Processing of Condition Monitoring Annotations with BERT and Technical Language Substitution: A Case Study

Karl Löwenmark[1], Cees Taal[2], Joakim Nivre[3], Marcus Liwicki[1] and Fredrik Sandin[1]

[1] *Embedded Intelligent Systems Laboratory (EISLAB), Luleå University of Technology, 971 87 Luleå, Sweden, karl.ekstrom@ltu.se*

[2] *SKF Research & Technology Development, Meidoornkade 14, 3992 AE Houten, P.O. Box 2350, 3430 DT Nieuwegein, The Netherlands*

[3] *RISE Research Institutes of Sweden, Isafjordsgatan 22, 164 40 Kista, Sweden, P.O. Box 857, 501 15 Borås, Sweden*

Abstract

Annotations in condition monitoring systems contain information regarding asset history and fault characteristics in the form of unstructured text that could, if unlocked, be used for intelligent fault diagnosis. However, processing these annotations with pre-trained natural language models such as BERT is problematic due to out-of-vocabulary (OOV) technical terms, resulting in inaccurate language embeddings. Here we investigate the effect of OOV technical terms on BERT and SentenceBERT embeddings by substituting technical terms with natural language descriptions. The embeddings were computed for each annotation in a pre-processed corpus, with and without substitution. The K-Means clustering score was calculated on sentence embeddings, and a Long Short-Term Memory (LSTM) network was trained on word embeddings with the objective to recreate the output from a keyword-based annotation classifier. The K-Means score for SentenceBERT annotation embeddings improved by 40% at seven clusters by technical language substitution, and the labelling capacity of the BERT-LSTM model was improved from 88.3 to 94.2%. These results indicate that the substitution of OOV technical terms can improve the representation accuracy of the embeddings of the pre-trained BERT and SentenceBERT models, and that pre-trained language models can be used to process technical language.

1. Introduction

Condition monitoring is vital in the process industry to ensure safe production with minimal wastage and early stops. Monitoring is done by human analysts, assisted by a computerized maintenance management system to estimate the state of industry components and make maintenance decisions. Upon fault detection, maintenance decision or maintenance activity (e.g. component replaced), the outcome is sometimes stored as unstructured text in maintenance work orders (MWOs) and fault diagnosis annotations. These MWOs and annotations contain valuable condition monitoring process information that could be used for data analysis purposes, if the knowledge embedded in the text could be unlocked and integrated into a computerised system. Processing this text would thus improve feedback between analysts and systems, and facilitate learning-based implementations using process-specific technical knowledge.

Tranformer-based (Vaswani et al., 2017) language models such as BERT (Devlin et al., 2019) and GPT (Radford et al., 2019) have been successfully used in many natural language processing (NLP) domains, but are not yet as widely implemented in the technical language domain. Word representations in BERT-based models are typically computed as functions of entire input sequences (Peters et al., 2017, 2018). The contextual input sequence approach adopted from ELMo (Peters et al., 2018) has many advantages, such as dealing with polysemy – identical words having different meanings in different contexts – which can thus be handled, as the embedding for a specific word is different depending on its context.

Technical Language Processing (TLP) was introduced as a domain-driven approach to NLP in a technical engineering framework (Brundage et al., 2021) to help unlock the knowledge represented in technical text. The potential for implementations of embedding algorithms on technical language was investigated by Nandyala et al. (2021), where the challenges of technical language word representations are further discussed, and five suggestions of word representation sys-

tems are provided: TF-IDF, pre-trained Word2Vec (Bahdanau et al., 2015), pre-trained GloVe (Pennington et al., 2014), custom-trained Word2Vec, and pre-trained BERT (Devlin et al., 2019). The authors presented filtered word clusters and word and sentence similarities using these models with a TLP pipeline on 5,485 work orders for 5 excavators (Hodkiewicz et al., 2017).[1] Cadavid et al. (2020) applied CamemBERT, a French version of BERT, to estimate the criticality and duration of maintenance problems from operator descriptions in a French manufacturing company dataset.

Due to the success of the BERT architecture there are many derivative models. A popular model is RoBERTa (Liu et al., 2019), which is a retrained BERT model with different hyperparameters and a slightly different pre-training objective. Neither BERT nor RoBERTa are explicitly trained for sentence embeddings however, as both produce embeddings at a word-level. Typically, either the embedding of the last token (CLS) in the sequence, the mean of all embeddings in the sequence, or the max embedding, are used. Sentence-BERT (Reimers & Gurevych, 2019) is a BERT-based model fine-tuned for sentence similarity with siamese and triplet networks (Schroff et al., 2015), with output vectors being pooled to a fixed-size sentence embedding using either the output of the CLS token, the mean of each output embedding vector, or a max-over-time computation of the output vectors. Using SentenceBERT, it is possible to represent the semantics of sentences and complete annotations more accurately than with BERT, which performs better on word-level representations.

Another solution for language representation is a keyword-based labelling system taxonomy. Ottermo et al. (2021) worked with domain experts to build a taxonomy from 80 annotations for classification of failure events in oil and gas valves in Norway, creating named entities from technical text to symbolise fault cases. A keyword-based system can perform well on a limited data set without requiring labels, but requires human engineering and scales poorly due to the limits of pre-defined conditional statements and inherent complexity in language.

Embeddings from pre-trained language models scale better, but do not generalise well to completely new domains and require large data sets to train. Evaluating the performance of technical word embeddings is also challenging, as most NLP evaluations rely on human-labelled downstream tasks such as GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019), but no such datasets exist for technical language. However, a well-developed technical keyword-based labelling system could serve as one basis for evaluation, and out of vocabulary (OOV) technical terms can be substituted by adding technical taxonomies based on natural language, effectively combining the benefits of both systems. Thus, methods for the integration of OOV technical terms need to be developed and evaluated to further the integration of NLP models into the TLP framework.

[1] Also known as the Prognostics Data Library.

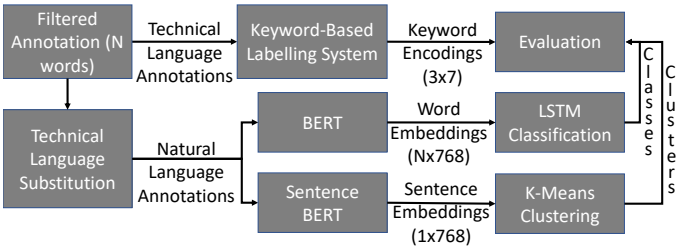

Figure 1. Overview of technical language processing steps, including technical language substitution and evaluation.

Here we investigate the effect of substituting technical terms with natural language descriptions or synonyms on the word and sentence representations generated by a pre-trained language model. In particular, we examine the technical language representations of BERT and SentenceBERT with and without technical language substitution through K-Means clustering, t-SNE (Hinton & Roweis, 2002) analysis and automatic labelling with a keyword-based system. The dataset used consists of real MWOs and annotations from vibration sensor condition monitoring of two large paper manufacturing plants in northern Sweden.

2. Method

Figure 1 gives an overview of the methodology, from annotations to clusters and evaluation. Filtered annotations are used as inputs to the technical language substitution block, presented in Section 2.1, where important technical words are identified and replaced with in-vocabulary language synonyms or descriptions. A pre-processed technical language annotation consisting of N words is fed directly into the keyword-based labelling system, outlined in Section 2.2, which produces labels that can serve as valuable evaluation insights for the unsupervised systems. The effect of the substitution on language model representations is evaluated through a supervised classification task and an unsupervised clustering task, further described in Section 2.3. Finally, the corpus from which annotations were filtered is described in Section 2.4.

2.1. Technical Language Substitution

The purpose of technical language substitution is to use human knowledge and language understanding to facilitate a training-free transfer to new domains where training-data is limited. Technical terms that are substituted are critically important for the condition monitoring-related semantics of the annotations, such as fault class, which directly guides further fault diagnosis. Prior to substitution, these terms are input as tokens that likely do not capture the semantics of the term, but post substitution, these terms are ideally transformed to accurate representations of the term meaning.

Analysing the vocabulary of a language model can be done by investigating its tokenizer, how it represents word strings with sparse vectors prior to embedding them. Older word vector representations tended to have a fixed vocabulary, being capable only of representing words seen during training, while all out-of-vocabulary (OOV) words are represented with the same ”unknown” token. Modern language models typically use techniques to maximise the coverage of all words in a corpus with a set amount of n-grams, through for instance subword tokenisation such as byte pair encoding (BPE) (Gage, 1994; Sennrich et al., 2015) or WordPiece (Schuster & Nakajima, 2012; Wu et al., 2016). The base BERT model and some of its derivatives use WordPiece, while the GPT series and for instance RoBERTa use BPE. Both subword tokenisation algorithms work by representing common words with a single token, while uncommon words are represented as a combination of tokens down to the base character level. For instance, using the WordPiece based *bert-base-uncased BertTokeniser* from the HuggingFace library, the word *academic* is tokenised directly to *[academic]*, while *academical* becomes *[academic, ##al]*. Thus, instead of having unique representations for every word ending, the endings can become unique tokens attached to the root words. However, the WordPiece splitting does not always produce intelligible tokens, such as for *remanufacture*, which is tokenised as *[re, ##man, ##uf, ##act, ##re]*, despite *[manufacture]* also existing as a token in the tokenizer vocabulary. Nonetheless, while some words might be split into unintelligible word pieces, no words are technically outside of the model's vocabulary.

Table 1 shows which technical terms were substituted and their substitution, translated from Swedish to English for reader convenience. Substitution can be done through paraphrasing – rewriting a technical concept with in-vocabulary words while maintaining semantics; synonym substitution – replacing an OOV-term with a semantically similar in-vocabulary term; or abbreviation expansion – substituting an OOV abbreviation with the in-vocabulary words that constitute it. Tests with fewer and with more substitutions were done, and the effect follows a pattern of more frequent and impactful keywords such as *BPFO* and *WO* having larger effect on the clusters and k-Means-score, and less frequent or impactful words such as *mms* and *ds* having smaller effect. The final keywords were chosen to be sufficiently numerous to clearly illustrate the different types impact of substitution, but limiting the number as to not obfuscate the impact through unnecessarily extensive input alterations. A more systematic study of the impact of various keyword combinations is a natural next step to further the understanding of the interaction between technical terms and natural language models.

Table 1. Technical terms and the natural language substitutions used.

technical terms	natural language substitution
bpfo	fault in the outer ring
bpfi	damage on the inner ring
sensor (givare)	sensor
wo	work order
looseness	distance that causes instability
dc	drying cylinder
env/envelope	measurement signal
gr	gear
mms	velocity measurement
fs	free side
ds	drive side

The natural language substitution was designed to be semantically similar to the technical terms, and pass through the tokeniser in a predictable manner, so that no substitution words were chopped up in a way which indicates that BERT has no prior experience of the term. Therefore, the Swedish word *givare*, which means sensor, was for instance just replaced with *sensor*, as SweBERT tokenises *sensor* as one token. The terms *BPFO* and *BPFI* are common fault types in ball-point bearings with distinct condition indicators in signals, but where fault severity is challenging to assess estimate even for expert analysts. Many technical terms are in effect abbreviations, such as *WO* for *work order*, or *dc* for *drying cylinder*. These can, in some cases such as *WO*, be directly substituted for their unabbreviated terms. Other terms, such as *GC* for *guide roller*, can be more difficult to simply expand. The Swedish word for roller, *vals*, is not encoded as one token, but rather as [*val, ##s*], which means whale's or election's. Even if *vals* was encoded as one token, it would likely refer to the more common meaning waltz, the dance, in pretraining rather than a roller. The issue of polysemy, multiple meanings of the same word, is circumvented in contextual language models by using the context as part of the word embedding input, but when neither the context nor the word meaning has been encountered during pre-training, the output embedding is unlikely to be a good representation of the underlying semantics.

Technical language substitution presents an opportunity to merge human knowledge of technical terms with the representational possibilities of natural language models. Unlike language model implementations on general-domain natural language, which can function without significant human interference on input language, this imposes an additional work load on NLP deployment. However, until a technical language model is developed on a massive technical corpus, or natural language models can be accurately fine-tuned on technical data, it can serve as a great addition to the TLP toolbox and for integration into NLP pipelines. Furthermore, defining technical terms in natural language can serve as a spring board for technical language fine tuning, through for instance contrastive learning (Giorgi et al., 2021).

2.2. Keyword-Based Labelling Model

In order to provide additional evaluation methods, a keyword-based labelling system was designed to output fault class and maintenance action labels for each annotation. Keywords were identified by analysing language distributions in collaboration with domain experts. Figure 2 shows the conditional co-occurence of the eight most common non-stopword words in the dataset, calculated from the probability of a target word (rows) appearing in the dataset given the context word (columns). The co-occurence illustrates certain properties of the dataset, such as the sparseness between some technical terms, and the co-dependence of others. For instance, the word *written* has a probability of 1 to appear with *WO*, and is thus always used in the same annotation as the word *WO*, which is due to a common phrase in the dataset being *WO written*. However, the word *WO* is also used without *written* as the context in about half of the annotations, as can be seen from the lower probability in the *WO* row. Hence, including *written* in the keyword-based labelling system would accurately capture all instances of annotations explicitly informing *WO written*, but would exclude annotations that implicitly state the same semantics, such as *WO [number] on BPFO*. Therefore, it is more consistent to base the labelling system on *WO* than on *written*.

The low overlap between key terms indicating fault class, which is seen from for instance the lack of co-occurence between *play*, *sensor* and *BPFO*, indicates that fault annotations often mention at most one fault class. Therefore, a labelling system for fault class detection can be defined by searching for identified fault class keywords. If multiple keywords are found, they can either be concatenated to a new class, or the annotation can be dropped into a corpus with uncertain annotations, where we opted to choose the second option.

The keyword-based labelling system also outputs maintenance actions when applicable, and thus outputs different labels for *sensor replaced, WO [work order] written sensor replacement* and *sensor occasionally malfunctioning*. In the first example, it is an annotation indicating that the fault class *sensor* is remedied, so the corresponding signals should be treated as signals without sensor fault indicators. When *WO written* is a part of an annotation, it indicates a fault which has sufficient severity to warrant replacement. Comparatively, a fault class annotation that does not contain either an indication that it has been replaced or that it should be replaced is simply an annotation that the fault is present.

Table 2. Fault classes and maintenance actions in the keyword-based labelling system.

Fault Classes	BPFI, BPFO, Cable, Play, Imbalance, Disturbance, Sensor
Actions	[WO, replace, change], replaced

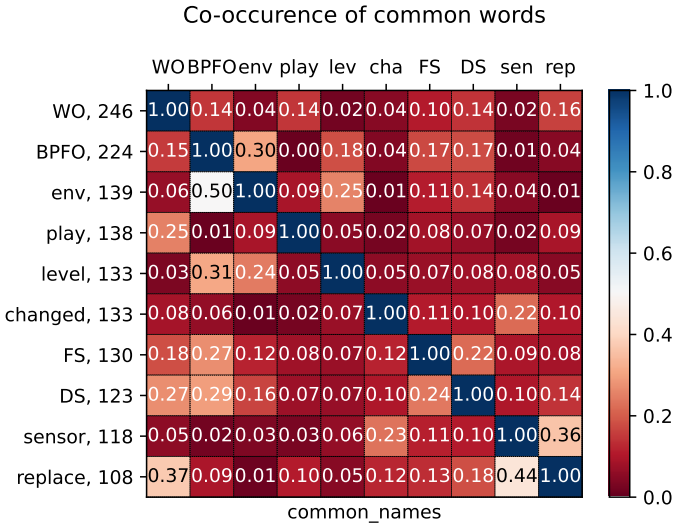

Figure 2. Conditional co-occurence of common words in the annotations, with stop words removed. The co-occurence is computed as the probability of the word in the row given the word in the column.

The keywords used in the labelling system are shown in Table 2. If exactly one fault class keyword was detected, the annotation was considered unambiguous and labelled as belonging to that fault class; if zero or multiple class keywords were detected, the annotation was considered ambiguous and put in another dataset. Maintenance actions were defined similarly, but ”WO”, ”replace” and ”change” were all projected to the ”WO” keyword. Annotations with at most one action keyword type present were considered unambiguous, and annotations with no action keywords were treated as belonging to a ”None” class. Only unambiguous annotations were then used in the evaluation step to ensure reliability of the labelling system.

2.3. Evaluation

Three methods for language model performance evaluation were devised. SentenceBERT was used to represent unsupervised language model understanding of complete annotations, while BERT was used together with a Long Short-Term Memory (LSTM) network (Hochreiter & Schmidhuber, 1997) to learn the combined representation through supervision by the keywords-based labelling system.

First, we investigate the embedding properties of SentenceBERT annotation embeddings visually through PCA and t-SNE dimension reduction techniques in an interactive plot where the annotation can be inspected by hovering over the embedded dot. The embedding dimension of 768 is reduced to 50 through PCA, then to two through t-SNE, as it is common practice to avoid applying t-SNE directly on data of dimensions higher than 50. Similar clusters were observed when t-SNE was applied directly on the data, although this required more computation power.

A visual inspection is difficult to quantify in proper evaluation numbers, and prone to subjective bias or focusing on the desired results, but can also serve as a guide to inspect the semantic distributions of annotation embeddings, and offer some insight into what words affect embedding placement in the 2D projection. However, the resulting 2D space can also be investigated quantitatively by clustering the projected embeddings through for instance K-Means, and evaluating the inherent clusterability of the space as well as the words and annotations used in each cluster. If the K-Means score, defined as the average distance between cluster data points and corresponding epicenters, increases (closer to zero) due to a change, then it stands to reason that the vector space is more separable if the score is normalised with regards to the scale of the K-Means space. If the clusters also form based on important technical words indicating fault class or maintenance action, then this improvement in separability is also an improvement in technical language representations. While the classes formed from K-Means might have no inherent overlap with those from the keyword-based labelling system, there is still merit in comparing these to see where the systems agree and disagree.

We also use BERT word embeddings as input to an LSTM network optimised to reproduce the keyword-based labels. The keyword-based labels labels are certainly not perfect representations of the desired encodings of technical annotations. However, it stands to reason that if the reproducibility of these labels is improved after technical language substitution, in spite of the fact that many technical terms substituted are also directly used as keywords for the labelling system, then the substituted BERT word embeddings are a better representation of the semantic space than the unsubstituted ones.

2.4. Dataset

Table 3 shows corpus properties of the original annotation dataset, a filtered version, a version with only unique annotations and finally annotations marked as clearly defined by the keyword-based system. The original annotations were directly extracted from the condition monitoring dataset, and the filtered annotations were computed by removing digits and special characters from the dataset, which reduces the average annotation length slightly. Duplicate annotations were removed to create a corpus with unique datasets, which has on average longer annotations due to many duplicates being "fault detected" annotations. The dataset was then finally split into clearly and unclearly defined annotations with regards to fault class, as decided by the keyword-based labelling system. The unclearly defined annotations feature on average more words, with on average longer annotations.

Table 3. Properties of differently filtered annotations.

Annotations set	#annotations	μ(annotations length)	σ(annotations length)	#words
Original	1975	6.19	6.50	3008
Filtered	1975	5.71	6.22	1929
Unique	1162	7.13	6.93	1929
Clearly defined	618	6.85	7.43	1111
Unclear	544	7.45	6.29	1334

Table 4. An example of annotations connected to a subasset in a paper machine (with anonymised names, X). This subasset has a higher than average number of annotations due to faults.

Fault	Date	Comment
None	09/20	The lubrication works as it should according to X. This roller is lubricated with oil. The bearing damage is clearly visible in env with many overtones but is at a very low level however it has increase a bit lately. Ground frequency for outer ring in mm/s also seen
BPFO	09/20	BPFO seen on FS. Write WO maybe??? Talked with X and he'll check lubrication.
BPFO	06/20	BPFO Indication DS. Low levels Keep watch.
None	03/20	Roller replaced.
None	12/19	WO written on bearing replacement drive side.
BPFO	12/19	BPFO on drive side. Keep watch.

The initial corpus consists of 1975 annotations with a total of 3009 unique tokens. Of the 3009 unique tokens, 1286 require wordpiece splitting for BERT to process, as explained in the previous section. The longest annotation is 106 words long, and the shortest is 1 word. The BERT model used as embedder has a maximum input length of 512. With the longest annotation at 106 words, and the tokenised version at most twice as long, all annotations were processable within this output limit. All annotation sentences were transformed from text to embedding space using the BERT tokeniser and the Swedish version of SentenceBERT (Rekathati, 2021).

Table 4 shows examples of annotations associated with a subasset consisting of two sensors, one on the locating (free, FS) and one on the non-locating (drive, DS) side. The first three notes are typical examples of fault detection, maintenance action and maintenance follow up. A BPFO is detected on the drive side, but the severity does not yet warrant a replacement. After 14 days, the analyst decides to write a work order for roller replacement. Three months later, the component is replaced and a follow-up annotation is written. This annotation is of critical importance for the possibility of technical language supervision (Löwenmark et al., 2021), as it indicates where the associated signal data should be treated as healthy again. However, only three months after this replacement a BPFO is spotted again, initially on the non-locating side. Upon further inspection after three months, it now appears primarily on the locating side, which shows the challenging task of analysing signals where faults propagate between sensors. This rapid recurrence of a new fault is unexpected, prompting another annotation describing a more detailed analysis of the component. Since the signal levels are low, the fault has not warranted a replacement even at the end

of the dataset (mid 2021), which showcases the importance of accurate fault severity assessment in minimising unnecessary maintenance actions and material wastage from premature replacements.

3. RESULTS

Figure 3 shows all outputs from the keyword-based labelling system on the filtered corpus, projected onto 2D-representations of SentenceBERT annotation embeddings with and without technical language substitution. The labels were produced as described in Section 2.1, and the 2D visualisation of the embedding space was produced through PCA dimension reduction to 50 dimensions, and t-SNE projection from 50 to 2 dimensions. While the system works identically on both annotations, the visualisation clearly shows that the clusters formed after technical language substitution correlate more with the labels produced by the keyword-based system.

Figure 4 shows 21 dimensional K-Means clustering applied on the same t-SNE projected embedding space as the second half in Figure 3. The labels are formed by searching for the five most common words in annotations belonging to each cluster. Due to BPFO being the most commonly mentioned fault class in the annotation set, it also becomes overrepresented in the label set. Comparing to the keyword-based labels from Figure 3, the large sets of BPFO labels (turquoise) has been split into multiple BPFO sets with small nuances in the semantics, such as whether the fault is of low levels or a WO should be written. However, BPFO is clearly over-

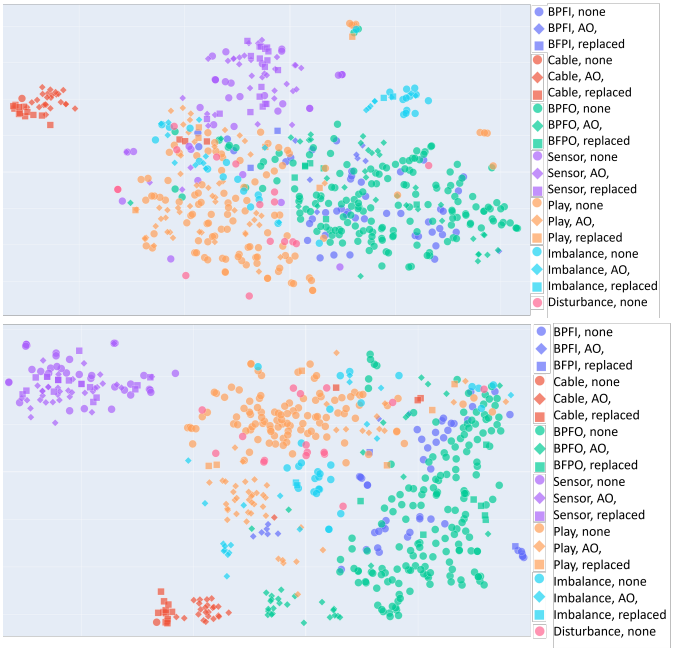

Figure 3. Two dimensional t-SNE transformation of annotation sentence embeddings, labelled by a keyword-based labelling system, without (top) and with (bottom) technical language substitution.

represented in the label set, which likely is due to the high number of BPFO cases, and the wide distribution in the cluster space, as can be seen in the keyword-based labels as well.

The projected embedding space had consistently average K-Means closer to zero for all K larger than 3, and performed significantly better at K larger than 5, as shown in Figure 5. The Figure shows K-Means scores for between 5 and 50 clusters, computed on all unique annotations in the dataset, plotted as a function of number of clusters. As expected, the K-Means score decreases as more clusters can be formed due to the shorter distances between points to their cluster epicenters. The K-Means score is on average 36% more negative before substitution for K larger than 5, and goes below 30% only for K = 11, 12 and 13 in this group.

The effect of technical language substitution was also evaluated by reproducing the output of the keyword-based labelling system from BERT word embedding inputs. The keyword-based labelling system produces annotation labels as outputs, which can be used as supervision signals. To widen the experimental scope, the base Swedish BERT model, KB-BERT (Malmsten et al., 2020), was used on a token level. Thus, an annotation consisting of five tokens resulted in a sequence of five 768 dimensional embeddings. These embeddings were then fed into an LSTM network, whose output was fed into a feed-forward neural network. Compared to SentenceBERT, this effectively allows the model to independently learn the ideal combination of word embeddings into dense representations of annotation semantics.

The LSTM model without technical language substitution obtained an accuracy of 88.3% on a randomly sampled test set when trained for 100 epochs, using the model with the highest validation set accuracy for testing, while the model with technical language substitution obtained an accuracy of 94.2% when trained with an identical set up, for a 5.9% difference. The number of erroneous predictions thus decreased from 11.7% to 5.8% for an error reduction of 50%.

Figure 6 shows an example of confusion matrices from the LSTM systems, where the bottom matrix is computed with substituted input. Due to random sampling of train, validation and test annotations, the number of annotations of each class can differ in each run, which is why multiple trials is important for reliability of the results. While the substituted input is clearly better suited for downstream NLP tasks, both models have the worst accuracy for *Play*, which likely appears in similar context as other faults more than any other fault class. Looking at Figure 3, some *Play* labels are far away from the main cluster in the t-SNE space, so it stands to reason that for the erroneous predictions the embeddings were poor indicators of the *Play* fault class.

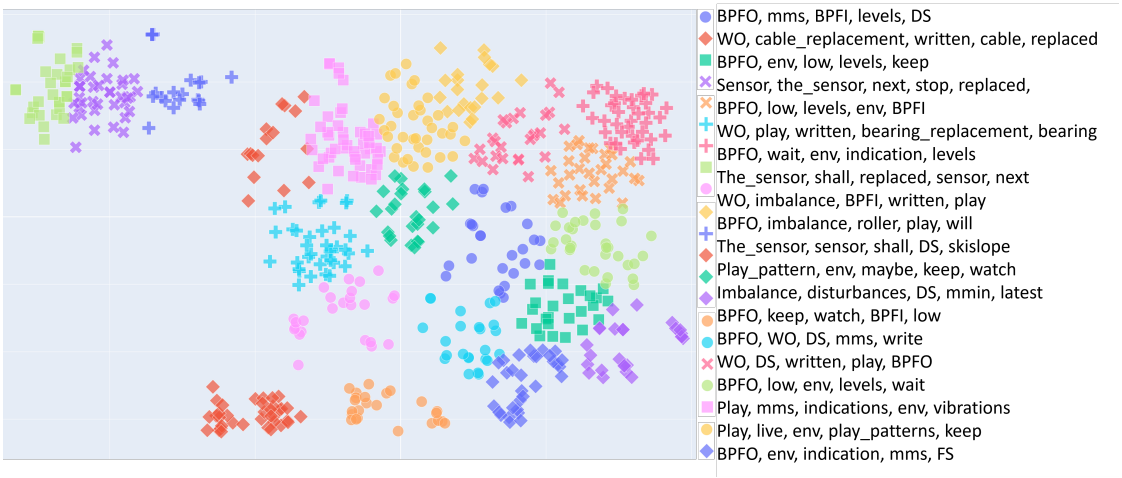

Figure 4. Two dimensional t-SNE transformation of annotation sentence embeddings with technical language substitution, classified with K-Means and labelled with the five most common words per cluster.

4. Discussion

The positive impact of technical language substitution was evaluated qualitatively in Figure 3 and quantitatively through the K-Means score in Figure 5, as well as the LSTM output reconstruction shown in 6. These results are proof of concept that substituting out-of-vocabulary words can improve the language model performance on other language domains. Figure 4 illustrates how well a K-Means algorithm can cluster annotation semantics based on SentenceBERT. Analysing the clusters quantitatively there are few instances where multiple fault classes are among the top five words, with the exception being BPFI which appears after BPFO and lacks its own clusters. The clusters also tend to form around words either indicating low severity, high severity or component replacement, but in a few cases contain both *replace* and *replaced*, words that indicate significant differences in the condition indicators. To deploy a language model on an annotation set, a human-in-the-loop approach where these differences are instructed to the TLP model might be necessary. Nonetheless, the results indicate that substitution significantly improves the language representation, and thus further improvements might facilitate language model deployment with less data engineering requirements.

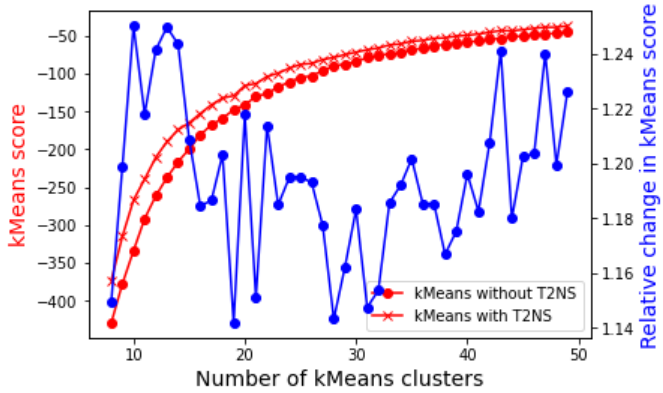

Figure 5. The K-Means score of K-Means clusters formed from SentenceBERT embeddings of unsubstituted (without T2NS) and substituted (with) annotations for a varying number of clusters, the relative improvement calculated as the old K-Means score divided by the new score.

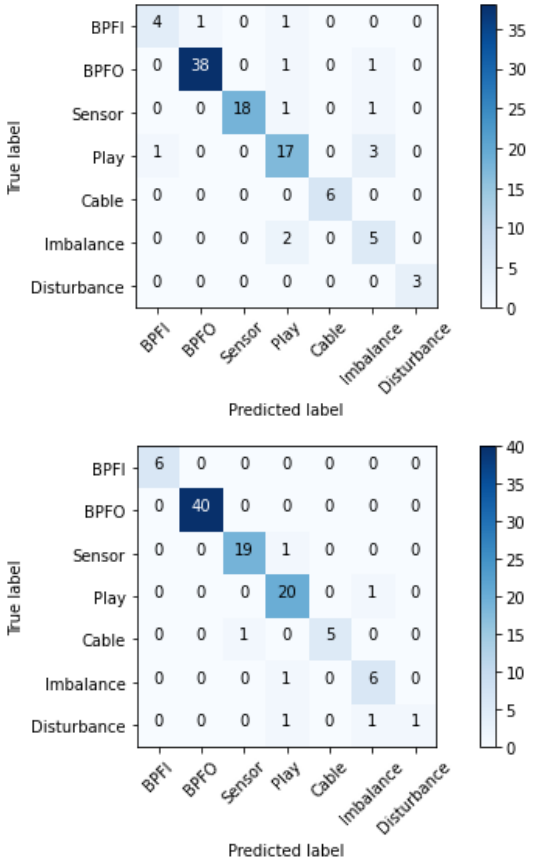

Figure 6. Confusion matrices from an LSTM with fully connected layers reconstructing the keyword-based labelling system classification without (top) and with (bottom) substitution on the same data split.

As no labelled ground truth evaluation dataset exists, multiple weaker evaluation methods were used to improve the reliabil-

ity of the results. If the results indicate similar conclusions despite the different evaluation methods, it is more likely that they correlate with some underlying truth which would otherwise be represented by ground truth labels. In principle, the normalised K-Means score only conveys information on the relative average distance between cluster data points and their epicenters. Randomly distributed data can in theory result in a lower K-Means score than perfectly clustered data if K is smaller than the number of clusters, for instance with two distinct separate clusters and $K = 1$. However, coupled with a visual inspection and comparison in Figure 3, it is clear the the reduction is due to improved representation rather than an under-fitted K. This is also evident from Figure 5, where the K-Means score is higher after substitution when $K = 1$ and 3, and where considerable improvement is seen first after $K = 6$. The improved performance of the LSTM-system with embedding inputs is also important for evaluation as it indicates a better distribution of the embedding space with regard to fault classes. It also reinforces the concept of assisting language model evaluation through a keyword-based system, as the substitution removes the keywords from the language model input space. Consequently, the improvements seen in the classification step must be from a superior distributional representation of the underlying semantics rather than a "short-circuit" mapping between keywords.

The lack of intrinsic or extrinsic evaluation tools or datasets for language models in the technical domain obfuscates potential improvements in technical language understanding. Cadavid et al. (2020) evaluated technical language representations using internal properties of the dataset, with equipment descriptions, symptoms and equipment importance as input, and type of disturbance (dominant or recessive) and maintenance workload (hours) as outputs. The existence of these two possible outputs facilitates an extrinsic evaluation, though one not necessarily in complete correlation with annotation semantics, as their results indicate that the TF-IDF-based methods by far outperform pre-trained CamemBERT and often perform just shy of fine-tuned CamemBERT. Thus, the increased natural language understanding of CamemBERT is either impeded by OOV technical terms, or the evaluation method has poor correlation with language understanding. Without access to resources such as GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019) from NLP, it is difficult to properly evaluate whether the language model is overfitted to the specific language distribution.

Nandyala et al. (2021) used an English dataset without obvious extrinsic evaluation tasks, and thus rely on quantitative evaluation in word similarity, sentence similarity and word cluster projections to compare their different distributional word vector models. These methods offer some initial insight into the workings of language models on technical language, but without quantitative values it is difficult to compare models and evaluation relies on subjective human judgement. Ideally, work towards creating a technical language version of GLUE and SuperGLUE can be initiated to further the research into adaption, pre-training and fine-tuning of language models on technical language.

5. Conclusion

This study has investigated the effect of substituting out-of-vocabulary technical terms with natural language descriptions on BERT and SentenceBERT word distributions To evaluate the models, a keyword-based labelling system was designed and used for visualisation and comparison with a K-Means clustering algorithm. Furthermore, the system was used to generate labels for optimization and testing of an LSTM with two output layers. The K-Means scores of clusters formed from t-SNE and PCA transformations of SentenceBERT representations of condition monitoring annotations were also computed and used as evaluation metrics. We contribute to the methodology of evaluating machine learning models' understanding of language, which was investigated by generating multiple evaluation methods where no ground truth is present.

There are many opportunities to bridge the gap between NLP successes and technical language challenges, but one major factor impeding this progress is the lack of standardised evaluation resources on technical language. For further research, we suggest the development of a technical version of the GLUE benchmark, and additional experimentation in transferring natural language understanding from large pre-trained language models to small technical language datasets through either data manipulation, fine-tuning or other transfer learning approaches. The effect of technical language substitution could also be further investigated through a systematic study of the effect of each substitution and different sets of substitutions, ideally on a dataset with many OOV technical terms but with some type of ground truth labels available for evaluation.

Acknowledgements

This work is supported by the Strategic innovation program Process industrial IT and Automation (PiIA), a joint investment of Vinnova, Formas and the Swedish Energy Agency, reference number 2019-02533. The technical term descriptions used in this work were kindly provided by Håkan Sirkka. We thank the members of the project reference group including Per-Erik Larsson, Kjell Lundberg, Håkan Sirkka and Peter Wikström, for valuable inputs.

References

Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *ICLR 2015.* (First published as pre-print on arXiv.)

Brundage, M. P., Sexton, T., Hodkiewicz, M., Dima, A., & Lukens, S. (2021). Technical language processing: Unlocking maintenance knowledge. *Manufacturing Letters*, *27*, 42–46.

Cadavid, J. P. U., Grabot, B., Lamouri, S., Pellerin, R., & Fortin, A. (2020). Valuing free-form text data from maintenance logs through transfer learning with CamemBERT. *Enterprise Information Systems*, *0*(0), 1–29.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019* (pp. 4171–4186). (First published as pre-print on arXiv.)

Gage, P. (1994). A new algorithm for data compression. *The C Users Journal archive*, *12*, 23-38.

Giorgi, J., Nitski, O., Wang, B., & Bader, G. (2021, August). DeCLUTR: Deep contrastive learning for unsupervised textual representations. In *ACL-IJCNLP 2021* (pp. 879–895). Online: Association for Computational Linguistics.

Hinton, G. E., & Roweis, S. (2002). Stochastic neighbor embedding. *NIPS 2002*, *15*.

Hochreiter, S., & Schmidhuber, J. (1997, 12). Long short-term memory. *Neural computation*, *9*, 1735-80.

Hodkiewicz, M. R., Batsioudis, Z., Radomiljac, T., & Ho, M. T. (2017). Why autonomous assets are good for reliability–the impact of 'operator-related component'failures on heavy mobile equipment reliability. In *Annual conference of the PHM Society* (Vol. 9).

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *ArXiv*, *abs/1907.11692*. (First published as pre-print on arXiv.)

Löwenmark, K., Taal, C., Schnabel, S., Liwicki, M., & Sandin, F. (2021). Technical language supervision for intelligent fault diagnosis in process industry. *arXiv e-prints*, arXiv:2112.07356.

Malmsten, M., Börjeson, L., & Haffenden, C. (2020, July). Playing with Words at the National Library of Sweden – Making a Swedish BERT. *arXiv e-prints*, arXiv:2007.01658.

Nandyala, A., Lukens, S., Rathod, S., & Agarwal. (2021, Jun). Evaluating word representations in a technical language processing pipeline. *PHM Society European Conference. 6*.

Ottermo, M. V., Håbrekke, S., Hauge, S., & Bodsberg, L. (2021, Jun). Technical language processing for efficient classification of failure events for safety critical equipment. *PHM Society European Conference. 6*.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In *EMNLP 2014* (pp. 1532–1543).

Peters, M. E., Ammar, W., Bhagavatula, C., & Power, R. (2017, July). Semi-supervised sequence tagging with bidirectional language models. In *ACL 2017* (pp. 1756–1765). Vancouver, Canada: Association for Computational Linguistics.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018, June). Deep contextualized word representations. In *NAACL-HLT 2018* (pp. 2227–2237). New Orleans, Louisiana: Association for Computational Linguistics.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners.

Reimers, N., & Gurevych, I. (2019, November). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *EMNLP-IJCNLP 2019* (pp. 3982–3992). Hong Kong, China: Association for Computational Linguistics. (First published as pre-print on arXiv.)

Rekathati, F. (2021). The KBLab blog: Introducing a Swedish sentence transformer.

Schroff, F., Kalenichenko, D., & Philbin, J. (2015, Jun). Facenet: A unified embedding for face recognition and clustering. *CVPR 2015*.

Schuster, M., & Nakajima, K. (2012). Japanese and korean voice search. In *ICASSP 2012* (p. 5149-5152).

Sennrich, R., Haddow, B., & Birch, A. (2015). Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. In I. Guyon et al. (Eds.), *NIPS 2017* (Vol. 30). Curran Associates, Inc. (First published as pre-print on arXiv.)

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., ... Bowman, S. (2019). Superglue: A stickier benchmark for general-purpose language understanding systems. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *NIPS 2019* (Vol. 32). Curran Associates, Inc.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. (2018, November). GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *EMNLP 2018* (pp. 353–355). Brussels, Belgium: Association for Computational Linguistics. ((First published as pre-print on arXiv.))

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., ... Dean, J. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv*, *abs/1609.08144*.