From description to code: a method to predict maintenance codes from maintainer descriptions

Srini Anand[1], **Rob Keefer**[2]
Illumination Works, LLC, Dayton, OH, USA
[1]Corresponding author
E-mail: [1]*srini.anand@ilwllc.com*, [2]*rob.keefer@ilwllc.com*

Received 5 July 2022; received in revised form 13 September 2022; accepted 1 October 2022
DOI https://doi.org/10.21595/marc.2022.22798

Abstract. Aircraft maintenance crews enter the actions performed, the time required to complete the actions, and process followed to complete the action into a system of record that may be used to support future important operational decisions such as part inventory and staffing levels. Unfortunately, the actions performed by maintainers may not align with structured, predetermined codes for such actions. This discrepancy combined with an overabundance of structured codes has led to incorrect and polluted maintenance data that cannot be used in decision making. Typically, the unstructured textual fields accurately record the maintenance action, but are inaccessible to common reporting approaches. The textual fields can be used to cleanse the structured fields, thereby making more data available to support operational decision making. This paper introduces a natural language processing pipeline to predict C-17 US Air Force maintenance codes from an unstructured, shorthand text record. This research aims to cleanse problematic structured fields for further use in operational efficiency and asset reliability measures. Novel use of text processing, extraction, clustering, and classification approaches was employed to develop a natural language processing pipeline suited to the peculiarities of short, jargon-based text. The pipeline evaluates the frequency of structured field values within the datase and selects an appropriate machine learning model to optimize the predictive accuracy. Three different predictive methods were investigated to determine an optimal approach: a Logistic Regression Classifier, a Random Forrest Classifier, and Unsupervised techniques. This pipeline predicted structured fields with an average accuracy of 93 % across the five maintenance codes.

Keywords: aircraft maintenance, maintenance codes, maintenance text, jargon-based text, natural language processing pipeline, NLP, tokenization, stemming, lemmatization, entity extraction, data cleansing.

1. Introduction

In recent years, natural language processing (NLP) has made significant advances across various tasks. These improvements were facilitated by improvements in models that predict words and sentences from surrounding context [1, 2]. These models are often successful when trained on "standard" text from English news or other literature. However, text encountered in technical applications such as maintenance and medical records differs significantly from these benchmarks. When traditional methods are applied to these data sets, performance often drops significantly [3, 4].

Maintenance staff enters maintenance records with the common assumption that a human will refer back to them and, therefore, a human will interpret them. This assumption leads to acronyms and jargon-filled text records that are difficult for a machine to process. Common problems with maintenance data include [5]:

– Technicians describe problems informally, leading to inconsistent/inaccurate data.
– Data is incomplete (lack of root cause in particular).
– Unstructured data is difficult to use in future diagnosis.

Aircraft maintenance records are no different from those reported in other studies. When an aircraft has an issue, a maintenance work order is entered into the maintenance management system. During aircraft downtime, maintainers diagnose and remediate the issue. The maintainer records the actions taken with coded fields and unstructured text upon completing the work order.

In the data used for this research, the unstructured text represented the work performed more accurately than the structured fields. SMEs estimate that the coded fields are accurate only 60 % of the time. Maintainers often enter the wrong code but accurately describe the action taken in the unstructured text. Unfortunately, downstream systems only use structured fields to inform critical decision-making, such as staffing and inventory needs.

Thus, we propose a human-in-the-loop method for predicting and cleansing the structured codes based on the unstructured text.

2. Maintenance data overview

Asset maintenance involves various stakeholders, such as asset owners, operators, and service providers. We call those who maintain these assets "maintainers". Typically, maintainers are technicians trained in the domain knowledge required to diagnose, repair, and maintain assets. This training includes clearly and concisely communicating with peers through common mental models and standard practices [6].

This communication is documented as both structured values and unstructured text. The unstructured text is usually in a free text format and includes jargon, abbreviations, and implicit relationships planned for human consumption.

In the case of aircraft maintenance, a pilot, crew member, or inspector may discover an issue that requires attention. The issue is documented as a maintenance work order. The details of a work order are entered into a maintenance management system and used to prioritize and schedule work items. The structured fields within the work order records are vital to operational efficiency, prediction of asset reliability, and identification of potential failures. In the case of the data addressed in this research, the structured fields were found to be incomplete, inaccurate, or lacking enough detail to provide accurate reports for use in decision-making.

The free text entered into the maintenance management system deviates from standard English text in a number of ways. The work order text tends to be much smaller and reflects a jargon-based shorthand notation rather than English text. Many of the words are domain-specific, including abbreviations or acronyms created by groups of maintainers. Punctuation does not follow conventional syntax and is typically used for domain-specific purposes. Others have reported similar issues with maintenance data [4, 7].

These deviations from the norm lead to gaps in the typical assumptions regarding natural language processing (NLP). While the number of maintenance records can be similar to the number of documents in an NLP corpus, the actual text provides minimal support for sentence structure, parts of speech, or other contexts often used in NLP methods. Thus, out-of-the-box preprocessing pipelines require modification.

One approach adopted the Python Natural Language Toolkit (NLTK) to include methods for replacing all forms of punctuation with a unique punctuation token and all numbers with an identification code token [7, 8]. Other approaches seek to bring structure to the unstructured data to bootstrap machine learning algorithms. In [8], the authors describe a method to identify elements with a common semantic class. Once a critical number of records have been processed, an unsupervised machine learning approach processed the remaining records.

In [5], an approach is proposed in which humans and an NLP-based system partner together to annotate unclassified text records. To maximize expected information gain, the authors propose a method for human-in-the-loop, automated tagging of concepts, essentially bringing structure to the unstructured data. Subject Matter Experts (SMEs) tagged unstructured data filtered through ML algorithms. This hybrid approach improved the classification of narratives, though it relied on humans for verification and disambiguation.

The approach presented in [7] applied information extraction techniques to helicopter maintenance data. Again, they identified similar problems in the narrative data, such as domain-specific jargon, abbreviations, and acronyms. This approach implemented partial parsing using Abney's techniques for handling nested syntactic structure to capture meaningful portions of data from the text [10].

Another study reported using NLP techniques with Marine V-22 maintenance data. In [8], the same problems are described with the maintenance narrative data already identified. The authors presented the results of modifications made to open-source tools that accommodated the domain-specific jargon, acronyms, and abbreviations. The approach incorporated a pipeline of parsing algorithms – three custom parsers followed by a default value. This approach improved the accuracy of the parsing mechanism.

3. Data analysis – exploratory data analysis

The maintenance records for this study included six structured fields and two unstructured text fields. The structured fields When Discovered Code (WDC), How Malfunctioned Code (HMC), Type Maintenance Code (TMC), Action Taken Code (ATC), and Work Unit Code (WUC).

The free text fields included a discrepancy record and a related maintenance corrective action record. The discrepancy records are short text typically consisting of a pilot or inspector's description of a quality problem on a specific aircraft. Maintenance action records include references to the procedures followed to remedy the discrepancy, most of which follow prescribed methods for remediation. For example, a discrepancy may be reported as "Turbulent landing. Suspect LT tire". The related maintenance action record may be reported as "R2 LT tire IAW TO-001-8327". Identifying the relationship between "LT tire" in both records is essential for automating data integrity checks and improving the efficiency of maintenance documentation.

Analysis of 222,614 maintenance action records identified 1,906,990 unique token sequences (i.e., n-grams) 1 to 4 tokens long. These one-word phrases ranged in size from 2 to 86 characters, most of which were between 2 and 20 characters. Of the one-word n-grams, 58 % were not English words (44,275 out of 76,271). Many of these non-English tokens were acronyms, codes (e.g., status, task order), misspellings, shortened phrases, and concatenated words.

Both Python Spacy and NLTK toolkits assume that the body of text for evaluation and extraction is longer than an abbreviated sentence found in maintenance action records. Since the maintenance records in this study utilize domain-specific terms and acronyms, a robust tagged or labeled data set is required for training entity extraction.

Extracted entities must be classified to identify relationships. Classifying the coded fields WUC, HOW MAL, and ATC based on the maintainer's narrative fields of a maintenance action poses a unique set of challenges because the narratives are short, acronym dense, and domain-specific. Fortunately, many maintenance actions are similar and provide context for similarity measures.

4. Approach

The maintenance code prediction engine was implemented as an NLP and ML-based pipeline, as illustrated in Fig. 1. The discrepancy and the maintainer's corrective action narrative pass through a custom tokenizer in this process. This tokenizer accommodates the characteristics of the jargon-rich text. The tokens were normalized by algorithms that identify synonyms and lemmas. Finally, these normalized tokens were used as input to a Term Frequency Inverse Document Frequency (TFIDF) model [11] for use by machine learning (ML) based classifier algorithms.

This approach facilitated the investigation into multiple approaches for processing maintenance narratives, leading to an optimal solution for the available data. Various text processing, extraction, clustering, and classification approaches were used to identify a process that led to a prediction accuracy of 93 % across the five predicted codes.

The entire pipeline was developed using Python 3 and open-source libraries. The ML classifier utilized supervised learning approaches. Over 1 million records were used to train the model, and over 300,000 records were used to test the model's accuracy.

The maintenance code prediction methods utilize the classification results and predict four codes for each of the five coded fields (ATC, HMC, WDC, TMC, and WUC).

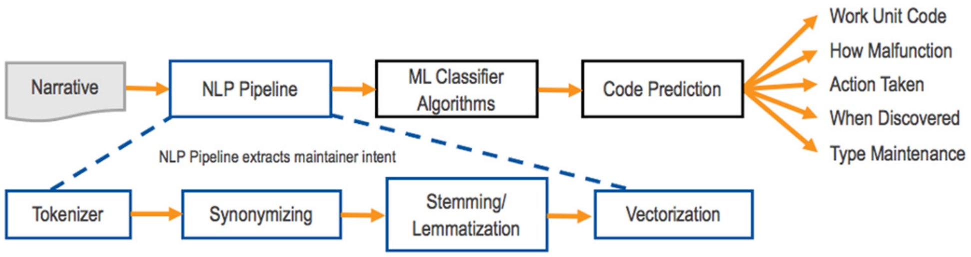


Fig. 1. Maintenance code prediction engine pipeline

4.1. Classification of data

The data used in this research contains over 1 million C-17 US Air Force maintenance data records. Each record contains the discrepancy narrative, maintenance narrative, and the coded fields assigned by the maintainer. Within these 1 million records, there exist 2,203 unique WUCs, 35 unique ATCs, 441 unique HMCs, 33 unique TMCs, and 35 unique WDCs. Thus, the prediction of an accurate WUC required more effort than training the other codes due to the significantly smaller value range to classify.

This code consists of five alphabetic and numeric characters and is used to identify the system, subsystem, and component on which the work was performed. The first two digits represent the system. The third digit adds another level of detail to identify the subsystem. The complete 5-digits provide the most detail by including the specific component worked on. Unfortunately, the WUC entered for a specific action performed often misrepresents the work performed. There are many causes for this, including user error, misunderstanding of the meaning of the codes, and misalignment of work performed with the code. Thus, when a work order is entered with 2- or 3-digit accuracy, it is difficult to draw specific conclusions for planning purposes.

All structured fields are used in operational reports, but the WUC is used predominately for inventory management, crew size estimates, and other important measures that affect operational decisions and efficiencies. Training ATC, HMC, TMC, and WDC records required significantly less processing than WUC to obtain favorable prediction results. Thus, much of our work is dedicated to processing 2-, 3-, and 5-digit WUCs.

Training on the entire dataset yielded a model with a WUC classification accuracy of ~70 %. Closer inspection revealed that certain WUCs occur at a higher frequency than others. Upon further investigation, it was discovered that only 612 unique WUCs occur in more than 100 instances, while the remaining 1,591 WUCs appear less than 100 times throughout the dataset. Such frequency discrepancies can lead to low accuracy levels in classifiers. With this in mind, the low-frequency WUCs were removed from the training data, which produced a significant improvement over the base to 89 % classification accuracy.

A second model was developed to address the remaining 1,591 low-frequency WUCs. The training set for this model removed the high-frequency WUCs and trained only low-frequency WUCs using the same classification methods with different parameters (see Table 1). Note that a Randomized Grid Search was used to identify the hyperparameters for each Stochastic Gradient Descent (SGD) model. (See the SGD section below for more details.)

This model produced a classification accuracy of 89 % for the low-frequency WUCs.

This discovery led to the combination of both models. The combination approach yielded a cohesive prediction tool with an overall accuracy of 90 %.

The following approach was used to combine the two models:

– The WUC for the low-density narratives was defined to be sparse. The high-frequency model was then trained on this new dataset such that a test narrative would be classified as either a high-density WUC or sparse.

– If the top prediction for the narrative was identified as sparse, the low-frequency model was used to further classify the narrative as a low-frequency WUC.

An enhanced NLP Pipeline and three different predictive methods were investigated to determine an optimal approach: a Logistic Regression Classifier [12], a Random Forrest Classifier [13], and Unsupervised techniques. These methods are outlined in the following sections.

4.2. Enhanced NLP pipeline

Early attempts to predict coded fields from the maintainer narrative used default settings for the vectorizer and classifier within the Python Scikit-learn libraries. The vectorization and classification steps were enhanced to form a consistent and high-performing Natural Language Processing pipeline. This pipeline is comprised of the following components: a tokenizer, a synonymizer, lemmatization, and vectorization. See Fig. 2.

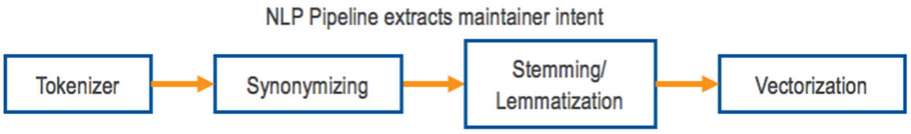


Fig. 2. Enhanced natural language processing pipeline

The tokenizer was enhanced with a token splitting pattern that split words but kept key phrases intact. Information-rich phrases such as Task Order (TO) numbers and Part numbers must remain untouched by the tokenizer.

Minor variations in tense can affect the outcome of an NLP pipeline. Maintainers use different forms of the same word in their documentation. For example, the root word perform could have other forms such as performs, performing, or performed. The intent is the same, but the vectorizer would treat these tokens as different words, increasing the number of unique tokens. The number of unique tokens increases the vector space, and the similar forms of the same token lead to a sparse vector space, thus lowering the ultimate accuracy of the classifiers. The process of Stemming and Lemmatization reduced these tokens to their root word, which in turn minimizes the list of unique words and keeps the vector space to a proper size.

Another enhancement to the pipeline was the addition of domain-specific stop words derived from the maintainer narrative. For example, the acronyms IAW (in accordance with) and TCTO (Time Compliance Technical Orders) occur regularly throughout the narrative text. While these are useful to locate TO numbers, they do not add additional meaning and can confuse the results of predicted coded fields.

A Doc2Vec vectorizer experiment showed no enhancement or increase in predictive capability relative to the TF-IDF vectorizer. Thus, the current implementation of the NLP pipeline includes the TF-IDF vectorizer.

A Grid Search optimizer [14] was used to obtain the most effective values for alpha and max_iter. This function extensively searches for the most effective hyperparameters' values within a specified numerical range. Specific values for alpha and max_iter were provided, and the resulting hyperparameter sweep generated a classification model for the WUC.

However, the computational time for the hyperparameter check was greater than 17 hours, or 1,026 minutes.

Further research into the available optimizers was conducted to develop a more efficient modeling process. The Randomized Grid Search [15] yielded similar results with considerably less computational time. Similar to SGD, Randomized Grid Search also accepts a numerical range of trial values, but unlike the conventional grid search, it randomizes them for a given number of iterations. This method allows for a more flexible, less computationally intensive hyperparameter search. Additionally, through random searching, the function can generate highly accurate models. For the purposes of the SGD classifier, the Randomized Grid Search produced a set of hyperparameters with similar accuracy to the conventional Grid Search. It found these values within a fraction of the computational time, approximately 105 minutes.

After stabilizing on values for the WUC, the Randomized Grid Search technique was used to find values for the other four coded fields. The Randomized Grid Search using 10,000 iterations is a favorable option, as it drastically lowers computational time while producing a highly accurate model.

4.3. Logistic regression classifier

Logistic regression predicts the probability of different outcomes. Unlike fields with binary outcomes (True/False, like/dislike, etc.), coded maintenance fields have multiple possible outcomes. The outcomes for each field fall into a definitive list of categorical values. As a result, a multinomial logistic regression approach was used. This method iterates over the data to determine the influence that an attribute has on determining the probability of the outcome. The output is the likelihood of each possible outcome based on the input data.

The Logistic Regression classifier requires the input of various hyperparameter values. Tuning or optimizing hyperparameters in machine learning models is a fundamental yet non-trivial problem; selected values must generate high accuracy models while minimizing training time. Optimization of two hyperparameter values for the SGD classifier was explored:

– Alpha: controls the regularization term and learning rate.
– Max_iter: maximum number of epochs, or number of passes through training data.

All other hyperparameters were set to default values.

4.4. Random forest classifier

Decision Trees are a popular method of classification, which use decisions about the data to create a tree of options from which to draw predictions. Decision Trees start with root nodes that are hierarchically split into branches based on features within the data and end with leaves, which are comprised of subsets of data that can no longer be divided. These end leaves contain the predicted values for input data. Random Forests are an ensemble method using multiple decision trees to find the best-predicted value for input data. This ensemble technique avoids overfitting the model to training data and thus tends to outperform individual Decision Trees.

A Random Forest classifier was used over multiple iterations to create several models for a branched classification approach. The Random Forest classifier was trained with the narratives and their system-level (two-digit) WUC family. Narratives were then classified from the system-level to subsystem-level (three-digit) WUC families. Lastly, the narratives were classified into specific five-digit WUC classes.

Note that a model was created for each two-digit and three-digit WUC family of the high-frequency WUCs. After training, test data was used by the first system-wide classifier to predict a two-digit WUC family for the test narrative. The resulting system-level WUC was used to classify the narrative into a three-digit model. This process uses the predicted three-digit family's model to classify each narrative to its full five-digit WUC.

This branched classification approach allows for computationally intensive training for extensive datasets in a relatively short period (~30-40 minutes) by dividing classification tasks into a larger number of detailed models rather than creating one large overarching model.

Preliminary results indicate that the branched approach predictions align with the ground truth WUC more than a model created using one overarching Random Forest classifier without branching models. These results demonstrate that the two and three-digit models have accuracy ratings greater than 90 %. Unfortunately, the five-digit WUC predictions drop to ~ 55 %. These models may require different parameters or data cleansing techniques to distinguish between narratives within WUC families easily.

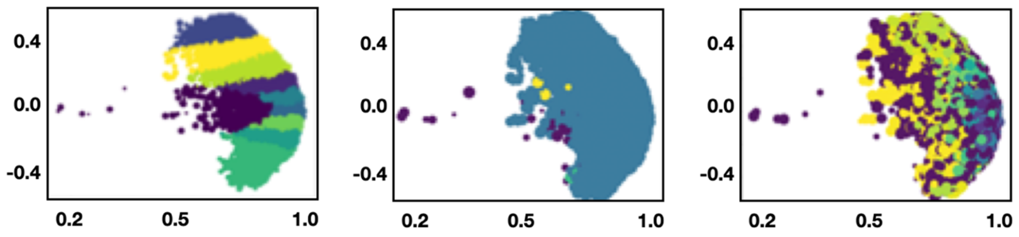
4.5. Unsupervised machine learning clustering techniques

Unsupervised learning is a method of machine learning in which no labels are present, and no training data is used. Instead, raw data is used to detect patterns within a large dataset. One such technique is clustering, in which the model determines differentiating features of data points and then creates groups, or clusters, of similar data points.

Different clustering techniques were utilized to identify functional clusters within the maintenance data. This exercise aimed to identify a method of classifying unlabeled or mislabeled narratives into the correct WUC family. This approach was comprised of the following steps:

1) Input the dataset into the model, including narrative and WUC labels.
2) Process data (tokenize, lemmatize, and vectorize narratives) for clustering.
3) Enrich the vectors by applying the TruncatedSVD dimensionality reduction method.
4) Cluster data into best fit groups.
5) Test cluster accuracy by predicting clusters for individual WUCs.

An ideal outcome would have been that the number of clusters found was equal to the number of unique WUCs in the dataset. Three different clustering algorithms were evaluated for this effort – k-means, DBSCAN, and OPTICS. These algorithms vary in approach and ability to find appropriate groupings.



a) K-Means b) DBSCAN c) OPTICS

Fig. 3. Clustering 0373 WUCs

Fig. 3 shows the output from the different approaches and their ability to discern narratives of WUC from one another. The OPTICS algorithm produced the best result, though only 34 % of the WUCs were correctly grouped together. Overall, this approach did not improve the WUC classification accuracy significantly. It also demonstrates that the narratives for some of the WUC families are very close to one another, leading to difficulty distinguishing one WUC from the other.

5. Results

The accuracy of the Maintenance Code Prediction Engine averages 93 % accuracy with the top four predictions across all five coded fields (ATC, HOW MAL, WDC, TMC, and WUC).

The Maintenance Code Prediction Engine averages 93 % accuracy with the top four predictions across all five coded fields. Fig. 4 illustrates the accuracy for each of the five coded fields. The WUC is the lowest-performing at 90 % accuracy in the top four predictions. The TM code is the best, with 97 % accuracy.

The average drops to 76 % across all five fields when only the top predicted code is considered. A core contributor to the lower accuracy of the top predicted code is the starting inaccuracy of the training data. It is believed that as the error rate in coded fields is reduced, the accuracy of this top value will improve.

The first column in Table 1 demonstrates the accuracy for each coded field when the top predicted value is used. Each column in the table contains the accuracy as the number of predicted values increases. The right column demonstrates the highest level of accuracy, with an average of 93 % across the five coded fields.

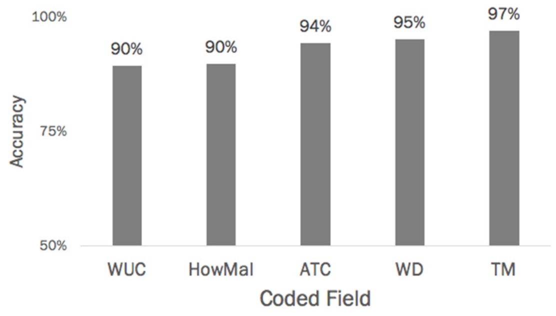


Fig. 4. Accuracy of top four predicted codes across five coded fields

Note that once the maintenance code prediction engine has been trained, it produces predictions with sub-second performance. Performance was tested with both the mobile app and the web application. Both reported sub-second responses for the predicted fields.

Table 1. Accuracy for each coded field with increasing number of predicted values

	1	2	3	4
WUC	72 %	81 %	86 %	90 %
How Mal	78 %	85 %	88 %	90 %
ATC	79 %	88 %	92 %	94 %
WD	75 %	87 %	92 %	95 %
TM	78 %	89 %	95 %	97 %

6. Discussion

The results obtained from this effort demonstrate that maintenance narratives contain sufficient information to predict coded fields used in maintenance systems. Table 1 demonstrates the levels of accuracy achieved for each field and the increase in accuracy as more predictions are considered.

This approach used stemming and lemmatization techniques to reduce the data's sparsity and increase similarity with the coded field descriptions. However, these techniques cannot recognize synonyms leading to a false reduction of similarity. In future efforts, we intend to extend this approach by utilizing word-embedding models trained to recognize synonyms, homonyms (the same word with a different meaning in varying contexts), hypernyms, and hyponyms. This modification will ensure that the resulting numerical representations closely resemble the original text and allow a high degree of match with the coded field descriptions.

Recall that the language used by the maintainers contains 58 % English phrases. The remaining text contains non-English words and phrases such as acronyms, codes (such as part numbers), and locations on aircraft. This jargon is specific to the U.S. Air Force and, at times, specific to a type of aircraft or geographic location. General purpose, off-the-shelf pre-trained word-embedding models such as BERT [1], Word2Vec [16], or Sentence Transformers [17] use publicly available datasets such as Wikipedia, Reddit comments, and Stack Exchange. While these approaches perform well on the English portion of the narratives, performance decreases when processing the jargon-based text. We propose to enhance the training of one or more of these models with additional information extracted from text relevant to the U.S. Air Force. Documents such as aircraft maintenance manuals, Work Unit Code manuals, and Technical Orders may provide the necessary and sufficient training material to improve performance. Our efforts will focus on developing a universal language model available for all aircraft types or understanding if a model for each type is necessary.

7. Conclusions

This work demonstrates the value of information in the textual narratives entered by the maintainers. It is possible to predict and cleanse structured codes within a record with high levels of accuracy. The process specified in this paper can be extended to other aircraft platforms and structured codes with minimal effort. The accuracy of the results depends on the variability of maintainer textual input and its relationship to the structured code. The textual input may not represent all variations captured within the coded field. Providing highly accurate input or corrections to user-selected codes will increase the overall quality of the data. This quality improvement will, in turn, increase the accuracy of downstream reports, decisions, and recommended actions.

As demonstrated in this paper, the effort of making predictions to cleanse the fields was encapsulated into a unique text processing pipeline. This pipeline predicted structured fields with an average accuracy of 93 % across the five maintenance codes. These predicted codes are more accurate than the data entered by maintainers and therefore are used to cleanse this data, leading to improved operational situational awareness. The pipeline comprises text cleansing via normalization, text processing, extraction, clustering, and classification approaches. This pipeline was architected to serve as a black box for repeatable situations and accommodate updates to each component independent of one another. While the unstructured text reflects the structured codes accurately, it is a jargon-based shorthand notation rather than English text. The internal architecture of the pipeline allows customization for new scenarios to accommodate the peculiarities of the short, jargon-based text.

The work presented in this paper demonstrates the possibility of using maintainer textual input to predict and cleanse structured, coded fields. The textual inputs reflect the nuances and jargon of each industry, but the text processing pipeline can be customized to handle these scenarios. Further research is warranted while converting the input to a numerical format representing the maintainers' intent.

Acknowledgements

The authors have not disclosed any funding.

Data availability

The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Conflict of interest

The authors declare that they have no conflict of interest.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv:1810.04805*, 2018.
[2] T. B. Brown et al., "Language models are few-shot learners," *Advances in Neural Information Processing Systems*, No. 33, pp. 1877–1901, 2020, https://doi.org/10.48550/arxiv.2005.14165
[3] B. Plank, "What to do about non-standard (or non-canonical) language in NLP," *arXiv:1608.07836*, 2016, https://doi.org/10.48550/arxiv.1608.07836
[4] Y. Gao, C. Woods, W. Liu, T. French, and M. Hodkiewicz, "Pipeline for machine reading of unstructured maintenance work order records," in *Proceedings of the 30th European Safety and*

Reliability Conference and 15th Probabilistic Safety Assessment and Management Conference (ESREL), 2020.

[5] T. Sexton, M. P. Brundage, M. Hoffman, and K. C. Morris, "Hybrid datafication of maintenance logs from AI-assisted human tags," in *2017 IEEE International Conference on Big Data (Big Data)*, Dec. 2017, https://doi.org/10.1109/bigdata.2017.8258120

[6] M. R. Hodkiewicz, "Maintainer of the future," *Australian Journal of Multi-Disciplinary Engineering*, Vol. 11, No. 2, pp. 135–146, 2015.

[7] A. Mckenzie, M. Matthews, N. Goodman, and A. Bayoumi, "Information extraction from helicopter maintenance records as a springboard for the future of maintenance text analysis," in *Trends in Applied Intelligent Systems*, pp. 590–600, 2010, https://doi.org/10.1007/978-3-642-13022-9_59

[8] H. Bokinsky et al., "Application of natural language processing techniques to Marine V-22 maintenance data for populating a CBM-oriented database," *AHS Airworthiness, CBM, and HUMS Specialists' Meeting*, 2013.

[9] Patrick Ziering, Lonneke van der Plas, and Hinrich Schütze, "Bootstrapping semantic lexicons for technical domains," in *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pp. 1321–1329, 2013.

[10] S. Abney, "Partial parsing via finite-state cascades," *Natural Language Engineering*, Vol. 2, No. 4, pp. 337–344, Dec. 1996, https://doi.org/10.1017/s1351324997001599

[11] G. Salton and M. Mcgill, *Introduction to Modern Information Retrieval*. McGraw-Hill, 1986.

[12] L. Bottou, "Stochastic learning," in *Advanced Lectures on Machine Learning*, Vol. 3176, pp. 146–168, 2004, https://doi.org/10.1007/978-3-540-28650-9_7

[13] D. Denisko and M. M. Hoffman, "Classification and interaction in random forests," *Proceedings of the National Academy of Sciences*, Vol. 115, No. 8, pp. 1690–1692, Feb. 2018, https://doi.org/10.1073/pnas.1800256115

[14] M. Feurer and F. Hutter, "Hyperparameter optimization," *Automated Machine Learning*, pp. 3–33, 2019, https://doi.org/10.1007/978-3-030-05318-5_1

[15] Bergstrajames and Bengioyoshua, "Random search for hyper-parameter optimization," *The Journal of Machine Learning Research*, Vol. 13, No. 2, Feb. 2012, https://doi.org/10.5555/2188385.2188395

[16] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv:1312.3005*, 2013.

[17] A. Vaswani et al., "Attention is all you need," *arxiv.1706.03762*, 2017, https://doi.org/10.48550/arxiv.1706.03762

Srini Anand received his Master's Degree in data science from Indiana University. His interest is in applying Natural Language Processing techniques to a wide variety of topic including topic identification, entity and key information extraction, and domain classification. He contributes to projects both in industry and government. As a principal data scientist, he founded the Illumination Works NLP Center of Excellence and acts as its lead.

Rob Keefer earned his Ph.D. in computer science form Wright State University. As the Director of Research and Innovation at Illumination Works, his research interests include natural language processing, image processing, and augmented reality. As an industry consultant, Rob has worked with over 30 different clients, including Cincinnati Children's Hospital, US Air Force, Raytheon, Major League Baseball, and FedEx Office. He has authored more than 30 publications and delivered over 25 different presentations at local, national, and international conferences.