξ_i, and hyper-parameter C to handle the non-separable cases we get:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{n}\sum_{i=1}^{n} \left(\frac{1}{2} \left\|\mathbf{w}^{(i)}\right\|^2 + C\xi_i \right) \\ s.t. \quad & y_i \left(\mathbf{w}^{(i)} \mathbf{x}_i + b \right) \geq 1 - \xi_i, i = 1 \ldots n \end{aligned} \tag{4.4}$$

Such optimisation problem can be solve effectively as standard SVM. We would also use the average norm in our formulation for a tractable optimisation algorithm.

4.3 Missingness-pattern Adaptive Model

In this section, we formulate our idea for binary classification, but it can also be extended to multi-class or regression tasks with associated objective functions. Given a data instance $(\mathbf{x}, \mathbf{m}, y)$ with feature vector $\mathbf{x} \in \mathbb{R}^d$, label $y \in \{-1, +1\}$ and $\mathbf{m} \in \mathbb{R}^{d'}$ an indicator vector represents its missingness

pattern. Without any prior knowledge, $\mathbf{m}$ will be a d-dimensional binary vector. Each bit of $\mathbf{m}$ indicates the missingness of the corresponding bit in $\mathbf{x}$. For example, $\mathbf{m} = [1, 0]$ indicates $\mathbf{x}$ is a 2-dimensional feature vector, and its second feature is missing. In some settings such as incomplete multi-view learning, features are missing group-wise, so $\mathbf{m}$ can serve as a group-wise indicator, thus making d' much smaller than d.

In order to treat missingness patterns adaptively, the linear decision function can be formulated as:

$$f(\mathbf{x}) = g(\mathbf{m})\mathbf{x}^o, \tag{4.5}$$

where $\mathbf{x}^o \in \mathbb{R}^d$ denotes the $\mathbf{x}$ after zero out the missing values. In this way, it is possible to apply different weight coefficients generated by $g(\mathbf{m})$ for data of different missingness patterns. g can be selected from a wide range of function classes. In this work, we adopt a simple yet efficient form of $g(\mathbf{m})$ given by:

$$g(\mathbf{m}) = (H\bar{\mathbf{m}})^\top \tag{4.6}$$

with $H \in \mathbb{R}^{d \times 2d'}$ serves as a dictionary for generating missingness-pattern-specific functions. $\bar{\mathbf{m}} = [\mathbf{m}^\top, (\mathbf{1}-\mathbf{m})^\top]^\top$ is an augmented vector generated by concatenating $\mathbf{m}$ and its element-wise logic NOT operation. In doing this, for every distinct missingness pattern $\mathbf{m}$, we have a corresponding weight vector generated by $H\bar{\mathbf{m}}$. Notice that we use $\bar{\mathbf{m}}$ instead of $\mathbf{m}$ to ensure that for every missingness pattern we select a fixed number of elements from H. Bias terms could also be incorporated into Eq.(4.5) by appending a constant feature to $\mathbf{x}^o$ and extend $\mathbf{m}$ and H accordingly. Thus the bias terms can also be adaptively fitted to missingness patterns. For notational simplicity, we omit them in our formulas.

In the spirit of large margin classifier, we can define a modified learning objective which is specialised for incomplete data. Given a set of n labelled

observations $\{(\mathbf{x}_i, \mathbf{m}_i, y_i)\}_{i=1}^n$, the learning objective is:

$$\begin{aligned} \min_{H} \quad & \frac{1}{n}\|M \odot (H\bar{M})\|_F^2 + \eta_1\|H\|_F^2 + \frac{\eta_2}{n}\sum_{i=1}^{n}\xi_i, \\ s.t. \quad & y_i(\bar{\mathbf{m}}_i^\top H^\top \mathbf{x}_i^o) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, 2, \cdots, n, \\ & \text{rank}(H) \leq k, M = [\mathbf{m}_1, \cdots, \mathbf{m}_n], \bar{M} = [\bar{\mathbf{m}}_1, \cdots, \bar{\mathbf{m}}_n], \end{aligned} \tag{4.7}$$

with $\|\cdot\|_F$ and $\odot$ denotes the Frobenius norm and the Hadamard product respectively. ξ_i is the slack variable for the margin. η_1, η_2 and k are hyper-parameters. The first term is the approximate denominator for instance-based margins (Chechik et al. 2008) calculated from the average norm as we discussed in the preliminaries. Because each instance has its own observable part, we should calculate the margin regarding observable part only. Therefore, we use the mask matrix M to zero out the weights in $H\bar{M}$ corresponding to missing features. We do not adopt the exact instance-based margin here since it brings difficulties in optimisation (Chechik et al. 2008). We also introduce η_1 to constraint the Frobenius norm of H and fix it to be a small constant.

Eq.(4.5) allows us to define a decision function for every missingness pattern while connecting them through a low-rank matrix H. The low-rank constraint introduces correlations between models for different missingness patterns, so that facilitates the learning of models related to some rare missingness patterns.

Our idea can also be applied to many existing neural networks with minimal modification. Assume the output of a neural network with complete data can be expressed as $\hat{y} = f(\mathbf{x}; \theta)$ with θ denote parameters of the network. We can adjust the weight of observed features by missingness pattern, which gives the output $\hat{y} = f((H\bar{\mathbf{m}}) \odot \mathbf{x}^o; \theta)$. The learning objective can be formulated as:

$$\min_{U,V,\theta} \sum_{i=1}^{n} \mathcal{L}\big(y_i, f((U^\top V\mathbf{m}_i) \odot \mathbf{x}_i^o; \theta)\big) \tag{4.8}$$

where $\mathcal{L}$ is the loss function and we incorporate the rank constrain by decomposing H into product of $U^\top$ and V with $U \in \mathbb{R}^{k\times d}$ and $V \in \mathbb{R}^{k\times 2d'}$. U

and V would be learned together with the network's parameters θ in an end-to-end manner. The motivation behind the formula is clear - we can adjust the importance of observed features when some other features are missing.

4.4 Generalisation Error Bound Analysis

In this section, we theoretically analyse the generalisation error of our linear model. We give a rather general bound on the generalisation error based on the growth function. The bound also supports the low-rank constraint in our model.

We firstly introduce some common settings in this section. A labelled training set is given by $D = \{(\mathbf{x}_i, \mathbf{m}_i, y_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \mathcal{X}$ with $\mathcal{X}$ a subset of $\mathbb{R}^d$, $y_i \in \{-1, +1\}$ and $\mathbf{m}_i \in \{0, 1\}^{d'}$ the missingness indicator vector. We assume that training data are drawn independently and identically distributed (i.i.d.) (for non-i.i.d case more information regarding the data distribution is required (Mohri & Rostamizadeh 2008)) according to some unknown distribution $\mathcal{D}$ and denote $D \sim \mathcal{D}^n$. Let the hypothesis set $\mathcal{F}$ be a family of functions mapping $\mathcal{X}$ to $\{-1, +1\}$ defined by $\mathcal{F} = \{\mathbf{x} \mapsto (H\bar{\mathbf{m}})^\top \mathbf{x}^o : rank(H) \leq k\}$. The empirical error of a hypothesis $f \in \mathcal{F}$ over the training set D is defined as $\widehat{R}_D(f) = \frac{1}{n}\sum_{i=1}^n 1_{f(\mathbf{x}_i)\neq y_i}$ where $1_{f(\mathbf{x}_i)\neq y_i} = 1$ if $f(\mathbf{x}_i) \neq y_i$ and 0 otherwise. The generalisation error of f is defined by $R_{\mathcal{D}}(f) = \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}}\left[1_{f(\mathbf{x})\neq y}\right]$.

We start with a bound on the generalisation error $R_{\mathcal{D}}(f)$ given by (Mohri, Rostamizadeh & Talwalkar 2018, Corollary 3.9).

Lemma 4.1 *(Mohri et al. 2018, Corollary 3.9) For any $\delta > 0$, with probability at least $1 - \delta$, for any $f \in \mathcal{F}$,*

$$R_{\mathcal{D}}(f) \leq \widehat{R}_D(f) + \sqrt{\frac{2\log \Pi_{\mathcal{F}}(n)}{n}} + \sqrt{\frac{\log \frac{1}{\delta}}{2n}}, \tag{4.9}$$

where $\Pi_{\mathcal{F}}(n)$ is the growth function for the hypothesis set $\mathcal{F}$ with n samples.

The growth function $\Pi_{\mathcal{F}}(n)$ is the maximum number of distinct sign-patterns on n samples that can be produced with functions in $\mathcal{F}$. We will give the bound and formal definition on $\Pi_{\mathcal{F}}(n)$ latter.

We restate the following Lemma (Bartlett, Harvey, Liaw & Mehrabian 2019, Lemma 17) for bounding the growth function:

Lemma 4.2 *(Bartlett et al. 2019, Lemma 17) Let $P_1, P_2, ..., P_n$ be n real polynomials in l real variables, and suppose the degree of each P_i does not exceed t. If $n \geq l$ then $s(P_1, P_2, ..., P_n) \leq 2(2ent/l)^l$ with $s(P_1, P_2, ..., P_n)$ denotes the total number of sign-patterns of the polynomials $P_1, P_2, ..., P_n$.*

Lemma 4.2 provides a bound for sign patterns of polynomials. This bound assumes $P_i \neq 0$. This coincides with most of the practical cases. If we would like to consider a more complete setting that allows $P_i = 0$, we can set $\text{sign}(0) = 1$ and follow the results in (Alon 1995, Proposition 5.5) to obtain $s(P_1, P_2, ..., P_n) \leq (8ent/l)^l$.

We then give the definition of the growth function $\Pi_{\mathcal{F}}(n)$ and its bound by following theorem.

Theorem 4.3 *The growth function $\Pi_{\mathcal{F}}(n)$ of hypothesis set $\mathcal{F}$ on n samples is defined and bounded by:*

$$\Pi_{\mathcal{F}}(n) = \max_{\{\mathbf{x}_1,...,\mathbf{x}_n\} \subseteq \mathcal{X}} \left| \left\{ \big(sign(f(\mathbf{x}_1)), ..., sign(f(\mathbf{x}_1))\big) : f \in \mathcal{F} \right\} \right| \leq 2(\frac{2ent}{l})^l, \tag{4.10}$$

where $t = 2$ and $l = k(d + 2d')$.

Proof 4.3 *We use the definition of growth function following (Mohri et al. 2018, Definition 3.6), where $\Pi_{\mathcal{F}}(n)$ is the maximum number of distinct ways in which n points can be classified using hypotheses in $\mathcal{F}$.*

Consider $f(\mathbf{x}_1), ..., f(\mathbf{x}_n)$ to be n real polynomials. Because $rank(H) \leq k$, H can be decomposed into product of $U^\top$ and V with $U \in \mathbb{R}^{k \times d}$ and $V \in \mathbb{R}^{k \times 2d'}$. Treat elements of U and V as variables, so that the degree of each polynomial $f(\mathbf{x}_i)$ is 2 and we have $k(d + 2d')$ variables. Apply Lemma

4.2, and notice that the total number of sign-patterns equals to the number of distinct ways n points can be classified, we complete the proof.

Substitute the result of theorem 4.3 into Lemma 4.1, we immediately arrive at the following corollary.

Corollary 4.4 *For any $f \in \mathcal{F}$ and $\delta > 0$, following generalisation error bound holds with probability at least $1 - \delta$:*

$$R_{\mathcal{D}}(f) \leq \widehat{R}_D(f) + \sqrt{\frac{2k(d+2d')\log\frac{4en}{k(d+2d')} + \log 4}{n}} + \sqrt{\frac{\log\frac{1}{\delta}}{2n}}, \tag{4.11}$$

The rank k of H, the feature dimension d, the dimension d' of missingness indicator vector $\mathbf{m}$ and the sample size n jointly represent the upper bound of generalisation error in above corollary. Clearly this bound decreases when sample size n increases. A low-dimensional feature vector $\mathbf{x}$ and a low-dimensional missingness pattern indicator vector $\mathbf{m}$ are both beneficial to the model generalisation. It also shows that appropriately constrain the rank k of H can be helpful to reduce the error.

4.5 Efficient Training Procedure

The optimisation of Eq.(4.8) is based on stochastic gradient descent with PyTorch (Paszke, Gross, Chintala, Chanan, Yang, DeVito, Lin, Desmaison, Antiga & Lerer 2017) implementation. We discuss the learning problem with regard to Eq.(4.7) in this section. It is non-convex due to the rank constraint. Notice that H can be decomposed as $H = U^\top V$ with $U \in \mathbb{R}^{k\times d}$ and $V \in \mathbb{R}^{k\times 2d'}$. Then the loss function associated with Eq.(4.7) is convex regarding U with fixed V and vice versa. We can optimise them alternatively until convergence. A straightforward way to minimise the loss function is through the subgradient method. We fix some subgradient oracles for U and

V as:

$$g_U = \frac{2}{n} V\bar{M}\left(M^\top \odot \left(\bar{M}^\top V^\top U\right)\right) + 2\eta_1 VV^\top U - \frac{\eta_2}{n}\sum_{i\in\mathcal{I}_{sv}} y_i V\bar{\mathbf{m}}_i \mathbf{x}_i^{o\top} \quad (4.12)$$

$$g_V = \frac{2}{n} U\left(M \odot \left(U^\top V\bar{M}\right)\right)\bar{M}^\top + 2\eta_1 UU^\top V - \frac{\eta_2}{n}\sum_{i\in\mathcal{I}_{sv}} y_i U\mathbf{x}_i^o\bar{\mathbf{m}}_i^\top \quad (4.13)$$

where $\mathcal{I}_{sv}$ denotes indices of support vectors. Given the subgradients, we can optimise U with fixed V and optimise V with fixed U iteratively until convergence. Below we will introduce the subroutine for optimising U with fixed V.

We adopt the Restarted SubGradient method (Yang & Lin 2018) in the training process. Let α_s and T_s be the step-size and number of iterations in stage s. In each stage, we adopt the following update rule:

$$U_s^{t+1} = U_s^t - \alpha_s \frac{g_{U_s^t}}{\left\|g_{U_s^t}\right\|_F},\ t = 1, \cdots, T_s, \quad (4.14)$$

and output $U_{s+1}^1 = \arg\min_{U\in\{U_s^1,\ldots,U_s^{T_s+1}\}} F(U)$, where $F(U)$ is the loss function associated with U.

With α_1 and T_1 set appropriately, α_s and T_s are updated by:

$$\alpha_{s+1} = \frac{1}{2}\alpha_s,\ T_{s+1} = \frac{1}{2}T_s, \quad (4.15)$$

Algorithm 4.1 presents the training procedure for optimising U. Calculating the subgradient requires linear time regarding n, d, d' and square time regarding the rank k.

4.6 Proof of Convergence

In this section, we discuss the rate of convergence of Algorithm 4.1. The key factor that influences the overall convergence is the convergence of subroutines to optimise U and V. We will discuss the rate of convergence regarding optimisation of U given V. For optimisation of V, a similar result holds, and we omit the details here.

Algorithm 4.1 Subroutine for optimising U

Input: U_1^1, V, the number of stages S.

Output: U_{S+1}^1.

Initialisation: $\epsilon_0 = F(U_1^1)$. Calculate C, γ, L_Φ, L_h, α_1, T_1.

for $s = 1$ *to* S **do**

  $\alpha_s = (\frac{1}{2})^{s-1}\alpha_1$; $T_s = 2^{s-1}T_1$;

  for $t = 1$ *to* T_s **do**

    Calculate U_s^{t+1} by Eq.(4.14);

  $U_{s+1}^1 = \arg\min_{U \in \{U_s^1, \ldots, U_s^{T_s+1}\}} F(U)$;

Our loss function is non-Lipschitz and can not be guaranteed to be strongly-convex regarding U. These are often required for deriving a convergence rate for subgradient methods. Thanks to the work in Necoara, Nesterov & Glineur and Grimmer, we can show that this algorithm gives an ϵ-approximate solution in $O(\frac{1}{\epsilon})$ iterations.

We first present our main theorem regarding the rate of convergence.

Theorem 4.5 *Let* $\epsilon_0 = F(U_1^1)$ *and* F^* *be the minima of* $F(U)$. *Let* $\gamma = \max(\sqrt{8L_\Phi}, 8L_h)$, $C = \frac{1}{\eta_1 \sigma_{min}^2(V)^+}$ *where* $\sigma_{min}(V)^+$ *is the smallest non-zero singular value of* V. *Set*

$$\begin{aligned} \alpha_1 &= \frac{\epsilon_0}{\gamma\sqrt{\eta_2}}, \\ T_1 &= \lceil \frac{1}{\epsilon_0} \max(\sqrt{\frac{8L_\Phi}{9}}\eta_2 C\gamma, 8L_h\sqrt{\eta}_2 C\gamma) \rceil, \end{aligned} \tag{4.16}$$

where $\lceil \cdot \rceil$ *denotes the ceiling function, then Algorithm 4.1 requires* $O\big(\frac{\sqrt{\eta_2}C\gamma}{\epsilon}\max(\sqrt{\frac{8}{9}L_\Phi\eta_2}, 8L_h)\big)$ *iteration complexity with total* $S = \lceil log_2(\frac{\epsilon_0}{\epsilon}) \rceil$ *stages to output* U *that satisfies* $F(U) - F^* \leq \epsilon$.

Proof sketch: We will firstly bound the Euclidean distance between U and the optimal solution U^*, with the difference of loss function $F(U) - F^*$. Then we can substitute this upper bound recursively into the upper bound

regarding subgradient method provided in (Grimmer 2019), which completes the proof.

Firstly, we give following lemma to bound the distance between U and the optimal solution U^*.

Lemma 4.6 *Denote by $\mathcal{U}^*$ the optimal set contains all minimisers of F. Let U^* denote the element in $\mathcal{U}^*$ which is closest to U. The following holds:*

$$\|U - U^*\|_F^2 \leq C(F(U) - F^*) \tag{4.17}$$

with a constant $C = \frac{1}{\eta_1 \sigma_{min}^2(V)^+}$ and F^* is the minimal value of F.

Proof 4.6 *Our loss function regarding $H = U^\top V$ has the form of:*

$$K(H) = \frac{1}{n}\|M \odot (H\bar{M})\|_F^2 + \eta_1 \|H\|_F^2 + \frac{\eta_2}{n}\sum_{i=1}^{n} \ell(y_i, \bar{\mathbf{m}}_i^\top H^\top \mathbf{x}_i^o) \tag{4.18}$$

Clearly $K(H)$ is a ρ-strongly convex function with $\rho \geq 2\eta_1$. Following proof of (Necoara, Nesterov & Glineur 2019, Theorem 8), the set of optimal solutions regarding minimising F is $\mathcal{U}^* = \{U : U^\top V = \Omega^*\}$. Given V and U, by definition of U^* we have:

$$U^* = \min_{U' \in \mathcal{U}^*} \|U' - U\|_F^2 \tag{4.19}$$

From KKT conditions of Eq.(4.19) we know $\mathbf{u}_i^* - \mathbf{u}_i + V\boldsymbol{\beta}_i = \mathbf{0}$ where $\mathbf{u}_i^*$, $\mathbf{u}_i$ and $\boldsymbol{\beta}_i$ denote i-th column vectors of U^*, U and related Lagrange multipliers respectively. This implies $\mathbf{u}_i^* - \mathbf{u}_i \in \mathrm{Im}(V)$. From Courant-Fischer theorem we know:

$$\|V^\top \mathbf{u}_i - V^\top \mathbf{u}_i^*\|_2 \geq \sigma_{min}(V)^+ \|\mathbf{u}_i - \mathbf{u}_i^*\|_2 \tag{4.20}$$

Apply Eq.(4.20) to every column of U we get:

$$\|V^\top U - V^\top U^*\|_F^2 \geq \sigma_{min}^2(V)^+ \|U - U^*\|_F^2 \tag{4.21}$$

By definition of strongly-convex function:

$$K(H_1) \geq K(H_2) + \langle k(H_2), H_1 - H_2 \rangle + \frac{\rho}{2}\|H_1 - H_2\|_F^2 \tag{4.22}$$

where $k(H_2) \in \partial K(H_2)$ is any subgradient of K at H_2. Let $H_1 = U^\top V$ and $H_2 = (U^*)^\top V$, and notice that $K(U^\top V) = F(U)$. We have:

$$F(U) \geq F^* + \left\langle Vk((U^*)^\top V)^\top, U - U^* \right\rangle + \frac{\rho\sigma^2_{min}(V)^+}{2}\|U - U^*\|_F^2 \tag{4.23}$$

Because $V\partial K((U^*)^\top V)^\top = \partial F(U^*)$. According to optimality conditions of subgradient method, we can choose $Vk((U^*)^\top V)^\top = \mathbf{0} \in \partial F(U^*)$. Thus,

$$\frac{\rho\sigma^2_{min}(V)^+}{2}\|U - U^*\|_F^2 \leq F(U) - F^* \tag{4.24}$$

Because $\rho \geq 2\eta_1$,

$$\|U - U^*\|_F^2 \leq \frac{1}{\eta_1\sigma^2_{min}(V)^+}(F(U) - F^*) \tag{4.25}$$

which completes the proof.

Our loss function has the form of $F(U) = \Phi(U) + h(U)$ with:

$$\Phi(U) = \frac{1}{n}\|M \odot (U^\top V\bar{M})\|_F^2 + \eta_1\|U^\top V\|_F^2 \tag{4.26}$$

$$h(U) = \frac{\eta_2}{n}\sum_{i=1}^{n}\ell(y_i, \bar{\mathbf{m}}_i^\top V^\top U\mathbf{x}_i^o) \tag{4.27}$$

where ℓ is the hinge loss define by $\ell(y, \hat{y}) = \max(0, 1 - y\hat{y})$. One can easily verify that $\Phi(U)$ has L_Φ-Lipschitz gradient and $h(U)$ is an L_h-Lipschitz function. Then another useful Lemma we will use recursively is:

Lemma 4.7 *With one stage in Algorithm 4.1, we have*

$$\min_{t=1\ldots T_s}\{F(U_s^t) - F^*\} \leq \frac{L_\Phi}{2}\left(\frac{\|U_s^1 - U^*\|_F^2}{2T_s\alpha_s} + \frac{\alpha_s}{2}\right)^2 + 2L_h\left(\frac{\|U_s^1 - U^*\|_F^2}{2T_s\alpha_s} + \frac{\alpha_s}{2}\right) \tag{4.28}$$

Lemma 4.7 is proved in (Grimmer 2019) by firstly applying (Grimmer 2019, Lemma 2.3) to our loss function $F(U)$ and then applying (Grimmer 2019, Theorem 1.2).

Combine Lemma 4.6 and Lemma 4.7 and notice that $F(U_s^1) \geq F(U_s^1) - F^* \geq 0$ we get:

$$\min_{t=1...T_s}\{F(U_s^t) - F^*\} \leq \frac{L_\Phi}{2}\left(\frac{CF(U_s^1)}{2T_s\alpha_s} + \frac{\alpha_s}{2}\right)^2 + 2L_h\left(\frac{CF(U_s^1)}{2T_s\alpha_s} + \frac{\alpha_s}{2}\right) \tag{4.29}$$

We assume that $F(U_s^1) \leq \eta_2$. This could be easily guaranteed by setting $U = 0$ at initialisation so that $F(0) \leq \eta_2$. When $\eta_2 \geq 1$, set the step size α_s and number of iteration T_s as:

$$\alpha_s = \frac{F(U_s^1)}{\gamma\sqrt{\eta}_2} \tag{4.30}$$

$$T_s = \lceil \frac{1}{F(U_s^1)} \max(\sqrt{\frac{8L_\Phi}{9}}\eta_2 C\gamma, 8L_h\sqrt{\eta}_2 C\gamma)\rceil \tag{4.31}$$

We can obtain $\min_{t=1,...,T_s+1}\{F(U_s^t) - F^*\} \leq \frac{F(U_s^1)}{2}$. We choose the best U in stage s as the initial values in stage $s+1$ following $U_{s+1}^1 = \arg\min_{U\in\{U_s^1,...,U_s^{T_s+1}\}} F(U)$, so that $\|U_{s+1}^1 - U^*\|_F^2 \leq C(F(U_{s+1}^1) - F^*) \leq \frac{CF(U_s^1)}{2}$. Applying the inequality recursively, we get $\alpha_{s+1} = \frac{\alpha_s}{2}$, $T_{s+1} = 2T_s$, $\min_{t=1,...,T_S+1}\{F(U_S^t) - F^*\} \leq \frac{F(U_1^1)}{2^S}$. In order to get U that satisfies $F(U) - F^* \leq \epsilon$, Algorithm 4.1 requires $S = \lceil log_2(\frac{\epsilon_0}{\epsilon})\rceil$ stages with $\epsilon_0 = F(U_1^1)$. Summing up the iterations for all stages and noticing that it is a geometric series, gives the iteration complexity $O\big(\frac{\sqrt{\eta_2}C\gamma}{\epsilon}\max(\sqrt{\frac{8}{9}L_\Phi\eta_2}, 8L_h)\big)$.

When $\eta_2 < 1$, the iteration complexity will be smaller than $O\big(\frac{C\gamma}{\epsilon}\max(\sqrt{\frac{8}{9}L_\Phi}, 8L_h)\big)$ to satisfy $F(U) - F^* \leq \epsilon$. This can be verified by setting $\eta_2 = 1$ in Eq.(4.30), Eq.(4.31) and substituting them recursively into Eq.(4.29). Thus we complete the proof.

Theorem 4.5 shows that Algorithm 4.1 has sublinear convergence rate.

4.7 Experiments

In this section, we present experiments on some real datasets with internally missing attributes as well as artificially missing entries.

Table 4.1: Summary of datasets

Dataset	Instances	Features	% of Missing
bands	539	19	5.38%
hepatitis	155	19	5.67%
horse	368	22	23.80%
mammographic	961	5	3.37%
pima	768	8	12.24%

4.7.1 Linear Model

We apply our method learned through Eq.(4.7) on several real datasets retrieved from UCI repository (Dua & Graff 2017) with internally missing attributes. Table 4.1 summarises the datasets. We randomly split the dataset into 70% training set and 30% test set. To consider a more general case, we randomly removed 30% of the values in the training set and test set. As a result, the missing rate would be higher than 30% for all datasets, and the missingness mechanisms are more complex than the original datasets.

We considered methods with publicly available codes. These baselines like zero, mean and KNN imputation are quite general methods that have been widely used in many CBM tasks (Bennane & Yacout 2012). We compared our method with the following baselines:

- **Flag**: Additional binary features indicating the missingness pattern of a given instance were concatenated to the original feature vector. The missing values of the original feature vector were set to zero.
- **Zero**: Missing values were set to zero.
- **Mean**: Missing features were set to averages of corresponding features from other instances that were not missing.
- **KNN**: Missing features of an instance were filled with means of those features calculated from the K-nearest neighbours of this instance. The

neighbourhood was measured using Euclidean distance with observed features. The K was chosen from {3, 4, 5}.

- **GMM**: Missing values in the training set were filled in an iterative way between two steps: (1) learning a GMM with the filled data and (2) re-filling missing values using components' means, weighted by the posterior probabilities of related components generated the sample. For the test set, we used the learned GMM to iteratively fill the missing values until convergence according to step (2). We chose the number of the mixture components from {3, 4, 5}. This idea is similar to that in (Ghahramani & Jordan 1994, Ouyang et al. 2004, Chechik et al. 2008).

- **MICE**: MICE iteratively imputed each missing feature by regression model based on other features (Azur, Stuart, Frangakis & Leaf 2011). We chose the linear regression to fit the models.

- **geom**: This method was proposed by Chechik et al.. It considers sample-specific margins. We used the iterative algorithm as suggested there with 5 iterations. The parameter C were selected from $\{10^{-5},...,10^{5}\}$.

- **karma**: This algorithm was presented in (Hazan, Livni & Mansour 2015). It trained a classifier under the low-rank assumption of data. The parameters γ and C were selected from {1, 2, 3, 4} and $\{10^{-5},...,10^{5}\}$.

We combined the Flag, Zero, Mean, KNN, GMM and MICE with Support Vector Machines (SVM) and chose the parameter C for SVM from $\{10^{-5},...,10^{5}\}$. Data were normalised to zero mean and unit covariance after imputation for imputation-based methods and normalised based on observed features for geom, karma and our method. We fixed $\eta_1 = 10^{-6}$ for our method. η_2 and k were chosen from $\{10^{-5},...,10^{5}\}$ and $\{2, 4, ..., d\}$ where d is the feature dimension of related dataset. All the hyper-parameters are selected based on 5-fold cross-validation on training sets.

Experiment results are presented in Table 4.2. We repeated the experiments 5 times to report the classification accuracy with mean and standard deviation.

Our method achieved the best accuracy on all 5 datasets with additional 30% entries removed. In general, our method is better than imputation methods, because inaccurate imputation could deteriorate the downstream classification task. Our method also outperforms Flag, which indicates that simply adding the missingness pattern as additional features is not as good as our strategy. These datasets contain inherent missing features, and we also removed some values randomly. These factors make the missingness mechanism complicated, and it is hard to learn a universal model that fits all these heterogeneous missingness patterns. Our method tries to adaptively apply the classifiers specialised to different missingness patterns, which makes it capable of learning some missingness-pattern specific classifiers.

We present experiments with the original datasets here in Table 4.3. The original datasets contain internally absent attributes. Our model consistently outperforms other baselines except on pima dataset. The performance gaps between all models are relatively small due to the low missing percentages.

4.7.2 Neural Networks

We compare our method with other baselines based on neural networks. The experiments were conducted on three datasets:

- **Sensorless Drive Diagnosis** (Bayer, Enge-Rosenblatt, Bator & Mönks 2013): This dataset is retrieved from UCI repository (Dua & Graff 2017). It consists of 11 classes, 58509 instances, and each instance has 49 features. The datasets were randomly split into 50% training set and 50% test set. We randomly selected 25% of the training data as the validation set.

- **MNIST** (LeCun, Cortes & Burges 2010): This is a dataset for classification of handwritten digits. The dataset contains 784 features and

Table 4.2: Classification accuracy (mean±std) with additional 30% entries removed for all datasets. The best results are bold and the second best are underlined.

	Dataset				
Method	bands	hepatitis	horse	mammographic	pima
Flag	0.583±0.006	<u>0.845±0.016</u>	0.825±0.007	0.764±0.006	0.737±0.022
Zero	<u>0.597±0.022</u>	0.842±0.017	0.816±0.015	0.761±0.018	0.736±0.010
Mean	0.586±0.008	0.843±0.017	0.822±0.013	<u>0.774±0.009</u>	0.740±0.002
MICE	0.575±0.027	0.774±0.044	0.712±0.041	0.772±0.016	0.738±0.023
GMM	0.572±0.021	0.825±0.037	0.805±0.013	0.768±0.012	0.742±0.021
KNN	0.592±0.016	0.812± 0.037	<u>0.836±0.026</u>	0.762±0.006	<u>0.747±0.009</u>
geom	0.575±0.023	0.834±0.025	0.819±0.022	0.762±0.009	0.742±0.006
karma	0.551±0.040	0.817±0.032	0.759±0.022	0.759±0.014	0.740±0.009
Ours	**0.648±0.021**	**0.868±0.009**	**0.840±0.011**	**0.776±0.010**	**0.756±0.006**

Table 4.3: Classification accuracy (mean±std) on original datasets. The best results are bold.

	Dataset				
Method	bands	hepatitis	horse	mammographic	pima
Flag	0.617±0.000	**0.872±0.000**	0.864±0.000	0.778±0.000	0.783±0.000
Zero	0.606±0.002	0.851±0.000	0.838±0.000	0.796±0.000	0.801±0.000
Mean	0.611±0.002	**0.872±0.000**	0.847±0.000	0.796±0.000	0.792±0.000
MICE	0.617±0.000	0.809±0.000	0.856±0.000	0.785±0.000	0.775±0.000
GMM	0.594±0.013	**0.872±0.021**	0.841±0.011	0.779±0.010	0.787±0.023
KNN	0.593±0.010	0.847±0.009	0.847±0.000	0.775±0.002	**0.805±0.000**
geom	0.605±0.000	**0.872±0.000**	0.865±0.000	0.789±0.000	0.792±0.000
karma	0.611±0.040	0.809±0.000	0.838±0.000	**0.799±0.000**	0.797±0.005
Ours	**0.678±0.005**	**0.872±0.000**	**0.876±0.007**	**0.799±0.001**	0.791± 0.002

has a training set of 60000 examples and a test set of 10000 examples. We randomly selected 20% of data in the training set as the validation set.

- **Avila** (De Stefano, Maniaci, Fontanella & di Freca 2018): The Avila data set has been extracted from 800 images of the "Avila Bible", an XII century giant Latin copy of the Bible. The prediction task consists in associating each pattern to a copyist. It consists of 12 classes, 20867 instances, and each instance has 10 features.

karma and geom cannot be applied to neural networks, so we omit them here. MICE cannot scale to MNIST dataset due to the high dimensionality of feature vectors. We compared an additional method proposed recently in (Śmieja, Struski, Tabor, Zieliński & Spurek 2018) and named it **PMNN**. The number of components of GMM for PMNN was chosen from {3, 4, 5}. We did not compare with other neural networks for classification since they required complete instances for training. We compared all the baselines based on a multilayer perceptron (MLP) consists of 3 ReLU hidden layers with 100 neurons per layer. We used the cross-entropy loss as the loss function in training. All hyper-parameters is selected based on the validation set. The range of hyper-parameters was similar to the linear model except that k was chosen from $\{2^1, 2^2, ..., 2^{\log_2 d}\}$ where d is the feature dimension. Because these datasets were complete, we randomly removed 10%, 30%, 50%, 70%, 90% of values in them. We repeated this procedure 5 times to report the classification accuracy with mean and standard deviation.

Table 4.4, Table 4.5 and Table 4.6 show the results of our method together with some baselines. The results show the advantage of our method over classical imputation methods and PMNN. Notice that PMNN produced a poor result when the missing ratio was low. PMNN is required to fit a GMM together with the neural network, but the GMM of PMNN is only trained with incomplete instances. Unlike GMM for imputation, where all data are used to fit the GMM, their model cannot be trained well when the percentage of missing is low. Flag shows good performance on Sensorless

Table 4.4: Classification accuracy (mean±std) on Sensorless Drive Diagnosis dataset. The best results are bold and the second best are underlined.

	Percentage of missing				
Method	10%	30%	50%	70%	90%
Zero	0.908±0.001	0.852±0.011	0.769±0.005	0.618±0.005	0.317±0.002
Mean	0.947±0.004	0.907±0.001	0.816±0.003	0.650±0.005	0.329±0.003
MICE	0.717±0.001	0.422±0.009	0.483±0.010	0.322±0.007	0.197±0.007
GMM	0.938±0.002	0.890±0.005	0.805±0.007	0.601±0.007	0.327±0.003
KNN	0.936±0.004	0.847±0.003	0.725±0.003	0.398±0.004	0.215±0.005
Flag	<u>0.970±0.001</u>	<u>0.925±0.001</u>	<u>0.834±0.002</u>	<u>0.677±0.003</u>	<u>0.345±0.004</u>
PMNN	0.230±0.001	0.886±0.001	0.781±0.001	0.649±0.002	0.318±0.001
Ours	**0.976±0.001**	**0.940±0.001**	**0.858±0.002**	**0.695±0.002**	**0.351±0.002**

Table 4.5: Classification accuracy (mean±std) on MNIST dataset. The best results are bold and the second best are underlined.

	Percentage of missing				
Method	10%	30%	50%	70%	90%
Zero	0.957±0.001	0.942±0.002	0.918±0.002	0.863±0.003	0.688±0.003
Mean	0.964±0.001	<u>0.951±0.001</u>	<u>0.933±0.001</u>	<u>0.891±0.003</u>	<u>0.727±0.004</u>
GMM	0.963±0.002	0.925±0.003	0.806±0.011	0.636±0.006	0.379±0.012
KNN	<u>0.965±0.001</u>	0.941±0.002	0.864±0.001	0.703±0.023	0.223±0.012
Flag	0.962±0.001	0.935±0.002	0.908±0.003	0.847±0.012	0.360±0.045
PMNN	0.933±0.001	0.910±0.002	0.883±0.003	0.842±0.002	0.700±0.004
Ours	**0.970±0.001**	**0.958±0.001**	**0.940±0.001**	**0.900±0.002**	**0.739±0.004**

Table 4.6: Classification accuracy (mean±std) on Avila dataset. The best results are bold and the second best are underlined.

	Percentage of missing				
Method	10%	30%	50%	70%	90%
Zero	0.722±0.002	0.630±0.005	0.553±0.006	<u>0.496±0.004</u>	<u>0.433±0.002</u>
Mean	0.718±0.005	0.630±0.005	<u>0.556±0.003</u>	0.492±0.003	0.433±0.002
MICE	0.717±0.005	0.618±0.005	0.435±0.007	0.422±0.002	0.412±0.001
GMM	0.722±0.004	<u>0.633±0.004</u>	**0.557±0.003**	0.470±0.002	0.432±0.002
KNN	<u>0.746±0.004</u>	0.620±0.002	0.536±0.006	0.474±0.003	0.426±0.002
Flag	0.713±0.005	0.630±0.004	0.555±0.003	0.491±0.002	<u>0.433±0.002</u>
PMNN	0.333±0.004	0.445±0.003	0.526±0.003	0.473±0.004	0.412±0.001
Ours	**0.765±0.002**	**0.646±0.004**	0.548±0.003	**0.496±0.003**	**0.434±0.001**

Drive Diagnosis dataset. However, its performance is limited on MNIST dataset, possibly because of the higher dimensionality of MNIST dataset. This indicates that the missingness patterns can be important in learning with incomplete data, but should be wisely incorporated into the model equation. Our model consistently outperforms other baselines, which verifies the effectiveness of our strategy to adjust the importance of present features by the missingness patterns.

4.8 Conclusion

This chapter proposed a general method for learning with incomplete data, where data of different missingness patterns are treated differently in the model level. This reduces the competition between data of different missingness patterns in training. A linear model was proposed that can be adaptively applied to data with different missingness patterns. Analysis of error bound justifies this model in the linear case. Analysis of generalisation for the neural network is a challenging topic that being postponed to future work. Our experiment results verified the effectiveness of our model empirically. The dimension of missingness indicator vectors plays an important role in the computation complexity and generalisation error. Our future work will focus on how to develop a lower-dimension representation for missingness indicator vectors. Although we do not impute the missing data for the proposed method, it does not conflict with imputation methods. How to combine various imputation methods with our model is another interesting future work.

Chapter 5

Sample-adaptive Multiple-kernel Learning for Learning with Multi-source Data

In practice, CBM data can be generated from multiple sources with different format and frequency. For different pieces of equipment, the available data sources can differ. Some of data sources can be absent due to transmission error, human misplay, or sensor malfunction. This chapter introduced the sample-adaptive multiple-kernel learning for the combination of multi-source data. Similar to Chapter 3, a case study was conducted on the multi-source data generated by railway points of a large-scale railway network.

5.1 Introduction

A railway junction is controlled jointly by one or more ends of points. They work together to control the routes of trains. We treat the set of railway points in a railway junction as a whole to predict their failures.

Apart from the delay and cancellation of trains, failure of points can also

cause severe economic loss and casualties. Railway points count for almost half of all train derailments in the UK (Ishak, Dindar & Kaewunruen 2016). On the morning of 12 December 1988, Clapham Junction rail crash [1] killed 35 people and injured 484 people. More than 20% of incidents in Sydney Trains rail network were caused by points failures. Maintaining railway points safe, and forecasting the incoming failure are vital tasks for reliable rail transportation.

Routine maintenance is usually performed on railway points to ensure the correctness and reliability of them. Such work is done by field engineers to inspect and test the equipment at a fixed time interval. However, this strategy cannot catch the rapid change of equipment status. For example, when extreme weather occurs, points often degrade faster than usual. As a result, they are more likely to fail soon. Instead of relying on passive routine maintenance, we could benefit more from predictive maintenance - which flexibly arranges the maintenance work according to the running condition of equipment.

Forecasting the failures is a critical step in predictive maintenance. Some research has been conducted on this topic (Camci et al. 2016, García Márquez et al. 2010, Oyebande & Renfrew 2002, Tao & Zhao 2015, Yilboga et al. 2010). Delicate sensors usually serve as data collectors for voltages, currents and forces in related work. Installation of sensors incurs costly labour and material expenses, as well as the possibility of sensor malfunction. Adding sensors for in-service equipment would also induce disruption to traffic. This is especially unacceptable for a large and busy rail network. These make the prediction with sensors' data expensive, or even infeasible. On the contrary, one can easily collect heterogeneous data from other sources such as weather, movement logs, and equipment details without an additional hardware upgrade.

Gathering available data from multiple sources enriches our knowledge on the working status of points. However, this also brings extra problems.

[1] `https://en.wikipedia.org/wiki/Clapham_Junction_rail_crash`

Firstly, data collected from different sources are often in incompatible formats, and they play different roles in revealing the condition of equipment. Secondly, we are not guaranteed that data are always intact - even for a single source. Actually, in most case, we can only feed incomplete data into our model. Besides, our data were collected upon 350 sets of railway points. They are possibly located in a rural area, city centre, or from a different point of view, bridges, tunnels. They can also be of various types and made by different manufacturers. These add up to the difficulties in designing models. To summarise, we are faced with three main challenges here:

- How to combine information from multiple sources efficiently and effectively?
- How to deal with missing data?
- How to consider the distinct and shared properties between different sets of railway points simultaneously?

To address these challenges, this chapter proposed a novel multiple kernel learning algorithms. This method was developed based on the multiple-kernel learning framework (Gönen & Alpaydın 2011). Multiple-kernel learning has attracted much attention over the last decade. It has been regarded as a promising technique for combining multiple data channels or feature subsets (Xu, Jin, Yang, King & Lyu 2010), which exactly meets our requirements. Different kernel mapping functions were applied to our data from different sources. Besides, we also concatenated all the data to form a data source so that the inter-source correlations could be found. An adaptive kernel weight determined by both properties of each individual set of railway points and the missingness pattern of data makes our model robust, effective and unique.

The main contributions of this work can be shown in the following aspects:

- A universal framework was provided to predict points' failure with multi-source data. Our data are easy to obtain for most of the rail networks over the world without a hardware upgrade, and thus could be used in many other rail networks.

- This work firstly introduces missingness-pattern-adaptive kernel weight into existing multiple-kernel learning framework.
- With a sample adaptive kernel weight, the proposed model can capture the distinct and share properties of different railway points.
- An optimisation algorithm was developed to optimise the proposed model. Through random feature approximation together with mini-batch gradient descent, the proposed method can be applied on large datasets.
- Experiments were conducted on a real-world dataset collected from a wide range of railway points over three years. The results clearly show the effectiveness of the proposed method.

5.2 Backgrounds

We give a brief introduction to failure prediction of railway points and the formulation of multiple-kernel learning (MKL) algorithm.

5.2.1 Failure Prediction of Railway Points

Knowing that railway points directly affect the capacity and reliability of rail transport, some research has been conducted on failure prediction of railway points (Camci et al. 2016, García Márquez et al. 2010, Oyebande & Renfrew 2002, Tao & Zhao 2015, Yilboga et al. 2010). Sensor data such as voltages, currents and forces were widely used in these works. They were collected in laboratories or from site sensors. These data would require a high sampling rate and lead to difficulties in both transmission and storage. Despite the success shown in these methods, they can be impractical in the real application.

Limited work explored the prediction task with data from another source. Weather plays a significant role in the probability of failure

(Hassankiadeh 2011), and has been used to predict the total number of failed turnout systems in a railway network (Wang, Xu, Tang, Yuan & Wang 2017). Note that this work could not locate the exact fault railway points, it only estimated the total number of failures in a large system. Apart from weather data, equipment logs were also valuable information for foreseeing the failures of related equipment (Sipos, Fradkin, Moerchen & Wang 2014). Logs can be generated by sensors, software applications and even maintenance records (Li, Zhang, Wu & Kirsch 2018), reflecting the working condition of a piece of equipment in a different view. In (Li, Zhang, Wu & Kirsch 2018), maintenance logs were used to forecast the failure between two scheduled maintenance.

Many of above-mention methods used support vector machines (SVM) (Chang & Lin 2011) for their models. They mainly focused on data from one source. A natural extension is to use multiple-kernel learning to formulate our multi-source problem and increase prediction accuracy.

5.2.2 Preliminaries

In this section, we introduce the formulation of MKL, based on which we will formulate our models. Compared to deep neural networks, MKL enjoys better interpretability while requires less training data. This is important for CBM because the outputs of the model can also be interpreted to discover important factors that cause failures.

MKL is designed to deal with several feature sets, either generated from single data source with different kernel functions or different data sources with predefined kernel functions. The learning objective of MKL is then finding an optimal combination of these feature sets. Following our application, we will give the formulation of MKL assuming multiple data sources with predefined kernel functions are available.

For sample $\mathbf{x}_i = [\mathbf{x}_i^{(1)^\top}, \mathbf{x}_i^{(2)^\top}, \cdots, \mathbf{x}_i^{(s)^\top}]^\top$ consists of s feature subsets (formed by concatenation of feature vectors), by applying s mapping func-

tions to each subset, it takes the form of:

$$\phi(\mathbf{x}_i) = [\phi_1^\top(\mathbf{x}_i^{(1)}), \phi_2^\top(\mathbf{x}_i^{(2)}), \cdots, \phi_s^\top(\mathbf{x}_i^{(s)})]^\top, \tag{5.1}$$

where $\{\phi_m(\cdot)\}_{m=1}^s$ denote feature maps associated with m pre-defined base kernels $\{\kappa_m(\cdot,\cdot)\}_{m=1}^s$, e.g. identity map for the linear kernel. Given samples $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ with $y_i \in \{-1, +1\}$ the label for $\mathbf{x}_i$, commonly used MKL can be formulated as the following optimisation problem (Lanckriet et al. 2004):

$$\begin{aligned} \min_{\boldsymbol{\omega},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\left(\sum_{m=1}^{s} \|\boldsymbol{\omega}_m\|_2\right)^2 + C\sum_{i=1}^{n}\xi_i \\ s.t. \quad & y_i\left(\sum_{m=1}^{s} \boldsymbol{\omega}_m^\top \phi_m(\mathbf{x}_i^{(m)}) + b\right) \geq 1 - \xi_i, \\ & \xi_i \geq 0, \quad i = 1, 2, ..., n, \end{aligned} \tag{5.2}$$

where $\|\cdot\|_2$ is the Euclidean norm for vectors or can be replaced by norm defined in related Hilbert space. $\boldsymbol{\omega}_m$ is the weight vectors for mapped features $\phi_m(\mathbf{x}_i^{(m)})$. b is the bias term and C is a regularisation parameter for $\boldsymbol{\xi}$ which consists of slack variables. The decision score of the classifier on a sample $\mathbf{x}$ is given by:

$$f(\mathbf{x}) = \sum_{m=1}^{s} \boldsymbol{\omega}_m^\top \phi_m(\mathbf{x}_i^{(m)}) + b. \tag{5.3}$$

where the weight for each kernel is implicitly included in $\boldsymbol{\omega}_m$.

In order to develop adaptive kernel weight, it will be more comprehensive if we could express the kernel weight explicitly. According to (Rakotomamonjy, Bach, Canu & Grandvalet 2008), following equality holds:

$$\frac{1}{2}\left(\sum_{m=1}^{s} \|\boldsymbol{\omega}_m\|_2\right)^2 = \left\{\min_{\boldsymbol{\eta}} \frac{1}{2}\sum_{m=1}^{s} \frac{\|\boldsymbol{\omega}_m\|_2^2}{\eta_m} : \; s.t. \sum_{m=1}^{s} \eta_m = 1, \eta_m \geq 0, \forall m\right\} \tag{5.4}$$

This helps to rewrite the MKL formulation into below optimisation problem.

$$\begin{aligned} \min_{\{\boldsymbol{\omega}_m\}_{m=1}^s, b, \boldsymbol{\xi}, \boldsymbol{\eta} \in \Delta} \; & \frac{1}{2} \sum_{m=1}^{s} \|\boldsymbol{\omega}_m\|_2^2 + C \sum_{i=1}^{n} \xi_i, \\ s.t. \quad & y_i \left(\sum_{m=1}^{s} \sqrt{\eta_m} \boldsymbol{\omega}_m^\top \phi_m(\mathbf{x}_i^{(m)}) + b \right) \geq 1 - \xi_i, \\ & \xi_i \geq 0, \quad i = 1, 2, ..., n, \end{aligned} \tag{5.5}$$

$\boldsymbol{\eta} = [\eta_1, ..., \eta_s]$ contains the weights for combination of base kernels. For L_1-norm constraint on kernel weights, $\Delta = \{\boldsymbol{\eta} \in \mathbb{R}_+^s : \sum_{m=1}^s \eta_m = 1, \eta_m \geq 0\}$. Corresponding decision score of the classifier on a sample $\mathbf{x}$ is given by:

$$f(\mathbf{x}) = \sum_{m=1}^{s} \sqrt{\eta_m} \boldsymbol{\omega}_m^\top \phi_m(\mathbf{x}^{(m)}) + b. \tag{5.6}$$

Above formulation helps to write the kernel weights explicitly, and this enables us to design a new algorithm featuring a more flexible kernel weight.

For the failure prediction task, we will need the model to be sample-specific. This means we need to treat different sample adaptively, considering their missingness-patterns and specific sets of equipment that generated the data. The model equation of traditional MKL does not allow such flexibility, which inspires us to design a new model.

5.3 Problem Description

In this section, we describe our data and related application. Figure 5.1 shows the workflow of the proposed method.

5.3.1 Data Description

The railway points' equipment details, maintenance logs, movement logs and failure history were collected from Sydney Trains database in a time range from 01/01/2014 to 30/06/2017. These data were collected from 350 sets of railway points spread in a large area. The weather data were crawled from

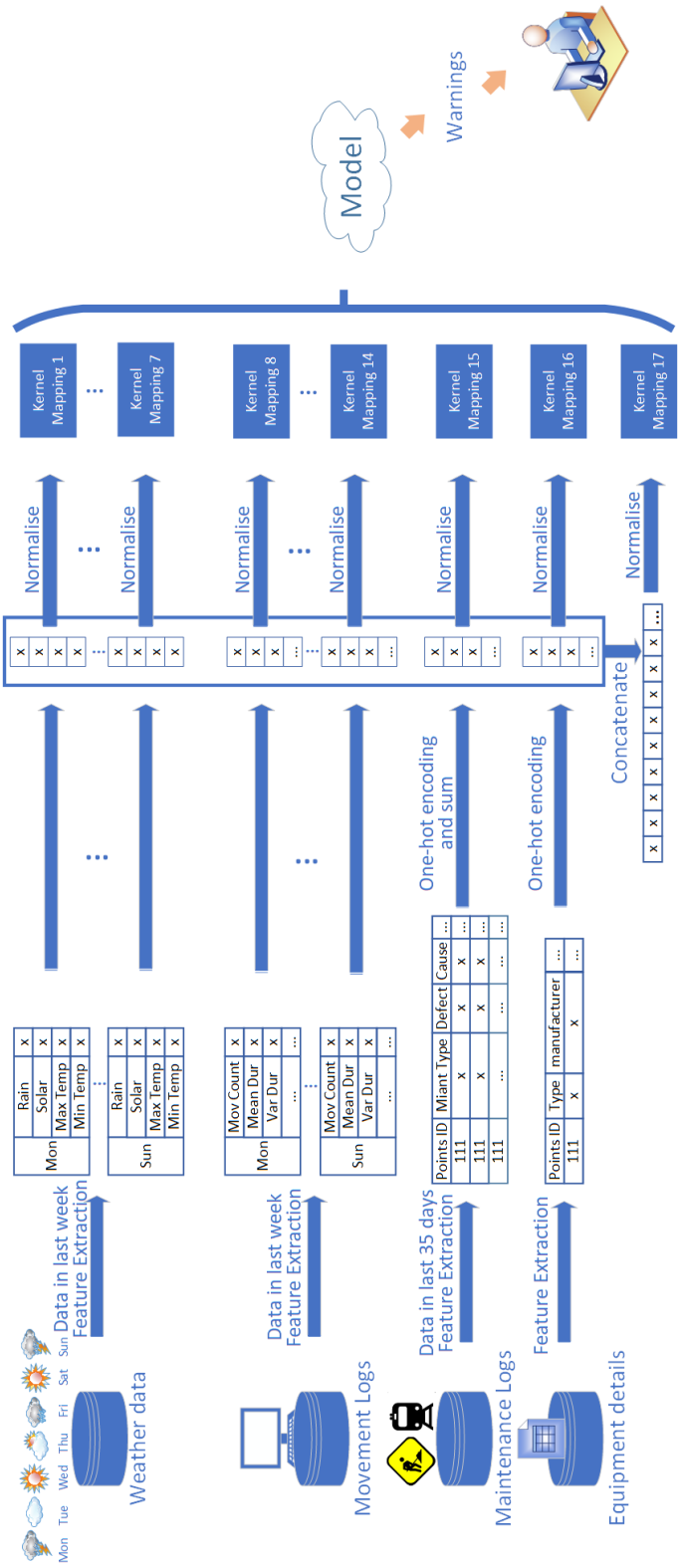


Figure 5.1: Workflow of our method.

Points ID	Occurred Date Time	Role	...
601	01/01/2020 11:00:00 AM	Primary Failed Equipment	...
602	01/01/2020 11:00:00 AM	Secondary Failed Equipment	...

Figure 5.2: A piece of IFMS data.

Australia Bureau of Meteorology[2] of the same time span. Below we are going to introduce their formats and features.

Infrastructure Failure Management System Database

Infrastructure Failure Management System (IFMS) Database stores history of failures of assets in Sydney Trains with timestamps. We extracted points' failures as part of our ground truth. Besides, some failures could be indirectly caused by railway points, and were recorded as "Secondary Failed Equipment". We filtered out these entries since these failures were more closely related to other equipment and require data related to them. A piece of IFMS data is presented in Figure 5.2.

Equipment Details

Equipment details data record the detailed parameters of every set of railway points, including Points ID, Manufacturer, Type and so on. A piece of data is presented in Figure 5.3. We use "-" to denote missing values. With the help of domain experts, we selected a subset of features from these columns, and they were all categorical variables. We transform these categorical data into numerical values through one-hot encoding.

[2] `www.bom.gov.au/climate/data/`

Points ID	Type	Manufacturer	Longitude	Latitude	...
601	SS	XX	151.1111	30.1111	...
602	KK	XX	-	-	...
603	KK	XX	150.1111	31.1111	...
...	...	...	...	...	...

Figure 5.3: A piece of equipment details.

Points ID	Finished Date	Maintenance Type	Defect	Cause	...
601	02/01/2017	RM	II	d	...
602	03/01/2017	RP	None	None	...
603	02/02/2017	RI	VI	e	...
...	...	...	...	...	...

Figure 5.4: A piece of maintenance log.

Maintenance Logs

Maintenance logs contain formatted historical maintenance logs of railway points. A subset of categorical features was extracted from them following advice by the domain experts. A piece of data is presented in Figure 5.4.

Movement Logs

Movement logs were automatically generated by Sydney Trains control system in a real-time manner. This system recorded states' changes of the railway points with timestamps in seconds. A piece of data is shown in Figure 5.5. We only list some of the event types here. Failures are reported in these logs as well. Some of the failures occurred in movement logs didn't appear in the IFMS database, for the reason that they recovered soon and didn't result in any significant incident. They were still real failures, and we included these failures in our ground truth. Sometimes workers were testing

Points ID	EventTime	EventText
601	02/01/2016 11:00:01 AM	STATE: GOING TO NORMAL
601	02/01/2016 11:00:06 AM	STATE: NORMAL
601	02/01/2016 11:30:00 AM	STATE: GOING TO REVERSE
...	...	...

Figure 5.5: A piece of movement log.

Date	Rain (mm)	Solar (MJ/m^{-2})	Max Temperature (°C)	Min Temperature (°C)
02/01/2016	0	29.1	30.9	21.3
03/01/2016	-	-	-	-
04/01/2016	12	25.4	39	19.4
...	...	...	...	...

Figure 5.6: A piece of weather data.

the points for preventative maintenance and this also generated failure logs. In this case, we ignore these failures to keep the ground truth clean.

Weather

Weather data were retrieved from the Australia Bureau of Meteorology. Our data were gathered from railway points spread in a large area, so weather conditions for them may vary. Our strategy was to download data from the nearest weather station according to the longitudes and latitudes provided by equipment details. Sometimes weather station would be closed for a while, and we were not able to find another station to substitute them in some situations. Some points lack geo-coordinates in Sydney Trains system. These cause the absence of weather data. Figure 5.6 shows a piece of weather data.

5.3.2 Problem Formulation

With data mentioned above in hand, we are going to make use of them to fulfil the prediction task. Essentially, this is a classification task. Since our data were generated from multiple sources, they came with different formats and sample frequencies. The two most important things are how we should aggregate our data from multiple sources and label them according to failure records.

Grouping and labelling data in a daily manner is an intuitive way. However, our data are highly imbalanced in label distribution. The number of days that failures occurred is about 4200, while our data include 454237 days summing over all railway points. This would produce a dataset contains only 0.9% positive samples if we give a label "1" to failures. Such imbalanced dataset would deteriorate the performance of the classifier.

Sydney Trains' train timetable shows cyclic patterns following calendar weeks (Gong, Li, Zhang, Liu, Zheng & Kirsch 2018), which will pose a periodic effect on the data as well. Therefore, we grouped our data according to calendar weeks. We gave label "1" to a week if any failure was recorded in IFMS or movement log of this week. As a result, our task is to predict whether there will be failures occur in any time of next week, depending on weather conditions, movement logs in this week and maintenance logs in a period of 35 days before next week. For maintenance logs, we extend the time range to 35 days since maintenances were often performed based on a monthly interval. We would also incorporate equipment details, and in general, they are independent of time. Figure 5.7 illustrates our data aggregation and labelling strategy. After some data cleaning, we finally generated 58833 samples, including 3900 positive samples.

Notice that in some cases we would lose the movement logs, for example, the influence of maintenance work. In these cases, we would only refer to logs in the IFMS database as failure indicators upon agreement with the domain experts.

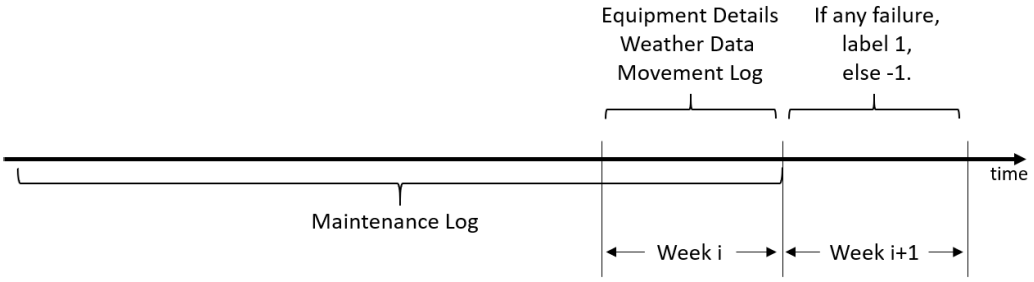


Figure 5.7: To forecast failures in week i+1, we use data from week i and maintenance logs in a 35-day interval before week i+1.

5.4 Methodology

5.4.1 Feature Extraction and Partition

Although we have grouped our data according to the above-mentioned criterion, we need to transform them further to form feature vectors.

For equipment details and maintenance logs data, we selected some columns following the advice of domain experts. Then we transform categorical variables into numerical values through one-hot encoding. For numerical variables in maintenance logs like cost, we firstly transform them into categorical variables "zero" or "nonzero", them transform them into numerical values through one-hot encoding as well. There could be several maintenances in the timespan we considered. If so, we summed up the transformed features.

For movement logs data, we extracted some statistical features of the movement for each set of railway points in every day, like mean of movements, variance of movements, count of movements and so on. Because there are 7 days per week, we would have 7 subsets of features for movement logs. Similarly, for weather data, we have 7 subsets for one week. This strategy is illustrated in Figure 5.1. Such partition allows us easily handle the missingness patterns in a daily format as we will introduce in detail in the next section. Table 5.1 summarises missing percentages of our data after such feature partition. The missingness of weather data is mainly due to

missing geo-coordinates of railway points or close of weather stations.

There are 16 feature subsets in total. By applying different kernel functions to different subsets, we can formulate our task as a multiple kernel learning problem for binary classification. In order to model the interaction among feature subsets, we also concatenated all feature subsets to form a long feature vector and applied a kernel function on it. Finally, we would get 17 kernels as our inputs. We term these feature subsets **channels**.

The missing probability for each data channel is not very high, but another fact is that 44% of samples are either missing one channel or more. Therefore, it is imperative for us to build a model that is suitable for prediction with incomplete samples.

5.4.2 Selecting Kernel Functions

After applying one-hot encoding, features generated from equipment details and maintenance logs data were often very sparse. We thus directly used linear kernel for these two data channels as recommended in literature (Li, Wang, Gu & Ling 2015, Fan, Chang, Hsieh, Wang & Lin 2008). For the remaining data channels consist of weather and movement logs of 7 days, we applied the commonly used radial basis function (RBF) kernels and set the bandwidth parameters for RBF based on a validation set to mitigate the risk of over-fitting. In the rare case, some channels of a sample were only partially missing. If so, we filled the missing part with means.

5.4.3 Missingness-pattern-adaptive Multiple-kernel Learning

To work with missing channels, a straightforward way is to learn separate kernel weights for each missingness pattern. However, there can be $\sum_{m=1}^{s} C_s^m$ missingness patterns if we have s channels, so it is possible that the data cannot cover every pattern. Besides, the data for one pattern can be less and contain only one type of label. Such a strategy also ignores the relationship

Table 5.1: Missing rates and dimensions of our data channels. 44% of samples are missing at least one channel.

Data		Missing Rate	Feature Dimension
Equipment Details		0%	450
Maintenance Logs		13%	365
Movement Logs	Monday	5%	30
	Tuesday	6%	30
	Wednesday	5%	30
	Thursday	5%	30
	Friday	7%	30
	Saturday	8%	30
	Sunday	10%	30
Weather	Monday	26%	4
	Tuesday	26%	4
	Wednesday	26%	4
	Thursday	25%	4
	Friday	25%	4
	Saturday	25%	4
	Sunday	25%	4

between missingness patterns. A likely choice would be to adjust the kernel weights according to missingness patterns.

In order to allow adaptive kernel combination, we firstly modify the decision function in Eq. (5.6) for a sample $\mathbf{x}$ with s channels into following form:

$$f(\mathbf{x}) = \sum_{m=1}^{s} \eta_m(\mathbf{x}) \left\langle \boldsymbol{\omega}_m, \phi_m(\mathbf{x}^{(m)}) \right\rangle + b, \tag{5.7}$$

with $\langle \cdot, \cdot \rangle$ denotes the inner product of vectors and

$$\eta_m(\mathbf{x}) = p_m \mathbf{v}_m^\top \sum_{j=1}^{2s} p_j \mathbf{v}_j, \tag{5.8}$$

where $\mathbf{p} = [p_1, p_2, \cdots, p_{2s}]^\top$ is a binary vector generated by one-hot encoding

on the missingness pattern for sample $\mathbf{x}$ (similar to $\bar{\mathbf{m}}$ defined in Chapter 4). We introduce $V = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_{2s}] \in \mathbb{R}^{k \times 2s}$ with latent dimension k to represent embedding matrix for missingness patterns. By Eq. (5.8), we express the kernel weights as a second order polynomial mapping from missingness patterns $\mathbf{p}$ with the coefficients given by related inner product of vectors in V. We give a simple example here to explain how we generate $\mathbf{p}$. Assume we have three data channels but for a sample the second one is missing, then:

$$\mathbf{p} = [1, 0, 1, 0, 1, 0]^\top. \tag{5.9}$$

The first and third "1" mean we have first and third feature subsets for this sample. The fifth "1" serves as a complementary feature for missing channel 2. By doing so, the absence of a channel would make its kernel weight zero and influence the kernel weights of other presented channels. Notice that the matrix V can also be interpreted similarly as a dictionary matrix for generating the kernel weights. This explains why the vector $\mathbf{p}$ is of $2s$ dimensionality. This is to guarantee that equal number of elements are selected from V for all missingness patterns.

The motivation behind this is that we want to collect information from the missingness pattern of each sample. Eq. (5.8) also indicates that the kernel weight for a channel is decided by "seeing" the existence of other channels' data.

With similar notation to Eq. (5.5), the optimisation problem after introducing adaptive kernel weight can be expressed as:

$$\begin{aligned} \min_{\{\boldsymbol{\omega}_m\}_{m=1}^s, b, \boldsymbol{\xi}, V} \quad & \frac{1}{2} \sum_{m=1}^{s} \|\boldsymbol{\omega}_m\|_2^2 + C_1 \sum_{i=1}^{n} \xi_i + C_2 \|V\|_F^2 \\ s.t. \quad & y_i \left(\sum_{m=1}^{s} \eta_m(\mathbf{x}_i) \boldsymbol{\omega}_m^\top \phi_m(\mathbf{x}_i^{(m)}) + b \right) \geq 1 - \xi_i, \\ & \xi_i \geq 0, \quad i = 1, 2, ..., n, \end{aligned} \tag{5.10}$$

where C_1 and C_2 are two regularisation parameters. $\|\cdot\|_F^2$ denotes the Frobenius norm. We add a regularisation term on V to prevent it from being

arbitrary scaled up due to the norm constraint on $\boldsymbol{\omega}_m$. Note that we no longer require the vectors for kernel weights lies in a simplex as in the original MKL algorithm.

Theorem 5.1 *Adopting an adaptive kernel weight in Eq.(5.8) would lead to a positive semi-definite kernel for MKL, with the kernel matrix K_η given by:*

$$K_\eta = \sum_{m=1}^{s} \Big(\big((V^\top V P) \odot P \big)^\top \mathbb{I}_m \mathbb{I}_m^\top \big((V^\top V P) \odot P \big) \Big) \odot K_m, \tag{5.11}$$

$\odot$ stands for the Hadamard product. $P = [\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_n]$ with each column vector $\mathbf{p}_i \in \{0,1\}^{2s}$ denotes the missingness pattern for sample i. $\mathbb{I}_m$ is a length-$2s$ indicator vector with only m-th element being 1. $\{K_m\}_{m=1}^s$ is the kernel matrix related to mapping $\{\phi_m(\cdot)\}_{m=1}^s$, and if m-th data channel is missing for a sample then we set $\phi_m(\cdot)$ to be a zero map for this sample.

Proof 5.1 *According to the following decision function,*

$$\begin{aligned} f(\mathbf{x}) &= \sum_{m=1}^{s} \eta_m(\mathbf{x}) \left\langle \boldsymbol{\omega}_m, \phi_m(\mathbf{x}^{(m)}) \right\rangle + b, \\ \eta_m(\mathbf{x}) &= p_m \mathbf{v}_m^\top \sum_{j=1}^{2s} p_j \mathbf{v}_j, \end{aligned} \tag{5.12}$$

we can rewrite it into an equivalent from by defining a new kernel function, with the feature map given by:

$$\hat{\phi}(\mathbf{x}) = [\eta_1(\mathbf{x})\phi_1(\mathbf{x}^{(1)}), ..., \eta_s(\mathbf{x})\phi_s(\mathbf{x}^{(s)})], \tag{5.13}$$

so that

$$f(\mathbf{x}) = \langle \hat{\boldsymbol{\omega}}, \hat{\phi}(\mathbf{x}) \rangle + b, \hat{\boldsymbol{\omega}} = [\boldsymbol{\omega}_1, ...\boldsymbol{\omega}_s], \tag{5.14}$$

For two samples $\mathbf{x}_1$ and $\mathbf{x}_2$, we have

$$\langle \hat{\phi}(\mathbf{x}_1), \hat{\phi}(\mathbf{x}_2) \rangle = \sum_{m=1}^{s} \eta_m(\mathbf{x}_1)\eta_m(\mathbf{x}_2) \cdot \langle \phi_m(\mathbf{x}_1^{(m)}), \phi_m(\mathbf{x}_2^{(m)}) \rangle \tag{5.15}$$

and notice that $\langle \phi_m(\mathbf{x}_1^{(m)}), \phi_m(\mathbf{x}_2^{(m)}) \rangle$ *is the* (i,j)*-th element of* K_m *and similarly 5.15 generates the* (i,j)*-th element of* K_η*. Rewrite the inner product in 5.15 for all samples collectively in a matrix form we can get*

$$K_\eta = \sum_{m=1}^{s} \Big(\big((V^\top V P) \odot P \big)^\top \mathbb{I}_m \mathbb{I}_m^\top \big((V^\top V P) \odot P \big) \Big) \odot K_m. \tag{5.16}$$

Following Schur product theorem (Zhang 2006), Hadamard product of two positive semi-definite matrices is positive semi-definite, and sum of positive semi-definite matrices is positive semi-definite.

Clearly, $\big((V^\top V P) \odot P \big)^\top \mathbb{I}_m \mathbb{I}_m^\top \big((V^\top V P) \odot P \big)$ *is the product of a matrix with its transpose so that it is positive semi-definite. Thus, in order to show 5.16 is positive semi-definite we only need to show* K_m *is positive semi-definite.*

When there is no missing data, K_m *is calculated from a predefined kernel function, and we can choose many available kernels like Gaussian kernel to satisfy that* K_m *being positive semi-definite.*

In our case, if m*-th data channel is missing in a sample, then we set* $\phi_m(\cdot)$ *to be a zero map for this sample. This will zero out corresponding diagonal elements of the kernel matrix, as well as corresponding rows and columns of the kernel matrix. For example, when* m*-th data channel is missing in* i*-th sample, the* i*-th column and row of* K_m *will be all zeros. To show* K_m *is still positive semi-definite, we can re-organise the row and columns to put* i*-th column and row to last row and column. Then according to Schur complement condition for positive semi-definiteness,* K_m *is a positive semi-definite matrix. Thus* K_η *is positive semi-definite, and we complete the proof.*

Theorem 5.1 shows that our adaptive kernel weight in theory will lead to a positive semi-definite kernel. It also shows that this problem is hard to solve in dual form because of the complicated form of K_η in Eq. (5.11).

5.4.4 Sample-adaptive Multiple-kernel Learning

If we train a unified model for all sets of railway points, we will possibly ignore some peculiarities of them even though we have included equipment details as features. Training separate models for each set of railway points performed even worse as we observed in initial experiments, because the data for each set of railway points are often insufficient for learning a robust model. These motivated us to modify our method so that it can be adjusted to fit each set of railway points, while still can use all the data together for training the model. We revised the kernel weight in Eq.(5.8) into the following format for a sample $\mathbf{x}$:

$$\eta_m(\mathbf{x}) = p_m \mathbf{v}_m^\top \sum_{j=1}^{2s} p_j \mathbf{v}_j a_j, \tag{5.17}$$

where we add a new vector $\mathbf{a} = [a_1, a_2, \cdots, a_{2s}]^\top$ to represent a unique embedding for the set of railway points that generated sample $\mathbf{x}$.

Substitute Eq. (5.17) into Eq. (5.7), we observe that the term p_m could be omitted from Eq. (5.17) if we set $\phi_m(\cdot)$ to be a zero map for data with absent m-th data channel, so we omit p_m for simplicity of notation. If we have T sets of railway points, then we will introduce the matrix $A = [\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_T] \in \mathbb{R}^{2s \times T}$ with T the total number of sets of railway points. Each column vector in A stands for a embedding of a set of railway points. Let $q(\mathbf{x}_i)$ be the mapping which maps $\mathbf{x}_i$ to index of railway points that generated the data $\mathbf{x}_i$. Eq. (5.17) can be rewritten into matrix form for sample $\mathbf{x}_i$:

$$\eta_m(\mathbf{x}_i) = \mathbb{I}_m^\top V^\top V \left(\mathbf{p}_i \circ \mathbf{a}_{q(\mathbf{x}_i)} \right), \tag{5.18}$$

where $\circ$ denotes the element-wise product of vectors.

With $\eta_m(\mathbf{x}_i)$ given in Eq. (5.18), corresponding optimisation problem

becomes:

$$\begin{aligned}\min_{\{\boldsymbol{\omega}_m\}_{m=1}^s, b, \boldsymbol{\xi}, V, A} \; & \frac{1}{2}\sum_{m=1}^{s}\|\boldsymbol{\omega}_m\|_2^2 + C_1\sum_{i=1}^{n}\xi_i + C_2\|V\|_F^2 + C_3\|A - \mathbf{1}_{2s\times T}\|_F^2, \\ s.t. \quad & y_i\left(\sum_{m=1}^{s}\eta_m(\mathbf{x}_i)\boldsymbol{\omega}_m^\top\phi_m(\mathbf{x}_i^{(m)}) + b\right) \geq 1 - \xi_i, \\ & \xi_i \geq 0, \quad i = 1, 2, ..., n,\end{aligned} \tag{5.19}$$

where C_3 is a regularisation parameter and $\mathbf{1}_{2s\times T}$ is a matrix of shape $2s \times T$ containing all ones. Notice that when A is a matrix of all ones, Eq. (5.17) reduce to Eq.(5.8). In other words, when C_3 is large enough, the two models would be equivalent. This regularisation term ensures an appropriate variance of models among different sets of railway points. One can also prove that such adaptive weights also retain a positive semi-definite kernel, following the similar steps in prove of theorem 5.1.

5.4.5 Optimisation

In this section, we introduce the optimisation algorithm we used to train our models. We will focus on the problem defined by Eq.(5.19), as Eq.(5.10) could be treated as a simplified version of Eq.(5.19), where the matrix A consists of all ones.

As mentioned before, Eq.(5.10) and Eq.(5.19) are hard to optimise in dual form. What's more, we cannot fit such large data into memory if we pre-compute those 17 kernel matrices. Thanks to the random feature (RF) approximation (Rahimi & Recht 2008), we can take an explicit form of mapped features hence avoiding calculation of the kernel matrices. This also facilitates the optimisation in the primal, which is much simpler. Given $\mathbf{x} \in \mathbb{R}^d$ and a predefined parameter D, the mapped features associated with a RBF kernel could be approximated by:

$$\phi(\mathbf{x}) = \sqrt{\frac{1}{D}}\left[\sin\left(\mathbf{g}_1^\top\mathbf{x}\right), \cos\left(\mathbf{g}_1^\top\mathbf{x}\right), \cdots, \sin\left(\mathbf{g}_D^\top\mathbf{x}\right), \cos\left(\mathbf{g}_D^\top\mathbf{x}\right)\right]^\top, \tag{5.20}$$

Algorithm 5.1 Training Procedure with Mini-batch Subgradient method

Input: Dataset $\mathcal{X}$ collected from T sets of railway points. Latent dimension k for V. Number of random features $\{d_m\}_{m=1}^s$ for each kernel. Hyper-parameters C_1, C_2, C_3. Learning rate β. Batch size h. The number of batches $H = \lfloor \frac{n}{h} \rfloor$.

Initialise: $\{\boldsymbol{\omega}_m\}_{m=1}^s = \mathbf{0}$. $b = 0$. $A = \mathbf{1}_{2s \times T}$. V with values sampled from a uniform distribution $\mathcal{U}(0, 1)$.

for *Epoch* $= 0$ to M **do**
  Shuffle the samples in $\mathcal{X}$ randomly.
  Split $\mathcal{X}$ into batches $X_1, X_2, \cdots, X_H$.
  for $i = 1, 2, \cdots, H$ **do**
    Get the index set $\mathcal{I}$ for support vectors in X_i
    Update V with step-size β and sub-gradient in Eq. (5.23)
    Update A with step-size β and sub-gradient in Eq. (5.24)
    Update b with step-size β and sub-gradient in Eq. (5.25)
    Update $\{\boldsymbol{\omega}_m\}_{m=1}^s$ with step-size β and sub-gradient in Eq. (5.22).
  end for
end for

where the entries of $G = [\mathbf{g}_1, \cdots, \mathbf{g}_D] \in \mathbb{R}^{d \times D}$ are drown i.i.d. from a Gaussian distribution $\mathcal{N}(0, \sigma^{-2})$ with σ bandwidth of the RBF kernel. Many variants of RF approximation have been proposed in the literature. Here we implement the Fastfood (Le, Sarlós & Smola 2013) for its simplicity and efficiency in memory usage.

Our optimisation problem can be rewritten into following form with hinge

loss $L(x, y) = \max(0, 1 - xy)$:

$$\begin{aligned}\min \mathcal{L} =& \frac{1}{2}\sum_{m=1}^{s} \|\boldsymbol{\omega}_m\|_2^2 \\ &+ C_1 \sum_{i=1}^{n} L\left(y_i, \sum_{m=1}^{s} \eta_m\left(\mathbf{x}_i\right)\left\langle \boldsymbol{\omega}_m, \phi_m(\mathbf{x}_i^{(m)})\right\rangle + b\right) \\ &+ C_2 \left\|V\right\|_F^2 + C_3 \left\|A - \mathbf{1}_{2s\times T}\right\|_F^2, \\ \text{w.r.t.} \quad & \{\boldsymbol{\omega}_m\}_{m=1}^{s}, b, V, A,\end{aligned} \tag{5.21}$$

with $\eta_m(\mathbf{x}_i)$ defined in Eq.(5.18), we can calculate the sub-gradients regarding these variables and get:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\omega}_m} = \boldsymbol{\omega}_m - C_1 \sum_{i\in\mathcal{I}} y_i \mathbb{I}_m^\top V^\top V \left(\mathbf{p}_i \circ \mathbf{a}_{q(\mathbf{x}_i)}\right) \phi_m(\mathbf{x}_i^{(m)}), \tag{5.22}$$

$$\frac{\partial \mathcal{L}}{\partial V} = -C_1 V \sum_{i\in\mathcal{I}}\sum_{m=1}^{s} y_i \boldsymbol{\omega}_m^\top \phi_m(\mathbf{x}_i^{(m)}) \Big(\mathbb{I}_m \big(\mathbf{p}_i \circ \mathbf{a}_{q(\mathbf{x}_i)}\big)^\top + \big(\mathbf{p}_i \circ \mathbf{a}_{q(\mathbf{x}_i)}\big)\mathbb{I}_m^\top\Big) + 2C_2 V, \tag{5.23}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{a}_t} = -C_1 \sum_{i\in\mathcal{I}\cap\mathcal{T}_t}\sum_{m=1}^{s} \Big(y_i \boldsymbol{\omega}_m^\top \phi_m(\mathbf{x}_i^{(m)}) V^\top V \mathbb{I}_m\Big) \circ \mathbf{p}_i + 2C_3(\mathbf{a}_t - \mathbf{1}_{2s}), \tag{5.24}$$

$$\frac{\partial \mathcal{L}}{\partial b} = -C_1 \sum_{i\in\mathcal{I}} y_i, \tag{5.25}$$

where $\mathcal{I} = \{i | 1 - y_i f(\mathbf{x}_i) > 0\}$ is the index set for support vectors. $\mathcal{T}_t = \{i | q(\mathbf{x}_i) = t\}$ is the index set of samples generated by railway points t.

With gradients calculated as Eq. (5.22) - Eq. (5.25), we adopted Mini-batch gradient descent in optimisation. We trained the models for 50 epochs with a constant learning rate $\beta = 0.0001$ and batch-size 256. Using d_m to denote the dimension of random features for m-th kernel mapping, the computational complexity for calculating the gradients is $O(\sum_{m=1}^{s} d_m h + s^2 k)$, which depends linearly on batch-size h and can be computed efficiently. We summarise the training process in Algorithm 5.1.

5.5 Experiments

Our data were collected from 350 sets of railway points from 01/01/2014 to 30/06/2017, together with corresponding weather data downloaded from the Australia Bureau of Meteorology. There are 58833 samples, including 3900 failures. We named this dataset **Points-All**.

We also built a subset consists of data from 5 most "vulnerable" sets of railway points, i.e. those with most failure samples, and named it **Points-Subset**.

These datasets are imbalanced in label distribution. We have tried to re-weight the samples in training by the label frequency but saw no performance gains, so we did not adopt such a strategy. Table 5.2 summarises the statistics of our datasets.

5.5.1 Baselines, Evaluation Metrics and Parameter Setting

Baselines

To show the effectiveness of our approach, experiments were conducted with the following methods.

- MKL-ZF is the l_p-norm MKL method solved by the algorithm in (Kloft et al. 2011) with absent channels filled by zeros. We conducted experiments for p ranges in $[10^0, 10^1, 10^2, 10^3, 10^4]$.
- MKL-MF is similar to MKL-ZF but with absent channels filled by the averages.

Table 5.2: Dataset summary.

Dataset	#instances	#failures	#railway points	#incomplete instances
Points_All	58833	3900	350	25942
Points_Subset	905	183	5	98

- MVL-MKL firstly imputes the missing values by the method in (Xu et al. 2015*b*), and then applied l_p-norm MKL with the imputed data. (Xu et al. 2015*b*) is a competitive method for filling incomplete data similar to our case, so we included it in our baselines.

- Absent Multiple Kernel Learning (AMKL) (Liu, Wang, Yin, Dou & Zhang 2015) is a state-of-the-art method for MKL with missing kernels. We only compared with AMKL on Points-Subset because it cannot be scaled up to fit our Points-All dataset.

- Single Source Classifiers (SSC) are the classifiers applied to single-source data. For weather and movement logs data, there are still 7 data channels for each source. We use our method MAMKL as the classifier. For maintenance logs, equipment details and the data channel formed by concatenating all features, we filled the missing channels with means and then used kernel SVM (Chang & Lin 2011) for classification because these data sources only consist of one channel.

- Missingness-pattern-adaptive MKL (MAMKL) is the method proposed in this work with kernel weights given by Eq. (5.8).

- Sample Adaptive MKL (SAMKL) is the method proposed in this work with kernel weights determined by Eq. (5.17).

For fair of comparison, for all methods, we used RF approximation for RBF kernels, and we fixed the random seed to make them determined. As such, l_p-norm MKL could also be applied to our Points-All dataset without precomputed kernels.

Evaluation Metrics

We used the Area Under Receiver Operating Characteristic Curve (AUROC) and Area Under Precision-Recall Curve (AUPRC) as our performance metrics for all the methods. The AUPRC can be a better performance metric for imbalance data (Saito & Rehmsmeier 2015).

Parameter Setting

For all non-convex methods, we repeated them 10 times to report the results with means and standard deviations. For the Points-All dataset, we split it into 60% training data, 20% validation data and 20% test data. The linear kernel was used for the data channels from equipment details and maintenance logs. We set same bandwidth for RBF kernels on 7 data channels from weather data. The bandwidth is chosen from $[\sigma^{-2}, \sigma^{-1}, \sigma^{0}, \sigma^{1}, \sigma^{2}]$ according to the AUROC on validation data using SVM with sum of these 7 kernels as input. σ is the standard deviation of weather data. The same criterion was adopted to select the parameter of RBF kernels for 7 data channels from movement logs and 1 data channel from concatenated features. The dimensions of RFs for approximating RBF kernels were set to 1024, 2048 and 2048 for movement logs, weather and concatenated features respectively. All other parameters were chosen from some appropriately large ranges based on the AUROC of related methods on validation data. For Points-Subset, we randomly selected 80% data as training set and the remaining 20% as the test set. Parameters for them were decided by 5-fold cross-validation on the training set.

5.5.2 Results on Points-Subset Dataset

Table 5.3 shows the experiment results on Points-Subset dataset. l_p-norm MKL got inferior results when $p = 1$, for the reason that they tended to find a sparse combination of kernels. This would encourage the model to ignore the data channels as many as possible so that only partly information is utilised. It can be useful when some of the data channels are irrelevant to the prediction task. However, in our case, it does not perform well. This means our data channels carry complementary information, so only use some of them could not produce a promising result. Experiment results on SSC also verify our argument that only use data from one source is not enough.

The pre-filling method did not perform well, because filling the missing

Table 5.3: Experiment results on Points-Subset dataset. Best results are bold and the second best are underlined. The results are reported with means and standard deviations (mean±std) for non-convex methods.

Methods		AUROC	AUPRC
MKL-ZF	$p = 10^0$	0.737	0.436
	$p = 10^1$	0.921	0.791
	$p = 10^2$	0.902	0.784
	$p = 10^3$	0.920	0.789
	$p = 10^4$	0.921	0.790
MKL-MF	$p = 10^0$	0.646	0.289
	$p = 10^1$	0.923	0.800
	$p = 10^2$	0.887	0.770
	$p = 10^3$	0.887	0.767
	$p = 10^4$	0.906	0.780
MVL-MKL	$p = 10^0$	0.655±0.002	0.292±0.002
	$p = 10^1$	0.852±0.008	0.783±0.005
	$p = 10^2$	0.898±0.010	0.788±0.015
	$p = 10^3$	0.873±0.006	0.788±0.005
	$p = 10^4$	0.873±0.006	0.788±0.004
SSC	Movement Logs	0.663±0.001	0.380±0.001
	Weather	0.864±0.035	0.781±0.036
	Maintenance Logs	0.667	0.301
	Equipment Details	0.516	0.217
	All Concatenated	0.669	0.376
AMKL		0.736	0.463
MAMKL		<u>0.942±0.005</u>	<u>0.831±0.016</u>
SAMKL		**0.947±0.007**	**0.840±0.011**

data in advance and used them in training will possibly introduce another source of error and propagate to following MKL task. The better performance of AMKL over imputation-based baselines proves that filling the absent data channel in advance is not an optimal solution. Although AMKL does not involve the imputation process and appropriately takes into account the missingness patterns in training, it does not distinguish data with different missingness patterns in testing. A fixed kernel weight for all samples is adopted, and it is based on l_1-norm MKL. These factors limited its performance as it tended to ignore some of the data channels similar to l_1-norm MKL.

MAMKL outperformed AMKL, and this verifies the importance and effectiveness of missingness-pattern-adaptive kernel weight. SAMKL further promotes the results of MAMKL based on sample-adaptive kernel weights. It is clear that the proposed method outperforms other baselines in terms of both AUROC and AUPRC. We attribute the improvement to the combination of multi-source data and the sample adaptive kernel weights. By appropriately setting the threshold, the SAMKL can achieve 87.5% precision and 71.7% recall. For comparison, the FrFM-EUC model proposed in Chapter 3 can obtain 73.1% precision, 76.9% recall, 0.898 AUROC and 0.801 AUPRC on the Points-Subset dataset. This further verified the effectiveness of the FrFM-EUC model and the combination of multi-source data.

5.5.3 Results on Points-All Dataset

Table 5.4 shows the experiment results on Points-All dataset. By training on all data, we also included some sets of railway points with only a few failure records. The proportion of incomplete samples is also higher than that in Points-Subset dataset. These added up to our difficulties in predicting the failures. As shown in Table 5.4, in contrast to their results on Points-Subset dataset, results of l_p-norm MKL with $p = 1$ is often better. This means traditional MKL cannot fully exploit the merits of multiple kernels, because when $p = 1$ most of the kernel weights will be zeroed out. Our method

Table 5.4: Experiment results on Points-All dataset. Best results are bold and the second best are underlined. The results are reported with means and standard deviations (mean±std) for non-convex methods.

Methods		AUROC	AUPRC
MKL-ZF	$p = 10^0$	0.699	0.218
	$p = 10^1$	0.691	0.199
	$p = 10^2$	0.696	0.205
	$p = 10^3$	0.690	0.196
	$p = 10^4$	0.692	0.197
MKL-MF	$p = 10^0$	0.698	0.223
	$p = 10^1$	0.684	0.204
	$p = 10^2$	0.687	0.204
	$p = 10^3$	0.682	0.198
	$p = 10^4$	0.668	0.176
MVL-MKL	$p = 10^0$	0.678±0.001	0.168±0.002
	$p = 10^1$	0.671±0.001	0.159±0.001
	$p = 10^2$	0.670±0.001	0.159±0.001
	$p = 10^3$	0.672±0.002	0.158±0.001
	$p = 10^4$	0.674±0.002	0.159±0.003
SSC	Movement Logs	0.546±0.010	0.093±0.001
	Weather	0.677±0.003	0.197±0.008
	Maintenance Logs	0.567	0.098
	Equipment Details	0.517	0.085
	All Concatenated	0.622	0.133
MAMKL		<u>0.721±0.002</u>	<u>0.261±0.009</u>
SAMKL		**0.734±0.002**	**0.270±0.002**

consistently outperforms other baselines on both AUROC and AUPRC, and see the improvement compared to SSC. Notice that SAMKL is much better than MAMKL in this dataset, which again verifies the effectiveness of sample adaptive kernel weight. This could guarantee a reliable warning regarding failures predicted by our model. By appropriately setting the threshold, the SAMKL can achieve 50.8% precision and 15.0% recall. The FrFM-EUC model proposed in Chapter 3 can similarly achieve 55.3% precision, 11.3% recall, 0.726 AUROC and 0.271 AUPRC on the Points-All dataset.

For each set of railway points, the number of samples is usually less than 180. Only several failures are observed for some points. We have also tried to trained many classifiers each for one set of railway points, but the results were unsatisfactory, so we did not list them here.

5.6 Conclusion

This chapter designed a novel approach for combining incomplete multi-source data to predict the failure of railway points. It was developed based on the multiple kernel learning framework but went a step further by exploiting the missingness patterns and sample-specific features. With the involvement of domain experts, we grouped our data weekly and split each week into a daily format to form 17 data channels and built 17 kernels. In this format, we can express the missingness patterns of samples clearly. After that, a missingness-pattern-adaptive MKL was put forward to leverage the information carried by missingness patterns. Through taking the distinct properties of each set of railway points into account, the prediction results were further improved by SAMKL algorithm. Experiments show that the proposed model can output reliable warnings for railway points, and can predict the failures precisely for those frequently-failed railway points.

There are still some practical problems remain to be solved: 1) how to efficiently and effectively clean the noisy data, which includes wisely relabelling a part of the data with the help of domain experts; 2) how to incorporate

time-series processing models for better failure prediction with multi-source information; 3) how to leverage domain knowledge in the optimisation of models, e.g. include domain knowledge as constraints in model optimisation. These problems give several interesting future research directions.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis presented several general methods for data analytics for CBM. We firstly abstracted three research questions on failure prediction, regarding sparse high-dimensional data, incomplete data and multi-source data, respectively. Focused on each research question, this thesis proposed corresponding methods and discussed them in each chapter.

Chapter 3 proposed FrFMs for mining the maintenance logs of equipment. The maintenance logs are usually of high-dimensionality and sparse, which causes difficulties in hand-crafting features for effective failure prediction. The FrFM model helped to automatically discover the important feature-interactions for failure prediction, and it was further enhanced with field information. The field information was used to constrain the embedding vectors in each field. Two variants of the FrFMs were proposed - based on Euclidean distance and Cosine distance respectively. The proposed method was tested with railway points' maintenance logs and saw improvements over competitive baselines.

As missing data is a pervasive problem in real-world CBM data, Chapter 4 proposed a general method for learning with incomplete data. When there are multiple missingness patterns in the dataset, a single model would likely

be a compromise between optimal models regarding each missingness pattern. Therefore, Chapter 4 focused on how to adaptively adjust the model according to the missingness patterns. By assuming all models associated with corresponding missingness patterns were generated from a dictionary and their missing pattern vectors, we could effectively learn an adaptive model considering the incomplete data. A non-linear model was also similarly proposed with neural networks serving as the backbone. Experiments on several public datasets demonstrated the effectiveness of the proposed method.

Chapter 5 proposed a sample-adaptive multiple-kernel learning method for multi-source data. The proposed algorithms were able to deal with incomplete and heterogeneous data. We further incorporated an embedding matrix for the embedding of each set of railway points so that we could treat them differently while using the data altogether in training. Experiments on real data proved the effectiveness of this method on both vulnerable sets and all sets of railway points.

6.2 Future Directions of Data Analytics for CBM

As a rather big topic, data analytics for CBM also involves dealing with time series. Although with a lot of related work on time-series (Brockwell, Davis & Calder 2002), the time series in CBM could be much more complicated. For example, how to combine multi-source time series data, and simultaneously deal with unevenly sampled data is a challenging issue.

Another important research direction will be how to learn an accurate model under imperfect supervision. In many cases, CBM data are not perfectly labelled, or only partly labelled, including the cases of mistakenly-labelled data and missing-label data, and this brings difficulties for leaning a good model. How to design an effective learning scheme under such insufficient supervision and with the special structure of CBM data is another

challenging task. Some work related to this topic includes active learning (Cohn, Ghahramani & Jordan 1996, Tong & Koller 2001). Active learning can help select the most important data to be labelled for learning an accurate predictive model. Thus, we can use related algorithms to select a small but important portion of data and ask for domain experts' help on labelling. Most of real-world CBM data are not generated for training a machine learning model. Even though they come with labels, the labels can be very noisy. Some recent work introduced the idea of relabelling (Zhao, Sukthankar & Sukthankar 2011, Lin, Mausam & Weld 2016). Relabelling part of the data is also a possible solution for some noisy CBM data.

Last but not least, how to integrate different models for CBM is an important problem. As presented in Chapter 5, multiple data channels would generate heterogeneous feature sets, and thus the models applied to these feature sets should ideally be channel-specific. Depending on the characteristics and formats of the feature sets, e.g. images, texts, categorical, numerical etc., appropriate models should be applied and integrated together. In this process, automatic model selection and hyper-parameter optimisation (Kotthoff, Thornton, Hoos, Hutter & Leyton-Brown 2017) will greatly facilitate the model design. Another type of model integration will be in a higher-level, like the ensemble learning (Zhang & Ma 2012), to leverage the advantages of different models. These two types of integration would possibly boost the model performance, but may also make it harder to interpret the results. An interpretable result is important for CBM tasks, so when integrating the models, it is better to take the interpretability into account.

Bibliography

Abdelgayed, T. S., Morsi, W. G. & Sidhu, T. S. (2017), 'Fault detection and classification based on co-training of semisupervised machine learning', *IEEE Transactions on Industrial Electronics* **65**(2), 1595–1605.

Afkanpour, A., György, A., Szepesvári, C. & Bowling, M. (2013), A randomized mirror descent algorithm for large scale multiple kernel learning, *in* 'International Conference on Machine Learning', PMLR, Atlanta, Georgia, USA, pp. 374–382.

Alon, N. (1995), 'Tools from higher algebra', *Handbook of combinatorics* **2**, 1749–1783.

Althloothi, S., Mahoor, M. H., Zhang, X. & Voyles, R. M. (2014), 'Human activity recognition using multi-features and multiple kernel learning', *Pattern Recognition* **47**(5), 1800–1812.

Azur, M. J., Stuart, E. A., Frangakis, C. & Leaf, P. J. (2011), 'Multiple imputation by chained equations: what is it and how does it work?', *International Journal of Methods in Psychiatric Research* **20**(1), 40–49.

Bartlett, P. L., Harvey, N., Liaw, C. & Mehrabian, A. (2019), 'Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks.', *Journal of Machine Learning Research* **20**(63), 1–17.

Batista, G. E. & Monard, M. C. (2002), 'A study of k-nearest neighbour as an imputation method.', *HIS* **87**(251-260), 48.

Bayer, C., Enge-Rosenblatt, O., Bator, M. & Mönks, U. (2013), Sensorless drive diagnosis using automated feature extraction, significance ranking and reduction, *in* '2013 IEEE 18th Conference on Emerging Technologies & Factory Automation', IEEE, pp. 1–4.

Bennane, A. & Yacout, S. (2012), 'Lad-cbm; new data processing tool for diagnosis and prognosis in condition-based maintenance', *Journal of Intelligent Manufacturing* **23**(2), 265–275.

Blondel, M., Fujino, A., Ueda, N. & Ishihata, M. (2016), Higher-order factorization machines, *in* 'Advances in Neural Information Processing Systems', pp. 3351–3359.

Blum, A. & Mitchell, T. (1998), Combining labeled and unlabeled data with co-training, *in* 'Proceedings of the 11th Annual Conference on Computational Learning Theory', pp. 92–100.

Boyd, K., Eng, K. H. & Page, C. D. (2013), Area under the precision-recall curve: point estimates and confidence intervals, *in* 'Joint European Conference on Machine Learning and Knowledge Discovery in Databases', Springer, pp. 451–466.

Brockwell, P. J., Davis, R. A. & Calder, M. V. (2002), *Introduction to time series and forecasting*, Vol. 2, Springer.

Bucak, S. S., Jin, R. & Jain, A. K. (2013), 'Multiple kernel learning for visual object recognition: A review', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **36**(7), 1354–1369.

Buuren, S. v. & Groothuis-Oudshoorn, K. (2010), 'mice: Multivariate imputation by chained equations in r', *Journal of Statistical Software* pp. 1–68.

Caetano, L. F. & Teixeira, P. F. (2015), 'Optimisation model to schedule railway track renewal operations: a life-cycle cost approach', *Structure and Infrastructure Engineering* **11**(11), 1524–1536.

Camci, F., Eker, O. F., Başkan, S. & Konur, S. (2016), 'Comparison of sensors and methodologies for effective prognostics on railway turnout systems', *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit* **230**(1), 24–42.

Cao, B., Zhou, H., Li, G. & Yu, P. S. (2016), Multi-view machines, *in* 'Proceedings of the Ninth ACM International Conference on Web Search and Data Mining', pp. 427–436.

Chang, C.-C. & Lin, C.-J. (2011), 'Libsvm: a library for support vector machines', *ACM TIST* **2**(3), 27.

Chang, Y.-W., Hsieh, C.-J., Chang, K.-W., Ringgaard, M. & Lin, C.-J. (2010), 'Training and testing low-degree polynomial data mappings via linear svm', *Journal of Machine Learning Research* **11**(Apr), 1471–1490.

Chechik, G., Heitz, G., Elidan, G., Abbeel, P. & Koller, D. (2008), 'Max-margin classification of data with absent features', *Journal of Machine Learning Research* **9**(Jan), 1–21.

Chen, M., Weinberger, K. Q. & Blitzer, J. (2011), Co-training for domain adaptation, *in* 'Advances in Neural Information Processing Systems', pp. 2456–2464.

Chen, X., Zheng, Y., Wang, J., Ma, W. & Huang, J. (2019), Rafm: Rank-aware factorization machines, *in* 'International Conference on Machine Learning', pp. 1132–1140.

Chin, W.-S., Zhuang, Y., Juan, Y.-C. & Lin, C.-J. (2015), A learning-rate schedule for stochastic gradient methods to matrix factorization, *in* 'Pacific-Asia Conference on Knowledge Discovery and Data Mining', Springer, pp. 442–455.

Cohn, D. A., Ghahramani, Z. & Jordan, M. I. (1996), 'Active learning with statistical models', *Journal of Artificial Intelligence Research* **4**, 129–145.

Davis, J. & Goadrich, M. (2006), The relationship between precision-recall and roc curves, *in* 'Proceedings of the 23rd International Conference on Machine Learning', pp. 233–240.

De Stefano, C., Maniaci, M., Fontanella, F. & di Freca, A. S. (2018), 'Reliable writer identification in medieval manuscripts through page layout features: The "avila" bible case', *Engineering Applications of Artificial Intelligence* **72**, 99–110.

Dekel, O., Shamir, O. & Xiao, L. (2010), 'Learning to classify with missing and corrupted features', *Machine learning* **81**(2), 149–178.

Dheeru, D. & Karra Taniskidou, E. (2017), 'UCI machine learning repository'.
URL: *http://archive.ics.uci.edu/ml*

Dhlamini, S. M., Nelwamondo, F. V. & Marwala, T. (2006), 'Condition monitoring of hv bushings in the presence of missing data using evolutionary computing', *WSEAS Transactions on Power Systems* **1**(2), 280–287.

Dick, U., Haider, P. & Scheffer, T. (2008), Learning from incomplete data with infinite imputations, *in* 'Proceedings of the 25th International Conference on Machine Learning', ACM, pp. 232–239.

Dong, J.-J., Tung, Y.-H., Chen, C.-C., Liao, J.-J. & Pan, Y.-W. (2011), 'Logistic regression model for predicting the failure probability of a landslide dam', *Engineering Geology* **117**(1-2), 52–61.

Dua, D. & Graff, C. (2017), 'UCI machine learning repository'.
URL: *http://archive.ics.uci.edu/ml*

Duchi, J., Hazan, E. & Singer, Y. (2011), 'Adaptive subgradient methods for online learning and stochastic optimization', *Journal of Machine Learning Research* **12**(Jul), 2121–2159.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R. & Lin, C.-J. (2008), 'Liblinear: A library for large linear classification', *JMLR* **9**, 1871–1874.

Fronza, I., Sillitti, A., Succi, G., Terho, M. & Vlasenko, J. (2013), 'Failure prediction based on log files using random indexing and support vector machines', *Journal of Systems and Software* **86**(1), 2–11.

García, F. P., Pedregal, D. J. & Roberts, C. (2010), 'Time series methods applied to failure prediction and detection', *Reliability Engineering & System Safety* **95**(6), 698–703.

García Márquez, F. P., Roberts, C. & Tobias, A. M. (2010), 'Railway point mechanisms: condition monitoring and fault detection', *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit* **224**(1), 35–44.

Ghahramani, Z. & Jordan, M. I. (1994), Supervised learning from incomplete data via an em approach, *in* 'Advances in Neural Information Processing Systems', pp. 120–127.

Goldberg, A., Recht, B., Xu, J., Nowak, R. & Zhu, J. (2010), Transduction with matrix completion: Three birds with one stone, *in* 'Advances in Neural Information Processing Systems', pp. 757–765.

Gönen, M. & Alpaydin, E. (2008), Localized multiple kernel learning, *in* 'Proceedings of the 25th International Conference on Machine learning', ACM, pp. 352–359.

Gönen, M. & Alpaydın, E. (2011), 'Multiple kernel learning algorithms', *JMLR* **12**(Jul), 2211–2268.

Gong, Y., Li, Z., Zhang, J., Liu, W., Zheng, Y. & Kirsch, C. (2018), Network-wide crowd flow prediction of sydney trains via customized online non-negative matrix factorization, *in* 'Proc. 27th ACM International Conference on Information and Knowledge Management', ACM, pp. 1243–1252.

Goodfellow, I., Mirza, M., Courville, A. & Bengio, Y. (2013), Multi-prediction deep boltzmann machines, *in* 'Advances in Neural Information Processing Systems', pp. 548–556.

Grimmer, B. (2019), 'Convergence rates for deterministic and stochastic subgradient methods without lipschitz continuity', *SIAM Journal on Optimization* **29**(2), 1350–1365.

Guclu, A., Yilboga, H., Eker, Ö. F., Camci, F. & Jennions, I. K. (2010), 'Prognostics with autoregressive moving average for railway turnouts'.

Guo, H., Tang, R., Ye, Y., Li, Z. & He, X. (2017), Deepfm: a factorization-machine based neural network for ctr prediction, *in* 'Proceedings of the 26th International Joint Conference on Artificial Intelligence', pp. 1725–1731.

Guo, Y., Sun, Y., Li, L. & Tang, X. (2019), 'Reliability assessment for multi-source data of mechanical parts of civil aircraft based on the model', *Journal of Mechanical Science and Technology* **33**(7), 3205–3211.

Hardoon, D. R., Szedmak, S. & Shawe-Taylor, J. (2004), 'Canonical correlation analysis: An overview with application to learning methods', *Neural Computation* **16**(12), 2639–2664.

Hassankiadeh, S. J. (2011), 'Failure analysis of railway switches and crossings for the purpose of preventive maintenance', *Transport Science* .

Hazan, E., Livni, R. & Mansour, Y. (2015), Classification with low rank and missing data., *in* 'ICML', pp. 257–266.

He, Q., Li, H., Bhattacharjya, D., Parikh, D. P. & Hampapur, A. (2015), 'Track geometry defect rectification based on track deterioration modelling and derailment risk assessment', *Journal of the Operational Research Society* **66**(3), 392–404.

He, X. & Chua, T.-S. (2017), Neural factorization machines for sparse predictive analytics, *in* 'Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval', pp. 355–364.

Hoi, S. C., Jin, R., Zhao, P. & Yang, T. (2013), 'Online multiple kernel classification', *Machine Learning* **90**(2), 289–316.

Hong, L., Doumith, A. S. & Davison, B. D. (2013), Co-factorization machines: modeling user interests and predicting individual decisions in twitter, *in* 'Proceedings of the 6th ACM International Conference on Web Search and Data Mining', pp. 557–566.

Hotelling, H. (1992), Relations between two sets of variates, *in* 'Breakthroughs in statistics', Springer, pp. 162–190.

Ishak, M. F., Dindar, S. & Kaewunruen, S. (2016), Safety-based maintenance for geometry restoration of railway turnout systems in various operational environments, *in* 'Proc. 21st National Convention on Civil Engineering'.

Jin, R., Hoi, S. C. & Yang, T. (2010), Online multiple kernel learning: Algorithms and mistake bounds, *in* 'International conference on algorithmic learning theory', Springer, pp. 390–404.

Juan, Y., Zhuang, Y., Chin, W.-S. & Lin, C.-J. (2016), Field-aware factorization machines for ctr prediction, *in* 'Proceedings of the 10th ACM Conference on Recommender Systems', pp. 43–50.

Kabir, G., Demissie, G., Sadiq, R. & Tesfamariam, S. (2015), 'Integrating failure prediction models for water mains: Bayesian belief network based data fusion', *Knowledge-Based Systems* **85**, 159–169.

Kankanala, P., Das, S. & Pahwa, A. (2013), 'Adaboost(+): An ensemble learning approach for estimating weather-related outages in distribution systems', *IEEE Transactions on Power Systems* **29**(1), 359–367.

Kloft, M., Brefeld, U., Laskov, P., Müller, K.-R., Zien, A. & Sonnenburg, S. (2009), Efficient and accurate lp-norm multiple kernel learning, *in* 'Advances in Neural Information Processing Systems', pp. 997–1005.

Kloft, M., Brefeld, U., Sonnenburg, S. & Zien, A. (2011), 'Lp-norm multiple kernel learning', *Journal of Machine Learning Research* **12**(Mar), 953–997.

Kobayashi, K., Kaito, K. & Lethanh, N. (2012), 'A bayesian estimation method to improve deterioration prediction for infrastructure system with markov chain model', *International Journal of Architecture, Engineering and Construction* **1**(1), 1–13.

Kolda, T. G. & Bader, B. W. (2009), 'Tensor decompositions and applications', *SIAM Review* **51**(3), 455–500.

Kotthoff, L., Thornton, C., Hoos, H. H., Hutter, F. & Leyton-Brown, K. (2017), 'Auto-weka 2.0: Automatic model selection and hyperparameter optimization in weka', *Journal of Machine Learning Research* **18**(1), 826–830.

Lai, P. L. & Fyfe, C. (2000), 'Kernel and nonlinear canonical correlation analysis', *International Journal of Neural Systems* **10**(05), 365–377.

Lanckriet, G. R., Cristianini, N., Bartlett, P., Ghaoui, L. E. & Jordan, M. I. (2004), 'Learning the kernel matrix with semidefinite programming', *Journal of Machine learning research* **5**(Jan), 27–72.

Le, Q., Sarlós, T. & Smola, A. (2013), Fastfood-approximating kernel expansions in loglinear time, *in* 'Proc. 30th International Conference on Machine Learning', Vol. 85.

Le Son, K., Fouladirad, M. & Barros, A. (2016), 'Remaining useful lifetime estimation and noisy gamma deterioration process', *Reliability Engineering & System Safety* **149**, 76–87.

LeCun, Y., Cortes, C. & Burges, C. (2010), 'Mnist handwritten digit database', *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist* **2**.

Lei, Y., Binder, A. & Kloft, M. (2016), Localized multiple kernel learning a convex approach, Technical report, University of Waikato Hamilton New Zealand.

Li, H., Huang, H.-Z., Li, Y.-F., Zhou, J. & Mi, J. (2018), 'Physics of failure-based reliability prediction of turbine blades using multi-source information fusion', *Applied Soft Computing* **72**, 624–635.

Li, J. R., Khoo, L. P. & Tor, S. B. (2006), 'Rmine: a rough set based data mining prototype for the reasoning of incomplete data in condition-based fault diagnosis', *Journal of Intelligent Manufacturing* **17**(1), 163–176.

Li, S. C.-X., Jiang, B. & Marlin, B. (2019), Misgan: Learning from incomplete data with generative adversarial networks, *in* 'International Conference on Learning Representations'.

Li, X., Gu, B., Ao, S., Wang, H. & Ling, C. X. (2017), Triply stochastic gradients on multiple kernel learning, *in* 'Proc. 33rd Conference on Uncertainty in Artificial Intelligence'.

Li, X., Wang, H., Gu, B. & Ling, C. X. (2015), Data sparseness in linear svm, *in* 'Proc. 24th International Joint Conference on Artificial Intelligence', pp. 3628–3634.

Li, Z., Zhang, J., Wu, Q. & Kirsch, C. (2018), Field-regularised factorization machines for mining the maintenance logs of equipment, *in* 'Australasian Joint Conference on Artificial Intelligence', Springer, pp. 172–183.

Liang, Y., Zhang, Y., Xiong, H. & Sahoo, R. (2007*a*), An adaptive semantic filter for blue gene/l failure log analysis, *in* '2007 IEEE International Parallel and Distributed Processing Symposium', IEEE, pp. 1–8.

Liang, Y., Zhang, Y., Xiong, H. & Sahoo, R. (2007*b*), Failure prediction in ibm bluegene/l event logs, *in* 'Seventh IEEE International Conference on Data Mining', IEEE, pp. 583–588.

Lin, C. H., Mausam, M. & Weld, D. S. (2016), Re-active learning: Active learning with relabeling., *in* 'AAAI', pp. 1845–1852.

Lin, K.-Y., Wang, C.-D., Meng, Y.-Q. & Zhao, Z.-L. (2017), Multi-view unit intact space learning, *in* 'International Conference on Knowledge Science, Engineering and Management', Springer, pp. 211–223.

Liu, C., Zhang, T., Zhao, P., Zhou, J. & Sun, J. (2017), Locally linear factorization machines, *in* 'IJCAI', pp. 2294–2300.

Liu, F., Zhou, L., Shen, C. & Yin, J. (2013), 'Multiple kernel learning in the primal for multimodal alzheimer's disease classification', *IEEE Journal of Biomedical and Health Informatics* **18**(3), 984–990.

Liu, X., Wang, L., Yin, J., Dou, Y. & Zhang, J. (2015), Absent multiple kernel learning., *in* 'AAAI', pp. 2807–2813.

Liu, X., Wang, L., Zhang, J. & Yin, J. (2014), Sample-adaptive multiple kernel learning., *in* 'AAAI', pp. 1975–1981.

Liu, X., Zhu, X., Li, M., Tang, C., Zhu, E., Yin, J. & Gao, W. (2019), Efficient and effective incomplete multi-view clustering, *in* 'Proceedings of the AAAI Conference on Artificial Intelligence', Vol. 33, pp. 4392–4399.

Liu, Z.-g., Pan, Q., Dezert, J. & Martin, A. (2016), 'Adaptive imputation of missing values for incomplete pattern classification', *Pattern Recognition* **52**, 85–95.

Lu, W., Yu, Y., Chang, Y., Wang, Z., Li, C. & Yuan, B. (2020), A dual input-aware factorization machine for ctr prediction, *in* 'Proceedings of the 29th International Joint Conference on Artificial Intelligence'.

Mathew, B. S. & Isaac, K. P. (2014), 'Optimisation of maintenance strategy for rural road network using genetic algorithm', *International Journal of Pavement Engineering* **15**(4), 352–360.

Moghaddass, R. & Zuo, M. J. (2014), 'An integrated framework for online diagnostic and prognostic health monitoring using a multistate deterioration process', *Reliability Engineering & System Safety* **124**, 92–104.

Moghaddass, R., Zuo, M. J. & Zhao, X. (2013), Modeling multi-state equipment degradation with non-homogeneous continuous-time hidden semi-markov process, *in* 'Diagnostics and Prognostics of Engineering Systems: Methods and Techniques', IGI Global, pp. 151–181.

Mohri, M. & Rostamizadeh, A. (2008), 'Rademacher complexity bounds for non-iid processes', *Advances in Neural Information Processing Systems* **21**, 1097–1104.

Mohri, M., Rostamizadeh, A. & Talwalkar, A. (2018), *Foundations of machine learning*, MIT press.

Molodova, M., Li, Z., Nunez, A. & Dollevoet, R. (2013), Monitoring the railway infrastructure: Detection of surface defects using wavelets, *in* '16th International IEEE Conference on Intelligent Transportation Systems', IEEE, pp. 1316–1321.

Necoara, I., Nesterov, Y. & Glineur, F. (2018), 'Linear convergence of first order methods for non-strongly convex optimization', *Mathematical Programming* pp. 1–39.

Necoara, I., Nesterov, Y. & Glineur, F. (2019), 'Linear convergence of first order methods for non-strongly convex optimization', *Mathematical Programming* **175**(1-2), 69–107.

Núñez, A., Hendriks, J., Li, Z., De Schutter, B. & Dollevoet, R. (2014), Facilitating maintenance decisions on the dutch railways using big data:

The aba case study, *in* 'IEEE International Conference on Big Data', IEEE, pp. 48–53.

Ohadi, A. & Micic, T. (2011), 'Stochastic process deterioration modelling for adaptive inspections', *Applications of Statistics and Probability in Civil Engineering* pp. 1085–1091.

Ortiz, E. M., Babbar, A., Syrmos, V. L., Clark, G. J., Vian, J. L. & Arita, M. M. (2008), Multi source data integration for aircraft health management, *in* '2008 IEEE Aerospace Conference', IEEE, pp. 1–12.

Ouyang, M., Welsh, W. J. & Georgopoulos, P. (2004), 'Gaussian mixture clustering and imputation of microarray data', *Bioinformatics* **20**(6), 917–923.

Oyebande, B. & Renfrew, A. (2002), 'Condition monitoring of railway electric point machines', *Iee Proceedings-Electric Power Applications* **149**(6), 465–473.

Ozenne, B., Subtil, F. & Maucort-Boulch, D. (2015), 'The precision–recall curve overcame the optimism of the receiver operating characteristic curve in rare diseases', *Journal of Clinical Epidemiology* **68**(8), 855–859.

Pan, J., Xu, J., Ruiz, A. L., Zhao, W., Pan, S., Sun, Y. & Lu, Q. (2018), Field-weighted factorization machines for click-through rate prediction in display advertising, *in* 'Proceedings of the 2018 World Wide Web Conference', pp. 1349–1357.

Park, D. H., Jung, G. M. & Yum, J. K. (2000), 'Cost minimization for periodic maintenance policy of a system subject to slow degradation', *Reliability Engineering & System Safety* **68**(2), 105–112.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L. & Lerer, A. (2017), 'Automatic differentiation in pytorch'.

Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T. & Efros, A. A. (2016), Context encoders: Feature learning by inpainting, *in* 'Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition', pp. 2536–2544.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, E. (2011), 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research* **12**, 2825–2830.

Peng, M., Li, X., Li, D., Jiang, S. & Zhang, L. M. (2014), 'Slope safety evaluation by integrating multi-source monitoring information', *Structural Safety* **49**, 65–74.

Podofillini, L., Zio, E. & Vatn, J. (2006), 'Risk-informed optimisation of railway tracks inspection and maintenance procedures', *Reliability Engineering & System Safety* **91**(1), 20–35.

Qiang, R., Liang, F. & Yang, J. (2013), Exploiting ranking factorization machines for microblog retrieval, *in* 'Proceedings of the 22nd ACM International Conference on Information & Knowledge Management', pp. 1783–1788.

Qiao, S., Shen, W., Zhang, Z., Wang, B. & Yuille, A. (2018), Deep co-training for semi-supervised image recognition, *in* 'Proceedings of the European Conference on Computer Vision', pp. 135–152.

Qin, C., Wang, L., Zhang, Y. & Fu, Y. (2019), Generatively inferential co-training for unsupervised domain adaptation, *in* 'Proceedings of the IEEE International Conference on Computer Vision Workshops', pp. 0–0.

Rahimi, A. & Recht, B. (2008), Random features for large-scale kernel machines, *in* 'Advances in Neural Information Processing Systems', pp. 1177–1184.

Rakotomamonjy, A., Bach, F. R., Canu, S. & Grandvalet, Y. (2008), 'Simplemkl', *Journal of Machine Learning Research* **9**(Nov), 2491–2521.

Rakotomamonjy, A. & Chanda, S. (2014), 'Lp-norm multiple kernel learning with low-rank kernels', *Neurocomputing* **143**, 68–79.

Rama, D. & Andrews, J. D. (2013), 'A reliability analysis of railway switches', *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of rail and rapid transit* **227**(4), 344–363.

Rendle, S. (2010), Factorization machines, *in* '2010 IEEE International Conference on Data Mining', IEEE, pp. 995–1000.

Rendle, S., Gantner, Z., Freudenthaler, C. & Schmidt-Thieme, L. (2011), Fast context-aware recommendations with factorization machines, *in* 'Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval', pp. 635–644.

Robles-Velasco, A., Cortés, P., Muñuzuri, J. & Onieva, L. (2020), 'Prediction of pipe failures in water supply networks using logistic regression and support vector classification', *Reliability Engineering & System Safety* **196**, 106754.

Sahoo, D., Hoi, S. C. & Li, B. (2014), Online multiple kernel regression, *in* 'Proc. 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', ACM, pp. 293–302.

Saito, T. & Rehmsmeier, M. (2015), 'The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets', *PloS One* **10**(3), e0118432.

Salfner, F. & Tschirpke, S. (2008), Error log processing for accurate failure prediction., *in* 'WASL'.

Saxena, A., Goebel, K., Simon, D. & Eklund, N. (2008), Damage propagation modeling for aircraft engine run-to-failure simulation, *in* '2008 International Conference on Prognostics and Health Management', IEEE, pp. 1–9.

Shafiee, M., Patriksson, M. & Chukova, S. (2016), 'An optimal age–usage maintenance strategy containing a failure penalty for application to railway tracks', *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit* **230**(2), 407–417.

Shen, Y. & Chen, T. (2018), Online ensemble multi-kernel learning adaptive to non-stationary and adversarial environments, *in* 'Proc. 21st International Conference on Artificial Intelligence and Statistics', Vol. 84.

Shen, Y., Chen, T. & Giannakis, G. B. (2018), Online multi-kernel learning with orthogonal random features, *in* 'IEEE International Conference on Acoustics, Speech and Signal Processing', IEEE, pp. 6289–6293.

Shivaswamy, P. K., Bhattacharyya, C. & Smola, A. J. (2006), 'Second order cone programming approaches for handling missing and uncertain data', *Journal of Machine Learning Research* **7**(Jul), 1283–1314.

Sindhwani, V., Niyogi, P. & Belkin, M. (2005), A co-regularization approach to semi-supervised learning with multiple views, *in* 'Proceedings of ICML Workshop on Learning with Multiple Views', Vol. 2005, Citeseer, pp. 74–79.

Sipos, R., Fradkin, D., Moerchen, F. & Wang, Z. (2014), Log-based predictive maintenance, *in* 'Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', pp. 1867–1876.

Śmieja, M., Struski, L. u., Tabor, J., Zieliński, B. & Spurek, P. a. (2018), Processing of missing data by neural networks, *in* 'Advances in Neural Information Processing Systems', pp. 2719–2729.

Sonnenburg, S., Rätsch, G., Schäfer, C. & Schölkopf, B. (2006), 'Large scale multiple kernel learning', *Journal of Machine Learning Research* **7**(Jul), 1531–1565.

Su, Z., Núñez, A., Baldi, S. & De Schutter, B. (2016), Model predictive control for rail condition-based maintenance: A multilevel approach., *in* 'ITSC', pp. 354–359.

Su, Z., Núñez, A., Jamshidi, A., Baldi, S., Li, Z., Dollevoet, R. & De Schutter, B. (2015), Model predictive control for maintenance operations planning of railway infrastructures, *in* 'International Conference on Computational Logistics', Springer, pp. 673–688.

Suh, M.-k., Woodbridge, J., Lan, M., Bui, A., Evangelista, L. S. & Sarrafzadeh, M. (2011), Missing data imputation for remote chf patient monitoring systems, *in* '2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society', IEEE, pp. 3184–3187.

Tao, H. & Zhao, Y. (2015), 'Intelligent fault prediction of railway switch based on improved least squares support vector machine', *Metallurgical and Mining Industry* **7**(10), 69–75.

Tong, S. & Koller, D. (2001), 'Support vector machine active learning with applications to text classification', *Journal of Machine Learning Research* **2**(Nov), 45–66.

Tsuda, Y., Kaito, K., Aoki, K. & Kobayashi, K. (2006), 'Estimating markovian transition probabilities for bridge deterioration forecasting', *Structural Engineering/Earthquake Engineering* **23**(2), 241s–256s.

Twala, B. (2009), Robot execution failure prediction using incomplete data, *in* '2009 IEEE International Conference on Robotics and Biomimetics', IEEE, pp. 1518–1523.

Wang, G., Xu, T., Tang, T., Yuan, T. & Wang, H. (2017), 'A bayesian network model for prediction of weather-related failures in railway turnout systems', *Expert Systems with Applications* **69**, 247–256.

Wang, G., Xu, T., Wang, H. & Zou, Y. (2016), Adaboost and least square based failure prediction of railway turnouts, *in* 'International Symposium on Computational Intelligence and Design', Vol. 1, IEEE, pp. 434–437.

Wang, J., Li, C., Han, S., Sarkar, S. & Zhou, X. (2017), 'Predictive maintenance based on event-log analysis: A case study', *IBM Journal of Research and Development* **61**(1), 11–121.

Wang, S., Zhou, M., Fei, G., Chang, Y. & Liu, B. (2018), 'Contextual and position-aware factorization machines for sentiment classification', *arXiv preprint arXiv:1801.06172* .

Weiyu, C., Yanyan, S. & Linpeng, H. (2020), Adaptive factorization network: Learning adaptive-order feature interactions, *in* 'The Thirty-Fourth AAAI Conference on Artificial Intelligence', pp. 3609–3616.

White, M., Zhang, X., Schuurmans, D. & Yu, Y.-l. (2012), Convex multi-view subspace learning, *in* 'Advances in Neural Information Processing Systems', pp. 1673–1681.

Williams, D., Liao, X., Xue, Y. & Carin, L. (2005), Incomplete-data classification using logistic regression, *in* 'Proceedings of the 22nd International Conference on Machine learning', ACM, pp. 972–979.

Xia, J., Zhang, S., Cai, G., Li, L., Pan, Q., Yan, J. & Ning, G. (2017), 'Adjusted weight voting algorithm for random forests in handling missing values', *Pattern Recognition* **69**, 52–60.

Xiao, J., Ye, H., He, X., Zhang, H., Wu, F. & Chua, T.-S. (2017), Attentional factorization machines: learning the weight of feature interactions via attention networks, *in* 'Proceedings of the 26th International Joint Conference on Artificial Intelligence', pp. 3119–3125.

Xu, C., Tao, D. & Xu, C. (2013), 'A survey on multi-view learning', *arXiv preprint arXiv:1304.5634* .

Xu, C., Tao, D. & Xu, C. (2015*a*), 'Multi-view intact space learning', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **37**(12), 2531–2544.

Xu, C., Tao, D. & Xu, C. (2015*b*), 'Multi-view learning with incomplete views', *IEEE Transactions on Image Processing* **24**(12), 5812–5825.

Xu, Z., Jin, R., Yang, H., King, I. & Lyu, M. R. (2010), Simple and efficient multiple kernel learning by group lasso, *in* 'ICML', Omnipress, pp. 1175–1182.

Yamada, M., Lian, W., Goyal, A., Chen, J., Wimalawarne, K., Khan, S. A., Kaski, S., Mamitsuka, H. & Chang, Y. (2017), Convex factorization machine for toxicogenomics prediction, *in* 'Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', pp. 1215–1224.

Yang, C., Lu, X., Lin, Z., Shechtman, E., Wang, O. & Li, H. (2017), High-resolution image inpainting using multi-scale neural patch synthesis, *in* 'Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition', pp. 6721–6729.

Yang, J., Tian, Y., Duan, L.-Y., Huang, T. & Gao, W. (2012), 'Group-sensitive multiple kernel learning for object recognition', *IEEE Transactions on Image Processing* **21**(5), 2838–2852.

Yang, T. & Lin, Q. (2018), 'Rsg: Beating subgradient method without smoothness and strong convexity', *Journal of Machine Learning Research* **19**(6), 1–33.

Yeh, C.-Y., Huang, C.-W. & Lee, S.-J. (2011), 'A multiple-kernel support vector regression approach for stock market price forecasting', *Expert Systems with Applications* **38**(3), 2177–2186.

Yilboga, H., Eker, Ö. F., Güçlü, A. & Camci, F. (2010), Failure prediction on railway turnouts using time delay neural networks, *in* '2010 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications', IEEE, pp. 134–137.

Yoon, J., Jordon, J. & Schaar, M. (2018), Gain: Missing data imputation using generative adversarial nets, *in* 'International Conference on Machine Learning', pp. 5675–5684.

Yu, Y., Wang, Z. & Yuan, B. (2019), An input-aware factorization machine for sparse prediction, *in* 'Proceedings of the 28th International Joint Conference on Artificial Intelligence', AAAI Press, pp. 1466–1472.

Zhai, Y., Ong, Y.-S. & Tsang, I. W. (2014), 'The emerging" big dimensionality"', *IEEE Computational Intelligence Magazine* **9**(3), 14–26.

Zhang, C. & Ma, Y. (2012), *Ensemble machine learning: methods and applications*, Springer.

Zhang, F. (2006), *The Schur complement and its applications*, Vol. 4, Springer Science & Business Media.

Zhang, L., Shen, W., Huang, J., Li, S. & Pan, G. (2019), 'Field-aware neural factorization machine for click-through rate prediction', *IEEE Access* **7**, 75032–75040.

Zhang, X. & Gao, H. (2012), 'Determining an optimal maintenance period for infrastructure systems', *Computer-Aided Civil and Infrastructure Engineering* **27**(7), 543–554.

Zhao, J., Xie, X., Xu, X. & Sun, S. (2017), 'Multi-view learning overview: Recent progress and new challenges', *Information Fusion* **38**, 43–54.

Zhao, L., Sukthankar, G. & Sukthankar, R. (2011), Incremental relabeling for active learning with noisy crowdsourced annotations, *in* '2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing', IEEE, pp. 728–733.

Zheng, Z., Lan, Z., Park, B. H. & Geist, A. (2009), System log pre-processing to improve failure prediction, *in* '2009 IEEE/IFIP International Conference on Dependable Systems & Networks', IEEE, pp. 572–577.