TODO TITLE

A report submitted to the University of Manchester for the degree of Bachelor of Science in the Faculty of Science and Engineering

Author: Sai Putravu
Student id: 10829976
Supervisor: TODO

2025

School of Computer Science

Contents

List of Figures

List of Tables

Abbreviations and Acronyms

Alphabetically sort this

BERT	Bidirectional Encoder Representation from Transformers.
CBM	Condition-based maintenance policy.
CNN	Convolutional Neural Network.
DBSCAN	Density-Based Spatial Clustering of Application with Noise.
DL	Deep Learning.
EPB1	Extracted proton beam line 1.
EPB2	Extracted proton beam line 2.
ELMo	Embeddings from Language Models.
FLD	First-line diagnosis system.
GPT	Generative Pre-trained Transformer.
HDBSCAN	Hierarchical Density-Based Spatial Clustering of Application with Noise.
HDD	Hard-Disk Drive.
HEDS	High energy drift space.
IoT	Internet of Things.
LEBT	Low energy beam transport line.
LINAC	Linear Accelerator.
FAP	Fault Analysis Pathway.
LSA	Latent Semantic Analysis.
ML	Machine Learning.
NLP	Natural Language Processing.
PCA	Principle Component Analysis.
PdM	Predictive maintenance policy.
PLM	Pre-trained Language Model.
PvM	Preventative maintenance policy.
R2F	Run-to-failure maintenance policy.
RFQ	Radio-frequency quadrupole.
RF	Random Forest.
SAFE	Supervised Aggregative Feature Extraction.
SMART	Self-monitoring and reporting technology.
STFC	Science and Technology Facilities Council.
SVD	Singular Value Decomposition.
SVM	Support Vector Machine.
TS-1	Target Station 1.
TS-2	Target Station 2.
t-SNE	t-distributed Stochastic Neighbour Embedding.
UKRI	United Kingdom Research and Innovation.
UMAP	Uniform Manifold Approximation and Projection.
MCR	Main Control Room.

Abstract

TODO

Chapter 1

Introduction

This project concerns the use of statistical and Machine Learning models to augment the predictive maintenance process at the STFC Rutherford Appleton Laboratory's ISIS Neutron and Muon research facility. The ISIS Neutron and Muon facility is a research centre for physical and life sciences, owned and operated by the STFC, a council that forms the UK Research and Innovation. In order to produce beams of neutrons and muons, allowing scientists to study materials at the atomic level, large and complex structures and machinery are required. The facility has a wealth of instrumentation taking multitudes of measurements to ensure that proper maintenance is completed in a timely manner.

Finish this off

: Introduce the sections of the paper.

- Refine this after coming back
- Section 2,
- Section 3,
- ...

1.1 Motivation

Motivate the research project.

: The things in this section will include

- Introduce the problem: Auto-categorisation and label inference
- Highlight the research aims. This should highlight the vastly open nature of the project and then hone in on the particular issue I am tackling. (i.e. to progress the state of PdM ...)
- Identify the data input, expected output, data shape and explain why this motivates the project

1.2 Overview of the ISIS Research Facility

The STFC Rutherford Appleton Laboratory's ISIS Neutron and Muon research facility is a research centre for physical and life sciences. This is owned and operated by the STFC, which is a council that forms the United Kingdom Research and Innovation (UKRI). The UKRI is a public body of the United Kingdom's Government that directs funding for research and innovation through the science budget of the Department for Science, Innovation and Technology. ISIS was designed in the 1970s and early 1980s, with the core of the design being a strong-focusing machine with size radio frequency accelerating cavities to provide an average beam current of $200\mu A$ [52]. According to ([52]), with the introduction of ISIS and other neutron sources, rapid development of neutron instrumentation was stimulated. Through time, many components instrumentation have been added such as the second target station - the current schematic representation of the facility can be seen in Figure 1.1.

This section details the various stages of the ISIS facility and discusses the current maintenance procedures.

1.2.1 The end-to-end production of neutrons and muons

The ISIS facility, described in A Practical Guide to the ISIS Neutron and Muon Source (ISIS Neutron and Muon Source [18]), is comprised of four major stages: (1) the ion source, (2) the radio-frequency quadrupole (RFQ), (3) linear accelerator (LINAC) and (4) the synchrotron. These components combined produce a 800 MeV proton beam that is directed to either target station 1 (TS-1) or target station 2 (TS-2). Starting with H^- ions (protons with two orbiting electrons), the path taken through the four stages is shown in Figure 1.2. Throughout the entire H^- or proton acceleration process, the beam must be kept under good vacuum and, to do so, many tens of vacuum pumps are maintaining the vacuum between 10^{-8} and 10^{-9} of atmospheric pressure.

The first stage of the ISIS machine is the ion source, which generates the negative hydrogen (H^-) ions, which are then accelerated through the RFQ and LINAC. This ion source is a pulsed source of H^- ions, which is ionised from a stream of Hydrogen gas by arranging for an electric discharge to take place between an anode and a cathode inside. According to the guide, published by the ISIS facility, around 20ml of hydrogen gas per minute is continuously delivered to the ion source from a hydrogen gas bottle. Once the H^- ions emerge from the

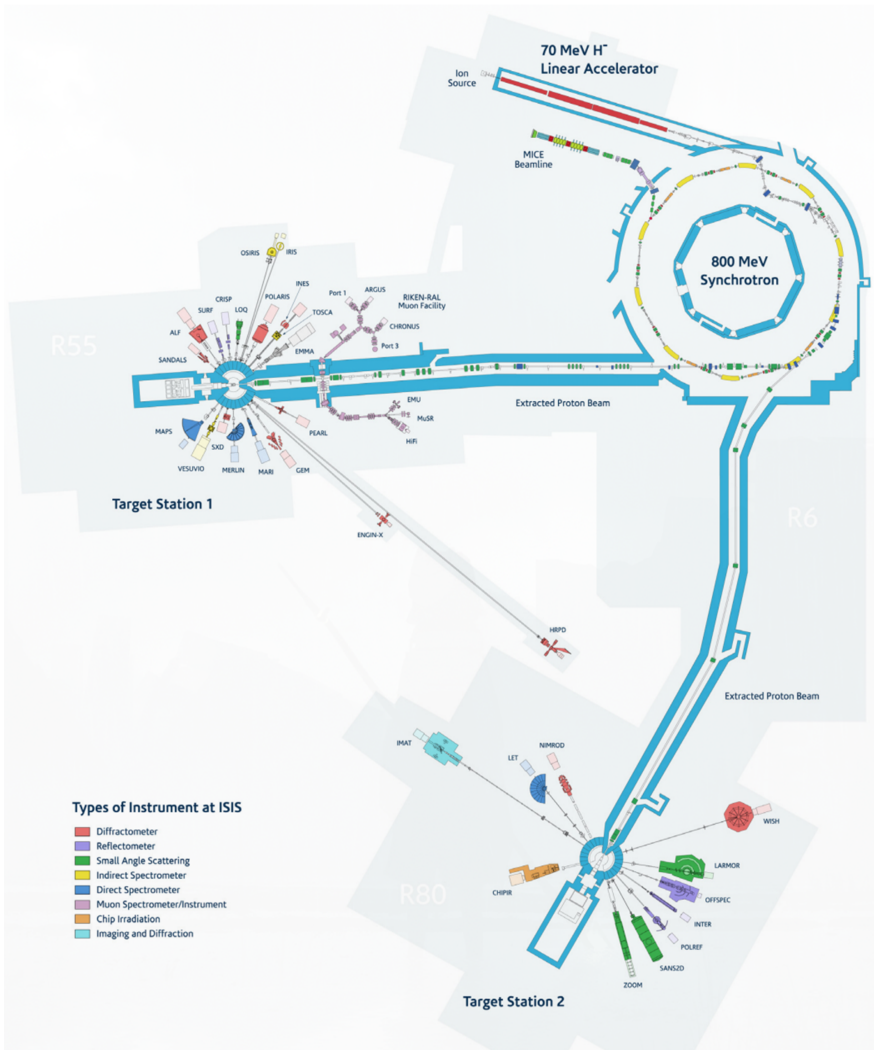

Figure 1.1: The schematic representation of the physical layout of ISIS. The light grey areas are footprints of the buildings. Source: (Thomason [52]).

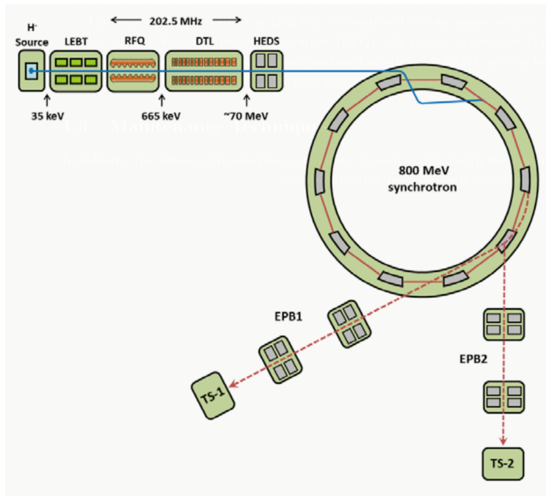

Figure 1.2: ISIS ion source and chain of accelerators, with H^- ions in blue and protons in red. Not to scale. Source: (ISIS Neutron and Muon Source [18]).

ion source, they have an energy of 35 keV and are presented to the RFQ via the Low Energy Beam Transport line (LEBT). The LEBT prevents the low-energy H^- beam from increasing in size (due to the mutual repulsion of the ions) and also incorporates a beam stop. This is essentially just a remotely removable sheet of metal that physically blocks the beam.

Next, the ISIS RFQ uses high-intensity radio frequency electric fields that focus, bunch and accelerate this H^- beam. The RFQ is located roughly one to two metres downstream from the ion source and immediately upstream of the LINAC and the beam leaves the RFQ at 665 keV. One of the roles of the RFQ is to effectively decouple the ion source from the LINAC, reducing the extent of variations in the beam in the LINAC, synchrotron and targets.

The LINAC consists of four accelerating tanks which progressively accelerate the beam from 665 keV to 70 MeV using high-intensity radio-frequency fields. The whole structure is arranged purposefully to be highly resonant (with a Q factor [29] of roughly 50000) at the operating radio-frequency. The tank structures are resonant at a sufficient enough level that their dimensions must be determined extremely precisely. This is done through controlling the temperature of a tank cooling water to significantly less than 1°C which essentially eliminates the effects of thermal expansion and contraction. It should be noted that, as the second electrons making up the H^- ions are not tightly bound, if the vacuum inside the LINAC tanks is faulty, then there could be a significant increase in radioactivity and beam loss. It is therefore important to maintain vacuum levels at the correct thresholds, mentioned in the guide. Beam loss monitors are installed over the length of the LINAC to provide metrics and warning signals when excessive losses occur. The output from the LINAC is a stream of H^- ions lasting around 200 μs.

The ion beam from the LINAC is injected into the synchrotron via the High Energy Drift Space (HEDS) which is a roughly 50m beam transport line. To enter the synchrotron, charge-exchange injection [3] occurs and the H^- ions are stripped of their electrons to become protons.

Charge-exchange injection is chosen as it is the most efficient way of taking and wrapping a long string of particles around the circumference of a synchrotron.

Once the protons at 70 MeV enter the synchrotron, a machine made up of many large magnets with a 26m circular orbit of radius and 163m circumference, they are accelerated to 800 MeV, 86% of the speed of light. The synchrotron is made up of ten sections called 'superperiods' each of which bends the beam at an angle of 36° using the magnets. These magnets further prevent beam expansion. The guide describes the LINAC to be thought of as a 'once-through' machine and the synchrotron as a 'thousands-of-times-around' machine. Additionally, in the synchrotron, there are four beam loss monitors to each superperiod to provide alerts when excessive beam losses arise due to poor synchronism. Every time the protons complete one full cycle, they pick up roughly 0.1 MeV of energy. Therefore, it takes just under 8000 turns for the protons to be accelerated to 800 MeV. Further configuration details can be found in the guide. The operational success of a synchrotron depends on good synchronism as the protons must be kept in the same orbit whilst being accelerated. To do this, the magnetic fields in the magnets and strength of the accelerating voltage have to increase with time to match the accelerating protons. After the injection from the LINAC is complete, a low-level accelerating voltage is applied, gradually separating the constant ring of charge into two separate bunches, which are a little more than $0.1\mu s$ apart. Finally, to send the two proton bunches to a particular target station, there exist three kicker magnets which deflect the bunches upwards. These are then bent up into the respective extracted proton beam lines (EPB1 or EPB2).

These two extracted proton beam lines deliver, consisting of a long series of magnets for directing, bending and steering, the proton bunches from the synchrotron to the respective target stations (TS-1 and TS-2). Both EPB1 and EPB2 lines are enclosed in thick steel and concrete shielding to account for cases of complete beam loss. Again, beam loss monitors are positioned along both lines to provide warnings of excessive beam losses.

There are two neutron producing targets on ISIS, which convert the high-energy proton beam as many neutrons as possible in the smallest amount of volume. This means choosing a target material with a high atomic number and highest neutron fluxes [45]. As described in the guide, ISIS uses tantalum-clad tungsten targets. These neutron-producing targets are surrounded by four moderators (a liquid hydrogen moderator at 100 K, liquid hydrogen moderator at 20 K and two water moderators at 300 K) and a beryllium reflector [43]. Around 20m upstream from the TS-1 neutron producing target, the ISIS Intermediate Target is incorporated in EPB1. This is a roughly 1cm thick graphite target that is placed directly in the beam to produce pions that decay into muons. These muons are selected by the muon beam line which transports these to various instrumentation.

With the detailed description of the steps required for the production of neutrons and muons, sourced from the guide, we highlight some key takeaways. There are an incredible number of points of failure in the facility due to the highly precise and radioactive nature task. The ISIS team have taken many precautions such as fault detection mechanisms (i.e. the beam loss monitors), breakdown counter-measures (i.e. the thick layer of steel and concrete in the EPB1 and EPB2 lines). However, as there are a large amount of components that can go wrong and issues that may arise, regular maintenance is required.

1.2.2 Maintenance at ISIS

As detailed in the 33-year historical account of the ISIS facility (Thomason [52]), the ISIS operations occur in cycles, periods of roughly 30-50 days where the machine runs constantly without breaks. Roughly 10 days before the scheduled start of a cycle, and gaps between cycles typically range from 1 week to 3 months. In addition, typically every four years, shutdowns scheduled for 6-9 months occur for major maintenance and upgrade work. On the ground, day-to-day operations are run from the Main Control Room (MCR) by the ISIS crew which consists of 6 shift teams of the following roles: (1) duty officer, (2) assistant duty officer and (3) duty technician. Downtime, quantified as the amount of time the machine is out-of-use, is a major consideration when deciding maintenance schedules. Many factors affect downtime such as having a robust plan, the day-to-day operators knowledge of the system and adequate inventories of spares [52]. Figure 1.3 shows the ratio of downtime to active machine operation since 2016, with over half the lifetime of the machine being down. In other words, over half the time, the machine is not active and is undergoing maintenance. This can be seen as a trade-off, between keeping the machine operative versus reducing failures through planned maintenance. The major maintenance strategies and their trade-offs are further explored in Section 1.3.

Add the graph, if you can ever figure out the operating cycles before 2016.

To help reduce downtime, over the last few years, a first-line diagnosis (FLD) system was introduced, presented in (Asim Yaqoob [4]). The FLD system helps reduce downtime by providing expert guidance on fault diagnosis and resolution, which has been shown to improve the dissemination of knowledge from experts to operations. The FLD utilises Fault Analysis Pathways (FAPs), which provide structural links between ISIS subsystems. This allows users of the system to access granular sub-systems' local documentation minimising file hunting and saving time and effort. An example FAP can be seen in Figure 1.4.

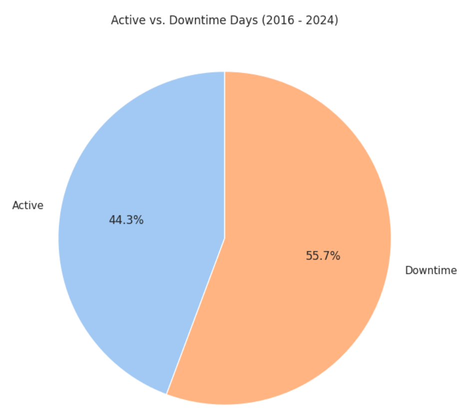

Figure 1.3: A visualisation of the machine downtime as opposed to time operating, according to the ISIS operational cycle. Data source: (ISIS Neutron and Muon Source [19])

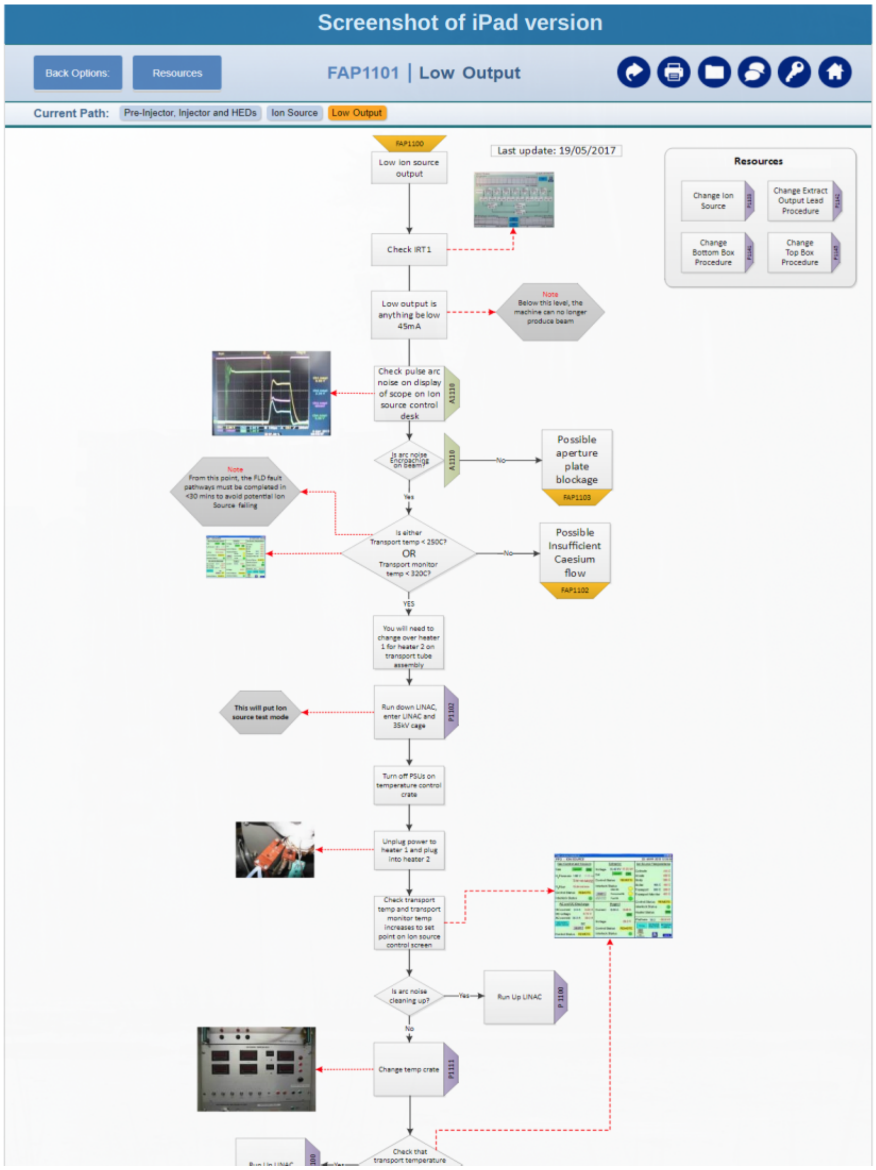

Figure 1.4: Example FAP1101, screenshotted from an iPAD version of FLD version 2.2, which shows the FAP for pre-injector, injector and high energy drift space. Source: (Asim Yaqoob [4])

1.2.3 Datasets

As mentioned previously, the ISIS research

: Introduce the research topic. The things in this section will include

- Talk about the ISIS research facility - DONE
- Talk about the Operational Cycle for ISIS - SORT OF DONE (graph too)
- Talk about the ISIS Crew and importance of having trained staff on premises. - SORT OF
- Talk about the Lost time and why it is important to minimise this for the ISIS research facility. - SORT OF
- Describe the first-line diagnosis system (FLD) and FAPs. - DONE
- Talk about the Datasets, operalog

1.3 Maintenance Techniques

In industry, the uptime of production systems are strongly coupled with the equipment maintenance. So much so that what was once considered a 'necessary evil' is now seen as a 'profit contributor' to be able to maintain a competitive edge [55, 16]. For facilities aiming to provide systems for research, maintenance impacts the downtime and cost of running. As a result, both to minimise unexpected downtime and provide a competitive edge, many industrial applications collect vast quantities of data during the entire life cycle of the system. This large amount of data may include information about processes, events and alarms [11] which occur along the industrial production line, collected by different equipment. The equipment may be located in different locations in the sub-components of the larger system or even different sub-components themselves.

talk about maintenance itself in a lot more depth

In the literature, various terms and categories of maintenance arise each with differing strategies [49, 32, 48]. Thus, while there exists some disagreement in nomenclature, we consider the four categories presented in (Susto et al. [49]). The four maintenance policy categories are as follows, noting that each policy has, uniquely, their own benefits and drawbacks:

1. Run-to-failure (R2F) maintenance: Continual usage of the system until failure. Restoration is performed at the point of noticing failure condition. This is the simplest approach and typically the most costly method as it reduces the facility's availability and requires a complete replacement of parts.
2. Preventative maintenance (PvM): Otherwise referred to as scheduled maintenance, performing maintenance at regular intervals to increase longevity of the component or in anticipation of the end of expected life of the component. While this typically prevents many errors, it wastes maintenance cycles when systems are perfectly healthy. Hence, causing unnecessary downtime and cost.
3. Condition-based maintenance (CBM): Taking the action to perform maintenance on equipment through monitoring various health characteristics and metrics of the com-

ponents of the system. This approach requires continuous monitoring and, thus, allows for close to instant response on maintenance only when required. However, a drawback of this policy is that one cannot plan maintenances in advance.

4. Predictive maintenance (PdM): Otherwise referred to as statistical-based maintenance, only performs maintenance actions when determined necessary. Prediction tools are utilised to implement forward-planning and scheduling systems, using statistical inference methods. However, if these statistical inferences are not accurate, the whole system suffers which inevitably leads to additional downtime and costs.

It should be noted, that several sources conflate CBM and PdM [32]. As in (Susto et al. [49]), where they are given as separate categories, we follow suit.

The PdM strategy stands out in the four categories presented as, given a statistical inference model that is able to detect faults accurately, this policy optimises the trade-off between improving equipment condition, reduce failure rates for equipment and minimising maintenance costs [11]. This technique enables one to apply foresight for pre-emptive scheduling of large-scale maintenance. As pointed out in Section (...), the ISIS facility aims to strike a balance between PvM, CBM and PdM through periods of large-scaled scheduled maintenance and collection of high quantities of metrics. This is done through the careful coordination between cycle scheduling, day-to-day crew-based monitoring and the FLD [52].

Implement the ISIS version of this in the background. Talk about the FIRST LINE DIAGNOSIS system

fill this

In industry, many maintenance strategies prefer using PdM whilst experimenting with a variety of statistical inference and artificial intelligence modelling approaches [32, 20]. Some examples from [11] are listed in Table 1.1 which highlights the trend in the industry towards more accurate, ML-based approaches.

Table 1.1: Examples of applications of PdM for industrial maintenance strategies.

Type	**Description**	**Reference**
Statistical	Application of SAFE to deal with PdM problems characterised by time-series data. The approach is tested on a real-life dataset of the semiconductor ion implantation process.	(Susto and Beghi [48])
ML	Application of SVM classification for fault prediction of rail networks, with discussion on using the model in optimising trade-offs related to maintenance schedule and costs.	(Li et al. [25])
ML	Audio analysis on IoT devices, enabling acoustic event recognition for machine diagnosis. This paper describes designing an end-to-end system, utilising CNN-based classification.	(Pan et al. [35])
ML	Utilisation of RF decision trees trained on SMART data to predict reliability of HDD in real-time.	(Su and Huang [47])

1.4 Sentence Similarity

Think of a good transition between PdM and Doc Similarity

Sentence similarity, otherwise referred to as document similarity, is the (NLP) task of computing the quantification of the similarities between two sentences, documents or texts. This task is motivated by the increasingly large amount of digitisation of human languages (and data, in general), calling for the need to understand similarity between various texts [39]. Examples of the use-cases of sentence similarity include: detection of academic malpractice via plagiarism [27, 5] and text summarisation [2, 22, 21]. According to [39], there are two main types of sentence similarities: (1) lexical similarity and (2) semantic similarity. The former is a computation of the equality between the lexicon of two sentences (i.e. a purely syntactical view), as opposed to the latter being a comparison between the semantics. Further, the type we focus on, semantic similarity can be split into three types:

- String-based similarity: Measures similarity directly between two strings, accounting for string sequences and character composition. These can be fine-grained, i.e. character-based; coarse-grained, i.e. term-based; or a hybrid mixture of both [59].

- Knowledge-based similarity: Measures the degree to which two sentences are related, utilising semantic networks (i.e. knowledge graphs). Examples of Knowledge-based similarity approaches include WordNet [7], the most popular type of approach.

- Corpus-based similarity: Premised on a provided corpus, a large database of text to derive inferences from. Methods of this type require the development statistical or DL models that train on the provided corpus and estimate the similarity between two sentence-pair inputs. Popular examples include traditional statistical models, such as LSA [23] and SVD [46] as well as word embedding models (utilising ML), such as Word2Vec [6], GloVe [37] and fastText [30].

Most of the models mentioned above require some numerical representation of the text to be able to apply mathematical procedures for similarity calculation. Computing this representation involves converting unstructured textual data into one or more vectors. Typically, this process includes (1) general pre-processing steps such as stop-word removal, case normalisation, parts-of-speech tagging, lemmatisation, and tokenisation [50]; and (2) applying an embedding model, either to a single token (word embedding) or to a sequence of tokens (sentence embedding). Step (1) can be seen as a feature extraction step applied on the unstructured textual data, where feature extraction is the process of extracting the most useful components of the data [41]. For example, part-of-speech tagging can be seen as introducing non-trivial features to some token through extracting the surrounding context.

This representation is known as an embedding, with the span of the possible vectors referred to as the embedding space. The dimension of the span is termed the embedding dimension. This is an important concept because the characteristics of the embedding space influence the model's ability to capture syntactic and semantic meaning in text, as the embedding space itself encodes this information. This can be seen in the Word2Vec model, described in (Bojanowski et al. [6]), which shows different embedding dimensions produce different results. Another conclusion that can be drawn from this paper is that, if embedding space is not constructed to maximise the meaning of texts, the accuracy of model predictions tends to deteriorate.

Therefore, the problem of sentence similarity can be directly mapped from the problem of sentence embedding (otherwise referred to as text embedding), where text embedding is the (NLP) task of learning a high-dimensional embedding space representation. Various aspects of text embedding are more thoroughly covered in Section 2.1. However, with the advent of the transformer architecture [54] and rise of the large language models, text embedding has been increasingly solved using DL models with high parameter counts [10] and considering extremely large token sequences. Nowadays, word embedding models are considered obsolete with (Cao [10]) only considering these models second-generation. Further, the paper states newer generations fall into the following categories:

- Third-generation: contextualised embeddings. These models dynamically account for contexts, encoding them into the embedding space. Examples of models include ELMo [42], GPT [38] and BERT [13]. As these models are trained to both understand some embedding space and generate natural language text, they are canonically referred to as language models.

- Fourth-generation: universal text embeddings. The generation which is currently state-of-the-art, with the aim of developing a unified model which is able to address multiple downstream tasks. Examples of models in this generation, making progress towards unification include Gecko [24], Multilingual e5 text embeddings [56], Nomic [34] and many more.

Second-, third- and fourth-generation text embedding models are used frequently in PdM for applications such as insight extraction [1, 53] and clustering intents from unstructured text data [33]. Sources of natural language datasets, in industrial applications typically arise from operational or managerial log files which document aspects such as failures, resolutions and comments similar to the ISIS facility failure logs ([**?**] see Section **??**). Advanced text embedding models enable for semi- or fully automatic insight retrieval and auto-categorisation, enabling intuitive understanding of the textual datasets potentially highlighting patterns in failure.

cite and fill when section exists

1.5 (maybe) Clustering

Talk about clustering lit. rev.

Think whether it is useful to present literature review in this section.

: The things in this section will include

- (DONE) Looking at general predictive maintenance
- (Done) Looking at general predictive maintenance in industrial applications
- (Done) Similar pairwise sentence similarity literature
- Similar literature in text clustering
- (sort of DONE) Similar literature in specifically sentence clustering in industrial applications

Think of a good transition between Sentence Similarity and Clustering

Chapter 2

Natural Language Processing Background

This chapter delves into the technical background required in understanding and appreciating the approach proposed in Chapter 3. Firstly, in Section 2.1.1, we discuss the technical details of the (third-generation text embedding) BERT model and its family of encoder-only transformers [13]. Specifically we further explore two improvements over BERT (XLNet [58] and MPNet [44]). Then, in Section 2.1.1, we explore the state-of-the-art, fourth-generation Nomic [34] architecture. After, we cover two methods of dimensionality reduction (PCA [36, 17] and UMAP [28]) motivated by the need to visualise samples from the high-dimensional embedding spaces of the aforementioned models, in Section 2.2. Finally, we present three clustering algorithms - with one supervised (k-Medoids []) and two unsupervised (DBSCAN [15] and HDBSCAN [9]) in addition to four clustering evaluation metrics. The clustering metrics we look at are: (1) Inertia [], (2) Silhouette [40], (3) Davies-Bouldin Index [12], (4) Calinski-Harabasz Index [8].

find the reference for this

maybe talk about optuna, if we use it.

double check all citations here are not empty

: Describe the various technical factors required before attempting to understand the methodology. The things in this section will include

- Discuss sentence embedding, similarity measures: BERT, RoBERTA, MPNet, XLNet, NOMIC.
- Dimensional reduction techniques and need for them (UMAP, PCA, t-SNE).
- Clustering methods: kmedoids, DBSCAN, DBSCAN*/HDBCAN
- Clustering evaluation methods: todo I don't remember these off the top of my head
- Maybe briefly touch on Optuna?

2.1 Sentence Embedding

Briefly touched on in Section 1.4, sentence embedding (otherwise known as text embedding) is the NLP task of computing some high-dimensional embedding vector-space representation for unstructured text data. This numerical representation should encode the semantic and syntactic meaning of the text and establish meaningful relationships. For example, the sentence 'I like dogs' should have the opposite representation to 'I hate dogs'. However these sentences should be more related than to the sentence 'My house was destroyed in an earthquake'. Deep learning sentence embedding models are now seen to be state-of-the-art as they are able to extract features automatically and more effectively than manual efforts, when supported with large quantities of data [26].

Maybe add this as an image illustration

2.1.1 BERT-family Transformers

Pre-trained language models (PLMs), such as BERT [13], have been widely successful in a wide range of NLP tasks through fine-tuning [14, 31]. Fine-tuning is the process of re-training a PLM on specialised tasks, leveraging the model's base knowledge, by applying perturbations to the pre-trained model parameters through gradient descent learning algorithms. These language models, trained on finding an embedding space for natural language as well as stochastically generating tokens to mimic natural language, often provide a great platform to perform task-specific model fine-tuning. Platforms such as HuggingFace [57], allow authors to upload these pre-trained model parameters which in-turn allows researchers to download them for them fine-tuning. Moreover, task-specific fine-tuned models parameters are uploaded, downloaded and shared on these platforms. These fine-tuned models are useful for researchers whose focus lies outside of optimising these model parameters. In this section, we talk about the BERT-family of language models.

BERT

The Bidirectional Encoder Representations from Transformer (BERT) language model introduced in 2018 by Google AI Language team in (Devlin et al. [13]) was designed to learn deep bidirectional representations from unlabelled text. It achieved this by learning the left and right context in every layer of the model. The model implementation can be split into two phases: (1) the pre-training phase; and (2) the fine-tuning phase. This allows the model architecture to remain common, with many down-stream NLP tasks benefiting from a single PLM. An example illustration can be seen in Figure 2.1, where a single model can be fine-tuned on many down-stream tasks by simply replacing the output layer. The model architecture is a multi-layer Transformer encoder based on the original paper (Vaswani et al. [54]) which is bidirectional by nature. As the transformer architecture is a very well researched architecture, well represented in the literature and slightly out-of-scope for this paper, we refer the reader to the original paper.

sprinkle some images in here.

The pre-training phase can be split into two further training tasks:

- Masked Language Model - During training, a portion of the input tokens are 'masked' and the model predicts these masked tokens based on the remaining context, using

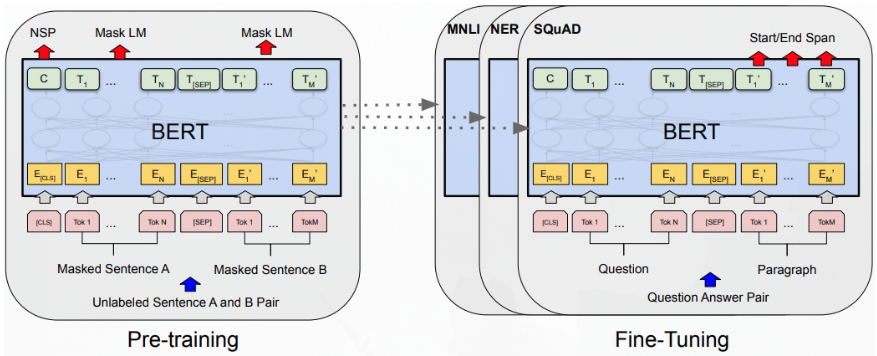

Figure 2.1: The overall pre-training and fine-tuning phases for BERT. For different downstream tasks, notice the same architecture, except output layer, is used. Source: (Devlin et al. [13]).

cross-entropy loss[60]. In the literature, this is referred to as a Cloze task [51]. In the experiments conducted by (Devlin et al. [13]) 15% of tokens in each sequence are randomly selected for masking. However, to mitigate the mismatch between pre-training and fine-tuning phases, a training data generator replaces each chosen masked tokens with: (1) the special token `[MASK]` 80% of the time; (2) a random token 10% of the time; and (3) the original token itself 10% of the time.

- Next Sentence Prediction - In this task, the model is trained on a binary classification task prediction the relationship between sentence pairs. Given two sentences `A` and `B` either (1) sentence `B` is the actual sentence that follows `A` 50% of the time or (2) sentence `B` is randomly selected from the corpus 50% of the time.

The Google AI Language team further fine-tune BERT across 11 NLP tasks, showing that BERT out-performed the state-of-the-art models at its point in time.

As previously mentioned, a benefit of using PLMs such as BERT for NLP tasks (such as text embedding) is the ability to off-load the computationally intensive pre-training and any initial fine-tuning stages to another party. Although, this means that there now exists an implicit trust on that third-party's model parameters. For example, the third-party could be motivated factors to censor certain phrases, artificially injecting testing data into training to increase performance metrics or maliciously change the embedding space. These are all factors which influence our decision on using a PLM and, thus, PLMs with transparently documented data sources, pre-training and fine-tuning should be preferred.

XLNet and MPNet

Since the release of the Google AI Language team's BERT, there have been some developments on further improving the performance of PLMs based on BERT's architecture. Namely, we touch on two: XLNet [58] and MPNet [44].

One downside of BERT, briefly touched on earlier, is the discrepancy between the pre-training and fine-tuning phases. Although BERT utilises bidirectional context for reconstruction of the special `[MASK]` token during pre-training, this special token does not exist at the fine-tuning phase. Furthermore, the predicted tokens are masked in the *input* thus BERT assumes the predicted tokens are independent to the surrounding context [58]. XLNet proposes

:

- Mathematics of BERT (detailed in XLNet paper)
- Mathematics of XLNet (detailed in XLNet paper)
- Autoregressive vs autoencoder
- Architectural changes in XLNet

Nomic

:

- Mention nomic is pitched as the first fully reproducable open-source, open-weights, open-data text embedding model
- Detail BERT archietcture adaptations
- Higher masking rate
- AdamW optimiser, learning rate with linear warmup.
- (weakly supervised) Pre-training: consistency filtering, curated long context text pairs, gradcache, mixed precision training.
- Prefixes on tasks: Symmetric category; asymmetric category.
- Supervised contrastive fine-tuning: datasets; learning rate. Randomly sampled mined negatives.
- talk about results, especially performance on long context

2.2 Dimensionality Reduction

:

- Talk why dimensionality reduction is needed: visualisation, curse of dimensionality, clustering? Figure out how to structure this well for the story.
- Talk about PCA and UMAP.

2.2.1 PCA

:

- Talk about PCA mathematics and why its used for dim reduction
- Talk about elbowing and finding the most optimal (?)

2.2.2 UMAP

:

- Talk about UMAP, the algorithm, the assumptions, the constraints
- Use the visualisation of the elephant here.

2.3 Clustering

:

- Talk about what clustering does, i.e. looking at trying to categorise or something
- Talk about the curse of dimensionality and why clustering does not perform well.
- Talk about k-Medoids (supervised), DBSCAN and HDBSCAN (unsupervised).

2.3.1 k-Medoids

:

- Talk about algorithm
- Talk about the parameters any effects when tuning these parameters.
- Talk about drawbacks and benefits of this approach, citations needed.

2.3.2 DBSCAN

:

- Talk about algorithm
- Talk about the parameters any effects when tuning these parameters.
- Talk about drawbacks and benefits of this approach, citations needed.

2.3.3 HDBSCAN

:

- Talk about algorithm
- Talk about the parameters any effects when tuning these parameters.
- Talk about drawbacks and benefits of this approach, citations needed.

2.4 Clustering Evaluation

:

Maybe this should be later on in the methodology?

- Motivate the need for clustering evaluation.
- Just highlight mathematics of each algorithm and mention the need for them.

2.4.1 Inertia

2.4.2 Silhouette

2.4.3 Davies-Bouldin Index

2.4.4 Calinski-Harabasz Index

2.5 Maybe Optuna

Chapter 3

Automatic Label Generation

: Describe the methods and procedures used. The things in this section will include.

- Explaining data format and data visualisation: wordcloud.
- Data cleaning steps, including removing key words such as Ion Source.
- Text preprocessing steps (cleaning) and computational challenges (tensorflow).
- Choosing the best sentence embedding transformer: MPNET, NOMIC.
- Data visualisation (before and after sentence embedding): similarity visualisation, explain unique sentences, token length distribution.
- Motivate why clustering in higher dimensions performs worse
- UMAP, PCA, t-SNE comparison. Motivate using UMAP.
- UMAP hyperparameter optimisation.
- Performing clustering with kmedoids, dbscan, hdbscan.
- Using optuna.
- Evaluation of results and choosing the best model (and arguing why hdbscan is the best by looking at the variance of dbscan and inflexibility of kmedoids)
- Touch on the production of a CLI application that allows you to mix and match various parts of the pipeline. Motivate the need for command line tool.

Chapter 4

Results and Discussion

: Describe the results and analyse the results

- Analyse the word cloud.
- Analyse the sentence embedding results.
- Analyse UMAP vs. PCA vs. t-SNE qualitatively and later quantitatively (compared to the clustering).
- Anaylse the UMAP hyperparameter optimisation qualitatively, mention that we use Optuna.

Chapter 5

Conclusion

:

- Definitely talk about clear and transparent PLMs and malicious stuff (see BERT section).

Bibliography

[1] PY Abijith, Piyush Patidar, Gaurav Nair, and Rohan Pandya. Large language models trained on equipment maintenance text. In *Abu Dhabi International Petroleum Exhibition and Conference*, page D021S065R003. SPE, 2023.

[2] Ramiz M Aliguliyev. A new sentence similarity measure and sentence based extractive technique for automatic text summarization. *Expert Systems with Applications*, 36(4): 7764–7772, 2009.

[3] C Ankenbrandt, C Curtis, C Hojvat, RP Johnson, C Owen, C Schmidt, L Teng, and RC Webber. H- charge exchange injection systems. In *11th International Conference on High-Energy Accelerators: Geneva, Switzerland, July 7–11, 1980*, pages 260–271. Springer, 1980.

[4] Asim Yaqoob. First line diagnosis at ISIS, 10 2017. URL https://indico.cern.ch/event/558933/contributions/2724364.

[5] Kensuke Baba, Tetsuya Nakatoh, and Toshiro Minami. Plagiarism detection using document similarity based on distributed representation. *Procedia computer science*, 111: 382–387, 2017.

[6] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146, 2017.

[7] Alexander Budanitsky and Graeme Hirst. Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and other lexical resources*, volume 2, pages 2–2, 2001.

[8] Tadeusz Caliński and Jerzy Harabasz. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27, 1974.

[9] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 160–172. Springer, 2013.

[10] Hongliu Cao. Recent advances in text embedding: A comprehensive review of top-performing methods on the mteb benchmark. *arXiv preprint arXiv:2406.01607*, 2024.

[11] Thyago P Carvalho, Fabrízzio AAMN Soares, Roberto Vita, Roberto da P Francisco, João P Basto, and Symone GS Alcalá. A systematic literature review of machine learning methods applied to predictive maintenance. *Computers & Industrial Engineering*, 137: 106024, 2019.

[12] David L Davies and Donald W Bouldin. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2):224–227, 1979.

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[14] Sergey Edunov, Alexei Baevski, and Michael Auli. Pre-trained language model representations for language generation. *arXiv preprint arXiv:1903.09722*, 2019.

[15] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[16] Maurizio Faccio, Alessandro Persona, Fabio Sgarbossa, and Giorgia Zanin. Industrial maintenance policy development: A quantitative framework. *International Journal of Production Economics*, 147:85–93, 2014.

[17] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

[18] ISIS Neutron and Muon Source. ISIS A Practical Guide to the ISIS neutron and Muon source, 4 2021. URL https://www.isis.stfc.ac.uk/Pages/News21_PracticalGuide.aspx.

[19] ISIS Neutron and Muon Source. ISIS beam operations, 4 2024. URL https://www.isis.stfc.ac.uk/Pages/beam-status.aspx.

[20] Ali Jezzini, Mohammad Ayache, Lina Elkhansa, Bassem Makki, and Maya Zein. Effects of predictive maintenance (pdm), proactive maintenace (pom) & preventive maintenance (pm) on minimizing the faults in medical instruments. In *2013 2nd International conference on advances in biomedical engineering*, pages 53–56. IEEE, 2013.

[21] Taeho Jo. K nearest neighbor for text summarization using feature similarity. In *2017 International Conference on Communication, Control, Computing and Electronics Engineering (ICCCCEE)*, pages 1–5. IEEE, 2017.

[22] Sushil Kumar and Komal Kumar Bhatia. Semantic similarity and text summarization based novelty detection. *SN Applied Sciences*, 2(3):332, 2020.

[23] Thomas K Landauer, Peter W Foltz, and Darrell Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284, 1998.

[24] Jinhyuk Lee, Zhuyun Dai, Xiaoqi Ren, Blair Chen, Daniel Cer, Jeremy R Cole, Kai Hui, Michael Boratko, Rajvi Kapadia, Wen Ding, et al. Gecko: Versatile text embeddings distilled from large language models. *arXiv preprint arXiv:2403.20327*, 2024.

[25] Hongfei Li, Dhaivat Parikh, Qing He, Buyue Qian, Zhiguo Li, Dongping Fang, and Arun Hampapur. Improving rail network velocity: A machine learning approach to predictive maintenance. *Transportation Research Part C: Emerging Technologies*, 45:17–26, 2014.

[26] Hong Liang, Xiao Sun, Yunlei Sun, and Yuan Gao. Text feature extraction based on deep learning: a review. *EURASIP journal on wireless communications and networking*, 2017:1–12, 2017.

[27] Romans Lukashenko, Vita Graudina, and Janis Grundspenkis. Computer-based plagiarism detection methods and tools: an overview. In *Proceedings of the 2007 international conference on Computer systems and technologies*, pages 1–6, 2007.

[28] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[29] HT Michael. Electronic circuits: fundamentals and applications, 2006.

[30] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[31] Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56(2):1–40, 2023.

[32] R Keith Mobley. *An introduction to predictive maintenance*. Elsevier, 2002.

[33] Giancarlo Nota, Alberto Postiglione, and Rosario Carvello. Text mining techniques for the management of predictive maintenance. *Procedia Computer Science*, 200:778–792, 2022.

[34] Zach Nussbaum, John X Morris, Brandon Duderstadt, and Andriy Mulyar. Nomic embed: Training a reproducible long context text embedder. *arXiv preprint arXiv:2402.01613*, 2024.

[35] Zhaotai Pan, Yi Ge, Yu Chen Zhou, Jing Chang Huang, Yu Ling Zheng, Ning Zhang, Xiao Xing Liang, Peng Gao, Guan Qun Zhang, Qingyan Wang, et al. Cognitive acoustic analytics service for internet of things. In *2017 IEEE International Conference on Cognitive Computing (ICCC)*, pages 96–103. IEEE, 2017.

[36] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11): 559–572, 1901.

[37] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[38] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[39] T Nora Raju, PA Rahana, Raichel Moncy, Sreedarsana Ajay, and Sindhya K Nambiar. Sentence similarity-a state of art approaches. In *2022 International Conference on Computing, Communication, Security and Intelligent Systems (IC3SIS)*, pages 1–6. IEEE, 2022.

[40] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[41] Mark Sammons, Christos Christodoulopoulos, Parisa Kordjamshidi, Daniel Khashabi, Vivek Srikumar, and Dan Roth. Edison: Feature extraction for nlp, simplified. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4085–4092, 2016.

[42] Justyna Sarzynska-Wawer, Aleksander Wawer, Aleksandra Pawlak, Julia Szymanowska, Izabela Stefaniak, Michal Jarkiewicz, and Lukasz Okruszek. Detecting formal thought disorder by deep contextualized word representations. *Psychiatry Research*, 304:114135, 2021.

[43] BP Sharma. Nuclear reactors: Moderator and reflector materials. *Encyclopedia of Materials: Science and Technology*, pages 6365–6369, 2001.

[44] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. Mpnet: Masked and permuted pre-training for language understanding. *Advances in neural information processing systems*, 33:16857–16867, 2020.

[45] Rudi JJ Stammler and Máximo Julio Abbate. Methods of steady-state reactor physics in nuclear design. 1983.

[46] Josef Steinberger and Karel Ježek. Text summarization and singular value decomposition. In *Advances in Information Systems: Third International Conference, ADVIS 2004, Izmir, Turkey, October 20-22, 2004. Proceedings 3*, pages 245–254. Springer, 2005.

[47] Chuan-Jun Su and Shi-Feng Huang. Real-time big data analytics for hard disk drive predictive maintenance. *Computers & Electrical Engineering*, 71:93–101, 2018.

[48] Gian Antonio Susto and Alessandro Beghi. Dealing with time-series data in predictive maintenance problems. In *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–4. IEEE, 2016.

[49] Gian Antonio Susto, Alessandro Beghi, and Cristina De Luca. A predictive maintenance system for epitaxy processes based on filtering and prediction techniques. *IEEE Transactions on Semiconductor Manufacturing*, 25(4):638–649, 2012.

[50] Ayisha Tabassum and Rajendra R Patil. A survey on text pre-processing & feature extraction techniques in natural language processing. *International Research Journal of Engineering and Technology (IRJET)*, 7(06):4864–4867, 2020.

[51] Wilson L Taylor. "cloze procedure": A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433, 1953.

[52] JWG Thomason. The isis spallation neutron and muon source—the first thirty-three years. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 917:61–67, 2019.

[53] Juan Pablo Usuga-Cadavid, Samir Lamouri, Bernard Grabot, and Arnaud Fortin. Using deep learning to value free-form text data for predictive maintenance. *International Journal of Production Research*, 60(14):4548–4575, 2022.

[54] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[55] Geert Waeyenbergh and Liliane Pintelon. A framework for maintenance concept development. *International journal of production economics*, 77(3):299–313, 2002.

[56] Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Multilingual e5 text embeddings: A technical report. *arXiv preprint arXiv:2402.05672*, 2024.

[57] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[58] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

[59] Minghe Yu, Guoliang Li, Dong Deng, and Jianhua Feng. String similarity search and join: a survey. *Frontiers of Computer Science*, 10:399–417, 2016.

[60] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.