NLP Tools for Predictive Maintenance Records in MaintNet

Farhad Akhbardeh, Travis Desell, Marcos Zampieri
Rochester Institute of Technology, United States
{fa3019, tjdvse, mazgla}@rit.edu

Abstract

Processing maintenance logbook records is an important step in the development of predictive maintenance systems. Logbooks often include free text fields with domain specific terms, abbreviations, and non-standard spelling posing challenges to off-the-shelf NLP pipelines trained on standard contemporary corpora. Despite the importance of this data type, processing predictive maintenance data is still an under-explored topic in NLP. With the goal of providing more datasets and resources to the community, in this paper we present a number of new resources available in MaintNet, a collaborative open-source library and data repository of predictive maintenance language datasets. We describe novel annotated datasets from multiple domains such as aviation, automotive, and facility maintenance domains and new tools for segmentation, spell checking, POS tagging, clustering, and classification.

1 Introduction

Engineering systems are generating ever increasing amounts of maintenance records often recorded in the form of event logbooks. The analysis of these records are aimed to improve predictive maintenance systems reducing maintenance costs, helping to prevent accidents, and saving lives (Jarry et al., 2018). Predictive maintenance records are collected in multiple domains such as aviation, healthcare, and transportation (Tanguy et al., 2016; Altuncu et al., 2018). In this paper, we present new datasets in the aviation and automotive domains listed in Table 2.

Maintenance record datasets generally contain free text fields describing issues and actions, as in the instances presented in Table 1. Most standard NLP pipelines for pre-processing and annotation are trained on standard contemporary corpora (e.g. newspaper texts, novels) failing to address most of the domain specific terminology, abbreviations, and non-standard spelling present in maintenance records. To help support research in this area, the MaintNet[1] platform, a collaborative open-source library and data repository for predictive maintenance data, has been developed (Akhbardeh et al., 2020). In this paper, we present an evaluation of the tools available at MaintNet, as well as two new datasets included in the platform.

The main contributions of this paper are the following:

1. The creation of novel language resources (e.g. abbreviation lists, datasets, and termbanks) for technical language and predictive maintenance data in the aviation, automotive, and facility management domains. We present two new datasets with aviation and automotive safety records that have been recently collected and annotated and are now available at MaintNet.

2. The creation and development of manually curated gold standards that can be used to evaluate the performance of POS tagging and clustering/classification on technical logbook data.

3. The development and evaluation of a number of Python (pre-)processing tools available at MaintNet including stop word removal, stemmers, lemmatizers, POS tagging, and clustering. We carry out an evaluation of MaintNet's spell checkers and POS taggers comparing them to off-the-shelf NLP packages such as NLTK (Bird et al., 2009) and Stanford Core NLP (Manning et al., 2014), as well as clustering methods.

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

ID	Issue/Problem	Date	Action
111552	R/H FWD UPPER BAFL SEAL NEEDS TO BE RESECURED	7/2/2012	INSTALLED POP RIVET TO RESECURE R/H FWD BAF SEEAL.
111563	CAP SCREWE MISSING, L/H ENG #4 BAFLE	7/3/2012	INSTALLED NEW SCREW. CHKD ENG
111574	CYL #1 BAFFLE CRACKED AT SCREW SUPPORT & FWD BAFL BELOWE #1	7/2/2012	FABRICATED PATCHES OF LIKE MATERIAL & RIVETED IAW CESSN
111585	#3 FWD PUSH ROD TUBE GSK LEAKING @ EGNINE	7/2/2012	REMOVED & REPLACED #3 FWD PUSH ROD TUBE SEALS. LEAK CHE

Table 1: Four instances from one of MaintNet's aviation datasets.

Domain	Dataset	Inst.	Tokens	Code	Source
Aviation	Maintenance	6,169	76,866	*Avi-Main*	University of North Dakota Aviation Program
	Accident	5,268	162,533	*Avi-Acc*	Open Data by Socrata
	Safety	**25,558**	345,979	***Avi-Safe***	**Federal Aviation Administration**
Automotive	Maintenance	617	4,443	*Auto-Main*	Connecticut Open Data
	Accident	54,367	242,012	*Auto-Acc*	NYS Department of Motor Vehicles
	Safety	**5,456**	**137,038**	***Auto-Safe***	**Open Data DC**
Facility	Maintenance	87,276	2,469,003	*Faci-Main*	Baltimore City Maryland Preventive Maintenance

Table 2: Instances and tokens in each dataset in MaintNet. Two new datasets, (*Avi-safe* and *Auto-Safe*), displayed in bold.

2 Related Work

Research in predictive maintenance systems requires large, cleansed, and often annotated logbook data gathered in domains such as web information extraction, system maintenance (*e.g.*, aviation, wind turbines, automobiles), and healthcare (*e.g.*electronic health records).

In the domain of healthcare, Altuncu et al. (2018) analyzed health records of patient incidents provided by the UK National Health Service using a deep neural network with word embedding. Tixier et al. (2016) developed a system to analyze injury reports applying POS tagging and term frequency to extract keywords about injuries creating a dictionary of events to improve future safety management. Savova et al. (2010) applied off-the-shelf NLTK libraries on free-text electronic medical records for information extraction purposes.

In technical domains such as aviation, where MaintNet provides a primary resource, Tanguy et al. (2016) studied various available NLP techniques 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 Talismane NLP toolkit (Urieli, 2013) for incident feature extraction and training.

As to the problem of non-standard spelling, Siklósi et al. (2013), proposed a method of correcting misspelled words in clinical records by mapping spelling errors to a large database of correction candidates. However, due to the large number of abbreviations in medical records, they were limited to specific terms and the normalization had to be performed separately. de Amorim and Zampieri (2013) proposed a dictionary-based spell correction algorithm using a clustering technique by comparing various distance metrics to aim to lower the number of distance calculations while finding or matching target words for misspellings. With this in mind, in MaintNet we provide users with tools developed to deal with domain-specific misspellings and abbreviations.

3 MaintNet Features

In the next sub-sections we present the tools in resources available in MaintNet divided into language resources, pre-processing, and clustering, In addition to that, MaintNet provides various dynamic webpages for users to communicate with each other and with the project developers which work similarly to a forum or message board. We hope that MaintNet's community participation features will further facilitate discussion and research in this under explored domain.

3.1 Language Resources

MaintNet currently features seven English datasets from the aviation, automotive, and facility maintenance domains, which are presented in Table 2. This paper introduces two new datasets with aviation and automotive safety records in bold. These datasets were collected from the USA Federal Aviation Administration and Open Data DC respectively. The list of fields and data types in each dataset is presented in Table 3.

Code	**Fields and Data Types**
Avi-Main	problem, action (text), ata chapter code (int.), date opened/closed (date), identifier/work order (int.)
Avi-Acc	flight (date), type (text), summary of incident (text), record/flight (int.)
Avi-Safe	flight (date), indicated safety/damage type (text), safety remarks (text), flight phase (text), identifier number (int.)
Auto-Main	problem, action (text), reason (text), department (int.), date opened/closed (date), identifier/job number (int.)
Auto-Acc	reported accident (text), accident or injury type (text), date issued (date), record number (int.)
Auto-Safe	request type (text), comment/report (text), request identifier number (int.)
Faci-Main	problem, action, problem type (text), location (text), date opened/closed (date), identifier/Work number (int.)

Table 3: Fields and data types in MaintNet's datasets.

In Figure 1 we present a screenshot of one of MaintNet's datasets, the *Avi-Main* dataset, that can be accessed and searched through the platform. Predictive maintenance datasets are particularly hard to obtain due to the sensitive information they contain. Therefore, we work closely with the data providers to ensure that all confidential and sensitive information in all datasets remains anonymous. As a collaborative platform, MaintNet will be expanded with the collaboration from interested members of the NLP community.

MaintNet further provides the user with domain specific abbreviation dictionaries, morphosyntactic annotation, and term banks validated by domain experts. The morphosyntactic annotation contains the POS tag, compound, lemma, and word stems. Finally, the domain term banks contain a list of terms that are used in each domain along with a sample of usage extracted from the corpus.

3.2 Pre-processing and Tools

One of the bottlenecks of automatically processing logbooks for predictive maintenance system is that most of these datasets are not annotated with the reason for maintenance or a categorization of the issue type. 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 2. The Python scripts for all components in this pipeline are made available through Maintnet.

The process starts with text normalization, including lowercasing, stop word and punctuation removal, and treating special characters with NLTK's (Bird et al., 2009) regular expression library, followed by tokenization (NLTK tokenizer), stemming (Snowball Stemmer), and lemmatization (WordNet (Miller, 1992)). With use of the 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* (Rehurek and Sojka, 2010). 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 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 (Levenshtein, 1966). 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 its widely used in spell checking (de Amorim and Zampieri, 2013). The results of our method compared to other spellchecking techniques in random samples of 500 instances from each of the 5 datasets is presented in Table 4.

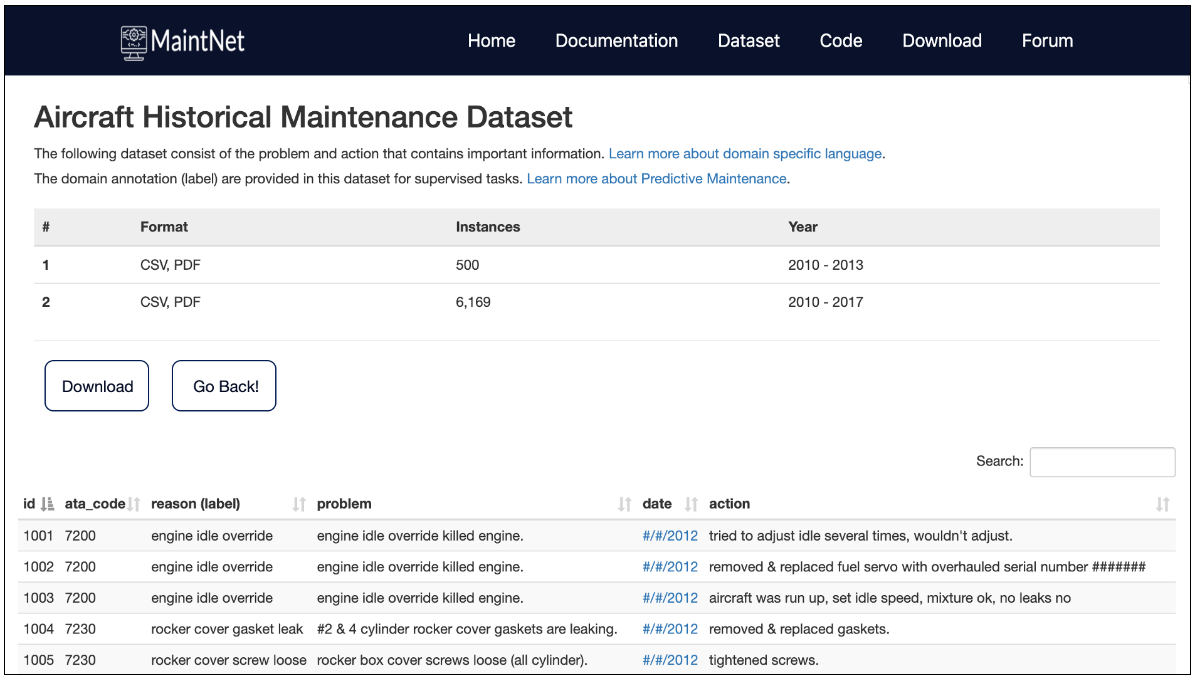
The results are reported in terms of success rate showing that the Levenshtein (Lev) algorithm

[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



Aircraft Historical Maintenance Dataset

The following dataset consist of the problem and action that contains important information. Learn more about domain specific language.
The domain annotation (label) are provided in this dataset for supervised tasks. Learn more about Predictive Maintenance.

#	Format	Instances	Year
1	CSV, PDF	500	2010 - 2013
2	CSV, PDF	6,169	2010 - 2017

id	ata_code	reason (label)	problem	date	action
1001	7200	engine idle override	engine idle override killed engine.	#/#/2012	tried to adjust idle several times, wouldn't adjust.
1002	7200	engine idle override	engine idle override killed engine.	#/#/2012	removed & replaced fuel servo with overhauled serial number #######
1003	7200	engine idle override	engine idle override killed engine.	#/#/2012	aircraft was run up, set idle speed, mixture ok, no leaks no
1004	7230	rocker cover gasket leak	#2 & 4 cylinder rocker cover gaskets are leaking.	#/#/2012	removed & replaced gaskets.
1005	7230	rocker cover screw loose	rocker box cover screws loose (all cylinder).	#/#/2012	tightened screws.

Figure 1: A screenshot of one of MaintNet's dataset webpages.

outperforms the Enchant (Ench), Pyspellchecker (Spell), and Autocorrect (Auto) spell checkers.

Code	Token	Miss	Ench	Spell	Auto	Lev.
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 4: Success rate of spell checkers on 500 instances per dataset. Token stands for total tokens and Miss stands for misspelled tokens.

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, as we discussed in 3.1, we created an exception list using developed morphosyntactic information for the WordNet lemmatizer to ignore mapping words which could be multiple parts of speech.

Finally, we have performed an extrinsic evaluation of MaintNet's pre-processing pipeline by evaluating its impact on POS tagging. To carry out this evaluation, we randomly selected 500 instances of the *Avi-Main* dataset to serve as our gold standard. A North-American English native speaker working in the project annotated the 500 instances using the Penn Treebank tagset. We make this gold standard available to the community in MaintNet.

We compared the performance of three available POS taggers: NLTK (Bird et al., 2009), Stanford CoreNLP (Manning et al., 2014) and TextBlob[6] trained on the raw and pre-processed versions the *Avi-Main* dataset and evaluated on raw and pre-processed versions of the gold standard. We present the results in Table 5 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% accuracy 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.

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

Table 5: Results of three POS taggers annotating raw and (pre-)processed versions of the gold standard.

[6] https://textblob.readthedocs.io/en/dev/

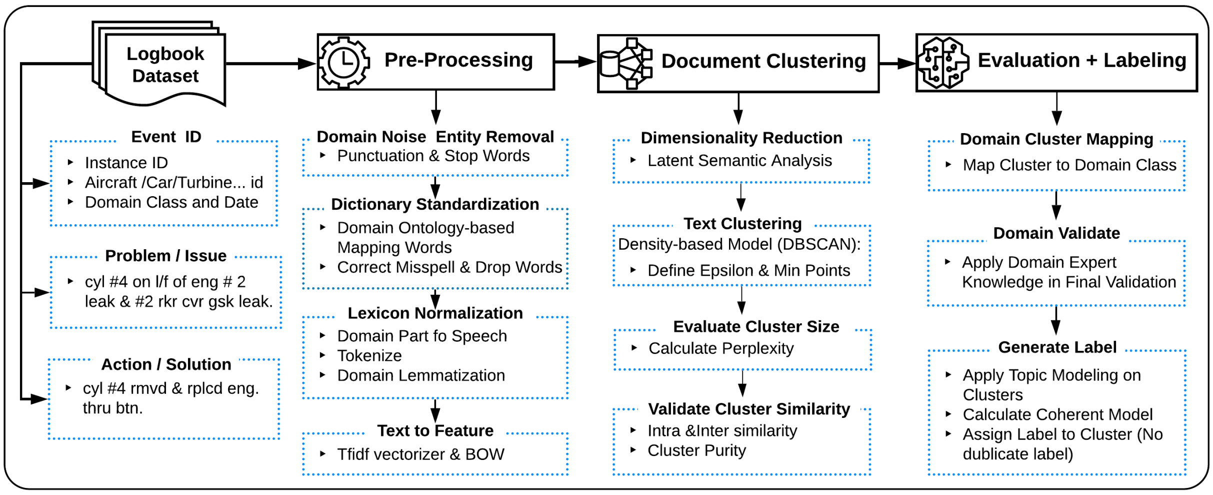


Figure 2: The components in MaintNet's processing and information extraction pipeline: pre-processing, document clustering, and evaluation.

3.3 Clustering

MaintNet also features implementations of popular clustering algorithms applied to logbook data that are made freely available to the research community. The motivation behind this is that most logbook data available is not annotated, which requires a domain expert to group instances into categories. Clustering techniques were used to help in this process.

We converted the terms and words into a numerical representation using libraries such as *tfidfvectorizer* (ElSahar et al., 2017) resulting in a large matrix of document terms (DT). We use truncated singular value decomposition (SVD) (ElSahar et al., 2017) known as latent semantic analysis (LSA), to perform a linear dimensionality reduction. We chose truncated SVD (LSA) over principal component analysis (PCA) (ElSahar et al., 2017) 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 (Jain, 2010), Density-Based Spatial Clustering of Applications with Noise (DBSCAN) (Ester et al., 1996), Latent Dirichlet Analysis (LDA) (Vorontsov et al., 2015), and hierarchical clustering (Aggarwal and Zhai, 2012). For comparison of the results, the silhouette and inertia metrics (Fraley and Raftery, 1998) were used to determine the number of clusters for k-means (both provided similar results), and perplexity (Fraley and Raftery, 1998) and coherence (Vorontsov et al., 2015) 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, Jaro, and cosine (Fraley and Raftery, 1998) to calculate intra- and inter-cluster similarity. The cosine similarity metric is commonly used and is independent of the length of document, while Jaro 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.

Finally, Figure 3 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 presented) 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.

The 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

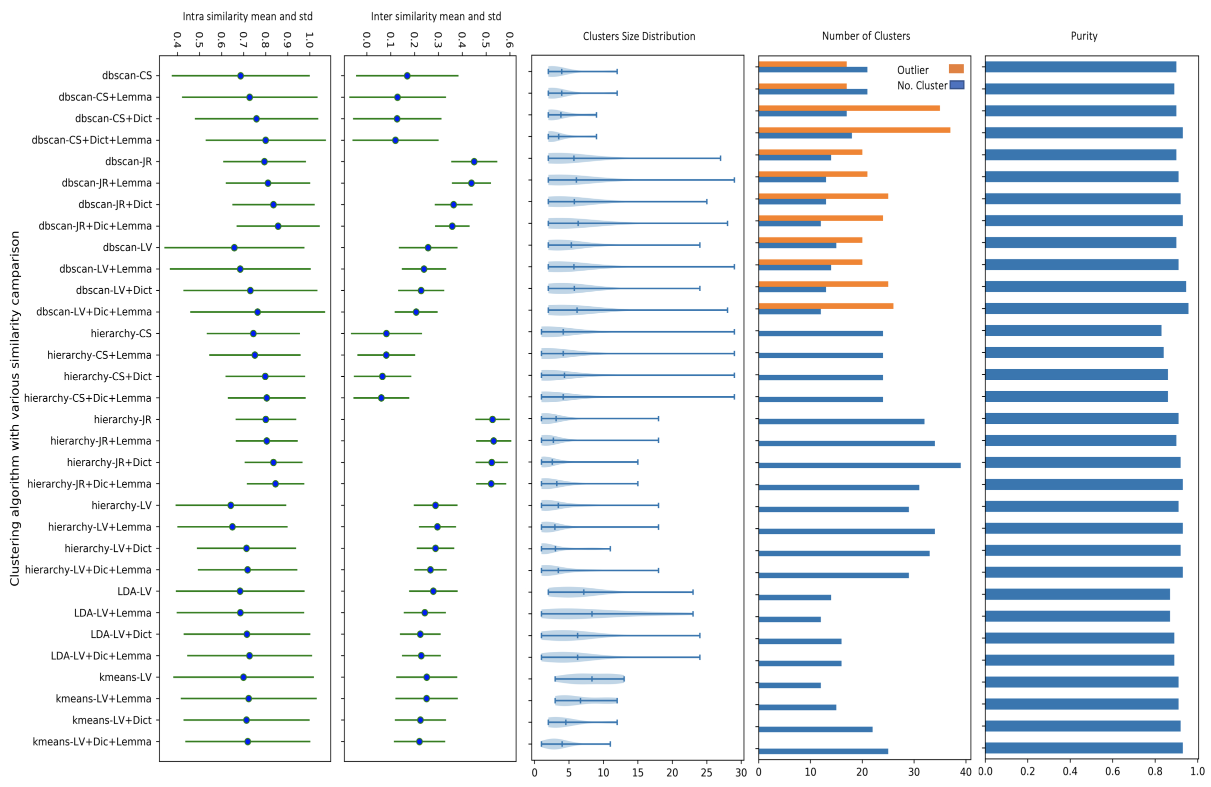


Figure 3: Results of the clustering methods. From left to right, calculated mean and standard deviation of intra- and inter-cluster similarity, cluster size distribution, number of clusters generated by each method and purity on *Avi-Main* dataset.

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 Conclusion

In this paper we evaluate the tools available in MaintNet, a collaborative open-source library for predictive maintenance language resources. MaintNet provides technical logbook datasets on multiple domains: aviation, automotive, and facility maintenance. A number of other important language resources such as abbreviation lists, morphosyntactic information lists, and termbanks have been developed and are also available through the platform. Text (pre-)processing tools developed in Python were evaluated and are also made available. These include spell checking, POS tagging, and document clustering.

Finally, we performed an intrinsic evaluation comparing the performance of several spell-checkers on five of the seven datasets in MaintNet and an extrinsic evaluation on raw and processed versions of the *Avi-Main* dataset on POS tagging. We showed an important increase in performance for all taggers we tested when using data processed with MaintNet's pre-processing pipeline. For the POS tagger comparison and clustering, we developed manually annotated gold standards which are also made available through the platform.

Acknowledgments

We thank the University of North Dakota's aviation program for providing the aviation maintenance records dataset.

We also thank Zechariah Morgain, the aviation domain expert who evaluated the results of our pre-processing techniques and clustering algorithms at many stages during this work, providing us with detailed information about these data sets.

References

Charu C. Aggarwal and ChengXiang Zhai. 2012. A survey of text clustering algorithms. In *Mining Text Data*.

Farhad Akhbardeh, Travis Desell, and Marcos Zampieri. 2020. MaintNet: A Collaborative Open-Source Library for Predictive Maintenance Language Resources. *arXiv preprint arXiv:2005.12443*.

M. Tarik Altuncu, Erik Mayer, Sophia N. Yaliraki, and Mauricio Barahona. 2018. From text to topics in healthcare records: An unsupervised graph partitioning methodology. *ArXiv*, abs/1807.02599.

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

Renato Cordeiro de Amorim and Marcos Zampieri. 2013. Effective spell checking methods using clustering algorithms. In *RANLP*.

Hady ElSahar, Elena Demidova, Simon Gottschalk, Christophe Gravier, and Frédérique Laforest. 2017. Unsupervised open relation extraction. *ArXiv*, abs/1801.07174.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*.

Chris Fraley and Adrian E. Raftery. 1998. How many clusters? which clustering method? answers via model-based cluster analysis. *Comput. J.*, 41:578–588.

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

Gabriel Jarry, Daniel Delahaye, Florence Nicol, and Eric Feron. 2018. Aircraft atypical approach detection using functional principal component analysis. In *SID*.

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

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *ACL*.

George A. Miller. 1992. Wordnet: A lexical database for english. *Commun. ACM*, 38:39–41.

Radim Rehurek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *LREC*.

Guergana K. Savova, James J. Masanz, Philip V. Ogren, Jiaping Zheng, Sunghwan Sohn, Karin Kipper Schuler, and Christopher G. Chute. 2010. 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.

Borbála Siklósi, Attila Novák, and Gábor Prószéky. 2013. Context-aware correction of spelling errors in hungarian medical documents. In *SLSP*.

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

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

Assaf Urieli. 2013. *Robust French Syntax Analysis: Reconciling Statistical Methods and Linguistic Knowledge in the Talismane Toolkit*. Ph.D. thesis, Université de Toulouse II le Mirail.

Konstantin Vorontsov, Oleksandr Frei, Murat Apishev, Peter Romov, and Marina Dudarenko. 2015. Bigartm: Open source library for regularized multimodal topic modeling of large collections. In *AIST*.