NLP and ML Methods for Pre-processing, Clustering and Classification of Technical Logbook Datasets

Farhad Akhbardeh
fa3019@rit.edu

NLP and ML Methods for Pre-processing, Clustering and Classification of Technical Logbook Datasets

by

Farhad Akhbardeh

A dissertation submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
in Computing and Information Sciences

B. Thomas Golisano College of Computing and
Information Sciences

Rochester Institute of Technology
Rochester, New York
July 2022

NLP and ML Methods for Pre-processing, Clustering and Classification of Technical Logbook Datasets

by

Farhad Akhbardeh

Committee Approval:

We, the undersigned committee members, certify that we have advised and/or supervised the candidate on the work described in this dissertation. We further certify that we have reviewed the dissertation manuscript and approve it in partial fulfillment of the requirements of the degree of Doctor of Philosophy in Computing and Information Sciences.

Dr. Travis Desell — Date
Dissertation Advisor

Dr. Marcos Zampieri — Date
Dissertation Co-advisor

Dr. Christian Newman — Date
Dissertation Committee Member

Dr. Steven Bethard — Date
Dissertation Committee Member

Dr. Zhong Chen — Date
Dissertation Defense Chairperson

Certified by:

Dr. Pengcheng Shi — Date
Ph.D. Program Director, Computing and Information Sciences

NLP and ML Methods for Pre-processing, Clustering and Classification of Technical Logbook Datasets

by

Farhad Akhbardeh

Submitted to the
B. Thomas Golisano College of Computing and Information Sciences Ph.D. Program in Computing and Information Sciences
in partial fulfillment of the requirements for the
Doctor of Philosophy Degree
at the Rochester Institute of Technology

Abstract

Technical logbooks are a challenging and under-explored text type in automated event identification. These texts are typically short and written in non-standard yet technical language, posing challenges to off-the-shelf NLP pipelines. These datasets typically represent a domain (a technical field such as automotive) and an application (*e.g.*, maintenance). The granularity of issue types described in these datasets additionally leads to class imbalance, making it challenging for models to accurately predict which issue each logbook entry describes. In this research, we focus on the problem of technical issue pre-processing, clustering, and classification by considering logbook datasets from the automotive, aviation, and facility maintenance domains. We developed Maint-Net, a collaborative open source library including logbook datasets from various domains and a pre-processing pipeline to clean unstructured datasets. Additionally, we adapted a feedback loop strategy from computer vision for handling extreme class imbalance, which resamples the training data based on its error in the prediction process. We further investigated the benefits of using transfer learning from sources within the same domain (but different applications), from within the same application (but different domains), and from all available data to improve the performance of the classification models. Finally, we evaluated several data augmentation approaches including synonym replacement, random swap, and random deletion to address the issue of data scarcity in technical logbooks.

Acknowledgments

I would like to thank my Ph.D. committee for their help and support. Without their guidance, I could not be able to finish this Ph.D. degree. Special thanks to my advisor Dr. Travis Desell and my co-advisor Dr. Marcos Zampieri for their advise in every phase of this Ph.D. study.

I would like to give special thanks to my Ph.D. committee, Dr. Christian Newman and Dr. Steven Bethard for their time reviewing my dissertation and providing me feedback. Furthermore, special thanks to Dr. Cecilia Alm for her collaboration in our research papers and for providing feedback as well.

I further thank the University of North Dakota aviation program for the aviation maintenance records dataset, especially the National General Aviation Flight Information Database (NGAFID) project, and also would like to thank domain expert Zechariah Morgan for his help during the evaluation of various methodologies developed for aviation resources.

This dissertation work is dedicated to my beautiful family and friends, especially my parents, my brothers, and my sister, and for their great support and inspiration during my Ph.D. years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Estimating downtime and performing timely maintenance is a key step to reducing costs and improving operational efficiency in various branches of engineering. Performing proper and timely maintenance is not only necessary to extend the life of machinery systems but also would further help to increase safety as well. *Predictive maintenance* techniques are applied to engineering systems to estimate when maintenance should be performed to minimize costs and expand machinery system's performance [20, 44], as well as mitigate risk and increase safety. Furthermore, predictive maintenance is a particularly relevant application used to mitigate problems ahead of fixed maintenance schedules [51].

A fixed maintenance schedule is a form of maintenance that experts (*e.g.*, mechanics, or engineers) perform in a scheduled format. Performing a fixed maintenance schedule has some drawbacks and may result in developing risks [92]. For instance, an expert may introduce unnecessary maintenance costs to other components (*e.g.*, engine) during the scheduled maintenance for fixing another specific part. So, the main purpose of predictive maintenance is to perform maintenance only when needed, which means performing maintenance should only be done after models predict specific problems or issues [108].

Predictive maintenance systems use machine learning to estimate maintenance operations using many sources of information such as historical maintenance records in the form of event logbooks [20]. Maintenance records are an important source of information for predictive maintenance [78]. These records are often stored in the form of technical logbooks in which each entry contains fields that identify and describe a maintenance issue (or problems) or a solution taken to address a safety problem (*e.g.*, replacing a part) written in non-standard language [5]. They are collected in many domains such as aviation, transportation, and healthcare [9, 113]. Being able to classify these technical events is an important step in the development of predictive maintenance systems. Figure 1.1 provides an overview of the process of utilizing various historical data such as technical logbooks to develop a predictive maintenance system to predict and further alert maintenance problems and failures before they happen. This consists of processing historical data (such as logbooks) to extract important event information, training event prediction models to monitor the condition of the equipment to identify certain failures prior to happening, and then providing alerts so that timely maintenance can be performed.

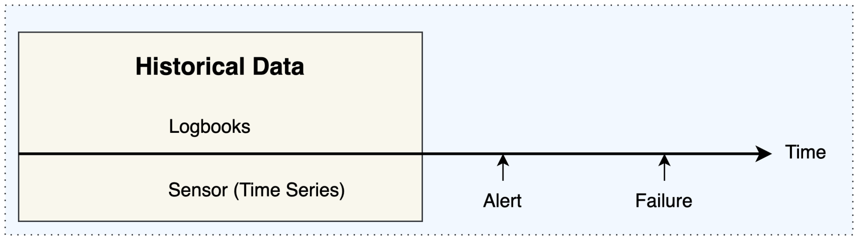


Figure 1.1: Process of utilizing historical data for performing predictive maintenance process.

In most technical logbooks, issues are manually labeled (assigned) by domain experts (*e.g.*, mechanics) making it possible to train systems to classify or cluster events by semantic similarity [78]. The issue type and descriptions include domain-specific technical language, abbreviations, and non-standard orthography, which off-the-shelf Natural Language Processing (NLP) models are unable to process as they are designed to work on standard data and are less effective on this data.

Furthermore, maintenance record data is typically proprietary and not made public. To the best of our knowledge, when we started the research that led to this Ph.D. thesis there was no aviation maintenance record dataset labeled with information such as issue and action codes or categories, requiring domain experts to manually annotate data for supervised document analysis and classification, an expensive and time-consuming process [1]. This has led to the development of domain-specific text pre-processing pipelines for logbook entries [6, 31].

1.1 Properties of Technical Logbooks

Classifying events in technical logbooks is a challenging problem for the NLP community for several reasons:

(a) The technical logbooks are written by various domain experts and contain short text entries with non-standard language including domain-specific abbreviated words (see Table 1.1 for examples), which makes them distinct from other short non-standard text corpora (*e.g.*, social media);

(b) Off-the-shelf NLP tools struggle to perform well on this type of data as they tend to be trained on standard contemporary corpora such as newspaper texts;

(c) Outside of the clinical and biomedical sciences, there is a lack of domain-specific, expert-based datasets for studying expert-based event classification, and in particular few resources are available for technical problem domains;

(d) Technical logbooks tend to be characterized by a large number of event classes that are highly imbalanced.

Original Entry	**Pre-processed Entry**
fwd eng baff seeal needs resecured	forward engine baffle seal needs resecured
both **engs**, all rocker **coveers** loose	both engines all rocker covers loose
r/h eng #3 intake **gsk** leaking	right engine number 3 intake gasket leaking
ck plow frame, bolts missing & **adj** brakes	check plow frame bolts missing adjust brakes
bird struck on **p/w** at **twy**. bird **rmvd**	bird struck on pilot window at taxiway. bird removed
diesel fuel leak under cab, **l/r** packer piston leaks	diesel fuel leak under cab lower right packer piston leaks
location **rptd** as **nm** from **rwy aprch** end	location reported as new mexico from runway approach end
cylinder #1 **baff** cracked at screw support	cylinder 1 baffle cracked at screw support

Table 1.1: Original and text-normalized example data instances illustrating that domain-specific terms (*baffle, engine*), abbreviations (*gsk - gasket, eng - engine, rwy - runway*), and misspellings (*seeal - seal*) are abundant in logbook data.

1.2 Research Goals

The research in this doctoral dissertation addresses these following research questions to overcome the discussed challenges:

RQ1: How well do state-of-the-art pre-processing techniques clean unstructured maintenance data? Can we develop better techniques to handle this type of data?

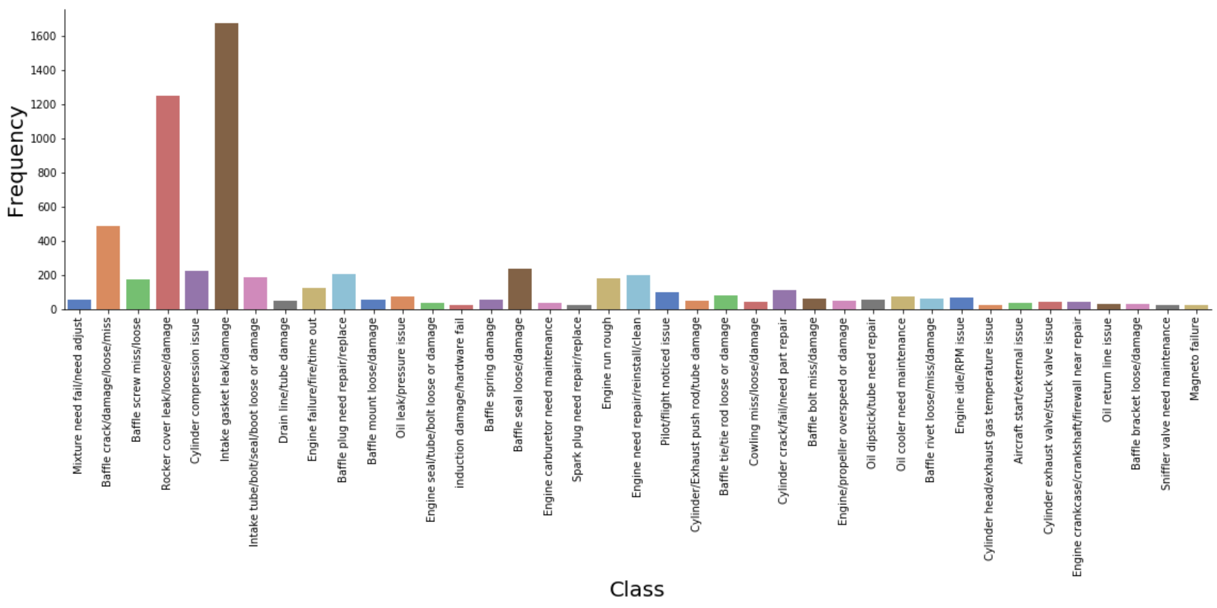


Figure 1.2: Number of instances in 39 unbalanced classes of the aviation maintenance (*Avi-Main*) dataset. (Published in [4])

RQ2: Using the proposed methods in this research work, which techniques are robust and the most effective in clustering/mining aviation maintenance textual data?

We addressed the aforementioned challenges (RQ1 and RQ2) with a special focus on clustering and annotating the logbook dataset using the developed domain-specific text pre-processing pipelines. We used various clustering algorithms to empirically compare the clustering outcome and evaluated them with the aid of a domain expert.

RQ3: To which extent does the class granularity and class imbalance present in technical logbooks impact technical event classification performance, and can a feedback loop for training data selection effectively address this issue?

We explored strategies to address the class imbalance in technical datasets. There is wide variation in the number of instances among the technical event classes examined in this research work, as shown in Figure 1.2 and Table 3.1. This extreme class imbalance is an obstacle when processing logbooks as it causes most learning algorithms to become biased, where they mainly predict the large classes [53]. To overcome this issue, we introduced a feedback loop strategy, which is a repurposing of a method used to address the extreme class imbalance in computer vision [19], and examined it for the classification of textual technical event descriptions. This technique is applied in the training of a suite of common classification models on seven predictive maintenance datasets representing the aviation, automotive, and facility maintenance domains [5].

RQ3.a: Which classification models are better suited to classify technical events for predictive maintenance across logbook datasets representing different technical domains?

We empirically compared the performance of different classification algorithms on seven logbook datasets from the aviation, automotive, and facilities domains.

RQ4: Which transfer learning approaches are better suited for classifying technical events for predictive maintenance across heterogeneous logbook datasets?

We explored transfer learning for domain adaptation in technical event classification to circumvent the limitation discussed regrading data scarcity which making it difficult to train robust predictive maintenance systems that require large amounts of data, for example, when using deep neural networks. Transfer learning techniques have been applied with great success to non-standard (*e.g.*, social media posts) text-based classification tasks such as sentiment analysis and offensive language identification [98, 114].

RQ4.a: How does the level of similarity between corpora impact the performance of transfer learning approaches for technical event classification?

We employed corpus similarity techniques to investigate the shared characteristics among these technical datasets, using multiple similarity methods.

RQ5: How well data augmentation techniques can impact classification models' performance while preserving the label information in logbook datasets?

We empirically examined the performance of various data augmentation methods on technical logbook data in the event classification task as well as evaluated the performance of utilized augmentation techniques using various metric-based methods.

1.3 Contributions

The main contributions of this doctoral thesis include:

1. The exploration of NLP methods for maintenance data (aviation maintenance in particular), which is an under-explored application:

1.a) The development of novel state-of-the art pre-processing techniques to clean unstructured maintenance data;

1.b) A comparison of clustering methods and similarity metrics for maintenance text mining;

1.c) The development of a collaborative open-source library for technical language resources (MaintNet) with a special focus on predictive maintenance;

2. Experimental results showing strong performance of the feedback loop in addressing the class imbalance problem in technical event classification across all datasets and models;
3. A thorough empirical evaluation of the performance of the technical event classifier considering multiple models and seven logbook datasets from three different domains;
4. A comprehensive study of domain adaptation under three conditions: (1) transfer within the domain, (2) transfer within the application, and (3) transfer over the global dataset;
5. The exploration of various data augmentation approaches for technical logbook dataset.

1.4 Publications

The following publications were produced as part of this doctoral research:

- Farhad Akhbardeh, Travis Desell, and Marcos Zampieri, "MaintNet: A Collaborative Open-Source Library for Predictive Maintenance Language Resources", In Proceedings of the 28th International Conference on Computational Linguistics (COLING), 2020.
- Farhad Akhbardeh, Travis Desell, and Marcos Zampieri, "NLP Tools for Predictive Maintenance Records in MaintNet", In Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing (AACL), 2020.
- Farhad Akhbardeh, Cecilia Alm, Marcos Zampieri, and Travis Desell "Handling Extreme Class Imbalance in Technical Logbook Datasets", In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP), 2021.
- Farhad Akhbardeh, Marcos Zampieri, Cecilia Alm, and Travis Desell "Transfer Learning Methods for Domain Adaptation in Technical Logbook Dataset", In Proceedings of the Language Resources and Evaluation Conference (LREC), 2022.

1.5 Overview of Dissertation Chapters

The remainder of this dissertation is organized as follows: Chapter 2 provides an overview on related work on technical logbooks, clustering, and classification in various NLP tasks. It further provides a summary of the research works in transformer language models, transfer learning, and data augmentation as well. Chapter 3 provides an overview of the datasets utilized in this research as well as the development of an open-source repository of language resources (MaintNet) for predictive maintenance and further discussion of the dataset challenges and pre-processing methods. Chapter 4 presents the details of the methods and results for text clustering (*e.g.*, DBSCAN, LDA) that are used to help in the process of grouping instances into categories and classification to handle the problem of class imbalance in technical data. Chapter 5 further investigated the benefits of using transfer learning methods (*e.g.*, transferring within a domain) for technical logbook datasets. Furthermore, Chapter 6 provides the details regarding the methods, experiments, and evaluations for various data augmentation techniques (*e.g.*, rule-based augmentation) that are examined in this thesis work. Finally, Chapter 7 provides a summary of the research performed in this dissertation and discusses the future direction of this research.

Chapter 2

Related Work

Chapter Introduction: This chapter reviews the literature regarding pre-processing and text normalization methods for domain-specific language. Furthermore, it includes various reviews of the existing methodologies to handle unstructured expert-based datasets (Section 2.1). Since most technical logbook datasets lack annotation, various relevant research works in text clustering methods and similarity metrics are investigated to help in the process of creating annotations. Furthermore, we study contemporary works on text classification and handling class imbalance problems (Section 2.2), followed by an overview of the domain adaptation, transformer-based language models and transfer learning approaches in various domains and datasets (Sections 2.3 and 2.4). Finally, recent works in data augmentation approaches are studied as well (Section 2.5).

2.1 Technical Logbook Datasets

Logbook data is gathered in many domains such as law enforcement, web information extraction, systems maintenance (*e.g.*, wind turbines [11], aviation [15], automobiles [96]), and electronic health to analyze critical factors in postoperative outcomes. However, the contents of logbooks are predominantly free text, and it is not straightforward to extract required information or which text normalization techniques are effective.

Text normalization techniques are used in various non-expert domain datasets such as social media to standardize unstructured data to process important events. Ritter *et al.* [101] proposed a novel open-source event extraction and supervised tagger for noisy microblogs (Twitter data). They used the Stanford Named Entity Recognizer (NER) and a Part of Speech Tagger (POS) to label word sequences and developed topic modeling techniques called *linkLDA* to extract events from the data. To identify events in the dataset, they used annotated data for training a tagger.

Cherry and Gue [26] applied word embedding-based modeling for information extraction on newswire and tweets, comparing named entity taggers and similarity techniques to improve their word embedding method. A major finding was that the cosine similarity metric was not sufficient for short sentences.

As dealing with non-standard expert textual data is also a difficult task in the medical domain, Deléger *et al.* [31] applied information extraction techniques on clinical reports to extract medication information. They used semantic extraction rule-based techniques to identify drug names and further drug-related information (*e.g.*, drug dosage) that was prescribed to patients. They analyzed reports using a sentence segmentation technique to divide them into meaningful parts, identifying the important topics per segment. Based on their findings, proper pre-processing techniques such as lexicon filtering improved performance.

Tixier *et al.* [117] developed a system to analyze injury reports by applying POS tagging and term frequency to extract keywords regarding injuries creating a dictionary of events to improve future safety management. Savova *et al.* [106] utilized (off-the-shelf) natural language processing toolkit (NLTK) libraries on free-text electronic medical records for information extraction purposes. They designed an architecture for clinical text analysis knowledge extraction (*cTAKES*), specific to the clinical domain, as it creates semantic annotations from patient records.

In the aviation domain, Tanguy *et al.* [113] studied various available NLP methods such as topic modeling to process aviation incident reports and extract useful information. They used standard NLP libraries to pre-process the data and then applied the *Talisman* NLP toolkit for incident feature extraction and training.

As to the problem of non-standard spelling, Siklosi *et al.* [109] proposed a method of correcting misspelled words in clinical records by mapping spelling errors to a large database of correction candidates. However, due to a large number of abbreviations in medical records, they were limited to specific terms and the normalization had to be performed separately.

Amorim *et al.* [30] proposed a dictionary-based spell correction algorithm utilizing a clustering approach by comparing various distance metrics to lower the number of distance calculations while finding or matching target words for misspellings. With this in mind, we developed a tool to deal with domain-specific misspellings and abbreviations (described in Section 3.2).

A major challenge in obtaining coherent document clusters from unlabeled data is determining a metric to quantify their accuracy, which involves calculating the difference between documents. Santhisree *et al.* [105] applied Cosine and Euclidean distance to measure the intra- and inter-cluster similarity generated from web usage data. Their goal was to achieve high intra-cluster and low inter-cluster similarity and they indicated that these similarity measurements helped to identify promising results. Amigó *et al.* [10] also analyzed various evaluation metrics to compare the quality of clusters generated with different text clustering techniques, and they suggested purity as the most popular evaluation metric.

Research Work	**Dataset(s)**	**Method(s) and Model(s)**
Ritter *et al.* [101]	Microblogs	linkLDA, POS, Stanford Named Entity Recognizer
Cherry and Gue [26]	News-wire and Tweets	Named Entity Tagger, Similarity
Deléger *et al.* [31]	Clinical Reports	Semantic Extraction Rule-based, Sentence Segmentation
Tixier *et al.* [117]	Injury Reports	POS Tagging, Term Frequency
Savova *et al.* [106]	Medical Records	Natural Language Processing Toolkit, *cTAKES*
Tanguy *et al.* [113]	Aviation Incident Reports	Topic Modeling, *Talisman* NLP Toolkit
Siklosi *et al.* [109]	Clinical Records	Misspelled Word Correction, Normalization
Amorim *et al.* [30]	Birkbeck Corpus	Dictionary-based Spell Correction
Santhisree *et al.* [105]	Web Usage	Cosine and Euclidean Distance

Table 2.1: Overview of related studies and developed method for various NLP techncial datasets.

As discussed and also outlined in Table 2.1, various studies developed techniques to process the standard or non-expert dataset. However, investigating NLP methods for pre-processing technical logbooks is less explored. Our developed pre-processing pipeline (described in Section 3.2) in this work addresses this important limitation.

2.2 Event Text Classification

Most expert-domain datasets containing events have focused on healthcare. For instance, Altuncu *et al.* [9] analyzed patient incidents in unstructured electronic health records provided by the U.K. National Health Service. They evaluated a deep artificial neural network model on the expert-annotated textual dataset of a safety incident to identify similar events that occurred.

Patrick *et al.* [91] proposed the cascade method of extracting the medication records such as treatment duration or reason, obtained from patient's historical records. Their approach for event extraction includes text normalization, tokenization, and context identification. A system using multiple features outperformed a baseline method using a bag of words model.

Yetisgen *et al.* [128] proposed a lung disease phenotypes identification method to prevent the use of a hand-operated identification strategy. They employed NLP pipelines including text pre-processing and further text classification on the textual reports to identify the patients with a positive diagnosis for the disease. Based on the outcome, they achieved notable performance by using n-gram features with a Maximum Entropy (MaxEnt) classifier.

There is also relevant research on event classification in social media. Hammer *et al.* [46] performed experimental work on Instagram text using weakly supervised text classification. Their approaches used word embeddings trained on an unstructured Instagram dataset to extract clothing brands based on the description that the users provided on the post. They used $200k$ Instagram posts with a total of 9 million comments, which were unlabeled and contained many special characters and informal words. To generate training data, they developed an unsupervised information extraction technique called "SemCluster" to pre-process this corpus, remove unnecessary data, and extract fashion attributes. They experimented with different similarity metrics such as Levenshtein to increase the accuracy of label extraction. In their approach, they used images that users posted on Instagram to validate the annotation process and then train a supervised generative model for text classification.

The problem of class imbalance has been studied in recent years for numerous NLP tasks. Madabushi *et al.* [115] studied automatic propaganda event detection from a news dataset using a pre-trained BERT model. They recognized that the BERT model had issues in generalizing. To overcome this issue, they proposed a cost-weighting method. Al-Azani *et al.* [8] analyzed polarity measurement in imbalanced tweet datasets utilizing features learned with word embeddings. Li *et al.* [65] studied the class imbalance problem in the task of discourse relation identification by comparing the accuracy of multiple classifiers. They showed that utilizing a unified method and further downsampling the negative instances can significantly enhance the performance of the prediction model on unbalanced binary and multi-class datasets.

Dealing with unbalanced classes is also well studied in sentiment classification. Li *et al.* [66] introduced an active learning method that overcomes the problem of class unbalance by choosing a significant sample of the minority class for manual annotation and majority class for automatic annotation to lower the amount of human annotation required. Furthermore, Damaschk *et al.* [28] examined techniques to overcome the problem of dealing with high-class imbalance in classifying a collection of song lyrics. They employed neural network models including a multi-layer perceptron and a Doc2Vec model in their experiments where their finding was that undersampling the majority class can be a reasonable approach to remove data sparsity and further improve the classification performance.

Li *et al.* [67] also explored the problem of high data imbalance using cross-entropy criteria as well as standard performance metrics. They proposed a loss function called Dice Loss that assigns equal importance to false negatives and false positives. In computer vision, Bowley *et al.* [19] developed an automated feedback loop method to identify and classify wildlife species from Unmanned Aerial Systems imagery, for training CNNs to overcome the unbalanced class issue. On their expert imagery dataset, the error rate decreased substantially from 0.88 to 0.05. So as shown in following Table 2.2, there are various methods developed to improve the classification models' performance as well as overcome the problem of class imbalance. This research work adapts the feedback loop strategy to the NLP problem of classifying technical events.

Research Work	**Dataset(s)**	**Method(s) and Model(s)**
Altuncu *et al.* [9]	Electronic Health Records	Deep Artificial Neural Network
Patrick *et al.* [91]	Patient's Historical Records	Cascade, Text Normalization, Context Identification
Yetisgen *et al.* [128]	Medical Reports	Phenotypes Identification, Maximum Entropy Classifier
Hammer *et al.* [46]	Instagram Text	Weakly Supervised Classification, Levenshtein, Generative Model
Madabushi *et al.* [115]	News	Automatic Event Detection, Pre-trained BERT, Cost-Weighting
Al-Azani *et al.* [8]	Tweet	Polarity Measurement, Word Embeddings
Li *et al.* [65, 66, 67]	Penn Discourse Treebank	Discourse Relation Identification, Unified, Downsampling, Dice Loss
Damaschk *et al.* [28]	Song Lyrics	Multi-layer Perceptron, Doc2Vec, Undersampling
Bowley *et al.* [19]	Wildlife Species	Automated Feedback Loop

Table 2.2: Overview of related studies of event classification and developed techniques and models.

2.3 Domain-specific Transfer Learning

2.3.1 Transfer Learning for NLP

Transfer learning strategies have been applied in various NLP tasks such as sentiment analysis to address key problems such as a deficit of labeled data. Studies have shown that training a model on one task or dataset (the *source*) and using transfer learning methods to transfer the knowledge to another task or dataset (the *target*) can improve performance compared to the model trained only on the target task with less data [88].

Tao *et al.* [114] proposed a transfer learning approach to overcome limited annotated data for aspect-based sentiment analysis tasks. Their analysis includes applying sentiment datasets from different domains to evaluate performance on sentence-level multi-label classification using the XLNet [127] and BERT [33] models, improving over baseline methods.

Terechshenko *et al.* [116] took advantage of transfer learning in political data analysis to overcome a limited dataset. They employed the XLNet [127] model to transfer learned knowledge to political science texts. Their experiment identified improvement on using a small source of a labeled dataset for transfer learning.

Zoph *et al.* [132] proposed utilizing the transfer learning method in the neural machine translation (NMT) task to improve the BLEU [89] score on multiple low-resource language datasets. Their approach consists of training the model on a large-scale French–English language source dataset and then transferring the learned parameters to the low-resource dataset including Uzbek–English for further training of the model. Their approach showed the overall performance improved by 5.6 points on BLEU score for the NMT task on 4 various low-resource pairs of datasets.

Transfer learning has also been used in healthcare to address the bottleneck of large labeled datasets and enable generalization capability. Romanov *et al.* [102] proposed a transfer learning technique to utilize the open-source Stanford Natural Language Inference dataset and medical terminologies in expert-annotated clinical data. They experimented with sequential inference using an LSTM in multiple layers. Their approach improved over previously reported methods on Natural Language Inference benchmarks.

Sun *et al.* [111] developed a BERT-based benchmark to evaluate PharmaCoNER biomedical dataset (target domain) using the transfer learning method. They examined two models of Multilingual-BERT and BioBERT. Their experiment showed a performance improvement in both models indicating that transferring learned knowledge from a large-scale trained domain to the target domain provides a useful approach for improving models' performance.

Dirkson *et al.* [34] proposed a transfer learning method to Twitter data associated with health to classify drug effects. They utilized a recurrent neural network architecture by Flair [3] having 512 hidden layers, yielding performance improvements over a baseline support vector classifier (SVC). Their finding was further that it can be beneficial to apply various domain-specific domain adaptation strategies.

2.3.2 Domain Adaptation

Employing domain adaptation to transfer learned knowledge of a source domain and improve performance in a target domain has also shown success in NLP problems. Axelrod *et al.* [13] investigated domain adaptation in statistical machine translation by utilizing instances from a general domain translation corpus of English and Chinese. They employed Moore and Lewis's [84] domain-specific models. The approach was successful with just a small subset of in-domain data.

Heilman *et al.* [47] utilized Daumé III's [29] domain adaptation in automatic short answer scoring and combined n-gram features and corpus similarity measures in the educational domain. Their results had high accuracy which approached human scores on the Beetle dataset that consists of student short answers to numerous questions.

Furthermore, several cross-domain adaptation approaches have been developed to leverage the knowledge learned in one domain to another. Peng and Dredze [94] studied transferring multi-task learning representations for sequence tagging using news and social media texts. They proposed a multi-task framework based on the BiLSTM model, which was capable of sharing the learner representation across tasks or domain datasets. The proposed framework achieved higher accuracy when applied to social media.

El Mekki *et al.* [35] proposed an unsupervised domain adaption approach utilizing pre-trained language models for cross-dialect sentiment analysis. They examined incorporating the Arabic dialects' fine- and coarse-grained taxonomies. In comparison to the zero-shot transfer using the BERT model, their approach showed roughly 20% performance improvement using the cross-dialect domain adaptation approach.

Bose *et al.* [18] proposed the unsupervised domain adaptation approach for cross-domain sentiment classification tasks by adapting the BERT variant model to the abusive language detection dataset. Their experiment utilizes the HateBERT model from Caselli *et al.* [21]'s work as well as further training with Reddit's large abusive language resource dataset using the Masked-Language Modeling (MLM) [33] approach. By analyzing the model outcome, they confirmed improvement in the model's performance on the target abusive language dataset in contrast to solely fine-tuning the HateBERT model on this dataset.

The many applications of transfer learning in NLP (as outlined in Table 2.3) confirm that transfer learning approaches are a promising strategy to overcome data scarcity. However, the use of

Research Work	**Dataset(s)**	**Method(s) and Model(s)**
Tao *et al.* [114]	Social Media	Aspect-based Sentiment Analysis, XLNet
Terechshenko *et al.* [116]	Political Science	XLNet
Zoph *et al.* [132]	French–, Uzbek–English	Neural Machine Translation, BLEU
Romanov *et al.* [102]	Natural Language Inference	Sequential Inference, LSTM
Sun *et al.* [111]	PharmaCoNER Biomedical	Multilingual-BERT, BioBERT
Dirkson *et al.* [34]	Twitter	Recurrent Neural Network Architecture (Flair)
Axelrod *et al.* [13]	English and Chinese	Statistical Machine Translation
Heilman *et al.* [47]	Beetle	Daumé III's Domain Adaptation
Peng and Dredze [94]	Social Media	Multi-task Learning, Sequence Tagging, BiLSTM
El Mekki *et al.* [35]	Arabic Dialects	Cross-dialect Sentiment Analysis
Bose *et al.* [18]	Reddit's Abusive Language	Cross-domain Sentiment Classification, HateBERT [21]

Table 2.3: Overview of various research studies of transfer learning and domain adaptation methods for NLP using various dataset types.

transfer learning to predictive maintenance datasets has not yet been explored. Our work fills this important gap providing empirical evidence of the feasibility of these methods when applied to technical logbook data.

2.4 Transformer-based Language Models

Since the transformer architecture proposed by Vaswani *et al.* [121] has shown significant model performance on machine translation tasks, several pre-trained language models (as shown in Sections 2.3.1 and 2.3.2) have been developed which achieve state-of-the-art results on various NLP tasks such as machine translation, named entity recognition, or sentiment analysis.

Background The transformer architecture (based on employing the attention mechanism) is composed of encoder and decoder blocks, as shown in Figure 2.1. The encoder element processes the input instance to transform it to the abstract representation and then passes it to the decoder element to generate an output instance. Further, the transformer architecture can be utilized in three various methods of encoder-decoder, encoder only, or decoder only, depending on the downstream task.

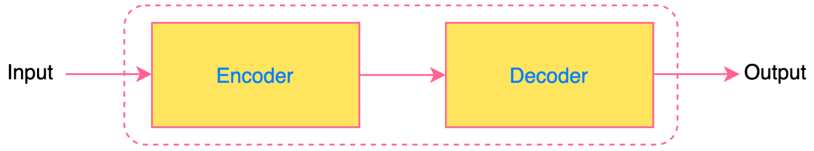


Figure 2.1: Original transformer-based encoder-decoder architecture [121].

The encoder block of the transformer as shown in Figure 2.2, is comprised of multi-head attention, residual and normalization, and a position-wise feed-forward network. Using the encoder-only model is employed in various state-of-the-art architectures such as BERT [33], where the given output of the encoder block is used as an input representation to various models for downstream tasks such as sentiment analysis, classification, or named entity recognition.

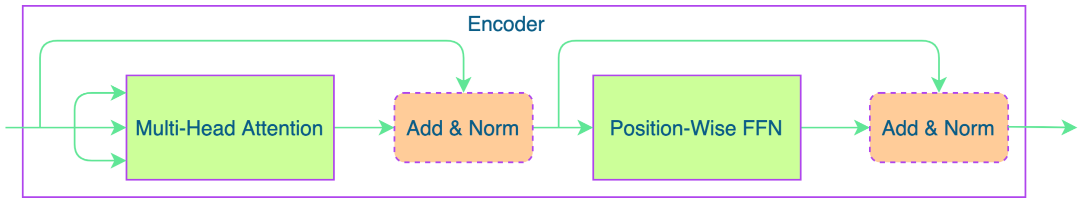


Figure 2.2: Overview of transformer architecture's encoder block [121].

The advantage of using the encoder element of the transformer architecture is to extract the contextual representation of the input instances where the various attention head and layer sizes also can be used. As also discussed in Section 2.3.2, various transformer-based models (*e.g.*, BERT) that use an encoder-only architecture achieved significant results while being applied to standard corpus datasets as these models are pre-trained on large standard datasets (*e.g.*, Wikipedia), and also further successfully adapted to other similar domains (*e.g.*, abusive language detection dataset [18]) and achieved better model performance for several NLP tasks [49] as well.

Also, pre-trained language models such as BERT [33] or XLNet [127] have shown notable performance improvement in other NLP tasks such as question answering, or next sentence prediction. Fine-tuning these language models on downstream tasks for instance sentiment analysis is a well-known method of improving model performance. However, these language models (*e.g.*, BERT) are initially pre-trained on large standard corpus (*e.g.*, English Wikipedia text), and fine-tuning them on expert or domain-specific datasets does not improve the performance significantly. Therefore, several adaptations of these language models [24, 48] have been developed by various researchers.

The variants of BERT also showed a comparable outcome by utilizing either less computational power or by having an optimized architecture. RoBERTa [70] has a modified training method that improves on the performance of the standard BERT model by increasing the number of iterations and data. DistilBERT [104] also achieved comparable performance to the BERT base model with 40% fewer parameters. ALBERT (architecture variant of BERT) [61] addressed the longer training time and memory limitations of BERT as well as enhanced training speed by utilizing factorized embedding parameterization and a cross-layer parameter sharing method.

ELECTRA [27] developed a new approach of pre-training instead of Masked-Language Modeling by replacing some tokens of instances with possible choices and showed comparable performance of the RoBERTa model with less computation time. A study of existing developed state-of-the-art transformer models also indicates how generalizable these pre-trained transformer models would be for technical language datasets and if their contextualized representation could improve the downstream task of event identification.

Chalkidis *et al.* [24] investigated various NLP approaches to adapt the BERT model to legal domain datasets and developed a variation of the BERT model called LEGAL-BERT. Their proposed approach relies on three distinct systematic analyses of utilizing BERT on various domain tasks. These methods include further pre-training BERT, pre-training BERT from the scratch, and using an off-the-shelf pre-trained BERT for a domain-specific legal dataset to examine performance. Their investigation showed an accuracy improvement by pre-training the BERT model.

Huang *et al.* [48] proposed ClinicalBERT where they adapted the original BERT to domain-specific clinical text by pre-training the model and further fine-tuning it to their clinical prediction task. Their proposed model showed performance improvements in multiple prediction tasks on clinical datasets such as intensive care unit notes and discharge reports, also evaluated by clinical domain metrics.

Pre-trained language models also showed remarkable performance on challenging downstream tasks such as online hate speech detection. Isaksen *et al.* [49] examines the impacts of transferring knowledge from the BERT model on identifying hateful, offensive, and common language. They fine-tuned the pre-trained BERT model with hateful and offensive language datasets such as those from Twitter. They also investigated the effect of further training the pre-trained BERT model with unlabeled domain-specific data. Their experiments showed an insignificant overall performance improvement in the natural language understanding task. Their key finding was that the standard BERT model pre-trained on standard English Wikipedia text where the language is formal does not relate well to the terms and words used in the hate speech dataset.

Further, there are various studies to improve transformer-based models. In a document-level context, Zhang *et al.* [129] extended a transformer model with a new context encoder approach. They employed a two-stage training approach that considered abundant sentence-level parallel text. They employed the NIST Chinese English dataset for the training process. Their approach showed significant model performance improvement over the baseline pre-trained transformer models.

There are various lighter transformer-based architectures that use lower computation resources than an original transformer architecture. Mehta *et al.* [79] developed a lightweight transformer-based architecture called DeLighT which utilizes fewer parameters but a deeper network compared to the standard transformer architecture developed by Vaswani *et al.* [121]. This architecture is capable of properly allocating the parameters within and also across the transformer block. The authors examined this model with various benchmarks (such as machine translation) and results show performance improvement compared to the standard transformer model by Vaswani *et al.* [121].

Research Work	**Dataset(s)**	**Method(s) and Model(s)**
Vaswani *et al.* [121]	WMT14 English-to-German	Transformer, Machine Translation, Encoder, Decoder
BERT [33]	GLUE Benchmark	Bidirectional, Encoder-only
Chalkidis *et al.* [24]	Legal Domain	LEGAL-BERT
Huang *et al.* [48]	Clinical Text (ICU)	ClinicalBERT
Isaksen *et al.* [49]	Twitter Hateful and Offensive	Pre-trained BERT, Natural Language Understanding
Zhang *et al.* [129]	NIST Chinese English	Two-stage Training, Pre-trained Transformers
Mehta *et al.* [79]	WMT14, WMT16	DeLighT Transformer

Table 2.4: Overview of multiple research that developed various transformer-based language models.

Many studies have shown notable performance of transformer-based models on various NLP tasks such as machine translation, named entity recognition, or sentiment analysis. The majority of these state-of-the-art models as summarized in Table 2.4, are pre-trained on standard language datasets (*e.g.*, Wikipedia) or adapted to other domains such as legal domain datasets (LEGAL-BERT). However, pre-training transformer-based language models for logbooks has not yet been explored.

2.5 Domain-specific Data Augmentation

As a common approach to dealing with limited labeled data, various methods have been studied over the years such as data augmentation to improve the performance of machine learning models. Data augmentation is a well-studied approach in computer vision and recently has been applied in various NLP tasks as well.

Kobayashi [56] proposed a contextual data augmentation approach using various given words by a bidirectional LSTM language model where the model was initially pre-trained on WikiText from a Wikipedia article. The approach is carried out by stochastically replacing the words in the instances with different words that are predicted by the language model. The author utilized instances from the Stanford Sentiment Treebank in their study and experimented on various text classification tasks. The proposed approach showed performance improvements for convolutional and recurrent neural network classifiers.

Wei *et al.* [125] developed the *easy* data augmentation technique for improving the performance of a text classification model using 4 operations consisting of random swap, random deletion, synonym substitution, and random addition. They examined this approach using recurrent neural networks on benchmark datasets such as Stanford Sentiment Treebank as well as customer reviews. This showed performance improvement on text classification tasks.

Furthermore, utilizing language models (*e.g.*, pre-trained models) for data augmentation has shown a significant performance increase in various NLP tasks. Kumar *et al.* [58] proposed a unified data augmentation approach using multiple pre-trained transformer models including BERT [33], GPT-2 [97], and BART [64]. Their approach consists of using the two techniques of prepending and expanding as a common way to condition a pre-trained language model on the dataset's class label information. They examined the approach on three various NLP tasks of question, intent, and sentiment classification utilizing a low-resource dataset. Their experiments showed an improved performance over pre-trained BART (Seq2Seq) compared to other data augmentation approaches. The key limitation of their approach was that the proposed method only works for a dataset that the model was already pre-trained on, where it may not work for other dataset types.

Zhou *et al.* [131] also proposed a data augmentation approach for cross-lingual Named Entity Recognition task based on Masked-Entity Language Modeling instead of using the standard Masked Language Modeling (MLM) [33] to efficiently preserve the label information. This approach is performed by masking the entity of training instances and then fine-tuning the standard Masked Language Model to produce new instances with swapped entities. They examined the proposed method on CoNLL dataset [118] and their results showed a significant performance improvement.

Based on the studied related work on data augmentation, various methods (as outlined in Table 2.5) have been developed to further improve machine learning model performance on various NLP tasks. However, per the discussion provided in Chapter 1 regarding the nature of the technical logbook data, data augmentation using pre-trained transformer models (which are most often trained on general (or standard) corpora (*e.g.*, Wikipedia)) may not be useful. The limitation of current

Research Work	**Dataset(s)**	**Method(s) and Model(s)**
Kobayashi [56]	SST	Contextual Data Augmentation, Bidirectional LSTM
Wei *et al.* [125]	SST, Social Media	*Easy* Data Augmentation
Kumar *et al.* [58]	SST-2, SNIPS, TREC	Unified Data Augmentation, BART, GPT-2, BERT
Zhou *et al.* [131]	CoNLL	Cross-lingual NER, Masked-Entity Language Modeling

Table 2.5: Overview of recent studies of developed data augmentation methods and strategies for various NLP tasks.

studies motivated us as a potential research to combine state-of-the-art knowledge and techniques and utilize optimal approach for augmenting the logbook data.

2.6 Chapter Summary

Technical logbooks dataset are an important source of information collected in various domains, however, processing these datasets is a challenging task. To investigate the methods and models that are proposed for processing various forms of domain-specific technical datasets, we studied the research works that aimed to develop techniques for handling technical datasets in domains such as healthcare, aviation, and social media. We presented recent methodologies developed for pre-processing and classification of domain-specific data. These studies show that investigating NLP methods for pre-processing technical logbooks is under-explored. Furthermore, we presented various studies on transfer learning approaches that were proposed for dealing with limited data in various domains, and we identified that the use of transfer learning to predictive maintenance datasets has not yet been explored. Additionally, we analyzed multiple research papers on pre-training transformer-based language models that previously showed notable performance on various NLP tasks. Finally, based on the current limitation of examining data augmentation techniques on technical logbooks, we investigated various suitable data augmentation approaches related to this work as well.

Chapter 3

MaintNet: Datasets and Tools for Predictive Maintenance

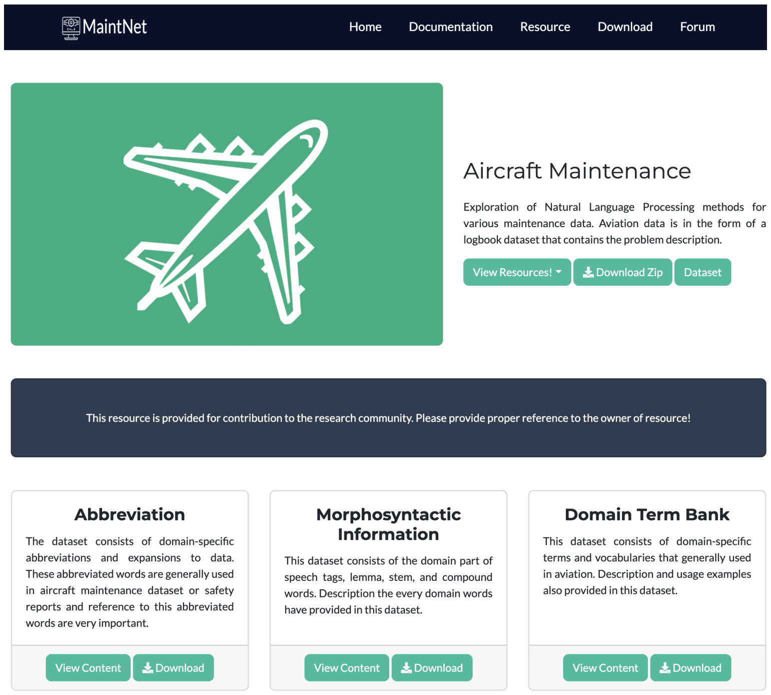
Chapter Introduction: This chapter introduces the technical datasets and tools utilized in this research work. As technical logbooks are an important source of information for predictive maintenance, Section 3.1 provides detail on technical logbooks and the developed open-source library for technical language resources (MaintNet). Sections 3.1.2 and 3.1.3 also provide discussion regarding technical logbook datasets' main features, challenges, and key characteristics. Finally, in Section 3.2, the details of handling unstructured technical logbook datasets using a developed pre-processing pipeline are provided as well.

3.1 Technical Event Datasets

As discussed in Chapter 1, maintenance record data is typically proprietary and to the best of our knowledge, until the start of this research project, there were no freely available tools and libraries developed to process technical logbook data. This challenge motivated us to provide an open-source repository to help encourage further study in this area, where we developed MaintNet[1], a collaborative, open-source library for technical language resources with a special focus on predictive maintenance data [5]. MaintNet is an exception. It is a unique open repository of predictive maintenance datasets from multiple domains, such as automotive, aviation and facility. As evidenced in

[1]`https://people.rit.edu/fa3019/MaintNet/`

Table 3.1, the datasets in MaintNet are, nonetheless, small – ranging from a few hundred instances for automotive maintenance to nearly 75,000 instances for facilities maintenance.



Aircraft Maintenance

Exploration of Natural Language Processing methods for various maintenance data. Aviation data is in the form of a logbook dataset that contains the problem description.

This resource is provided for contribution to the research community. Please provide proper reference to the owner of resource!

Abbreviation

The dataset consists of domain-specific abbreviations and expansions to data. These abbreviated words are generally used in aircraft maintenance dataset or safety reports and reference to this abbreviated words are very important.

Morphosyntactic Information

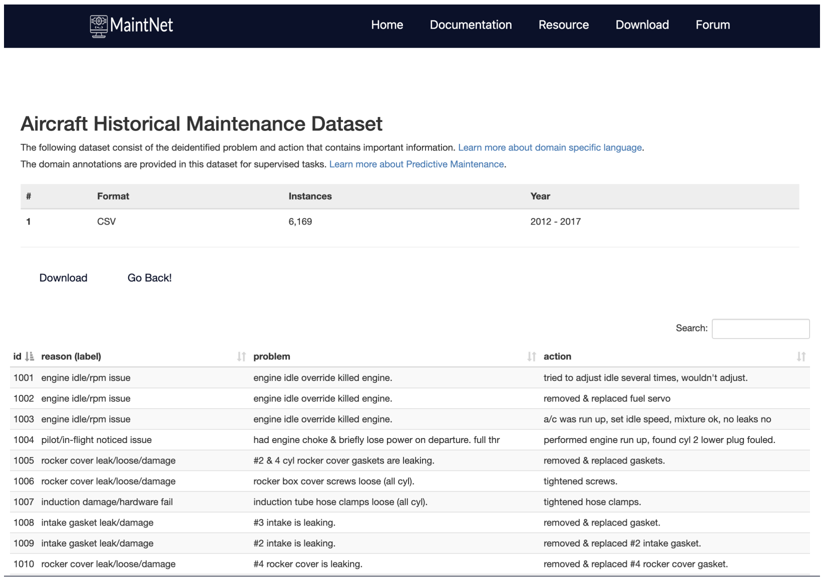
This dataset consists of the domain part of speech tags, lemma, stem, and compound words. Description the every domain words have provided in this dataset.

Domain Term Bank

This dataset consists of domain-specific terms and vocabularies that generally used in aviation. Description and usage examples also provided in this dataset.

Figure 3.1: Overview of MaintNet language resources that include technical data, domain-specific abbreviation dictionaries, morphosyntactic annotation, and term banks.

MaintNet Features: Predictive maintenance datasets are hard to obtain due to the sensitive information they contain. Therefore, we worked closely with the data providers to ensure that any confidential and sensitive information in the datasets remained anonymous. In addition to the datasets as shown in Figure 3.1, MaintNet further provides its users with domain-specific abbreviation dictionaries, morphosyntactic annotation, and term banks. The abbreviation dictionaries contain abbreviations validated by domain experts. The morphosyntactic annotations contain a part of speech (POS) tag, compound, lemma, and word stems. Finally, the domain term banks contain a collected list of terms that are used in each domain along with a sample of usage in the corpus. MaintNet also provides a webpage tool for users to communicate with each other and the project developers; as well as providing resources to share with the community (shown in Figure 3.3). We hope this will help further facilitate discussion and research contribution in this important and under-explored area of research. Furthermore, MaintNet provides users with corpora and a search feature (shown in Figure 3.2) that helps to identify domain terms or information through the platform.



Aircraft Historical Maintenance Dataset

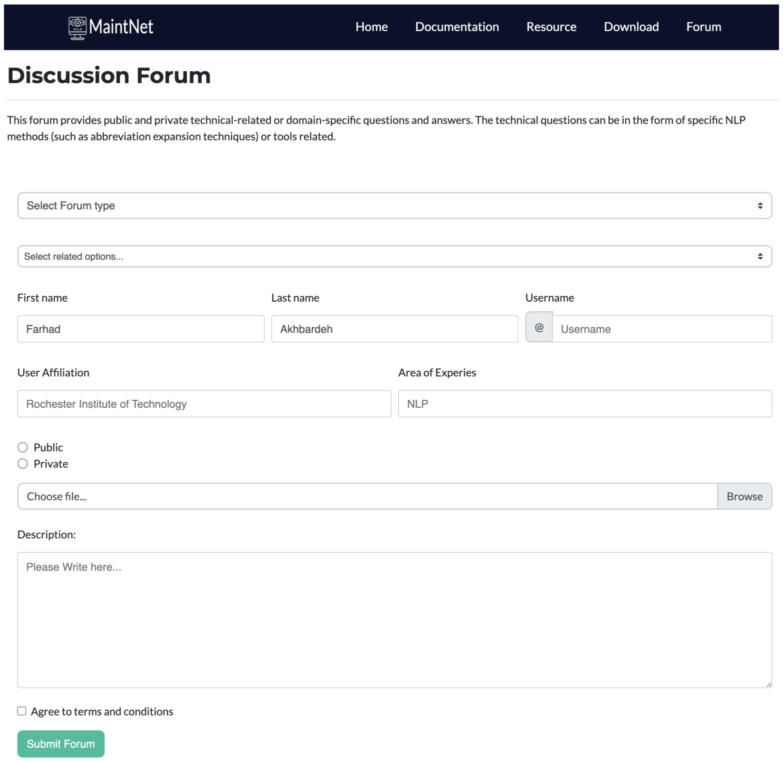
The following dataset consist of the deidentified problem and action that contains important information. Learn more about domain specific language.

The domain annotations are provided in this dataset for supervised tasks. Learn more about Predictive Maintenance.

#	Format	Instances	Year
1	CSV	6,169	2012 - 2017

id	reason (label)	problem	action
1001	engine idle/rpm issue	engine idle override killed engine.	tried to adjust idle several times, wouldn't adjust.
1002	engine idle/rpm issue	engine idle override killed engine.	removed & replaced fuel servo
1003	engine idle/rpm issue	engine idle override killed engine.	a/c was run up, set idle speed, mixture ok, no leaks no
1004	pilot/in-flight noticed issue	had engine choke & briefly lose power on departure. full thr	performed engine run up, found cyl 2 lower plug fouled.
1005	rocker cover leak/loose/damage	#2 & 4 cyl rocker cover gaskets are leaking.	removed & replaced gaskets.
1006	rocker cover leak/loose/damage	rocker box cover screws loose (all cyl).	tightened screws.
1007	induction damage/hardware fail	induction tube hose clamps loose (all cyl).	tightened hose clamps.
1008	intake gasket leak/damage	#3 intake is leaking.	removed & replaced gasket.
1009	intake gasket leak/damage	#2 intake is leaking.	removed & replaced #2 intake gasket.
1010	rocker cover leak/loose/damage	#4 rocker cover is leaking.	removed & replaced #4 rocker cover gasket.

Figure 3.2: Overview of MaintNet technical language dataset.



Discussion Forum

This forum provides public and private technical-related or domain-specific questions and answers. The technical questions can be in the form of specific NLP methods (such as abbreviation expansion techniques) or tools related.

First name	Last name	Username
Farhad	Akhbardeh	@ Username

User Affiliation	Area of Experies
Rochester Institute of Technology	NLP

Description:

Please Write here...

Figure 3.3: Overview of MaintNet discussion forum that helps the research community to collaborate regarding technical language resources.

In this work, we used a set of 7 logbook datasets (as shown in Table 3.1) from the aviation, automotive, and facility domains (available at MaintNet [5]). These datasets were: 1) Aviation maintenance (*Avi-Main)* which contains seven years of maintenance logbook reports describing problems and actions collected by the University of North Dakota aviation program on aircraft maintenance that were reported by the mechanic or pilot; 2) Aviation accident (*Avi-Acc)* which contains four years of aviation accident and reported damages; 3) Aviation safety (*Avi-Safe)* which contains eleven years of aviation safety and incident reports; Accidents were caused by foreign objects/birds during the flights which led to safety inspection and maintenance, where safety crews indicated the damage (safety) level for further analysis; 4) Automotive maintenance (*Auto-Main)* is a single year report with maintenance records for cars; 5) Automotive accident (*Auto-Acc)* which contains twelve years of car accidents and crash reports describing the related car maintenance issue and property damaged in the accident; 6) Automotive safety (*Auto-Safe)* which contains four years of noted hazards and incidents on the roadway from the driver; and 7) Facility maintenance (*Faci-Main)* which contains six years of logbook reports collected for building maintenance.

Dataset	**Instance**	**N. of Cls**	**Class Size Distribution**			
			Minimum	**Median**	**Average**	**Maximum**
Avi-Main	6,169	39	21	56	158	1,674
Avi-Acc	4,130	5	179	966	826	1,595
Avi-Safe	17,718	2	2,134	8,859	8,859	15,584
Auto-Main	617	5	23	48	123	268
Auto-Acc	52,707	3	1,085	11,060	17,569	40,562
Auto-Safe	4,824	17	86	213	284	678
Faci-Main	74,360	70	25	303	1,062	10,748

Table 3.1: The overview of number of classes (N. of Cls), and class size statistics: minimum, average, median, and maximum for each dataset. (published in [5])

3.1.1 Dataset Description

These technical logbooks consist of short, compact, and descriptive domain-specific English texts (for instance ranging between 2 and 20 tokens on aviation maintenance) including abbreviations, terminologies and domain-specific words. An example instance from Table 3.3, *r/h fwd upper baff seal needs to be resecured*, shows how the instances for a specific issue class are comprised from specific vocabulary (less ambiguity), and therefore contain a high level of granularity (level of description for an event from multiple words) [85]. Also, the instances such as "*eng light on, remoev hyd lines, leak note*" in aviation maintenance data (*Avi-Main*) or "*ckd fire ext throughout bldg*" in facility maintenance (*Faci-Main*) contains domain-specific abbreviations (*eng, hyd, bldg*), or misspelling (*remoev*) that briefly forms the description regarding specific event type.

Dataset	**Token Size Distribution**			
	Minimum	**Median**	**Average**	**Maximum**
Avi-Main	2	11	13	19
Avi-Acc	4	12	14	80
Avi-Safe	1	14	19	56
Auto-Main	2	4	7	32
Auto-Acc	2	4	5	6
Auto-Safe	1	24	25	55
Faci-Main	1	17	30	459

Table 3.2: Token size distribution: minimum, average, median, and maximum for each dataset.

Further, technical logbook datasets in these different domains contain terms and abbreviations that are similar or identical but with different meanings when appearing in a different domain, and are non-standard to typical pre-processing pipeline packages. For example, in the instance "*while in fl, after performing a few high power man*" where *fl* refers to the *flight level* and *man* refers to *manual*, rather than to their typical expansion (*Florida*) or lexical sense (*male individual*). Further highlighting the non-standard lexicon of these datasets, Figure 3.4 and 3.5 provides the top 10 most frequent words in the domains of aviation, automotive and facility on maintenance, accident and safety datasets.

An instance in the logbook can be formed as a complete description of the technical event (such as a safety or maintenance inspection) like: *#2 & #4 cyl rocker cover gsk are leaking*, or it might contain an incomplete description by solely referring to the damaged part/section of machinery (*hyd cap chck eng light on*) using few domain words. In either form of the problem description, the given annotation (label) is at the issue type-level, *e.g.*, *baffle damage.* Table 3.3 shows multiple examples with associated instances.

Table 3.1 and Table 3.2 present statistics for each dataset, in terms of the number of instances, number of classes, and the minimum, average, median and maximum class and token size distribution to represent how imbalanced the datasets are.

Example Instance of Technical Logbook Entry	Tech. Issue Label	Abbr., Misspelling, Term.
(1) **AFT** ON TAXI, **WING STRUECK FUEL** TRUCK, CHANDLER, AZ	SUBSTANTIAL DAMAGE	AFT, WING, STRUECK, FUEL
(1) AIRCRAFT ON **ROLLOUT**, **GEAR** COLLAPSED, MURFREESBORO, **TN**	UNKNOWN	ROLLOUT, GEAR, TN
(1) **AIRCCRAFT** ON LANDING, **GEAR** COLLAPSED, ABILENE, **TX**	MINOR DAMAGE	AIRCCRAFT, GEAR, TX
(1) LANDING AIRCRAFT LOST **ALTITUDE** WHILE TURNING BASE TO FINAL	SUBSTANTIAL DAMAGE	ALTITUDE
(1) AIRCRAFT RIGHT **GEAR** CATCH FIRE ON **RWY** DALLAS **TX**	MINOR DAMAGE	GEAR, RWY, TX
(2) **R/H FWD** UPPER **BAFF** SEAL NEEDS TO BE RESECURED	BAFFLE DAMAGE	R/H, FWD, BAFL
(2) BOTH **ENGS**, ALL ROCKER **COVEERS** LOOSE	ROCKER COVER LEAK	ENGS, ROCKER, COVEERS
(2) WHEN **RPM** SET IN **IDLE** RANGE, **ENG** KICKS & SPUTTERS AS IF FU	ENGINE FAILURE	RPM,ENG
(2) SMALL CRACK NOTED ON RIGHT **AFT BAFFL** OF RIGHT **ENG** ON **IB SID**	BAFFLE CRACK	BAFFL, ENG, IB, SID
(2) LEFT **ENG**, LEFT **AFT** BAFFLE **ANGLE BRACKET** IS CRACKED	BAFFLE BRACKET	ENG, AFT, ANGLE, BRACKET
(2) LEFT **ENG I/B** BAFFLE INTERCONNECT **ROD** BROKEN	BAFFLE LOOSE	ENG, I/B, BAFFLE, ROD
(2) OVERSPEED **ENGINE** BY 20 **RPM** FOR NO MORE THAN 10 SEC	ENGINE OVERSPEED	ENGINE, RPM
(2) **CYLINDER** #1 **BAFF** CRACKED AT **SCREW** SUPPORT	BAFFLE DAMAGE	BAFF, CYLINDER, SCREW
(3) **PM** SERVICES CHECK **TIRES** FOR LEAKS CHECK **PLOW BATT**	PM SERVICE	PM,TIRES, PLOW, BATT
(3) **DIESEL** FUEL LEAK UNDER **CAB**, **L/R PACKER PISTON** LEAKS	DRIVER REPORTED	DIESEL, CAB, L/R, PACKER
(3) CHECK **OIL** OR TRANS LEAK **SPINNER** LIGHT ADJ **CONV** CHAIN PLOW	DRIVER REPORTED	OIL, SPINNER,CONV
(3) CHECK **L/R** OUTER TIRE, AND **GAS PADDLE**	PM SERVICE	L/R, GAS, PADDLE
(3) **PLOW** DONT WORK ROAD CALL **CK BATTERY**	BREAKDOWN	PLOW,CK, BATTERY
(4) FAILURE TO YIELD RIGHT, **OVE** CORRECTING OVER **STEERING**	DRIVING ISSUE	OVE, STEERING
(4) MOTORISTS REGULARLY ILLEGAL **U-TURNS** IN **R/HOUR**	STOP SIGN RUNNING	U-TURNS, R/HOUR
(4) PRETTY CONSISTENT **SPEEDING** ALL HOURS OF THE DAY	SPEEDING	SPEEDING
(4) **CARS** TRYING TO GET TO THIS INTERSECTION ON THE REGULAR	BLOCKING CROSSWALK	CARS
(4) EXCESSIVE **SPEEDING** ALONG ARKANSAS	SPEEDING	SPEEDING
(5) THE **A/C** UNIT IN THE KITCHEN ON 3TH FLOOR **DMG**/LEAK	BUILDING PM	A/C, DMG
(5) RESET **BOILER** #2 **TMER**, CHECKED **BLDG**. THROUGHOUT	PREVENTIVE MAINT	BOILER, BLDG
(5) CLEANED AROUND THE EXTERIOR OF THE **BLDG**	SERVICE	BLDG
(5) **REPL** LEAKING THREE SECTION HOT WATER **BOILER**	BOILER	REPL, BOILER
(5) **REPL DOOR** CLOSER ON HALLWAY LEADING INTO THE FORENSIC BAY	DOOR	REPL, DOOR
(6) DISREGARDED THE **SIGNAL** OR REGISTRAR **SIGN**	UNKNOWN	SIGNAL, SIGN
(6) **BRAKE** FAILURE OR DEFECTIVE	NON-INCAPACITATING	BRAKE
(6) IMPROPER **LANE** USAGE	UNKNOWN	LANE
(7) ABNORMALITES NOTE **FAN BLAD** BEND OUTWARD POST FLIGHT **INSP**	CAUSED DAMAGE	FAN, BLAD,INSP
(7) **ENG** PARAMETERS NORMAL, BUT NEEDS **INSP**	NO DAMAGE	ENG, INSP
(7) ANGLE OF ATTACK **INDICATOR** BROKEN **RT** SIDE OF **A/C**	CAUSED DAMAGE	INDICATOR, RT, A/C

Table 3.3: Example instances of technical logbook entries spanning the aviation accident (1), aviation maintenance (2), automotive maintenance (3), automotive safety (4), facility maintenance (5), automotive accident (6), and aviation safety (7) datasets. Each instance shows how domain-specific terminology (Term.), abbreviations (Abbr.), and misspelled words (in bold font) are used by the domain expert, and also illustrates some of the event types covered.

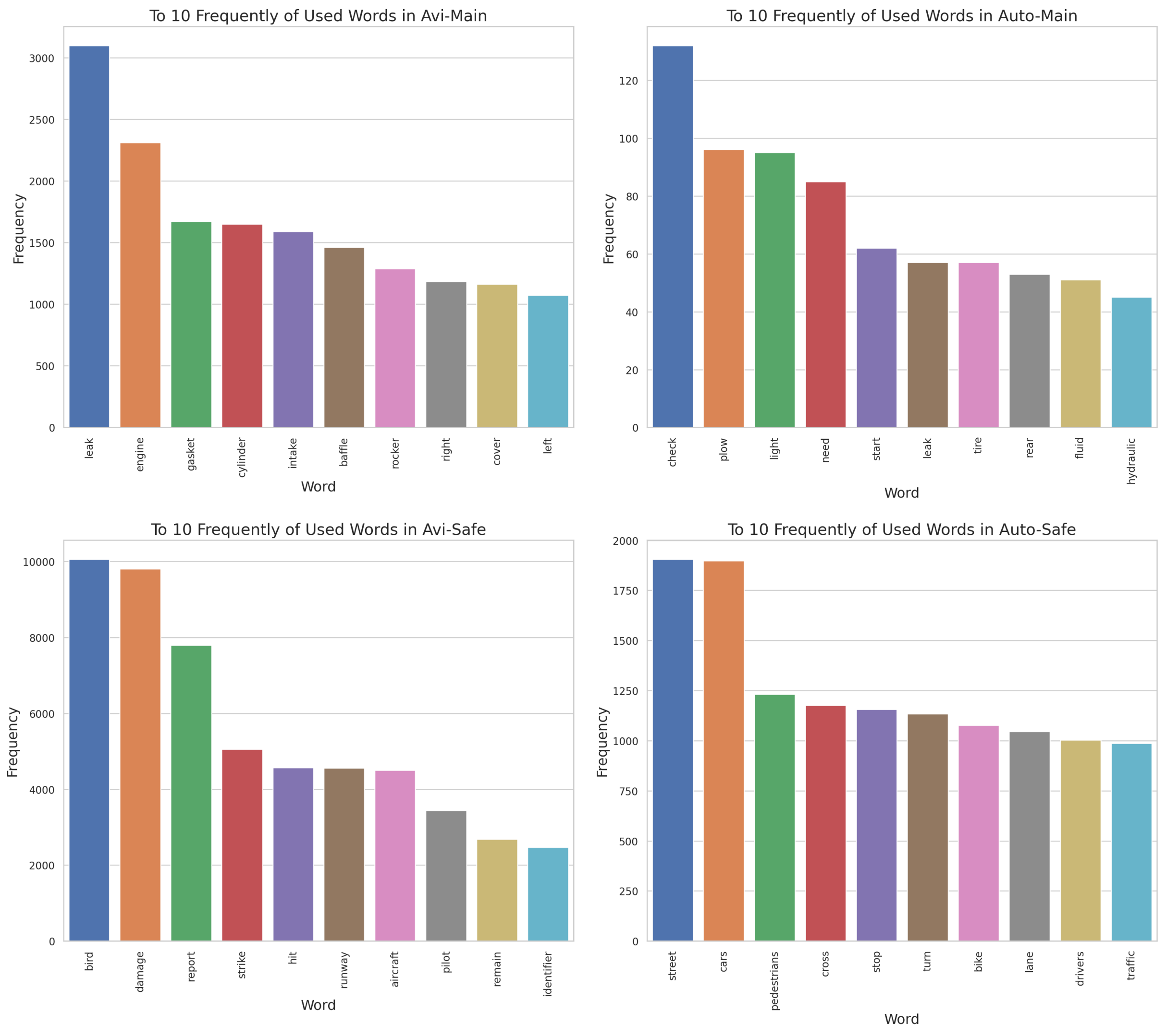


Figure 3.4: Top 10 frequency of words used in the aviation maintenance (Avi-Main), aviation safety (Avi-Safe), automotive maintenance (Auto-Main), and automotive safety (Auto-Safe) datasets representing the nature of such technical logbook data.

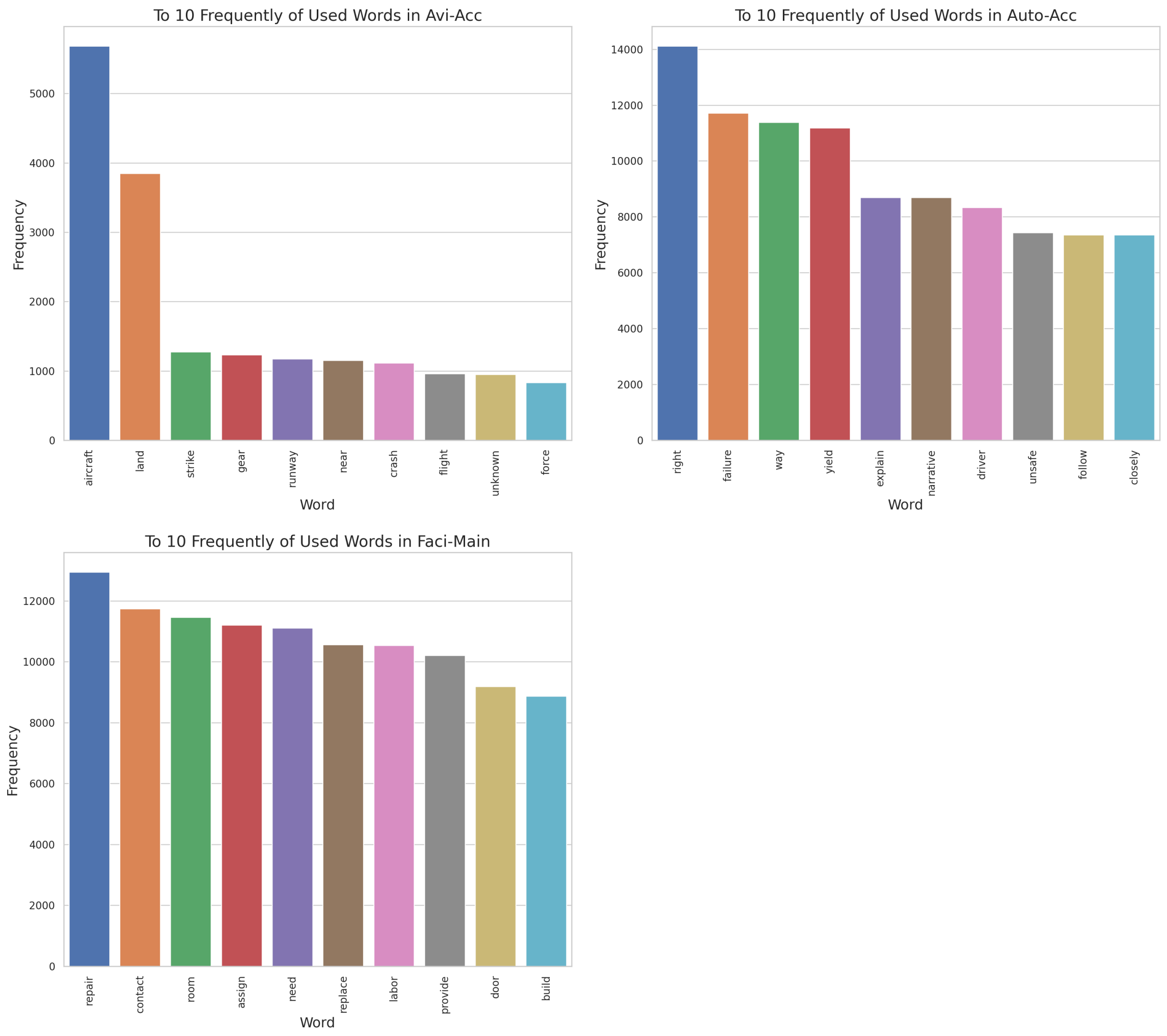


Figure 3.5: Top 10 frequency of words used in the aviation accident (Avi-Acc), automotive accident (Auto-Acc), and facility maintenance (Faci-Main) datasets representing the nature of such technical logbook data.

3.1.2 Dataset Characteristics

Further characteristics of these log entries include compound words (*antifreeze, engine-holder, drif-tangle, dashboard*). Many of these words (*e.g.*, a compound word: *dashboard*) essentially represent the items, or domain-specific parts used in the descriptions. Additionally, function words (*e.g.*, prepositions) are important and removing them could alter the meaning of the entry. The logbook datasets also have both the following shared and distinct characteristics:

Shared Characteristics: Each instance contains a descriptive observation of the issue and/or the suggested action that should be taken (*eng inspection panel missing screw*). Each instance also refers to maintaining a single event, which means the recognized problem applies to the only single-issue type. As an example, the instance *cyl #1 baff cracked at screw support & forward baff below #1* includes a combination of sequences that refers to the location and/or specific part of the machinery.

Distinct Characteristics: In each domain, terminologies, a list of terms, and abbreviations are distinct, and an abbreviation can have different expansion depending on the domain context [110], *e.g.*, *a/c* can mean *aircraft* in aviation and in the automotive domain *air conditioner*. However, the abbreviations and acronyms of the domain words (*e.g.*, *atc* - *air traffic control*) in these technical datasets should not be approached as a word sense disambiguation problem as they require character level expansion.

3.1.3 Technical Logbook Challenges

There are various key challenges related to technical logbook data that make it difficult for off-the-shelf NLP pipelines to handle, where most of these problems regarding the nature of the logbook data are discussed in Sections 3.1.1 and 3.1.2 in detail. In general language corpus (*e.g.*, news text, Wikipedia text), the instance usually follows standard formatting and structure that current NLP models (*e.g.*, pre-trained language models) can process properly. However, the written description in the technical logbook lacks such a standardized structure due to the different writing formats that domain experts use while describing the observed issue during an inspection.

Furthermore, the aforementioned non-standard format of technical logbooks poses various challenges to the model that we are developing where following is a list of the main challenges that need to be considered:

1. Utilizing various domain-specific abbreviations and acronyms that might be specific to each domain (*e.g.*, *AGL – Above Ground Level*) and dropping or substituting any character can alter the meaning (*e.g.*, *AGL* to *AL – Approach Lights* or *ALS – Approach Lighting System*);
2. Using uncommon syntax and parts of speech sequences (*e.g.*, *tires, lights testing showed multiple issues*) or contractions (*e.g.*, *needn't – need not*);
3. Using misspelling or dropped words in the description which can be inaccurately seen as abbreviations (*e.g.*, *fast* where dropped to: *fas – final approach segment*);
4. Using problem descriptions of varying length that can consist of a few tokens that describe the same issue (*e.g.*, *engine failed, engine not working properly*);

3.2 Technical Logbook Pre-processing

One of the bottlenecks of automatically processing logbooks for predictive maintenance systems is that most of these datasets are not annotated with the reason for maintenance or a categorization of the issue type. Furthermore, as discussed in Section 1.1, most standard NLP pipelines for pre-processing and annotation are trained on standard contemporary corpora (*e.g.*, newspaper texts, novels) struggle to address most domain-specific terminology, abbreviations, and non-standard spelling present in logbook datasets. To address this issue, we implemented several pre-processing steps to clean and extract as much information from logbooks as possible. The pipeline is shown in Figure 3.6.

The process starts with text normalization, including lowercasing, stop word and punctuation removal, and treating special characters with NLTK's [17] regular expression library, followed by tokenization (NLTK tokenizer), stemming (Snowball Stemmer), and lemmatization (WordNet [83]). With the use of collected morphosyntactic information, POS annotation is carried out with the NLTK POS tagger. Term frequency-inverse document frequency (TF-IDF) is obtained using the *gensim tfidf model* [99].

Our analysis of the logbooks found that many of the misspellings and abbreviations lead to incorrect or non-existent dictionary look ups. To overcome this issue, we explored various state-of-the-art

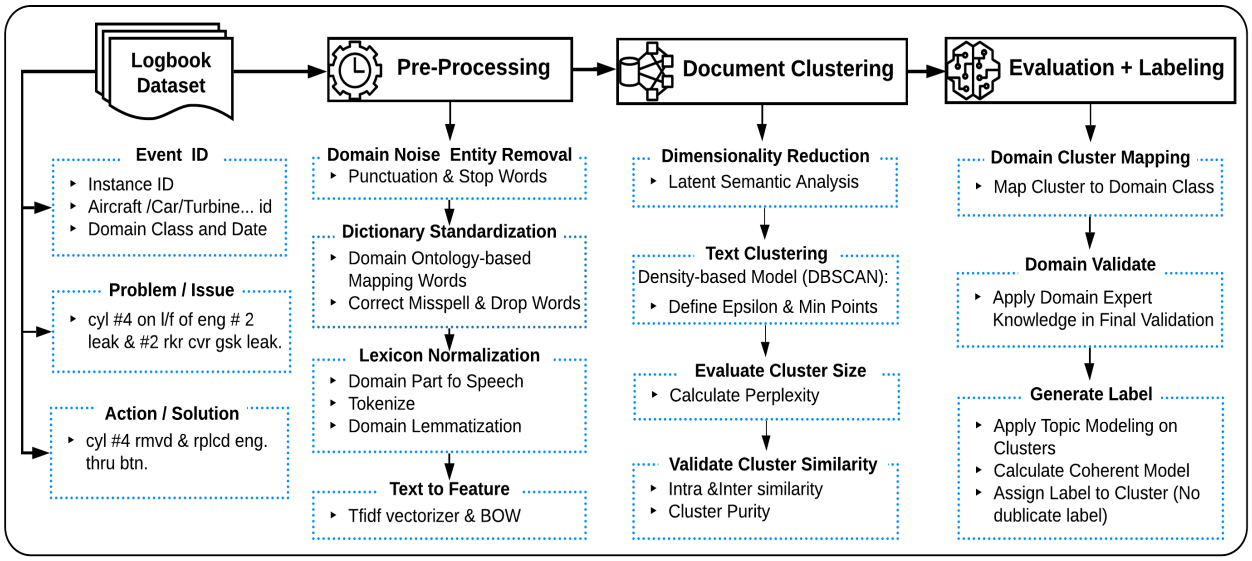


Figure 3.6: The components in MaintNet's processing and information extraction pipeline: pre-processing, document clustering, and evaluation. (published in [6])

spellcheckers including Enchant[2], Pyspellchecker[3], Symspellpy[4], and Autocorrect[5].

Given the inaccuracy of existing techniques, we developed methods of correcting syntactic errors, typos, and abbreviated words using a Levenshtein distance algorithm [63]. This method uses a dictionary of domain-specific words and maps the various possible misspelled words into the correct format by selecting the most similar word in the dictionary. The Levenshtein algorithm was chosen over other distance metrics (*e.g.*, Euclidian, Cosine) as it allows us to control the minimum number of string edits and it is widely used in spell checking [30].

WordNet was used to lemmatize the document, however, it requires defining a POS tagger parameter which we want to lemmatize (the WordNet default is "noun"). As the maintenance instances typically consist of verb, noun, adverb and adjective words that define a problem, action and occurrence, by using "verb" as the POS parameter, there is an issue of mapping important noun words such as "left" (*e.g.*, left engine) to "leave" or "ground" to "grind". To resolve this issue, we created an exception list using developed morphosyntactic information for the WordNet lemmatizer to ignore mapping words which could be multiple parts of speech.

[2]`https://www.abisource.com/projects/enchant/`
[3]`https://github.com/barrust/pyspellchecker`
[4]`https://github.com/wolfgarbe/SymSpell`
[5]`https://github.com/fsondej/autocorrect`

Finally, we have performed an extrinsic evaluation of the developed pre-processing pipeline by evaluating its impact on POS tagging. To carry out this evaluation, we randomly selected 500 instances of the aviation maintenance (*Avi-Main*) dataset to serve as our gold standard. A North-American English native speaker working for the project annotated the 500 instances using the Penn Treebank tagset. We made this gold standard available to the community in MaintNet [6]. We compared the performance of three available POS taggers: NLTK [17], Stanford CoreNLP [74] and TextBlob [7] trained on the raw and pre-processed versions of the *Avi-Main* dataset and evaluated on raw and pre-processed versions of the gold standard.

3.2.1 Results of Pre-processing

Table 3.4 presents the results of our proposed Levenshtein spellchecking method compared to other techniques in random samples of 500 instances from each of the 5 datasets. The results are reported in terms of success rate showing that the Levenshtein (Leven) algorithm outperforms the Enchant (Ench), Pyspellchecker (Spell), and Autocorrect (Auto) spell checkers.

Code	**Token**	**Miss**	**Ench**	**Spell**	**Auto**	**Leven**
Avi-Main	3299	289	86%	61%	73%	98%
Avi-Safe	6059	828	84%	56%	68%	91%
Auto-Main	2599	266	69%	27%	49%	95%
Auto-Acc	2422	169	87%	59%	77%	97%
Faci-Main	7758	926	83%	63%	59%	93%

Table 3.4: Success rate of spell checkers on 500 instances per dataset. Token stands for total tokens and Miss stands for misspelled tokens. (published in [6])

Table 3.5 presents the results of the performance comparison of the three available POS taggers in terms of accuracy. Stanford CoreNLP obtained the best results among the three POS taggers with 91% and 87% accuracy on the processed and raw versions of the data respectively. The results show an improvement of 4 percentage points in the performance of each of the three POS taggers when annotating MaintNet's pre-processed data confirming the importance of these pre-processing methods.

[6] `https://people.rit.edu/fa3019/MaintNet/datasets`

[7] `https://textblob.readthedocs.io/en/dev/`

POS Tagger	**Raw**	**Processed**	**Difference**
NLTK	77%	81%	+4%
Stanford	87%	91%	+4%
TextBlob	77%	81%	+4%

Table 3.5: Results of three POS taggers annotating raw and pre-processed versions of the gold standard. (published in [6])

3.3 Chapter Summary

As discussed in Chapter 1, technical logbooks are proprietary and very hard to obtain due to the unique nature and structure of these datasets. In this work, we developed an open-source repository of technical language resources (MaintNet) with a special focus on predictive maintenance data to address this challenge. MaintNet is an open repository that contains resources from various domains of aviation, automotive, and facility. MaintNet provides the user with several features such as domain-specific abbreviation dictionaries, morphosyntactic annotation, term banks, and a webpage tool for communicating with researchers about further discussion and research contribution. We also discussed the shared and distinct characteristics of these technical datasets. Based on the description provided in Section 3.1.3, technical logbooks contain non-standard languages and grammar that challenge off-the-shelf NLP pipelines to process. Therefore, we developed a new pre-processing pipeline to handle these unstructured technical datasets and extract as much information as possible from these datasets and utilize them in various NLP tasks (which answers RQ1). We further compared the performance of various POS taggers (NLTK, Stanford CoreNLP, and TextBlob) on the technical logbook dataset (Avi-Main) and the outcome indicated outperformance of Stanford CoreNLP compared to the other POS taggers.

Chapter 4

Technical Logbook Clustering and Classification

Chapter Introduction: This chapter provides details of methods and specific techniques used in this doctoral research work to extract important information from historical maintenance data and utilized in various methods. This includes implementing various text clustering techniques applied to technical logbooks to address the problem of lack of annotation in these datasets (Section 4.1). Furthermore, examining various strategies to address the extreme class imbalance in these technical datasets and comparing the performance of various event classification models that are discussed in Section 4.2.1 as well.

4.1 Technical Logbook Clustering

By employing the pre-processing pipeline discussed in Section 3.2, we cleaned the technical logbook datasets by converting non-standards to a standard format and then further implemented popular clustering algorithms and applied them to the datasets. The motivation behind this is that most logbook data available is not annotated, which requires a domain expert to group instances into categories and text clustering techniques were used to help in this process. Furthermore, it should be noted that evaluations of text clustering outcomes were performed as well to provide a comparison for which techniques could properly group these technical event descriptions in logbooks based on their similar semantic information. For instance, the desired outcome of clusters that could group problem/event descriptions such as "*cylinder cover leak*", "*cylinder intake leak*" and "*cylinder* *gasket leak*" together would be considered a suitable clustering technique's outcome.

We first converted the terms and words into a numerical representation using libraries such as *tfidfvectorizer* [93] resulting in a large matrix of document terms (DT). We used truncated singular value decomposition (SVD) [72] known as latent semantic analysis (LSA), to perform a linear dimensionality reduction. We chose truncated SVD (LSA) over principal component analysis (PCA) [36] in our system, due to the fact LSA can directly be applied to our *tfidf* DT matrix and it focuses on document and term relationships where PCA focuses on a term covariance matrix (eigendecomposition of the correlation).

We experimented with different 4 clustering techniques: k-means [50], Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [37], Latent Dirichlet Analysis (LDA) [122], and hierarchical clustering [2]. For comparison of the results, the silhouette and inertia metrics [40] were used to determine the number of clusters for k-means (both provided similar results), and perplexity [40] and coherence [122] scores were used for LDA. DBSCAN and hierarchical clustering do not require a predetermined number of clusters.

For evaluation, we used a standard measurement of cluster cohesion including high intra-cluster similarity and low inter-cluster similarity. We chose 3 different similarity algorithms including Levenshtein [57], Jaro-Winkler [124] , and Cosine [40] to calculate intra- and inter-cluster similarity. The cosine similarity metric is commonly used and is independent of the length of document, while Jaro-Winkler is more flexible by providing a rating of matching strings. We collected human annotated instances by a domain expert to serve as our gold standard, and these are provided on MaintNet to encourage research into improving unsupervised clustering of maintenance logbooks. We further utilized the purity metric for measuring the quality of the clusters by calculating the given number of accurately assigned documents divided by the total number of documents [75].

4.1.1 Results of Logbook Clustering

Figure 4.1 shows the empirical analysis of the four clustering techniques with and without our additional data pre-processing steps (Levenshtein-based dictionary spellchecking and the lemmatizer list previously discussed in Section 3.2) on the *Avi-Main* dataset. We examined the distribution of cluster sizes, the number of clusters, and the number of outliers (in the case of DBSCAN). Using a domain-based spellchecker and the modified lemmatizer list improved the purity and overall accuracy of the clusters by increasing the means of intra-cluster similarity and decreasing the means of inter-cluster similarity.

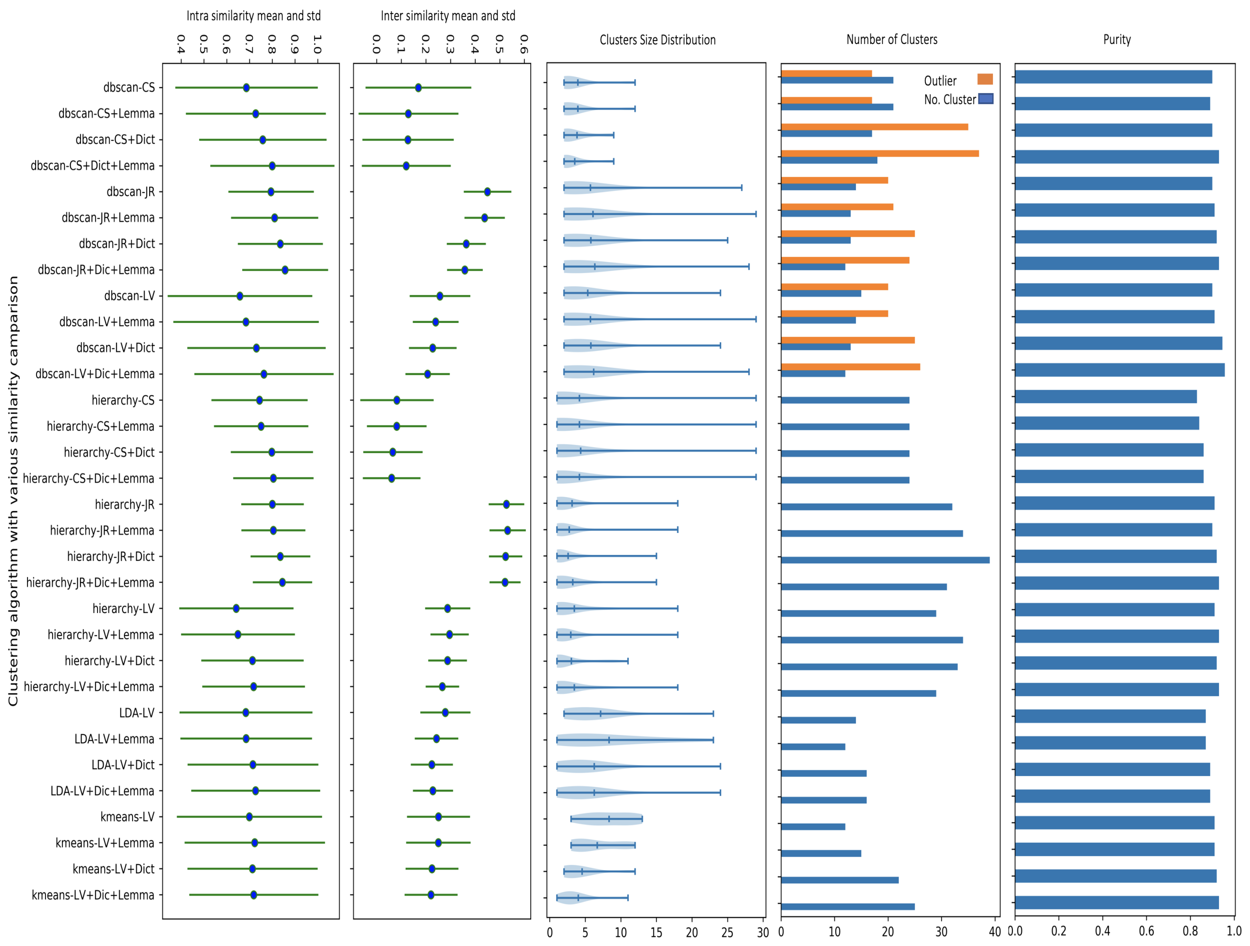


Figure 4.1: Results of the clustering methods. From left to right, calculated mean and standard deviation of intra- and inter-cluster similarity, cluster size distribution, number of clusters generated by each method and purity on *Avi-Main* dataset. (published in [6])

DBSCAN provided more accurate clusters in comparison to other algorithms while also detecting outliers, which could help identify if any new issues are introduced to the maintenance logs or if there are safety issues reported by the pilot during flight operation. K-means provided somewhat comparable results to DBSCAN, but it was not able to detect outliers and determining the number of clusters (K) is challenging, especially as this number may change over time as more issues are reported. Hierarchical clustering performed poorly, where similar issues were found to be distributed across different clusters. It was also more computationally expensive than the other methods. Clusters generated with LDA were better than hierarchical clustering, however LDA clustered some of the documents that contain the same equipment with different types of issues description together, resulting in clusters with a mixture of issue types.

4.2 Technical Event Classification

4.2.1 Handling Class Imbalance

Collecting additional data to augment datasets is a common approach for tackling the problem of skewed class distributions. However, as discussed earlier in Chapter 1, technical logbooks are proprietary and very hard to obtain. In addition, each domain captures domain-specific lexical semantics, preventing the use of techniques such as domain adaption [73] to apply a large class data from one technical domain to another. For example, instances that describe an *engine failure* in the aviation domain are distinct from *engine failure* instances reported in the automotive domain. In this research work, we applied five different methods for selecting training data for the models to analyze their effects on classification performance: (1) under(down)- and (2) over-sampling, (3) random down-sampling, (4) a feedback loop strategy, and (5) a baseline strategy which simply uses all available data.

Re-sampling Under- and over-sampling are re-sampling techniques [76] that were used to create balanced class sizes for model training. For over-sampling, instances of the minority classes are randomly copied so that all classes would have the same number of instances as the largest class. For under-sampling, observations are randomly removed from the majority classes, so that all classes have the same number of instances as the smallest class. For both approaches, we first divided our datasets into test and training sets before performing over-sampling to prevent contamination of the test set by having the same observations in both the training and test data.

Feedback Loop To address class imbalances in text classification, this work adapted the approach in Bowley *et al.* [19] from the computer vision domain. The goal of this approach is not only to alleviate the bias towards majority classes but also to adjust the training data instances such that the models are always being trained on the instances that was performing the worst on. It should be noted that this approach is very similar to *adaptive learning* strategies which have been shown to aid in human learning [52, 81].

Algorithm 1 presents pseudocode for the feedback loop. In this process, the active training data (the data used to actually train the models in each iteration of the loop) is continually resampled from the training data. The model is first initially trained with an undersampled number of random instances from each class, which becomes the initial active training data. The model $\mathcal{M}$ then

Algorithm 1 Feedback Loop Pseudocode

```
▷ Gets $\mathcal{MCS}$ random instances from each class
function SampleRandom($\mathcal{C}$, $\mathcal{MCS}$)
    Array $\mathcal{A}$
    for $i \leftarrow 1$ to size($\mathcal{C}$) do
        shuffle($\mathcal{C}_i$)
        $\mathcal{A} \leftarrow \mathcal{A} \cup$ getFirstN($\mathcal{MCS}$, $\mathcal{C}_i$)
    return $\mathcal{A}$

▷ Gets $\mathcal{MCS}$ instances from each class with the worst error
function Resample($\mathcal{C}$, $\mathcal{M}$, $\mathcal{MCS}$)
    Array $\mathcal{A}$
    for $i \leftarrow 1$ to size($\mathcal{C}$) do
        calculateError($\mathcal{C}_i$)
        sortByError($\mathcal{C}_i$)
        $\mathcal{A} \leftarrow \mathcal{A} \cup$ getFirstN($\mathcal{MCS}$, $\mathcal{C}_i$)
    return $\mathcal{A}$

Input: Training Data $\mathcal{D} = \text{Instance}(1, 2, \ldots, N)$
Input: Feedback Loop Iterations $\mathcal{FLI}$
Input: Epochs Per Loop Iteration $\mathcal{FLE}$
Input: Minimum Class Size $\mathcal{MCS}$

▷ Divide training data by class
Array $\mathcal{C} \leftarrow$ splitByClass($\mathcal{D}$)

▷ Get initial active training data $\mathcal{A}$ randomly
Array $\mathcal{A} \leftarrow$ SampleRandom($\mathcal{C}$, $\mathcal{MCS}$)
Model $\mathcal{M}$
for $l \leftarrow 1$ to $\mathcal{FLI}$ do
    ▷ Train the model for the number of epochs per iteration
    $\mathcal{M} \leftarrow$ train($\mathcal{M}$, $\mathcal{FLE}$, $\mathcal{A}$)
    ▷ Update the active training data
    $\mathcal{A} \leftarrow$ Resample($\mathcal{D}$, $\mathcal{M}$, $\mathcal{MCS}$)

Output: $\mathcal{M}$
```

performs inference over the entire training set, and then selects $\mathcal{MCS}$ instances from each class $\mathcal{C}_\text{i}$ which had the worst error during inference, where $\mathcal{MCS}$ is the minority (smallest) class size. The model is then retrained with this new active training data and the process of training, inference and selection of the $\mathcal{MCS}$ worst instances repeats for a fixed number of feedback loop iterations, $\mathcal{FLI}$. In this way the model is always being trained on the instances it has classified the worst.

To measure the effect of resampling the worst performing instances, the feedback loop approach was also compared to a random downsampling (DS) loop, where instead of evaluating the model over each instance and selecting the worst performing instances, $\mathcal{MCS}$ instances from each class are instead randomly sampled. As performing inference over the entire training set adds overhead, a comparison to the random DS loop method would show if performing this inference is worth the performance cost over simple random resampling. This approach is the same as Algorithm 1 except that `SampleRandom` is used instead of `Resample` in the feedback loop. Section 4.2.3 describes how the number of training epochs and loop iterations were determined such that all the training data selection methods are given a fair evaluation with the same amount of computational time.

Evaluation Metrics For imbalanced datasets, simply using precision, recall or F1 score metrics for the entire datasets would not accurately reflect how well a model or method performs, as they emphasize the majority classes. To overcome this, alternative evaluation metrics to handle the class imbalance problem were used, as recommended by [14]. Specifically, we report the models performance based on precision, recall, and F1 score by utilizing a macro-average over all classes, as this gives every class equal weight, and hence reveals how well the models and training data selection strategies perform.

4.2.2 Model Architecture and Training

Different machine learning methods were considered for technical event/issue classification (*e.g.*, engine failure, turbine failure). Each instance is an individual short logbook entry (approximately 12 words on average per instance including function words), as shown in Table 3.2. The methods used in this study were a deep neural network (DNN) [32], a Long Short-Term Memory (LSTM) [112], a recurrent neural network (RNN) [90], a convolutional neural network (CNN) [54], and BERT [33].

Deep Neural Network A deep artificial neural network (DNN), as described by Dernoncourt *et al.* [32], can learn abstract representation and features of the input instances that would help to achieve better performance on predicting the issue type in the logbook dataset. The DNN used was a 3 layer, fully connected feed forward neural network with an input embedding layer of dimension 300 and equal size number of words followed by 2 dense layers with 512 hidden units with ReLU activation functions followed by a dropout layer. Finally, we added a fully connected dense layer with size equal to the number of classes, with a SoftMax activation function.

Long Short-Term Memory An LSTM RNN was also used to perform a sequence-to-label classification. As described by Suzgun *et al.* [112] LSTM RNNs utilize several vector gates at each state to regulate the passing of data by the sequence which enhances the modeling of the long-term dependencies. We used a 3 layer LSTM model with a word embedding layer of dimension 300 and the equal size number of words followed by an LSTM layer with setting the number of hidden units equal to the embedding dimension, followed by a dropout layer. Finally, we added a fully connected layer with size equal to the number of classes, with a SoftMax activation function.

Convolutional Neural Network Convolutional neural networks (CNNs) have demonstrated exceptional success in NLP tasks such as document classification, language modeling, or machine translation [69,107]. As Xu *et al.* [126] described, CNN models can produce consistent performance when applied to the various text types such as short sequences. We evaluated a CNN architecture [54] with a convolutional layer, followed by batch normalization, ReLU, and a dropout layer, which was followed by a max-pooling layer. The model contained 300 convolutional filters with the size of 1 by n-gram length pooling with the size of 1 by the length of the input sequence, followed by concatenation layer, then finally connected to a fully connected dense layer, and an output layer equal to the size of the dataset class using a SoftMax activation function.

Bidirectional Encoder Representations We also evaluated using pre-trained uncased Bidirectional Encoder Representations (BERT) for English [33]. We fine-tuned the model, and used a word piece based BERT tokenizer for the tokenization process and the *RandomSampler* and *SequentialSampler* for training and testing respectively. To better optimize this model, a schedule was created for the learning rate that decayed linearly from the initial learning rate we set in the optimizer to 0.

4.2.3 Experimental Settings

Datasets and Baselines First, the technical text pre-processing pipeline [6] (discussed in Section 3.2) was applied, which comprises domain-specific noise entity removal, dictionary-based standardization, lexical normalization, part of speech tagging, and domain-specific lemmatization. We divided the datasets selecting randomly from each class independently to maintain a similar class size distribution, using 80% of the instances for training and 20% of the instances for testing data. For feature extraction, two methods were considered: a bag-of-word model (n-grams:1) [93] and pre-trained 300 dimensional GloVe word embeddings [95].

Hyperparameter and Tuning The coarse to fine learning (CFL) approach [62] was used to set parameters and hyperparameters for the DNN, LSTM, and CNN models. Experiments considered batch sizes of 32, 64, and 128, an initial learning rate ranging from 0.01 to 0.001 with a learning decay rate of 0.9, and dropout regularization in the range from 0.2 to 0.5 in all models, as well as ReLU and SoftMax activation functions [86], categorical cross-entropy [130] as the loss function, and the Adam optimizer [55] in the DNN, LSTM, CNN and BERT models. Based on experiments and network training accuracy, a batch size of 64 and drop out regularization of 0.3 was selected for model training.

Each model with each training data selection strategy was trained 20 times to generate results for each dataset. To ensure each training data selection strategy was fairly compared with a similar computational budget, the number of training epochs and loop iterations (if the strategy had a feedback or random downsampling loop) were adjusted so that the total number of training instances evaluations each model performed was the same. For each dataset, the number of forward and backward passes, 'T' for 100 epochs of the baseline strategy was used as the standard. As an example, Table 4.1 shows how many loop iterations, epochs per loop, and inference passes were done for each training data selection strategy on the *Auto-Safe* dataset. Given the differences between the min and max class sizes it was not possible to get exact matches but the strategies came as close as possible. We counted each inference pass for the feedback loop the same as a forward and backward training pass, which actually was a slight computational disadvantage for the feedback loop, as a forward and backward pass in training takes approximately 1x to 2x the time as an inference pass.

Dataset	L	EPL	LTI	INM	T
Baseline	1	100	3,859	0	385,900
Downsampling	1	329	1,173	0	385,917
Oversampling	1	42	9,214	0	386,988
Random DS Loop	33	10	1,173	0	387,090
Feedback Loop	25	10	1,173	3,859	389,725

Table 4.1: Details regarding different training process using the various methods for handling the unbalanced class in automotive safety (*Auto-Safe*) dataset with 17 total classes. Loop (L), Epochs Per Loop (EPL), Active Training instance Size (LTI), Inference for New Misclassified (INM) and Total Instances Evaluated (T). (Published in [4])

4.2.4 Results of Handling Class Imbalance

Table 4.2 shows a comparison between the baseline and the four different class balancing methods (over-sampling, under-sampling, the random down-sampling (DS) loop and the feedback loop). Based on these outcomes, the feedback loop strategy almost entirely outperforms the other methods over all datasets and models, showing that performing inference over the training set and reselecting the training data from the worst performing instances does provide a benefit to the learning process. A plausible explanation is that this strategy does not introduce bias into the larger class and also does not effect the minority class size distribution. It also does not waste training time on instances the model has already well learned.

Table 4.2 also shows the empirical analysis of the four classification models, with the model and training data selection strategy providing the overall best results shown in bold and italics. Using technical text pre-processing techniques described in Section 4.2.3, and the feedback loop strategy described in Section 4.2.1, the precision, recall, and F1 score improved compared to the baseline performance. The CNN model outperformed the other algorithms with improved precision, recall, and F1 score for almost all datasets except for *Avi-Main*, where BERT had the similar results, and *Auto-Main* where CNN and BERT tied. This is interesting, given the current popularity of the BERT model, however it may be due to the substantial lexical, topical, and structural linguistic differences between the technical logbook data and the English corpus that BERT was pre-trained on.

Furthermore, we conducted the Mann-Whitney U-test of statistical significance by using the F1 scores of each of the 20 repeated experiments of the classification models, using the baseline and the feedback loop approach as the two different populations. The outcomes are shown in Table 4.3, with the differences being highly statistically significant.

4.2.5 Discussion of Handling Class Imbalance

To investigate the optimal strategies for dealing with these imbalanced technical datasets, we studied various methods on how to process the data, extract features, and classify the type of event. Regarding the discussion provided in Chapter 3 about the nature of such a dataset, there are key challenges that effect the performance of employed algorithms. As discussed in Chapter 1, the extreme class imbalance observed in these technical datasets substantially affects learning algorithms' performance. To overcome this issue, we first explored oversampling and undersampling,

Dataset	Model	Baseline (%)			Down Sampling (%)			Over Sampling (%)			Random DS Loop (%)			Feedback Loop (%)		
		Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1
Avi-Main	DNN	0.90	0.89	0.89	0.67	0.78	0.70	0.90	0.90	0.90	0.90	0.90	0.90	**0.93**	**0.91**	**0.91**
	LSTM	0.84	0.85	0.84	0.81	0.83	0.81	0.85	0.84	0.84	0.84	0.84	0.84	**0.86**	**0.88**	**0.87**
	CNN	0.93	0.92	0.92	0.89	0.88	0.88	0.94	0.92	0.92	0.93	0.91	0.91	***0.95***	**0.94**	**0.94**
	BERT	0.93	0.93	0.93	0.85	0.86	0.85	0.94	0.94	0.94	0.94	0.93	0.93	***0.95***	***0.96***	***0.95***
Avi-Acc	DNN	0.47	0.44	0.43	0.35	0.45	0.35	0.48	***0.47***	0.47	0.50	0.44	0.46	***0.52***	0.45	***0.48***
	LSTM	0.38	0.37	0.37	0.35	0.35	0.35	0.39	**0.39**	0.39	0.38	**0.39**	0.38	**0.40**	**0.39**	**0.39**
	CNN	0.50	***0.49***	***0.49***	0.43	0.42	0.42	***0.52***	0.44	0.47	0.51	0.44	0.47	***0.52***	0.46	0.48
	BERT	0.48	0.42	0.44	0.41	0.40	0.40	0.50	0.44	0.46	0.50	0.44	0.46	**0.51**	**0.45**	**0.47**
Avi-Safe	DNN	0.43	0.41	0.41	0.36	0.36	0.36	0.50	0.50	0.50	0.50	0.49	0.49	**0.53**	**0.51**	**0.51**
	LSTM	0.47	0.46	0.46	0.43	0.42	0.42	**0.49**	**0.50**	**0.49**	0.48	0.46	0.47	**0.49**	**0.50**	**0.49**
	CNN	0.59	0.57	0.57	0.50	0.50	0.50	0.60	0.59	0.59	0.59	0.59	0.59	***0.62***	***0.61***	***0.61***
	BERT	0.50	0.50	0.50	0.44	0.46	0.44	0.54	0.54	0.54	0.53	0.53	0.53	**0.56**	**0.57**	**0.56**
Auto-Main	DNN	0.58	0.45	0.49	0.33	0.49	0.39	0.60	**0.55**	0.56	0.58	0.54	0.55	**0.61**	**0.55**	**0.57**
	LSTM	0.49	0.55	0.51	0.41	0.42	0.41	0.50	0.60	0.54	0.51	0.58	0.54	**0.53**	**0.61**	**0.55**
	CNN	0.61	0.61	0.61	0.53	0.53	0.53	0.64	***0.64***	***0.64***	0.63	**0.64**	0.63	***0.65***	***0.64***	***0.64***
	BERT	0.60	0.60	0.60	0.54	0.53	0.53	0.63	***0.64***	0.63	0.63	0.63	0.63	**0.64**	***0.64***	***0.64***
Auto-Acc	DNN	0.43	0.34	0.30	0.35	0.42	0.27	0.39	**0.42**	0.31	0.40	0.39	0.39	**0.48**	0.40	**0.40**
	LSTM	0.45	0.39	0.41	0.40	0.40	0.40	0.42	**0.41**	0.41	0.42	0.40	0.40	**0.48**	**0.41**	**0.44**
	CNN	0.46	0.43	0.44	0.44	0.41	0.42	0.49	0.50	0.49	0.50	0.51	0.50	**0.51**	***0.53***	***0.52***
	BERT	0.50	0.49	0.49	0.47	0.47	0.47	0.50	0.50	0.50	0.51	0.49	0.50	***0.52***	**0.51**	**0.51**
Auto-Safe	DNN	0.52	0.46	0.48	0.40	0.47	0.41	0.54	0.51	0.51	0.54	0.51	0.51	**0.55**	**0.52**	**0.53**
	LSTM	0.40	0.40	0.40	0.38	0.39	0.38	0.41	**0.42**	0.41	0.41	0.41	0.41	**0.43**	**0.42**	**0.42**
	CNN	0.59	0.58	0.58	0.52	0.51	0.51	0.59	***0.60***	0.59	0.59	0.59	0.59	***0.62***	***0.60***	***0.61***
	BERT	0.57	0.56	0.56	0.52	0.50	0.50	**0.58**	0.56	0.56	0.57	0.57	0.57	**0.58**	**0.59**	**0.59**
Faci-Main	DNN	0.57	0.48	0.50	0.33	0.40	0.34	0.56	0.48	0.50	0.57	0.50	0.53	**0.59**	**0.51**	**0.54**
	LSTM	0.56	**0.56**	0.56	0.53	0.52	0.52	0.59	0.55	0.56	0.59	**0.56**	0.57	**0.63**	**0.56**	**0.60**
	CNN	0.64	0.64	0.64	0.61	0.60	0.60	0.66	0.66	0.66	0.65	0.65	0.65	***0.69***	***0.67***	***0.68***
	BERT	0.63	0.64	0.63	0.60	0.60	0.60	0.65	0.64	0.64	0.64	0.65	0.64	**0.68**	***0.67***	**0.67**

Table 4.2: Comparison of handling class imbalance results for the 7 datasets, for the baseline and four methods to address class imbalance for the four evaluated models (DNN, LSTM, CNN and BERT). Each model's macro average performance is shown as precision (Pre), recall (Rec) and F1 score. The best results over the training data selection strategies are shown in bold, and the best results over all models are additionally in italics. (Published in [4])

Dataset	DNN	LSTM	CNN	BERT
Avi-Main	0.0020	0.0043	0.0002	0.0004
Avi-Acc	0.0011	0.0399	0.0103	0.0015
Avi-Safe	0.0000	0.0023	0.0059	0.0012
Auto-Main	0.0001	0.0181	0.0009	0.0004
Auto-Acc	0.0000	0.0055	0.0001	0.0161
Auto-Safe	0.0003	0.0106	0.0011	0.0083
Faci-Main	0.0002	0.0001	0.0003	0.0005

Table 4.3: Statistical significance of the various classification models between the Baseline approach and Feedback Loop approach F1 scores using the Mann-Whitney U test. Experiments indicate statistical significance with a p value of 0.05. (Published in [4])

which both result in balanced class sizes. Undersampling removed portions of dataset that could be important for certain technical events or issues, which resulted in underfitting and weak generalization for important classes. On the other hand, oversampling may introduce overfitting in the minority class, as some of the event types are very short tokens containing domain-specific words. Following this, to minimize the possibility of overfitting and underfitting, a random downsampling loop and a feedback loop were investigated to minimize bias in the training process. It was found that the added computational cost of the feedback loop inference was worth the reduction in training time it caused over the random downsampling loop.

The scarce data available in a dataset such as *Auto-Main* is certainly an issue for deep learning methods. Examining the accuracy improvement by using the proposed feedback loop strategy, requires incorporating more instances to the event classes. Similar to any supervised learning models, we noticed some limitations that could be addressed in future work. As shown in the previous Chapter (such as Table 3.3), logbook instances contain short text, and utilizing recurrent deep learning algorithms such as LSTM RNNs which are heavily based on the context leads to weak performance compared to other algorithms. One possible explanation is that logbooks with short instances (sequences) are not providing sufficient context for the algorithm to make better predictions. Another could be that RNNs are notoriously difficult to train [90], and the LSTM models may simply require more training time to achieve similar results. There is some evidence for this, as the dataset with the most instances, which also had the second largest number of tokens per instance on average was *Faci-Main*, which is the dataset which the LSTM model had the closest performance to the CNN and BERT models, and was also the only one which the LSTM model outperformed the DNN model.

The pre-trained BERT model provided a reasonable classification performance compared to the other deep learning models, however as BERT is pre-trained on standard language, the performance when applying to logbook data was not optimal. Training or fine-tunning BERT to technical logbook data is likely to improve performance as observed in the legal and scientific domains [16,24]. As training or fine-tuning BERT requires large amounts of data, a limitation for fine-tuning a domain-specific BERT is the amount of logbook data available.

4.3 Chapter Summary

This chapter focused on predictive maintenance and importance of logbook datasets for the technical event classification task. Technical logbook datasets contain important information regarding events that occurred and were reported by domain experts, however, most of these datasets are not annotated with the reason for maintenance or categorization of issue type to utilize for the development of predictive maintenance systems. To address this challenge, we implemented popular clustering algorithms including LDA, DBSCAN, Hierarchy, and K-means and applied them to the logbook datasets. We empirically compared the clustering techniques with various forms of pre-processing methods and we calculated the intra- and inter-cluster similarity to compare their accuracy. Based on our analysis, DBSCAN produced high quality of cluster while also detecting outliers which could be used to identify new issues when becomes available to the system (which answers RQ2). We further evaluated multiple strategies to address the extreme class imbalance in these datasets to address RQ3 and we showed that the feedback loop strategy performs best, almost entirely providing the best results for the 7 different datasets and 4 different models investigated. We empirically compared different classification algorithms (DNN, LSTM, CNN, and pre-tuned BERT) to address RQ3.a and the outcomes showed CNN model outperformance compared to the other classifiers. The methodology presented in this research could be applied to other maintenance corpora from a variety of technical domains.

Chapter 5

Transfer Learning of Technical Logbooks

Chapter Introduction: As discussed in Section 2.3, transfer learning approaches have been applied in various NLP tasks to address key problems such as limited data. As such, this chapter aims to provide details of exploring the use of transfer learning methods for domain adaptation in technical event classification to handle the problem of data scarcity, using a variety of technical logbook datasets. Further details of examining various transfer learning techniques as well as the experiments and strategies utilized in this research work are also provided in Sections 5.1 and 5.2. Furthermore, details of exploring multiple similarity measurement techniques to identify the key relationships in the technical logbook dataset and investigating the impact of similarity of these corpora on the performance of the event classification model are provided in Section 5.3 as well.

5.1 Transfer Learning Techniques

Text using a modest, small technical dataset, such as the automotive maintenance data, can limit a model's generalization capacity and performance. One potential solution could be to utilize a data augmentation approach to increase the dataset size by generating synthetic data. However, domain-specific datasets where each domain captures domain-specific lexical semantics – the case for technical logbooks as illustrated previously – prevents the use of techniques such as domain-discriminative data selection applied to the smaller domain data class [73]. Furthermore, these technical datasets are highly imbalanced.

Therefore, we studied four different methods of transfer learning (domain adaptation) using the seven domain-specific datasets and analyzed their effects on the performance of technical event classification on the target datasets: (1) a baseline strategy which simply trains the model on a single dataset, and then also strategies that (2) transfer to a dataset from other sources within the domain (but different applications), (3) transfer to a dataset from sources with the same application (but different domains), and (4) transfer from all other sources in the global dataset. Figure 5.1 provides an overview of the process of transferring from the source to the target dataset utilizing three transfer learning methods in this research work.

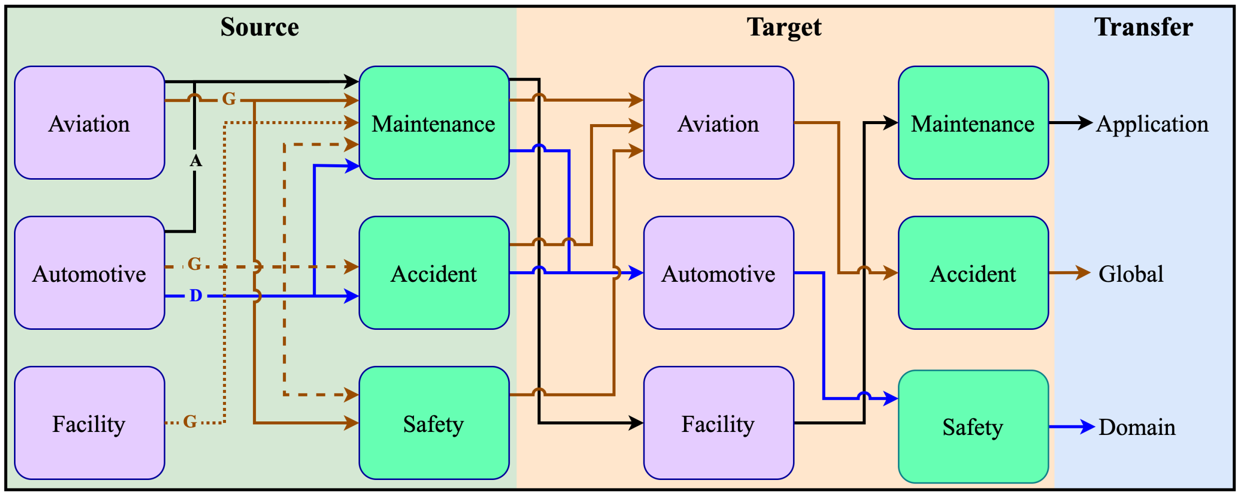


Figure 5.1: Process of transfer learning methods for technical logbooks by representing the three various approaches of transferring within the domain, transferring within an application, and transferring over the global dataset. The black color (A) represents to application, blue (D) represents the domain, and brown (G) represents the global transferring method. As an example of transferring within an application, we train the model on aviation maintenance (*Avi-Main*) and automotive maintenance (*Auto-Main*) source datasets and then take the trained model and transfer it to the facility maintenance (*Faci-Main*) data as the target dataset to perform further training and classification. (Published in [7])

Transferring within a Domain Transferring a learned model within a domain dataset can benefit the target domain dataset by utilizing knowledge learned from various domain datasets, as these corpora should have similar vocabularies. In this approach, we train the model on selected datasets within the domain, and then transfer the model to a different target dataset, where we continue to train the model to perform event classification. As an example, we can train the model on the aviation maintenance (*Avi-Main*) and aviation safety (*Avi-Safe*) source datasets and then take the trained model and transfer it to the aviation accident (*Avi-Acc*) data as the target dataset, to perform further training and classification.

Transferring within an Application While datasets within a domain stand to share similarities between their corpora, datasets that share an application may also stand to benefit in a similar manner, however with a different set of potential vocabulary. This strategy evaluates the impact of the shared knowledge within an application, where the model is first trained on source datasets from other domains which share the same application, and then transferred to the target dataset. For example, the model can be first trained on source dataset of aviation accidents (*Avi-Acc*) and then transfer to target domain of automotive accidents (*Auto-Acc*) for further training and classification.

Transferring over the Global Dataset Finally, to provide another option to address the potential that transfer learning may be improving performance simply because the model had more data to train on [80], a global transfer strategy is investigated. In this strategy, given the seven datasets available, we considered every single dataset as a target, and then initially train the model on every other dataset available.

5.2 Model Architecture and Training

For the event identification task, we considered a supervised machine learning method to classify the issue type (*e.g.*, cylinder damage, intake gasket leak). As mentioned above, the event description in a dataset contains short text and has a single event category. The machine learning model used in this study is a convolutional neural network (CNN) [54] model that has shown success in several NLP tasks such as question classification [107] or sentiment analysis [123], and that further is capable of providing suitable performance while applying it on various sequence types. Furthermore, we also evaluated it using pre-trained ALBERT [61] for English and we fine-tuned the model on the downstream task of event identification. We used the ALBERT transformer model in this work as has previously been shown to achieve high performance in various NLP tasks and benchmarks using less parameters than other transformers models, and that it also benefits from a cross-layer parameter sharing property [60].

We trained the CNN architecture proposed by Kim *et al.* [54] which consists of 100 one-dimensional convolutional filters with the size of multi n-gram lengths followed by ReLU activation, dropout layer, max-pooling layer with a size of 2 by the length of the input sequence and followed by a fully connected dense layer and output layer set to class size with SoftMax activation function.

As discussed in Chapter 1, the technical dataset classes are highly imbalanced in general. Figure 1.2 shows the *Avi-Main* dataset' imbalance classes as an example. These characteristics can cause the classification model to overgeneralize the majority class. To address this issue, we utilized the feedback loop strategy (discussed in Section 4.2.1) initially proposed by Bowley *et al.* [19] in computer vision and which has since been successfully adapted to the NLP domain. This approach not only mitigates the problem of the classifier preferring larger majority classes but also adjusts the training data such that the model keeps training on the worst performing training instances.

5.3 Datasets and Baselines

To address the issues with technical logbook datasets noted in Section 3.1.1, we utilized the text pre-processing pipeline [6] (discussed in Section 3.2) which is capable of domain-based abbreviation expansion, noise entity removal, lexical normalization, dictionary-based standardization, part of speech tagging, and domain-specific lemmatization. The dataset is divided into 80% for training and 20% for testing, and 100 dimensional word embeddings [82] were used for feature extraction.

Baseline The baseline strategy for training the models consisted of training the model on a single dataset (*e.g.*, *Auto-Safe*), and then performing the classification task on the 7 datasets. The baseline strategy does not contain any transfer learning approaches and solely uses the source dataset.

Technical Logbook Similarity Technical logbooks contain different instance sizes and token sizes, however as described in Section 3.1.1, they contain usually short instances, as well as specific words, terms, and abbreviations, where they have less ambiguity. These instances in the logbook dataset also share similar or dissimilar characteristics that would be used in specific terms or abbreviations. Each domain dataset has similar terminologies that are shared with other domains (*e.g., ft - feet*), however, some can share similar abbreviations by different semantics (*a/c* - in aviation domain: aircraft, in automotive and facility domain: air-conditioner). These shared similarities and features in the datasets could bring useful information when training the model on a specific dataset(s) and transferring the learned knowledge between within domains or dataset(s).

Instances in datasets such as those in the aviation domain, contain descriptions regarding problem types that are semantically similar to other domain's dataset such as the automotive domain (*e.g.*, *engine not start*, *cylinder compression issue*). Evaluation of instance similarity within the logbook dataset can help to interpret how data are semantically similar in either a word or sentence level. This can be done by measuring the inter-corpus similarity and identifying corpus homogeneity [22].

We experimented with applying four similarity measures including Levenshtein [57], Jaro-Winkler [124], Universal Sentence Encoder [23], Gensim Word2vec [100] to compare and extract key relations between instances in the dataset. In both the Universal Sentence Encoder and Word2vec model, we utilized the cosine similarity [75] for computations.

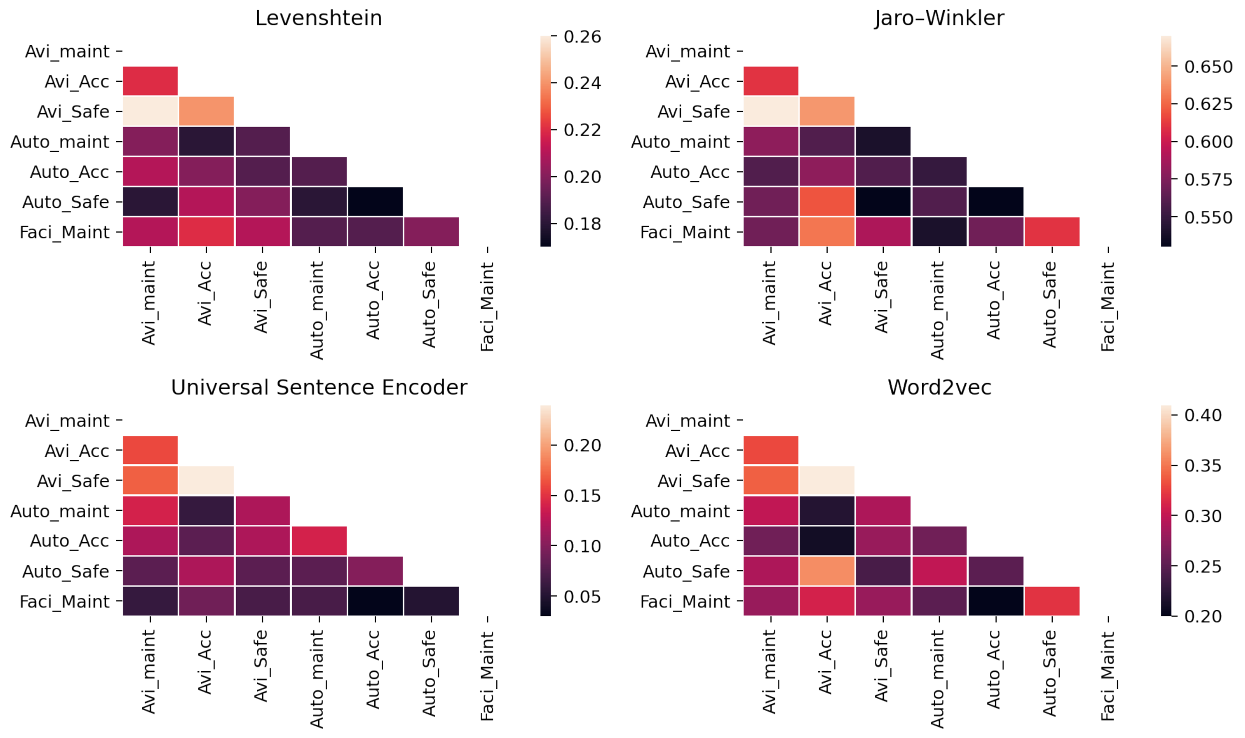


Figure 5.2: Heat maps of similarity scores for 4 algorithm including Levenshtein, Jaro-Winkler, Universal Sentence Encoder, and Word2vec applied on 500 random instances from each domain. The various color types shown in these figures describe when the similarities between a pair of datasets are lower or higher where the higher value (with lighter color) defines higher similarity, and lower value (with darker color) define low similarity. (Published in [7])

The corpus similarity experiment in this study has been done using random sets of 500 instances from each of seven dataset and we calculated the similarity measurements between instances in an inter-document form. This means every instance from a selected dataset was compared to the instances in the other remaining selected datasets to compute the distance. Figure 5.2 shows the findings of these analyses in heat maps and further discussion regarding the relationship between corpus similarity and domain adaptation is provided in Section 5.4.

5.4 Results of Transfer Learning of Technical Logbooks

This section provides a performance analysis of transfer learning between varying dataset types using the previously described CNN and ALBERT (transformer-based) model. Table 5.1 presents an evaluation of training only on the source dataset to transferring models trained on other datasets (either within the domain, within the application, or all other datasets) to that source dataset.

Dataset	Model	Baseline (%)			Domain (%)				Application (%)				Global (%)			
		P	R	F1	P	R	F1	S	P	R	F1	S	P	R	F1	S
Avi-Main	CNN	0.91	0.89	0.89	0.92	0.89	***0.90***	**0.0003**	0.89	0.88	0.88	0.1510	0.87	0.87	0.87	**0.0472**
	ALBERT	0.89	0.87	0.88	0.90	0.91	***0.90***	**0.0014**	0.88	0.87	0.87	0.1918	0.86	0.85	0.85	**0.0040**
Avi-Safe	CNN	0.88	0.85	0.86	0.89	0.87	***0.88***	**0.0336**	0.89	0.82	0.85	**0.0066**	0.88	0.82	0.85	**0.0028**
	ALBERT	0.87	0.84	0.85	0.88	0.86	**0.87**	**0.0124**	0.86	0.82	0.84	**0.0092**	0.85	0.85	0.84	**0.0163**
Avi-Acc	CNN	0.49	0.51	0.49	0.50	0.51	***0.50***	**0.0056**	0.48	0.51	0.48	**0.0293**	0.48	0.50	0.49	0.2982
	ALBERT	0.44	0.48	0.46	0.45	0.50	**0.47**	**0.0094**	0.42	0.48	0.45	**0.0187**	0.41	0.47	0.44	**0.0170**
Auto-Main	CNN	0.64	0.70	0.67	0.67	0.71	***0.69***	**0.0344**	0.64	0.68	0.66	**0.0246**	0.65	0.68	0.67	0.4548
	ALBERT	0.59	0.64	0.62	0.61	0.67	**0.64**	**0.0077**	0.58	0.61	0.60	**0.0170**	0.59	0.62	0.60	**0.0169**
Auto-Safe	CNN	0.50	0.45	0.46	0.53	0.49	***0.50***	**0.0002**	0.42	0.39	0.40	**0.0011**	0.44	0.40	0.41	**0.0171**
	ALBERT	0.48	0.46	0.46	0.50	0.48	**0.48**	**0.0026**	0.46	0.44	0.44	**0.0104**	0.47	0.42	0.43	**0.0029**
Auto-Acc	CNN	0.48	0.67	0.49	0.47	0.69	**0.50**	**0.0242**	0.47	0.68	**0.50**	**0.0372**	0.49	0.69	***0.52***	**0.0084**
	ALBERT	0.45	0.65	0.47	0.45	0.68	0.47	0.0513	0.46	0.67	**0.48**	**0.0291**	0.48	0.67	**0.48**	**0.0285**
Faci-Main	CNN	0.5	0.70	**0.56**	-	-	-	-	0.47	0.66	0.51	**0.0001**	0.45	0.65	0.49	**0.0001**
	ALBERT	0.55	0.69	***0.57***	-	-	-	-	0.51	0.67	0.54	**0.0187**	0.49	0.68	0.53	**0.0016**

Table 5.1: A performance comparison of the various transfer learning experiments. The average of the final models' performance across 10 repeated experiments is shown as precision (P), recall (R), F1 score, as well as statistical significance (S) using the Mann-Whitney U test. Results which outperform only training on the source dataset are in bold, and the best for a dataset are in bold and italics. Experiments which showed statistical significance with a p value of 0.05 are also in bold. (Published in [7])

Experimental Settings In the experimental process, we used the coarse to fine learning approach for optimizing parameters and hyperparameters [62]. Hyperparameters for training were determined by investigating batch sizes of 32, 64, and 128 [77], with an initial learning rate of 0.01 for CNN and 1e-5 for fine-tuning ALBERT model. Further, dropout regularization ranging from 0.2 to 0.4, and ReLU and SoftMax as an activation function [86], and Adam optimizer [55], and categorical cross-entropy [130] for the loss function were selected. Based on the experiment and model performance, dropout regulation with a rate of 0.2 and batch size of 32 were selected for the training.

Experimental Design First, the CNN and ALBERT (fine-tuned) models were trained 10 times on each source dataset, with the *baseline* (source) column reporting the average precision (P), recall (R) and F1 scores of those runs. Following this, 10 CNN and ALBERT (fine-tuned) models were trained on the other domains, and then each of those 10 models were transferred to the source dataset for further training (layer freezing was not used). These results are reported in the *domain*, *application* and *global* columns. Additionally, Mann-Whitney U-Tests of statistical significance were performed comparing the populations of final losses across the 10 repeated experiments of the source data, to the final losses of the 10 repeated experiments for each of the transferred runs, which are reported in the *S* columns. Based on these experiments, we observed performance improvements for each dataset.

5.5 Discussion of Transfer Learning of Technical Logbooks

All the datasets in the aviation domain had improved performance when being transferred to from within the domain with statistical significance, while their performance was degraded when transfer learning was performed across applications and from the global dataset. Similar results were found in the automotive domain, where the model performance improved across all datasets when using the within-domain transfer learning approach, however, interestingly, the automotive accident (*Auto-Acc*) dataset also achieved better model performance while transferring over the application and global datasets, with the best performance coming from the global dataset. For the other two datasets, within-domain transfer learning found the best results, also with statistical significance (which answers RQ4). For the facilities maintenance dataset (*Faci-Main*), within-domain transfer learning was not possible as there were no other datasets in the domain, and similar to the aviation dataset, transferring from the application and global datasets also reduced performance with statistical significance.

The transfer learning results provide some interesting insights into how transfer learning can be performed across varying technical logbook datasets. While in all cases, keeping the training data within a domain provided a statistically significant improvement, in the automotive accident dataset, utilizing all other training data provided the overall best results. This suggests that while for the majority of our datasets, adding in additional training data from outside of the domain only served to confuse the models, but that this is not always the case. Some datasets may still benefit from simply having more training data due to the nature of that particular classification problem, or perhaps due to a wider variety of tokens allowing for more similarity to other datasets.

Furthermore, to address RQ4.a, we examined similarity measurement techniques to identify the key relationships in the aviation, automotive, and facilities domains, as well as investigating the similarity of these corpora that might lead to lowering or improving the performance of event classification model. For this reason, we applied the Levenshtein and Jaro-Winkler similarity methods to compare the similarity of these corpora. However, to further extract the key attributes between these datasets, we employed the Universal Sentence Encoder and Word2vec model to semantically evaluate the instances based on their semantic meanings.

Based on the outcomes, we noticed high inter-corpus similarity within the aviation safety (*Avi-Safe*) and aviation maintenance (*Avi-Main*) datasets. The reason for this high inter-corpus similarity score could be the common domain terms and abbreviations that have been used in these datasets for instance "*eng was shut down and noticed slight vibration*" and "*right eng vibration with increasing power*" where the domain abbreviated word "*eng*" appeared in both aviation safety and aviation maintenance dataset respectively. Additionally, this could be related to the performance of transfer learning outcomes where sharing similar information within the domain leading performance improvement, however, this could require additional analysis of including more similar domain datasets (*e.g.*, additional aviation safety data) in the future, as well as a further evaluation of how models' performance can be further improved by transferring from more similar datasets. Furthermore, in comparison of the evaluated similarity methods, the Universal Sentence Encoder method provided outcomes which were more representative of the performance of transfer learning compared to the other methods.

5.6 Chapter Summary

As discussed in Section 2.3, transfer learning approaches have been applied in various NLP tasks (*e.g.*, machine translation, sentiment analysis) and achieved a model performance improvement in various studies. Therefore this research work compared transfer learning approaches for domain adaptation for event classification in logbook datasets. We acquired seven logbook datasets from three technical domains containing short instances with non-standard grammar and spelling, and many abbreviations.

To answer RQ4, we evaluated three domain adaption methods including (1) transferring within the domain, (2) transferring within the application, and (3) transferring over the global dataset compared to the baseline approach of training classification model on the single (source) domain dataset. Our results indicate that transferring within the domain dataset delivers the best performance across both CNN and ALBERT (transformer-based) models. Finally, to address RQ4.a we applied corpus similarity techniques to investigate shared characteristics among these technical datasets, using Levenshtein, Jaro-Winkler, Universal Sentence Encoder, and the Gensim Word2vec models. The outcome of similarities indicated high inter-corpus similarity within the domain dataset of aviation safety and aviation maintenance, which could also correlate to the performance of transfer learning. A comparison of the aforementioned similarity techniques also indicated that Universal Sentence Encoder provided outcomes more relates to the performance of transfer learning.

Chapter 6

Domain-specific Technical Logbook Augmentation

Chapter Introduction: The following chapter evaluates data augmentation techniques for technical logbook datasets to address the problem of data scarcity while further improving the event classification models' performance. Section 6.2 provides details of the various data augmentation techniques evaluated in this research work. As analyzing the performance of data augmentation techniques and identifying the proper method for technical logbook is important, and Section 6.3 provides the detail of various metric-based evaluation techniques examined in this research work. Additionally, a discussion of utilized event classification methods and experiments performed for augmentation of technical logbook datasets are provided in Sections 6.4 and 6.6.

6.1 Data Augmentation for NLP

Data augmentation is a known technique to introduce a modified version of the original data to the models in order to improve the generalization and further the model performance. Data augmentation techniques are also an important method known to cope with data scarcity [87, 103]. Data augmentation is also a widely used and known approach in various computer vision applications such as image cropping, rotation, or flipping and has been recently adapted to various NLP problems and tasks as well [38].

Based on the overview of current research works discussed in Section 2.5, these data augmentation techniques were developed to further improve machine learning model performance on various challenging NLP tasks such as question and answering. However, the nature of the technical logbook data prevents us from using data augmentation methods that were recently proposed based on pre-trained transformer models (which are most often trained on general (or standard) corpora (*e.g.*, Wikipedia)) [71] that may not be useful on applying to the technical logbook data.

Given the unique and proprietary nature of technical logbook data (which is a key issue related to the performance of the classification models), the augmentation techniques that we are choosing should be investigated properly for the nature of these datasets. Regarding the provided discussion in Section 2.5, rule-based data augmentation techniques such as *easy* data augmentation [125] were developed to properly augment the text data in the various NLP tasks [12,120]. However, employing these techniques requires making sure that they are not altering the actual problem definition in the logbook entries (*e.g.*, deletion of “noun”), and as well as preserving the semantics of the problem definition without removing domain-specific terms. These consist of replacing the domain keywords with proper and close meaning (*e.g.*, engine choked briefly - engine blocked shortly).

6.2 Data Augmentation Methods

There are several data augmentation techniques proposed in various research studies which have shown great success in NLP tasks such as text classification, sentiment analysis, and question answering [25, 42]. In these various studies, data augmentation methods are utilized to create further synthetic examples which help the model learn to generalize better. The following section provides the details of the augmentation techniques utilized in this research work.

Rule-based Augmentation Techniques A rule-based technique includes various text-based augmentation approaches to help perform certain operations on either word or characters to generate new augmented outputs. These operations include “Synonym Replacement”, “Random Swap”, or “Random Deletion” [39] and have been utilized in various NLP tasks. In this work, we employed the aforementioned rule-based augmentation strategies at the word level to augment technical logbook datasets for event identification tasks.

As shown in Algorithm 2, in the process after technical logbook datasets is divided into training and testing, for each augmentation strategy the instance of training data ($\mathcal{DM}$) were split into tokens ($\mathcal{T}_s$), and for instances that have a minimum specified number ($\mathcal{P}$) of tokens (tokens ≥ 2), a

Algorithm 2 Data Augmentation Pseudocode

```
Input: Training Data $\mathcal{D}$ = Instance $(1, 2, \ldots, N)$
Operation Type: $\mathcal{DA}$ = Synonym Replacement ($\mathcal{SR}$), Random Swap ($\mathcal{RS}$), Random Deletion ($\mathcal{RD}$)

function PerformAugment($\mathcal{T}_s$, $\mathcal{DA}$)
    for $i \leftarrow$ every($\mathcal{T}_s$) do
        if $\mathcal{DA}$ is $SR$ then
            $\mathcal{A} \leftarrow$ ReplaceSynonym($\mathcal{T}_s$)
        else if $\mathcal{DA}$ is $RS$ then
            $\mathcal{A} \leftarrow$ RandomSwap($\mathcal{T}_s$)
        else if $\mathcal{DA}$ is $RD$ then
            $\mathcal{A} \leftarrow$ RandomDelete($\mathcal{T}_s$)
    return $\mathcal{A}$

▷ Gets $\mathcal{DM}$ instance from training data ($\mathcal{D}$)
function EventAugmentation($\mathcal{DM}$, $\mathcal{P}$, $\mathcal{DA}$)
    Array $\mathcal{A}$, Token $\mathcal{T}$
    for $i \leftarrow$ every($\mathcal{DM}$) do
        $\mathcal{T}_s \leftarrow$ Split($\mathcal{DM}_i$)
        if $\mathcal{T}_s$ is $\geq \mathcal{P}$ then
            $\mathcal{A} \leftarrow$ PerformAugment($\mathcal{T}_s$, $\mathcal{DA}$)
        else$\mathcal{T}_s$ is $< \mathcal{P}$
            pass
    return $\mathcal{A}$
```

rule-based augmentation operation (*PerformAugment*) performed (using tokens and defined certain operation type) and used to train the models. In augmentation operations, the synonym replacement (*ReplaceSynonym*) approach consists of randomly replacing the synonym of the instances with the similar synonym from the lexical (English) database (*e.g.*, input instances of "*both engines not working*" in aviation domain would augmented as "*both engines not operating*"). The random swap (*RandomSwap*) approach performs by randomly replacing the position of the tokens in the instances (*e.g.*, input instances of "*excessive speeding along arkansas*" in automotive domain would augmented as "*along arkansas excessive speeding*"), where the Random deletion (*RandomDelete*) consists of removing the random set(s) of tokens from the input instances (*e.g.*, input instances of "*car rear bumper cover paint need coating*" in automotive domain would augmented as "*car rear bumper paint need coating*").

6.3 Evaluation of Data Augmentation Methods

Due to the unique nature of the technical logbook compared to standard corpora, the quality and properties of instances generated by the various data augmentation methods/operations may vary. This can also impact a model's performance on the downstream task of classification. To assess the efficiency of data augmentation methods and investigate the quality of the problem and label information adjustments, various qualitative evaluation approaches were considered in this research. Our evaluation process is composed of utilizing various metric-based evaluations which are known as standard NLP measurements. The following section provide the details of these evaluation methods.

Metric-based Evaluation Utilizing metric-based evaluations provides a cheap and efficient way to assess the quality of the generated instances, as compared to collecting human evaluations, which is an expensive process. There are various popular metrics that are available for Natural Language Generation (NLG) tasks including the BLUE [89], and ROUGE [68] to calculate the score of the generated/augmented instances. These two aforementioned approaches require two input instances of reference and candidate and will return a score that indicates how the candidate instance matches the input reference instance. The BLUE score (bilingual evaluation understudy) as initially was proposed for automatic evaluation of the quality of machine translation texts, recommended to be utilized in the data augmentation evaluation process by various research works as shown to perform well.

For additional comparison to these popular scores, utilizing the similarity-based methods including Levenshtein [57], Jaro-Winkler [124], Universal Sentence Encoder [23], and Gensim Word2vec [100] also considered to capture the word-level and sentence-level similarity of the augmented instance (candidate) to the original reference instances. For this form of evaluation, random samples of the generated datasets per method with their reference input instances were selected for calculating and comparison of the scores.

Finally, the outcomes obtained from these different metric-based evaluation approaches provide a reasonable assessment on which data augmentation method(s) is suitable for the nature of technical logbook data representing the various domain-specific abbreviations and terminologies.

6.4 Model Architecture and Training

Per discussion provided in Section 2.5, data augmentation techniques are used in various NLP downstream tasks such as Neural Machine Translation (NMT) to improve the performance of models. Therefore to further examine the performance of data augmentation methods on event identification tasks for the technical logbook datasets consisting of short text with a single event category, we considered a supervised machine learning method to classify the issue type in the technical logbook datasets (such as *baffle crack*) and further analyze the outcomes.

The machine learning models used in this study are a transformer-based BERT-Tiny, BERT-Small, and BERT-Medium [33] which have previously shown success in various NLP studies by properly identifying the context of the instances. Furthermore, based on the discussion provided in Chapter 1, as technical dataset classes are highly imbalanced, we utilized the feedback loop strategy [19] (discussed in Section 4.2.1) to overcome this problem. Further discussion regarding the pre-training process of these models using technical language data is provided in Section 6.5.1.

6.5 Datasets and Baselines

Regarding the descriptions provided in Sections 1.1 and 3.1.3, technical logbook datasets contain non-standard language, and to address this issue we used the text pre-processing pipeline [6] (discussed in Section 3.2) on logbook data that consists of domain-based abbreviation expansion, domain-based noise entity removal, lexical normalization, dictionary-based standardization, part of speech tagging, and domain-specific lemmatization. Furthermore, the dataset was divided into 80% for training and 20% for testing to prevent contamination of the test set. The augmentation techniques were only applied to the training set, with the augmentation technique being applied to each instance, which kept the training data size constant. For example, the 3,859 training instances in automotive safety (*Auto-Safe*) were replaced with modified versions of after performing the specified augmentation technique (*e.g.*, synonym replacement) on each instance, resulting in 3,859 modified instances.

6.5.1 Pre-training Transformer-based Language Models

As previously discussed in Section 2.4, various transformer-based language models using encoder-only architectures such as BERT achieved significant results on various downstream tasks while being applied to standard corpus datasets as well as their successful performance while being adapted to other domains (*e.g.*, ClinicalBERT [48]). Therefore in this research, we experimented with pre-training three transformer-based BERT models including BERT-Tiny, BERT-Small, and BERT-Medium [119] as these models have previously shown comparable outcomes on various NLP benchmarks such as GLUE while using less parameters. The number of layers for the BERT-Tiny model is 2, for the BERT-Small is 4, and for BERT-Medium is 8. The pre-training process was performed using the Masked Language Modeling objective on technical logbook datasets with the original BERT [33] network configuration.

Baseline The baseline approach consists of training the models on each technical logbook dataset (*e.g.*, *Avi-Acc*) without utilizing any data augmentation techniques (only using the source dataset) and then further performing the event classification task.

6.5.2 Technical Logbook Evaluation Setup

In the evaluation process, 100 random samples of original instances from domain-specific logbook data were selected as references and used for the input to data augmentation methods to compare their performance. Following this, three data augmentation techniques were utilized including synonym replacement, random swap, and random deletion, where 100 instances per technique were generated.

To perform the evaluation, input reference instances and generated output instances from each data augmentation method were used as evaluation references and candidates respectively. The following Table 6.1 provides the overview of the sample of input reference data shown in the *instance* column and their *augmented* instances from three data augmentation methods as shown in column *operation.*

The outcome of the evaluation using four various metric-based evaluation methods (also discussed in Section 6.3) including BLUE, Levenshtein, Jaro-Winkler, and Universal Sentence Encoder provided in the following Table 6.2. This evaluation outcome indicates how well the new augmented instance from each method is semantically similar to the original input instance. Also, it can help to identify more suitable data augmentation techniques to utilize in event identification tasks for technical logbooks.

Instance	**Operation**	**Augmented**
aircraft on taxi gear collapsed and ground looped	Synon. Repl.	aircraft on hack gear collapsed and ground looped
	Rand. Swap	aircraft on gear taxi ground and collapsed looped
	Rand. Delete.	aircraft on taxi collapsed and looped
exhust air leak check light	Synon. Repl.	exhust breeze leak check light
	Rand. Swap	exhust leak air check light
	Rand. Delete.	exhust air check light

Table 6.1: Example instances of technical logbook entries and their augmented form produced by domain-specific data augmentation techniques utilizing various approaches of synonym replacement (Synon. Repl.), random deletion (Rand. Delete.), and random swap.

Evaluation Method	**Synon. Repl (%)**	**Rand. Swap (%)**	**Rand. Delete(%)**
BLUE	0.45	0.40	0.38
Levenshtein	0.79	0.74	0.73
Jaro-Winkler	0.82	0.91	0.78
Universal Sentence Encoder	0.90	0.94	0.88
Gensim Word2vec	0.84	0.92	0.80

Table 6.2: Evaluation of domain-specific data augmentation techniques of synonym replacement (Synon. Repl.), and random deletion (Rand. Delete.) and swap using various metrics.

Based on the evaluation of similarity-based outcomes, including the Universal Sentence Encoder, we determined that the random swap technique produces outcomes semantically similar to the original input compared to the other techniques, however when evaluated with BLUE and Levenshtein, the outcomes are varied and shows high similarity for the synonym replacement method. This could be related to the fact that similarity methods such as Levenshtein consider the position of token when comparing two instances.

6.6 Results of Data Augmentation of Technical Logbooks

This section provides evaluation outcomes for examining the impact of data augmentation techniques on model performance utilizing various transformer-based language models including BERT-

Tiny, BERT-Small, and BERT-Medium (as discussed in Section 6.4). Table 6.3 presents these evaluations that include event identification using the baseline approach (solely trained on source data) and three data augmentation techniques of synonym replacement, random swaps, and random deletion. A further discussion of the experiment outcomes is provided in Section 6.7.

Dataset	Model	Baseline (%)			Synonym Repl. (%)				Random Swap (%)				Random Delete. (%)			
		P	R	F1	P	R	F1	S	P	R	F1	S	P	R	F1	S
Avi-Main	BERT-Tiny	0.90	0.89	***0.89***	0.88	0.87	0.87	**0.0423**	0.90	0.90	***0.89***	0.1043	0.86	0.84	0.84	**0.0022**
	BERT-Small	0.90	0.90	0.89	0.89	0.88	0.88	**0.0342**	0.90	0.91	***0.90***	**0.0198**	0.87	0.84	0.84	**0.0013**
	BERT-Medium	0.90	0.91	0.90	0.89	0.90	0.88	**0.0437**	0.91	0.92	***0.91***	**0.0288**	0.88	0.85	0.86	**0.0025**
Avi-Safe	BERT-Tiny	0.87	0.84	0.84	0.90	0.74	0.80	**0.0153**	0.91	0.82	***0.86***	**0.0405**	0.81	0.83	0.82	**0.0242**
	BERT-Small	0.89	0.85	0.86	0.83	0.88	0.84	**0.0268**	0.87	0.88	***0.87***	**0.0478**	0.83	0.83	0.82	**0.0094**
	BERT-Medium	0.89	0.87	0.87	0.90	0.86	0.86	**0.0441**	0.88	0.88	***0.88***	**0.0202**	0.84	0.83	0.83	**0.0056**
Avi-Acc	BERT-Tiny	0.51	0.47	***0.48***	0.53	0.46	0.47	0.0651	0.50	0.46	***0.48***	0.3087	0.57	0.43	0.43	**0.0267**
	BERT-Small	0.53	0.46	0.48	0.53	0.47	**0.48**	0.3387	0.51	0.49	***0.49***	**0.0378**	0.54	0.43	0.46	**0.0293**
	BERT-Medium	0.53	0.49	0.50	0.50	0.50	0.49	**0.0410**	0.52	0.51	***0.51***	**0.0319**	0.53	0.48	0.48	0.0867
Auto-Main	BERT-Tiny	0.60	0.65	0.61	0.62	0.64	***0.62***	**0.0285**	0.56	0.64	0.60	**0.0137**	0.55	0.58	0.55	**0.0087**
	BERT-Small	0.64	0.67	***0.63***	0.62	0.66	0.62	**0.0319**	0.60	0.66	0.62	**0.0269**	0.56	0.60	0.57	**0.0156**
	BERT-Medium	0.66	0.69	0.65	0.61	0.68	0.63	**0.0225**	0.66	0.71	***0.67***	**0.0205**	0.64	0.63	0.62	**0.0187**
Auto-Safe	BERT-Tiny	0.54	0.48	0.50	0.57	0.50	**0.52**	0.0104	0.57	0.52	***0.53***	**0.0045**	0.58	0.49	**0.51**	**0.0093**
	BERT-Small	0.56	0.50	0.52	0.58	0.53	***0.54***	**0.0268**	0.57	0.53	**0.53**	**0.0316**	0.57	0.51	**0.53**	**0.0436**
	BERT-Medium	0.56	0.52	0.53	0.58	0.52	**0.54**	**0.0376**	0.59	0.51	***0.55***	**0.0095**	0.58	0.52	0.53	0.0750
Auto-Acc	BERT-Tiny	0.55	0.54	***0.50***	0.51	0.55	0.49	**0.0433**	0.52	0.54	0.49	0.0594	0.43	0.52	0.39	**0.0020**
	BERT-Small	0.65	0.52	0.52	0.73	0.53	***0.54***	**0.0246**	0.73	0.54	***0.54***	**0.0205**	0.47	0.50	0.40	**0.0019**
	BERT-Medium	0.61	0.54	0.54	0.74	0.54	**0.55**	**0.0203**	0.74	0.55	***0.56***	**0.0186**	0.48	0.52	0.42	**0.0051**
Faci-Main	BERT-Tiny	0.67	0.61	0.62	0.68	0.64	***0.65***	**0.0115**	0.69	0.64	***0.65***	**0.0094**	0.70	0.59	0.60	**0.0141**
	BERT-Small	0.66	0.65	0.64	0.67	0.66	***0.66***	**0.0287**	0.70	0.63	**0.65**	**0.0310**	0.67	0.62	0.63	**0.0428**
	BERT-Medium	0.70	0.63	0.65	0.71	0.64	***0.66***	**0.0311**	0.70	0.65	***0.66***	**0.0433**	0.68	0.63	0.64	**0.0475**

Table 6.3: A performance comparison of the various text data augmentation methods. The average of the final models' performance is shown as precision (P), recall (R), F1 score, as well as statistical significance (S) using the Mann-Whitney U test with a p value of 0.05 are also in bold. The best outcome for a dataset are in bold and italics.

Experimental Settings The experimental process in this study was performed by utilizing the coarse to fine learning approach [62] to set parameters and hyperparameters for fine-tuning the BERT-tiny, BERT-Small, and BERT-Medium models. This experiment was considered by examining batch sizes of 32, and 64, with an initial learning rate of 1e-5 for fine-tuning all models where based on the experiment and model performance batch size of 32 was selected for training. Furthermore, ReLU and SoftMax as an activation function [86], Adam optimizer [55], as well as categorical cross-entropy [130] for the loss function, were selected for training the models.

Experimental Design In the training process, each model was trained 10 times using each specific data augmentation technique as well as a baseline approach (using only the source dataset) to generate results for each dataset. The training process of each augmentation technique was also consists of keeping the training data size constant and perform certain augmentation operations during the training of each model to provide a fair computation budget for each approach. The outcome of these experiments is reported by average precision (P), recall (R), and F1 scores of the runs in the *Baseline*, Synonym Replacement (*Synonym Repl*), *Random Swap*, and Random Deletion (*Random Delete.*) columns of Table 6.3. Furthermore, the *S* column in this Table indicates the outcome of Mann-Whitney U-Tests of statistical significance conducted by comparing the set of final losses of 10 repeated experiments of a baseline approach to the final losses of 10 repeated experiments of each data augmentation technique.

6.7 Discussion of Data Augmentation of Technical Logbooks

In order to examine the performance of data augmentation techniques on technical logbook data, we used three rule-based methods of synonym replacement, random swap, and random deletion. Based on the experiments and outcomes shown in Table 6.3, and also using the baseline approach which solely relies on source (original) data and other three augmentation approaches, we identified the various impacts on the event identification task.

We observed performance improvement with statistical significance on datasets such as aviation maintenance (*Avi-Main*), aviation safety (*Avi-Safe*) and automotive safety (*Auto-Safe*) utilizing the BERT-Medium model while applying the random swap techniques. However, we noticed that the logbook datasets such as automotive safety (*Auto-Safe*) and automotive accident (*Auto-Acc*) with BERT-Small model, their performance improved with the synonym replacement technique and in some cases, achieved an outcome similar to the baseline such as aviation accident (*Avi-*

Acc) with BERT-Tiny model. On the other hand, random deletion in some cases performed worst by achieving a lower performance with statistical significance, whereas only in automotive safety (*Auto-Safe*) performed similar results to baseline with BERT-Medium. This could be the reason that deletion of domain words could lead to loss of context in the instance which is important for the event descriptions.

Furthermore, to evaluate the quality of the generated instances from the aforementioned augmentation techniques, we performed the metric-based evaluation (also discussed in Section 6.3) where the outcomes are shown in Table 6.2 and indicate the overall high similarity of the augmented text specially in the case of random swap compared to the other methods (excluding the BLUE and Levenshtein). This could also refer to the question of whether modification of the instance could alter or preserve the problem definition. However, for further examining the evaluation, an expert-based (human) evaluation of these methods could help to identify if the semantics of the technical logbook data with various domain-specific terminologies changes or remain semantically similar to the original problem definition.

6.8 Chapter Summary

Data augmentation is known as an important approach to improving model performance and generalization in various NLP tasks. To answer RQ5, we explored three data augmentation techniques including synonym replacement, random swap, and random deletion to improve the event identification task using technical logbook datasets from three domains of aviation, automotive, and facility. Furthermore, to properly determine the optimal data augmentation techniques for technical logbooks containing domain-specific event descriptions with non-standard grammar and spelling, we computed four metric-based evaluations (BLUE, Levenshtein, Jaro-Winkler, and Universal Sentence Encoder) from original input data as a reference and augmented instances as a candidate. The outcome of a metric-based evaluation and empirical analysis of the event identification task indicated the random swap technique is a more suitable augmentation method compared to the other two utilized augmentation techniques of synonym replacement and random deletion.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This doctoral research focuses on the problem of pre-processing, clustering, and technical issue classification by considering technical logbook datasets from the automotive, aviation, and facilities maintenance domains. Technical logbook datasets are written by domain experts (*e.g.*, engineers) and contain short text entries with non-standard language (including domain-specific abbreviated words and terminologies), which makes it challenging for off-the-shelf NLP tools trained on standard contemporary data to process them. The problem description in technical logbooks consists of important information that requires to be processed for the development of an effective predictive maintenance system. To overcome the aforementioned challenge and further develop the methods to handle technical logbook datasets and perform event identification tasks, we addressed various research questions (RQ) in this thesis work.

To overcome the problem of non-standard language in technical logbooks, we addressed the proposed RQ1 ("How well do state-of-the-art pre-processing techniques clean unstructured maintenance data? Can we develop better techniques to handle this type of data?") by developing the pre-processing pipeline to clean these unstructured maintenance data. We further performed an intrinsic evaluation comparing the performance of various spell-checkers for the technical logbook dataset to convert the non-standard language to a standard format (Published in [6]).

Furthermore, as discussed in Section 3.2, most technical logbook datasets are not annotated with the reason for maintenance or categorization of the issue type. To address this challenge and answer the proposed RQ2 ("Using the proposed methods in this research work, which techniques are robust and the most effective in clustering/mining aviation maintenance textual data?"), various clustering methods and similarity metrics for maintenance text mining are compared as well to provide the annotation in the technical logbook datasets (Published in [6]).

As technical logbooks tend to be characterized by a large number of event classes that are highly imbalanced, we evaluated multiple strategies to address the extreme class imbalance in these technical datasets. We adapted the feedback loop approach in Bowley *et al.* [19] from the computer vision domain to address RQ3 ("To which extent does the class granularity and class imbalance present in technical logbooks impact technical event classification performance, and can a feedback loop for training data selection effectively address this issue?") and we showed that a feedback loop strategy performs the best (Published in [4]).

We also compared transfer learning approaches for domain adaptation for event classification in logbook datasets. We evaluated three-domain adaption methods including (1) transferring within the domain, (2) transferring within the application, and (3) transferring over the global dataset compared to the baseline approach of training classification model on the single (source) domain dataset to answer our RQ4 ("Which transfer learning approaches are better suited for classifying technical events for predictive maintenance across heterogeneous logbook datasets?"). Our results indicate that transferring within the domain dataset usually delivers the best performance (Published in [7]).

To further examine the transfer learning approaches and answer RQ4.a ("How does the level of similarity between corpora impact the performance of transfer learning approaches for technical event classification?"), we applied corpus similarity techniques to investigate shared characteristics among these technical datasets. We used four similarity methods including Levenshtein, Jaro-Winkler, Universal Sentence Encoder, and the Gensim Word2vec. Based on the outcome of these methods, we noticed the high inter-corpus similarity for within-domain datasets of aviation safety and aviation maintenance. Furthermore, the outcomes achieved by Universal Sentence Encoder were more relates to the performance of transfer learning compared to the other utilized methods (Published in [7]).

Furthermore, to answer RQ5 ("How well data augmentation techniques can impact classification models' performance while preserving the label information in logbook datasets?"), we examined the various data augmentation approaches including synonym replacement, random swap, and random deletion to improve the event identification task by utilizing technical logbook datasets. We further used three pre-trained machine learning models of BERT-Tiny, BERT-Small, and BERT-Medium.

We pre-trained these models using the Masked Language Modeling objective and fine-tuned them on the downstream task of event classification. To examine the quality of data augmentation techniques, we also computed four metric-based evaluations methods including BLUE, Levenshtein, Jaro-Winkler, and Universal Sentence Encoder. Based on the outcome of three utilized augmentation techniques, four metric-based evaluations and three models, we observed the performance improvement of classification models using random swap in most cases such as aviation safety (*Avi-Safe*) and aviation accident (*Avi-Acc*) data.

7.2 Future Work

The research work in this dissertation provides essential phases of processing technical language datasets as well as performing the downstream task of event identification. There are also additional approaches that could help provide a future direction to improving the various research limitations such as logbook data scarcity. The following sections provide an overview of these possible research directions:

7.2.1 Zero- and Few-Shot Learning for Technical Logbook

One of the suitable approaches as a future direction of this research that could be explored is zero-shot and few-shot learning classification approaches for technical logbooks that contain non-standard languages with domain-specific abbreviations and terminologies. A zero-shot and few-shot learning is a known technique in computer vision [45] and recently has been employed in various NLP research work [43, 59] to improve the model performance in downstream tasks with limited training data available. As technical logbook datasets are proprietary, utilizing zero-shot and few-shot learning methods would help to analyze if the models can perform well with the limited logbook data available.

Furthermore, as discussed in Section 3.1.3, technical logbooks also contain varying lengths of problem descriptions that can consist of a few or more tokens that describe the same issue (or may contain various domain terms in instances but are semantically similar). An investigation of zero-shot or few-shot learning with this nature of technical data would help to analyze how the performance of the model could improve when dealing with these challenging structures of data.

7.2.2 Domain Knowledge Fusion for Data Augmentation

An investigation of the benefit of incorporating the domain adaptation methods with data augmentation techniques would provide a proper comparison if this strategy can help the model in better generalization. The process in this approach includes utilizing unsupervised data from other related domains for augmenting the technical logbook dataset and further examining the performance of the approach on event identification tasks. Further, various data augmentation techniques such as rule-based, or model-based could be employed to construct new instances.

Alternatively, incorporating other rule-based textual augmentation approaches such as combining the multiple rule-based techniques (*e.g.*, "Synonym Replacement" with "Random Deletion") can be examined to compare how this approach would further benefit the model in better generalizing by improving the event classification performance. It should be noted that new instances that are generated using the aforementioned techniques (*e.g.*, combining two rule-based methods) should not alter the label information. To make sure these new instances are still semantically similar to input (source) data with the same label information, employing a similarity method is required to compare the output of every augmented instance with the original input instance.

7.2.3 Human Evaluation of Data Augmentation

As discussed in Section 6.3, the quality of instances generated by data augmentation techniques may vary and this can impact models' performance in various downstream NLP tasks. To assess the efficiency of data augmentation approaches, we employed various qualitative evaluation approaches and compared their outcomes (Section 6.5.2). However to further evaluate the utilized data augmentation methods (including the aforementioned methods in Section 7.2.2 as well), and examine how far the augmented instances varied from original input resources, an additional qualitative approach of human evaluation needs to be investigated for each technique that considers human-in-the-loop for evaluation.

This form of evaluation would be composed of collecting evaluations based on expert and nonexpert-based (English only) assessment categories: (*Domain experts)* for evaluation purposes needs to be chosen based on their expertise in the domain. The process would consist of providing them with a task that includes the questions (quality assessment based on fluency and adequacy [41]) and then collecting the final evaluation outcome. As technical logbook data are written by domain experts (*e.g.*, mechanics), this form of evaluation can be considered as a gold standard as this is a significant way to analyze if any specific data augmentation operation (*e.g.*, random deletion) can alter the semantic of the problem description and event label. In (*Nonexpert-based)* evaluation, all human annotators need to be chosen based on their expertise with the minimum of a master's degree in natural language processing, computer science, or other related fields.

Experimental Setup for Human Evaluation The annotation process would consist of giving the set of randomly generated instances and asking them (human evaluators) to evaluate the instances. In each evaluation (expert and nonexpert), random samples of augmented (generated) datasets (*e.g.*, 100) would be selected to assess the augmentation methods' performance. These random selections would consist of original reference instances from domain-specific logbook data that are used for the input of data augmentation methods, and every output of generated instances from each data augmentation method. To better quantify the performance of the data augmentation methods and reduce the annotator's bias, the given random sample of instances in each task needs to be randomized.

For both expert and nonexpert-based evaluation methods, three common task-based questions need to be asked (with one possible answer to choose out of two answers): 1) *Does the instance grammatically sound correct?* This question will define the fluency of the new augmented instances. 2) *Does the instance accurately define the given problem (issue) type(s)?* (*e.g.*, engine failure, speeding), and 3) *Does the given instance represent augmented or real format?*. To further assess human annotation and reliability of the rating (for more than 2 human evaluations), calculating intraclass correlation (for the inter-rater reliability) and as well as Cohen's kappa coefficient where the scores range between 0 with no agreement and 1 with the high agreement in both methods required.

7.3 Concluding Remarks

Predictive maintenance is a technique applied to engineering systems to estimate when maintenance should be performed to maintain equipment or software, and to be an enhancement to or even prevent the need for traditional scheduled maintenance. Performing maintenance only as needed can not only save costs but additionally improve safety as scheduled maintenance tasks do pose a risk of causing issues as routine maintenance is performed. Technical logbook datasets are a important source of information for developing predictive maintenance systems. These datasets contain descriptions regarding maintenance problems and actions performed to address these issues.

However, these datasets are typically written by domain experts in non-standard language and grammar. The nature of these datasets makes them challenging for off-the-shelf Natural Language Processing (NLP) models to process. These challenges includes using various domain-specific abbreviations, acronyms, and misspelling or dropped words in the description by domain experts (*e.g.*, mechanics). Furthermore, characteristics of these datasets make them distinct from other domain-specific datasets (*e.g.*, biomedical [31]) as well.

Therefore the research in this dissertation aimed to address these challenges by developing preprocessing tools and further examining various clustering and classification methods to extract information for predictive maintenance. The evaluated strategies such as transfer learning methods, or data augmentation techniques for this research showed how domain-specific data is different in nature compared to other natural language data. The methodologies evaluated in this research provided an important assessment of which methods are suitable to apply to domain-specific data containing similar nature and structures. Furthermore, the methodologies discussed as a future direction such as “domain knowledge fusion for data augmentation” could also apply to other domains that have limited available data to perform certain NLP downstream tasks such as event classification to improve model performance.

Bibliography

[1] Wathiq Abed, Sanjay Kumar Sharma, and Robert Sutton. Diagnosis of bearing fault of brushless DC motor based on dynamic neural network and orthogonal fuzzy neighborhood discriminant analysis. In *2014 UKACC International Conference on Control (CONTROL)*, pages 378–383, 2014.

[2] Charu C. Aggarwal and ChengXiang Zhai. A survey of Text Clustering Algorithms. In *Mining Text Data*, 2012.

[3] Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. FLAIR: An Easy-to-Use Framework for State-of-the-Art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[4] Farhad Akhbardeh, Cecilia Ovesdotter Alm, Marcos Zampieri, and Travis Desell. Handling Extreme Class Imbalance in Technical Logbook Datasets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4034–4045, Online, August 2021. Association for Computational Linguistics.

[5] Farhad Akhbardeh, Travis Desell, and Marcos Zampieri. MaintNet: A Collaborative Open-Source Library for Predictive Maintenance Language Resources. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 7–11, Barcelona, Spain, December 2020. International Committee on Computational Linguistics.

[6] Farhad Akhbardeh, Travis Desell, and Marcos Zampieri. NLP Tools for Predictive Maintenance Records in MaintNet. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference*

on Natural Language Processing, pages 26–32, Suzhou, China, December 2020. Association for Computational Linguistics.

[7] Farhad Akhbardeh, Marcos Zampieri, Cecilia Ovesdotter Alm, and Travis Desell. Transfer Learning Methods for Domain Adaptation in Technical Logbook Datasets. In *Proceedings of the Language Resources and Evaluation Conference*, pages 4235–4244, Marseille, France, June 2022. European Language Resources Association.

[8] Sadam Al-Azani and El-Sayed El-Alfy. Using Word Embedding and Ensemble Learning for Highly Imbalanced Data Sentiment Analysis in Short Arabic Text. *Procedia Computer Science*, Vol 109:359–366, December 2017.

[9] M. Altuncu, Erik Mayer, Sophia Yaliraki, and Mauricio Barahona. From free text to clusters of content in health records: an unsupervised graph partitioning approach. *Applied Network Science*, Vol 4, Jan 2019.

[10] Enrique Amigó, Julio Gonzalo, Javier Artiles, and M. Felisa Verdejo. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12:461–486, 2008.

[11] Jesse Andrawus, John Watson, Mohammed Kishk, and Allan Adam. The Selection of a Suitable Maintenance Strategy for Wind Turbines. *Wind Engineering*, 30, 12 2006.

[12] Mikko Aulamo, Sami Virpioja, Yves Scherrer, and Jörg Tiedemann. Boosting Neural Machine Translation from Finnish to Northern Sámi with Rule-Based Backtranslation. In *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 351–356, Reykjavik, Iceland (Online), May 31–2 June 2021. Linköping University Electronic Press, Sweden.

[13] Amittai Axelrod, Xiaodong He, and Jianfeng Gao. Domain Adaptation via Pseudo In-Domain Data Selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

[14] Siddhartha Banerjee, Cem Akkaya, Francisco Perez-Sorrosal, and Kostas Tsioutsiouliklis. Hierarchical Transfer Learning for Multi-label Text Classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics ACL*, pages 6295–6300, Florence, Italy, July 2019. Association for Computational Linguistics.

[15] Abdel Bayoumi, Nicholas Goodman, Ronak Shah, Les Eisner, Lem Grant, and Jonathan Keller. Conditioned-Based Maintenance at USC-Part I: Integration of Maintenance Management Systems and Health Monitoring Systems through Historical Data Investigation. In *Proceedings of AHS International Specialists' Meeting on Condition Based Maintenance*. Citeseer, 2008.

[16] Iz Beltagy, Kyle Lo, and Arman Cohan. SciBERT: A Pretrained Language Model for Scientific Text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China, November 2019. Association for Computational Linguistics.

[17] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly, 2009.

[18] Tulika Bose, Irina Illina, and Dominique Fohr. Unsupervised Domain Adaptation in Cross-corpora Abusive Language Detection. In *Proceedings of the Ninth International Workshop on Natural Language Processing for Social Media*, pages 113–122, Online, June 2021. Association for Computational Linguistics.

[19] Connor Bowley, Marshall Mattingly, Andrew Barnas, Susan Ellis-Felege, and Travis Desell. An analysis of altitude, citizen science and a convolutional neural network feedback loop on object detection in unmanned aerial systems. *Journal of Computational Science*, Vol 34:102 – 116, 2019.

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

[21] Tommaso Caselli, Valerio Basile, Jelena Mitrović, and Michael Granitzer. HateBERT: Retraining BERT for Abusive Language Detection in English. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 17–25, Online, August 2021. Association for Computational Linguistics.

[22] Gabriela Cavaglià. Measuring corpus homogeneity using a range of measures for inter-document distance. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC'02)*, Las Palmas, Canary Islands - Spain, May 2002. European Language Resources Association (ELRA).

[23] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. Universal Sentence Encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 169–174, Brussels, Belgium, November 2018. Association for Computational Linguistics.

[24] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. LEGAL-BERT: The Muppets straight out of Law School. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online, November 2020. Association for Computational Linguistics.

[25] Hao Chen, Rui Xia, and Jianfei Yu. Reinforced Counterfactual Data Augmentation for Dual Sentiment Classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 269–278, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[26] Colin Cherry and Hongyu Guo. The Unreasonable Effectiveness of Word Representations for Twitter Named Entity Recognition. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies HLT-NAACL*, pages 735–745, Denver, Colorado, May 2015. Association for Computational Linguistics.

[27] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *International Conference on Learning Representations*, 2020.

[28] Matthias Damaschk, Tillmann Dönicke, and Florian Lux. Multiclass Text Classification on Unbalanced, Sparse and Noisy Data. In *Proceedings of the First NLPL Workshop on Deep Learning for Natural Language Processing*, pages 58–65, Turku, Finland, September 2019. Linköping University Electronic Press.

[29] Hal Daumé III. Frustratingly Easy Domain Adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[30] Renato Cordeiro de Amorim and Marcos Zampieri. Effective Spell Checking Methods Using Clustering Algorithms. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, pages 172–178, Hissar, Bulgaria, September 2013. INCOMA Ltd. Shoumen, BULGARIA.

[31] Louise Deléger, Cyril Grouin, and Pierre Zweigenbaum. Extracting medical information from narrative patient records: The case of medication-related information. *Journal of the American Medical Informatics Association*, Vol 17:555 – 558, sep 2010.

[32] Franck Dernoncourt, Ji Young Lee, and Peter Szolovits. Neural Networks for Joint Sentence Classification in Medical Paper Abstracts. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2*, pages 694–700, Valencia, Spain, April 2017. Association for Computational Linguistics.

[33] 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*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[34] Anne Dirkson and Suzan Verberne. Transfer Learning for Health-related Twitter Data. In *Proceedings of the Fourth Social Media Mining for Health Applications (#SMM4H) Workshop & Shared Task*, pages 89–92, Florence, Italy, August 2019. Association for Computational Linguistics.

[35] Abdellah El Mekki, Abdelkader El Mahdaouy, Ismail Berrada, and Ahmed Khoumsi. Domain Adaptation for Arabic Cross-Domain and Cross-Dialect Sentiment Analysis from Contextualized Word Embedding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2824–2837, Online, June 2021. Association for Computational Linguistics.

[36] Hady Elsahar, Elena Demidova, Simon Gottschalk, Christophe Gravier, and Frederique Laforest. Unsupervised Open Relation Extraction. In *The Semantic Web: European Semantic Web Conference (ESWC) 2017 Satellite Events*, pages 12–16, Cham, 2017. Springer International Publishing.

[37] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.

[38] Marzieh Fadaee, Arianna Bisazza, and Christof Monz. Data Augmentation for Low-Resource Neural Machine Translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 567–573, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[39] Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. A Survey of Data Augmentation Approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online, August 2021. Association for Computational Linguistics.

[40] Chris Fraley and Adrian E. Raftery. How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis. *The Computer Journal*, 41(8):578–588, 1998.

[41] Markus Freitag, Ricardo Rei, Nitika Mathur, Chi-kiu Lo, Craig Stewart, George Foster, Alon Lavie, and Ondřej Bojar. Results of the WMT21 Metrics Shared Task: Evaluating Metrics with Expert-based Human Evaluations on TED and News Domain. In *Proceedings of the Sixth Conference on Machine Translation*, pages 733–774, Online, November 2021. Association for Computational Linguistics.

[42] Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. Soft Contextual Data Augmentation for Neural Machine Translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5539–5544, Florence, Italy, July 2019. Association for Computational Linguistics.

[43] Tianyu Gao, Adam Fisch, and Danqi Chen. Making Pre-trained Language Models Better Few-shot Learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online, August 2021. Association for Computational Linguistics.

[44] Hardik A. Gohel, Himanshu Upadhyay, Leonel Lagos, Kevin Cooper, and Andrew Sanzetenea. Predictive maintenance architecture development for nuclear infrastructure using machine learning. *Nuclear Engineering and Technology*, 52(7):1436–1442, 2020.

[45] Jiechao Guan, Zhiwu Lu, Tao Xiang, Aoxue Li, An Zhao, and Ji-Rong Wen. Zero and Few Shot Learning With Semantic Feature Synthesis and Competitive Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(7):2510–2523, 2021.

[46] Kim Hammar, Shatha Jaradat, Nima Dokoohaki, and Mihhail Matskin. Deep Text Mining of Instagram Data without Strong Supervision. In *International Conference on Web Intelligence (WI)*, pages 158 – 165, Santiago, Chile, dec 2018.

[47] Michael Heilman and Nitin Madnani. ETS: Domain Adaptation and Stacking for Short Answer Scoring. In *Second Joint Conference on Lexical and Computational Semantics (*SEM),*

Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013), pages 275–279, Atlanta, Georgia, USA, June 2013. Association for Computational Linguistics.

[48] Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. ClinicalBERT: Modeling Clinical Notes and Predicting Hospital Readmission. *arXiv preprint arXiv:1904.05342*, 2019.

[49] Vebjørn Isaksen and Björn Gambäck. Using Transfer-based Language Models to Detect Hateful and Offensive Language Online. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 16–27, Online, November 2020. Association for Computational Linguistics.

[50] Anil Kumar Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31:651–666, 2010.

[51] Gabriel Jarry, Daniel Delahaye, Florence Nicol, and Eric Feron. Aircraft atypical approach detection using functional principal component analysis. *Journal of Air Transport Management*, 84:101787, 2020.

[52] Philip Kerr. Adaptive learning. *English Language Teaching (ELT) Journal*, 70(1):88–93, 10 2015.

[53] Donghwa Kim, Deokseong Seo, Suhyoun Cho, and Pilsung Kang. Multi-co-training for document classification using various document representations: TF-IDF, LDA, and Doc2Vec. *Information Sciences*, Vol 477:15 – 29, 2019.

[54] Yoon Kim. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics.

[55] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[56] Sosuke Kobayashi. Contextual Augmentation: Data Augmentation by Words with Paradigmatic Relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[57] Stavros Konstantinidis. Computing the edit distance of a regular language. *Information and Computation*, 205(9):1307–1316, 2007.

[58] Varun Kumar, Ashutosh Choudhary, and Eunah Cho. Data Augmentation using Pre-trained Transformer Models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26, Suzhou, China, December 2020. Association for Computational Linguistics.

[59] Varun Kumar, Hadrien Glaude, Cyprien de Lichy, and Wlliam Campbell. A Closer Look at Feature Space Data Augmentation for Few-Shot Intent Classification. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 1–10, Hong Kong, China, November 2019. Association for Computational Linguistics.

[60] Po-Nien Kung, Tse-Hsuan Yang, Yi-Cheng Chen, Sheng-Siang Yin, and Yun-Nung Chen. Zero-Shot Rationalization by Multi-Task Transfer Learning from Question Answering. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2187–2197, Online, November 2020. Association for Computational Linguistics.

[61] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for Self-supervised Learning of Language Representations. In *International Conference on Learning Representations*, 2020.

[62] Kenton Lee, Luheng He, and Luke Zettlemoyer. Higher-Order Coreference Resolution with Coarse-to-Fine Inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2*, pages 687–692, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[63] Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.

[64] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics.

[65] Junyi Jessy Li and Ani Nenkova. Addressing Class Imbalance for Improved Recognition of Implicit Discourse Relations. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 142–150, Philadelphia, PA, U.S.A., June 2014. Association for Computational Linguistics.

[66] Shoushan Li, Shengfeng Ju, Guodong Zhou, and Xiaojun Li. Active Learning for Imbalanced Sentiment Classification. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 139–148, Jeju Island, Korea, July 2012. Association for Computational Linguistics.

[67] Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. Dice Loss for Data-imbalanced NLP Tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 465–476, Online, July 2020. Association for Computational Linguistics.

[68] Chin-Yew Lin. ROUGE: A Package for Automatic Evaluation of Summaries. In *Proceedings of the Association for Computational Linguistics (ACL) Workshop: Text Summarization Braches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[69] Junyang Lin, Qi Su, Pengcheng Yang, Shuming Ma, and Xu Sun. Semantic-Unit-Based Dilated Convolution for Multi-Label Text Classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4554–4564, Brussels, Belgium, October 2018. Association for Computational Linguistics.

[70] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*, 2019.

[71] Shayne Longpre, Yu Wang, and Chris DuBois. How Effective is Task-Agnostic Data Augmentation for Pretrained Transformers? In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4401–4411, Online, November 2020. Association for Computational Linguistics.

[72] Max Louwerse, Zhiqiang Cai, Xiangen Hu, Matthew Ventura, and Patrick Jeuniaux. Cognitively Inspired Nlp-based Knowledge Representations: Further Explorations of Latent Semantic Analysis. *International Journal on Artificial Intelligence Tools*, 15:1021–1040, 12 2006.

[73] Xiaofei Ma, Peng Xu, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. Domain Adaptation with BERT-based Domain Classification and Data Selection. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 76–83, Hong Kong, China, November 2019. Association for Computational Linguistics.

[74] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[75] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*, volume 39. Cambridge University Press, USA, 2008.

[76] Manolis Maragoudakis, Katia Kermanidis, Aristogiannis Garbis, and Nikos Fakotakis. Dealing with Imbalanced Data using Bayesian Techniques. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy, May 2006. European Language Resources Association (ELRA).

[77] Dominic Masters and C. Luschi. Revisiting Small Batch Training for Deep Neural Networks. *ArXiv*, abs/1804.07612, 2018.

[78] J.J. McArthur, Nima Shahbazi, Ricky Fok, Christopher Raghubar, Brandon Bortoluzzi, and Aijun An. Machine learning and BIM visualization for maintenance issue classification and enhanced data collection. *Advanced Engineering Informatics*, 38:101 – 112, 2018.

[79] Sachin Mehta, Marjan Ghazvininejad, Srinivasan Iyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. Delight: Deep and Light-weight Transformer. In *International Conference on Learning Representations*, 2021.

[80] Oren Melamud, Mihaela Bornea, and Ken Barker. Combining Unsupervised Pre-training and Annotator Rationales to Improve Low-shot Text Classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3884–3893, Hong Kong, China, November 2019. Association for Computational Linguistics.

[81] Carol Midgley. *Goals, goal structures, and patterns of adaptive learning.* Routledge, 2014.

[82] Tomas Mikolov, Kai Chen, G. Corrado, and J. Dean. Efficient Estimation of Word Representations in Vector Space. In *International Conference on Learning Representations (ICLR)*, 2013.

[83] George A. Miller. WordNet: A Lexical Database for English. *Commun. ACM*, 38:39–41, 1992.

[84] Robert C. Moore and William Lewis. Intelligent Selection of Language Model Training Data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[85] Rutu Mulkar-Mehta, Jerry Hobbs, and Eduard Hovy. Granularity in Natural Language Discourse. In *Proceedings of the Ninth International Conference on Computational Semantics*, IWCS '11, page 360–364, USA, 2011. Association for Computational Linguistics.

[86] Vinod Nair and Geoffrey E. Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27 th International Conference on Machine Learning*, Haifa, Israel, 2010. ICML.

[87] Itsuki Okimura, Machel Reid, Makoto Kawano, and Yutaka Matsuo. On the Impact of Data Augmentation on Downstream Performance in Natural Language Processing. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP*, pages 88–93, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[88] Sinno Jialin Pan and Qiang Yang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[89] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[90] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. PMLR, 2013.

[91] Jon Patrick and Min Li. A Cascade Approach to Extracting Medication Events. In *Proceedings of the Australasian Language Technology Association Workshop 2009*, pages 99–103, Sydney, Australia, December 2009.

[92] Martin Pech, Jaroslav Vrchota, and Jiří Bednář. Predictive Maintenance and Intelligent Sensors in Smart Factory: Review. *Sensors*, 21(4), 2021.

[93] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[94] Nanyun Peng and Mark Dredze. Multi-task Domain Adaptation for Sequence Tagging. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 91–100, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[95] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[96] Mahesh Pophaley and R. Vyas. Choice criteria for maintenance strategy in automotive industries. *International Journal of Management Science and Engineering Management*, 5:446–452, 05 2013.

[97] Alec Radford, Jeff Wu, R. Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. In *OpenAI Blog*, page 8, 2019.

[98] Tharindu Ranasinghe and Marcos Zampieri. Multilingual Offensive Language Identification with Cross-lingual Embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5838–5844, 2020.

[99] Radim Rehurek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the Language Resources and Evaluation (LREC) workshop on new challenges for NLP frameworks*, 2010.

[100] Radim Rehurek and Petr Sojka. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2), 2011.

[101] Alan Ritter, Mausam Mausam, Oren Etzioni, and Sam Clark. Open Domain Event Extraction from Twitter. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1104-1112:1104 – 1112, aug 2012.

[102] Alexey Romanov and Chaitanya Shivade. Lessons from Natural Language Inference in the Clinical Domain. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1586–1596, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[103] Víctor M. Sánchez-Cartagena, Miquel Esplà-Gomis, Juan Antonio Pérez-Ortiz, and Felipe Sánchez-Martínez. Rethinking Data Augmentation for Low-Resource Neural Machine Translation: A Multi-Task Learning Approach. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8502–8516, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[104] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108, 2019.

[105] K. Santhisree and A. Damodaram. CLIQUE: Clustering based on density on web usage data: Experiments and test results. *2011 3rd International Conference on Electronics Computer Technology*, 4:233–236, 2011.

[106] Guergana K. Savova, James J. Masanz, Philip V. Ogren, Jiaping Zheng, Sunghwan Sohn, Karin Kipper Schuler, and Christopher G. Chute. Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications. *Journal of the American Medical Informatics Association : JAMIA*, 17 5:507–13, 2010.

[107] Dinghan Shen, Martin Renqiang Min, Yitong Li, and Lawrence Carin. Learning Context-Sensitive Convolutional Filters for Text Processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1839–1848, Brussels, Belgium, October 2018. Association for Computational Linguistics.

[108] Michael Short and John Twiddle. An Industrial Digitalization Platform for Condition Monitoring and Predictive Maintenance of Pumping Equipment. *Sensors*, 19(17), 2019.

[109] Borbála Siklósi, Attila Novák, and Gábor Prószéky. Context-Aware Correction of Spelling Errors in Hungarian Medical Documents. In Adrian-Horia Dediu, Carlos Martín-Vide, Ruslan Mitkov, and Bianca Truthe, editors, *Statistical Language and Speech Processing*, pages 248–259, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[110] Richard Sproat, Alan W. Black, Stanley Chen, Shankar Kumar, Mari Ostendorf, and Christopher Richards. Normalization of non-standard words. *Computer Speech & Language*, 15(3):287 – 333, 2001.

[111] Cong Sun and Zhihao Yang. Transfer Learning in Biomedical Named Entity Recognition: An Evaluation of BERT in the PharmaCoNER task. In *Proceedings of The 5th Workshop on BioNLP Open Shared Tasks*, pages 100–104, Hong Kong, China, November 2019. Association for Computational Linguistics.

[112] Mirac Suzgun, Yonatan Belinkov, and Stuart M. Shieber. On Evaluating the Generalization of LSTM Models in Formal Languages. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2019*, pages 277–286, 2019.

[113] Ludovic Tanguy, Nikola Tulechki, Assaf Urieli, Eric Hermann, and Céline Raynal. Natural language processing for aviation safety reports: From classification to interactive analysis. *Computers in Industry*, 78:80–95, 2016.

[114] Jie Tao and Xing Fang. Toward multi-label sentiment analysis: a transfer learning based approach. *Journal of Big Data*, 7:1, 01 2020.

[115] Harish Tayyar Madabushi, Elena Kochkina, and Michael Castelle. Cost-Sensitive BERT for Generalisable Sentence Classification on Imbalanced Data. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 125–134, Hong Kong, China, November 2019. Association for Computational Linguistics.

[116] Zhanna Terechshenko, Fridolin Linder, Vishakh Padmakumar, Michael Liu, Jonathan Nagler, Joshua A Tucker, and Richard Bonneau. A Comparison of Methods in Political Science Text Classification: Transfer Learning Language Models for Politics. *Available at SSRN*, 2020.

[117] Antoine J.-P. Tixier, Matthew R. Hallowell, Balaji Rajagopalan, and Dean Bowman. Automated content analysis for construction safety: A natural language processing system to extract precursors and outcomes from unstructured injury reports. In *Automation in Construction*, 2016.

[118] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147, 2003.

[119] Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-Read Students Learn Better: On the Importance of Pre-training Compact Models. *arXiv: Computation and Language*, 2019.

[120] Francis M. Tyers. Rule-Based Augmentation of Training Data in Breton-French Statistical Machine Translation. In *Proceedings of the 13th Annual conference of the European Association for Machine Translation*, Barcelona, Spain, May 14–15 2009. European Association for Machine Translation.

[121] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[122] Konstantin Vorontsov, Oleksandr Frei, Murat Apishev, Peter Romov, and Marina Dudarenko. BigARTM: Open Source Library for Regularized Multimodal Topic Modeling of Large Collections. In *International Conference on Analysis of Images, Social Networks and Texts (AIST)*, 2015.

[123] Jin Wang, Liang-Chih Yu, K. Robert Lai, and Xuejie Zhang. Dimensional Sentiment Analysis Using a Regional CNN-LSTM Model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 225–230, Berlin, Germany, August 2016. Association for Computational Linguistics.

[124] Yaoshu Wang, Jianbin Qin, and Wei Wang. Efficient Approximate Entity Matching Using Jaro-Winkler Distance. In *WISE*, 2017.

[125] Jason Wei and Kai Zou. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China, November 2019. Association for Computational Linguistics.

[126] Jingyun Xu, Yi Cai, Xin Wu, Xue Lei, Qingbao Huang, Ho fung Leung, and Qing Li. Incorporating context-relevant concepts into convolutional neural networks for short text classification. *Neurocomputing*, 386:42 – 53, 2020.

[127] 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.

[128] Meliha Yetisgen-Yildiz, Cosmin Bejan, and Mark Wurfel. Identification of Patients with Acute Lung Injury from Free-Text Chest X-Ray Reports. In *Proceedings of the 2013 Workshop on Biomedical Natural Language Processing*, pages 10–17, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[129] Jiacheng Zhang, Huanbo Luan, Maosong Sun, Feifei Zhai, Jingfang Xu, Min Zhang, and Yang Liu. Improving the Transformer Translation Model with Document-Level Context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 533–542, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[130] Zhilu Zhang and Mert R. Sabuncu. Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 8792–8802, Red Hook, NY, USA, 2018. Curran Associates Inc.

[131] Ran Zhou, Xin Li, Ruidan He, Lidong Bing, Erik Cambria, Luo Si, and Chunyan Miao. MELM: Data Augmentation with Masked Entity Language Modeling for Low-Resource NER.

In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2251–2262, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[132] Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. Transfer Learning for Low-Resource Neural Machine Translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas, November 2016. Association for Computational Linguistics.