ENHANCING ELECTRONIC LOGBOOKS USING MACHINE LEARNING

Jennefer Maldonado*, Samuel Clark, Wenge Fu, Seth Nemesure
Brookhaven National Laboratory, Upton, New York, United States

Abstract

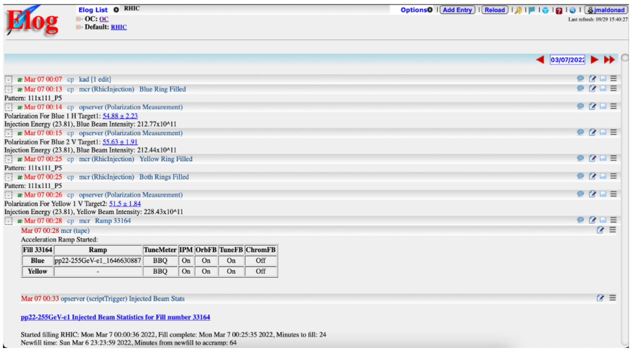
The electronic logbook (elog) system used at Brookhaven National Laboratory's Collider-Accelerator Department (C-AD) allows users to customize logbook settings, including specification of favorite logbooks. Using machine learning techniques, customizations can be further personalized to provide users with a view of entries that match their specific interests. We will utilize natural language processing (NLP), optical character recognition (OCR), and topic models to augment the elog system. NLP techniques will be used to process and classify text entries. To analyze entries including images with text, such as screenshots of controls system applications, we will apply OCR. Topic models will generate entry recommendations that will be compared to previously tested language processing models. We will develop a command line interface tool to ease automation of NLP tasks in the controls system and create a web interface to test entry recommendations. This technique will create recommendations for each user, providing custom sets of entries and possibly eliminate the need for manual searching.

INTRODUCTION

The electronic logbook (elog) system is used to record information related to machine and system operations as well as individual record keeping. Applications in the controls system send data, plots, and images to be uploaded. The system is shown in Fig. 1. Engineer and physicists often use the elog to document procedures and instructions. The more documentation we have available, the easier it is to diagnose new problems limiting down time and delays in science. The search feature in the system only provides entries with the exact search term the user enters. There are entries related to this search term that may not contain the term exactly. Natural language processing models can aid in this search process by analyzing similarity in entries and classifying them by topic group. If a user is interested in power supply failures and search the term "power supply", entries about magnet quenches without the search term will not be featured but are directly related. Analyzing and producing similarity metrics will help specific groups of people like operators, controls staff, and physicists narrow down entries of interest. The goal is to provide accurate results in order to improve productivity of users.

DATA PROCESSING

As users enter entries in the system, each entry is stored in a MySQL database. This enables storage of information dating back to 2013 when the system was first implemented. To access this stored data we use a tool developed to connect to our departmental databases. Along with each entry we have some meta data like the entry id, timestamp, author, and tag. This data is collected and stored in a Pandas dataframe. This dataframe is then processed to remove rows with empty content. The empty rows correspond to entries with no text, only include images, or have been deleted. After processing, all rows in the dataframe contain content that need to be formatted in a language model interpretable format. First the entries are tokenized. The goal of tokenizing text is to separate the sentences into a list of words the model can utilize better for training purposes. After tokenizing, lemmatization is applied to convert all words in inflected forms to the dictionary form. For example, the word log has variations such as "logs", "logging", "logged". Each inflected form found in the dataframe is transformed into the dictionary form of the word. In our example the dictionary form is log. To ensure the model focuses on collider accelerator topics the most common words in the English language are removed from the data. Examples of these words are "the", "in", "go", and "had". If these words are not removed from text there is a risk the model will correlate sentences containing common words rather than ones related to the elog entry topics. Punctuation is also removed from the text vectors. The histograms in Fig. 2 display the total word counts of the most used words in elog entries before and after the common words have been removed. Once the contents of entries are processed, we can use Gensim's Doc2Vec (D2V) model to predict similar entries.



```
Mar 07 00:07 cp kad [1 edit]
Mar 07 00:13 cp mcr (RhicInjection) Blue Ring Filled
Pattern: 111x111_P5
Mar 07 00:14 cp opserver (Polarization Measurement)
Polarization For Blue 1 H Target1: 54.88 ± 2.23
Injection Energy (23.81), Blue Beam Intensity: 212.77x10^11
Mar 07 00:15 cp opserver (Polarization Measurement)
Polarization For Blue 2 V Target1: 55.63 ± 1.91
Injection Energy (23.81), Blue Beam Intensity: 212.44x10^11
Mar 07 00:25 cp mcr (RhicInjection) Yellow Ring Filled
Pattern: 111x111_P5
Mar 07 00:25 cp mcr (RhicInjection) Both Rings Filled
Pattern: 111x111_P5
Mar 07 00:26 cp opserver (Polarization Measurement)
Polarization For Yellow 1 V Target2: 51.5 ± 1.84
Injection Energy (23.81), Yellow Beam Intensity: 228.43x10^11
Mar 07 00:28 cp mcr Ramp 33164
    Mar 07 00:28 mcr (tape)
    Acceleration Ramp Started:
```

Fill 33164	Ramp	TuneMeter	IPM	OrbFB	TuneFB	ChromFB
Blue	pp22-255GeV-e1_1646630887	BBQ	On	On	On	Off
Yellow	-	BBQ	On	On	On	Off

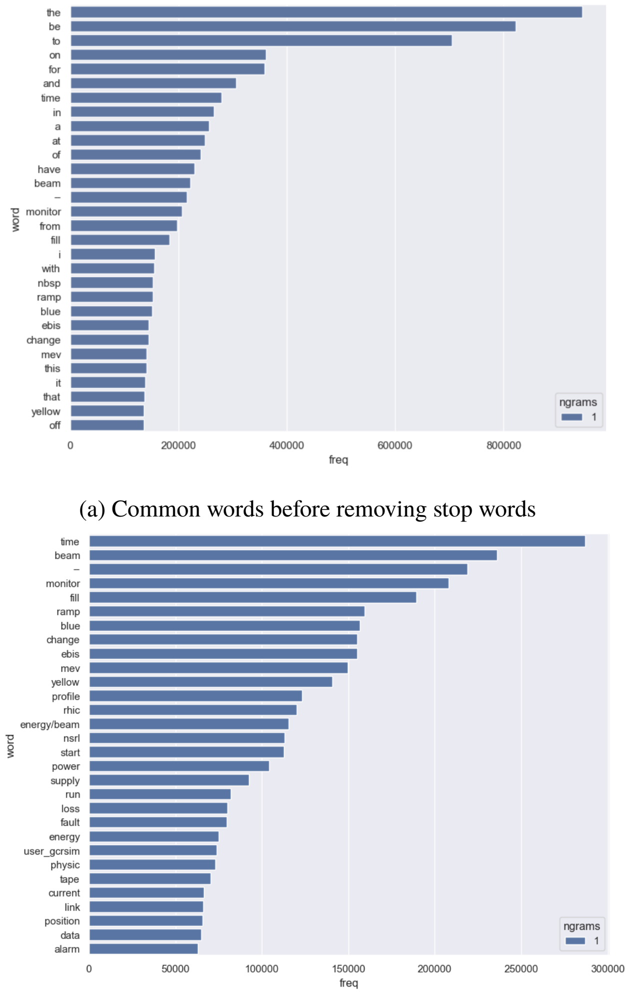
```
    Mar 07 00:33 opserver (scriptTrigger) Injected Beam Stats

    pp22-255GeV-e1 Injected Beam Statistics for Fill number 33164

    Started filling RHIC: Mon Mar 7 00:00:36 2022, Fill complete: Mon Mar 7 00:25:35 2022, Minutes to fill: 24
    Newfill time: Sun Mar 6 23:23:59 2022, Minutes from newfill to accramp: 64
```

Figure 1: Screenshot of elog interface.

NATURAL LANGUAGE MODELS

The Gensim package is a fast library for training large NLP models [1]. This package makes it easy to load a model and build a vocabulary from processed elog entry vectors. The model trained for approximately 1.5 hours on nearly a decades worth of entry data. This was done on a machine with a GPU and the total number of entries was about 1.5 million. Only 100 epochs were used for each prediction.

* jmaldonad@bnl.gov



(a) Common words before removing stop words

(b) Common words after removing stop words

Figure 2: Histograms of Word Counts from Elog Entries.

When the model finds similar documents compared to one entry the timing varies from 0.3 seconds to 1 second.

CLASSIFICATION

Two classification models were tested with the elog entry data and were analyzed for best performance. Metrics used to determine if the classifier was ideal for this data were the recall, precision, and f-score. High accuracy and low recall or precision is not an indicator of an appropriate model for a problem so these additional metrics must be included in the analysis.

Multiclass Classifier

There are tags in the elog system that help identify major topics in entries. Examples of these tags are *failure* and *machine setup*, that are used to help identify major topics for an entry. The tags are not used often unfortunately which makes the classification task at hand much more difficult. This classifer was built using predefined layers in Tensorflow [2]. Running this multiclass classifier with 19 tags and 1000 epochs produced a 79% training accuracy and 72% testing accuracy. The recall score was around 39%, the precision score was 47%, and an F Score of 0.42. Figure 3 displays the training and testing accuracy.

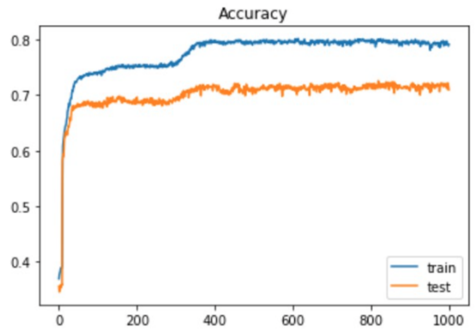


Figure 3: Training Accuracy for Multiclass Classifier.

Multinomial Naive Bayes

The multinomial naive bayes classifier implements the naive bayes technique for data with discrete features which is applicable to word counts in text classification [3]. The elog entries are transformed into word vector counts using scikit learn's CountVectorizer. [3] This improved recall, precision, and fscore to 74%, 66%, and 0.70 respectively.

TOPIC MODELING

Latent Semantic Analysis

Latent Semantic Analysis (LSA) was created to solve the problems lexical matching created back in the late 90's. At the time search engines were using lexical matching as the go to algorithm. Unfortunately, this caused a lot of irrelevant information to be returned in the search, missing the relevant materials [4]. LSA implements dimension reduction techniques. The data in the dataset was processed the same way as if the Doc2Vec model was being applied. A count

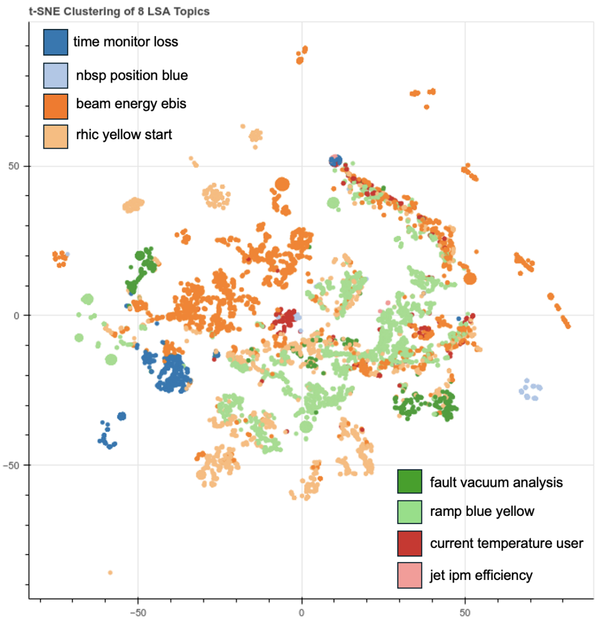


Figure 4: Clustering Results for LSA using t-SNE.

vectorizer was utilized to convert each elog entry unto a numerica vector. Scikit learn calls it's LSA technique truncated singular value decomposition (SVD). The goal of LSA is to find meanings and similarities of documents based on how frequently words appear as well as the location within the document. Once the data was transformed using the truncated SVD, ScikitLearn's T-distributed Stochastic Neighbor (t-SNE) Embedding [3] is used to cluster our data. The goal of t-SNE is to take the similarity or distance between the vector and predicted topic, thus providing a useful way to visualize this data [5]. As for any clustering techniques the goal is to have clearly defined groups of entries. The eight LSA topics in Fig. 4 would be difficult to identify by visual inspection.

Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) processes each document in the dataset as a vector of real numbers. These values represent the ratio of the counts of the words in the document. These vectors then represent entire documents. This approach was used in the early 2000s for internet search engines [6]. The scikit learn latent Dirichlet allocation function with eight topics was applied in addition to the count vectorizer. Transforming the matrix and applying t-SNE generates Fig. 5. The clustering results are more clear then LSA.

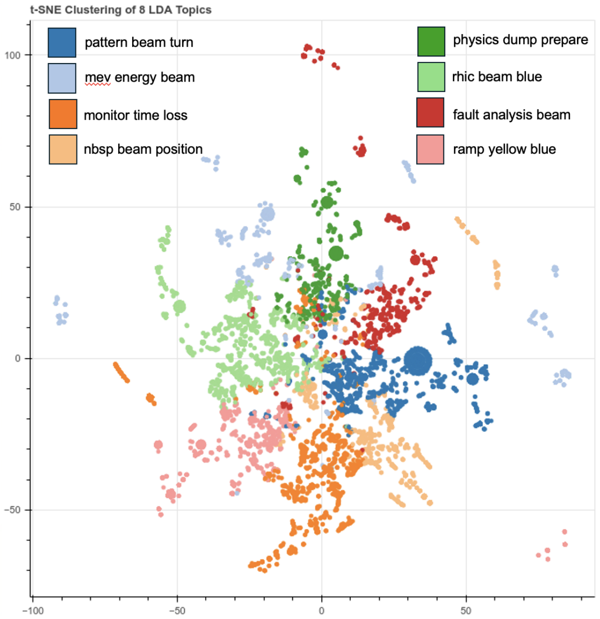


Figure 5: Clustering Results for LDA using t-SNE.

IMAGE PROCESSING

Applying optical character recognition (OCR) to images results in text data parsed out of pictures that are unable to be processed by NLP models. OCR was used to analyze the images attached to elog entries. The image data is retrieved similarly to the entry text. We utilized Keras OCR [7] to parse the text from the images. The package has a useful function to annotate images. The plot in Fig. 6 is a machine performance trend graph. The beam intensity on the y-axis is in scientific notation. This causes some confusion in the OCR algorithm. For example, the plus symbol is translated as the letter t and the number eight to a letter b. The y-axis title is vertically written along the graph axis and is also incorrectly identified. The retrieval of images from the elog system uses gif format. Keras OCR does not support this format so images must first be converted to one of Keras supported image formats. The controls system has utilities in most applications that allow users to upload snapshots directly to the elog. It may be more useful to retrieve the data straight from the applications and process it that way, but this was not analyzed yet.

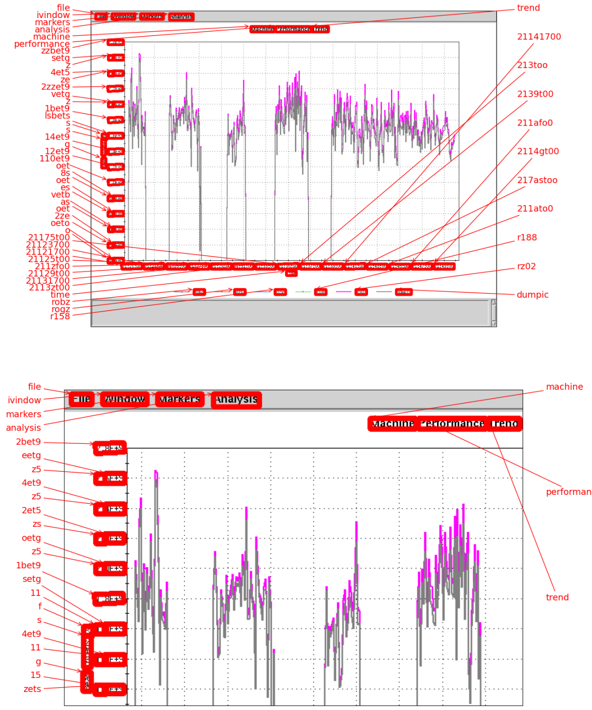


Figure 6: Keras OCR results on an elog image.

WORKFLOW

Since there are entries from the elog dating back to 2013, there are 1.5 million entries to process. The goal of this workflow is to ease training by only processing all of the entries once and then incrementally train the D2V model daily with the new entries. The daily number of entries uploaded to the elog varies from 50 to 1,300 depending on operating status. This reduces the load of the model training on all 1.5 million entries each time. After the model is updated users have the ability to do the perform actions using the most recent set of entries. These include identification of similar entries, sorting entries, and performance statistics. The classification model is trained and stored. The goal is to automate the script to run daily, providing users with the most up to date information.

FUTURE WORK

Web Interface

Ultimately these tools will be implemented into the elog system and the system search feature will run on these models. Until the controls team is confident with performance, a web based search engine will be created to test the search functionality. The website is in development and has a search box to enter text. The search text is parsed ready to be passed to the NLP package. There have been setbacks with loading and training the classification model between storage directories. Once this issue is resolved, development will continue on the web page.

Application Data

Proccessing images to be used for OCR is complicated. Some images have been produced in the unsupported gif format. This requires a potentially time consuming conversion process while increasing the need for additional resource allocations. Additionally, the system has a feature for application image dumps that includes meta data with the image. This allows contextual launching of applications through the elog interface. The path forward is to obtain the meta data from these special images and append it to existing elog entries or create new ones to be processed by the D2V model.

CONCLUSION

Natural language processing makes searching large databases of text much easier. The electronic logbook system is a good example of how these methods improve user's experience by providing better search results when compare to traditional regular expression searches. The LSA and LDA efforts helped understand topics and importance within elog entries, but the multinomial naive bayes classifier allows for more customization with existing elog tags. OCR is a useful technique but there are more useful techniques to be explored using the controls systems meta data. The immediate goal is to provide users with an updated web interface and collect user and system feedback related to performance and accuracy.

REFERENCES

[1] R. Řehůřek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora", in *Proc. LREC 2010 Workshop New Chall. NLP Framew.*, 2010, pp. 45–50. http://is.muni.cz/publication/884893/en

[2] Multiclass text classification with tensorflow, https://github.com/snymanje/MultiClass-Text-Classification-with-Tensorflow

[3] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python", *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011. https://www.jmlr.org/papers/v12/pedregosa11a.html

[4] S. T. Dumais, "Latent semantic analysis", *Annual Review of Information Science and Technology*, vol. 38, no. 1, pp. 188–230, 2004. doi:10.1002/aris.1440380105

[5] Topic modelling with lsa and lda, https://www.kaggle.com/code/rcushen/topic-modelling-with-lsa-and-lda/notebook

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation", *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003. https://www.jmlr.org/papers/v3/blei03a.html

[7] Keras-ocr, https://github.com/faustomorales/keras-ocr