February 2022

U.S. DEPARTMENT OF ENERGY

National Nuclear Security Administration

Advanced Natural Language Processing for Work Breakdown Structures

Technical Paper

Maura Lapoff
Technomics, Inc.

Table of Contents

ABSTRACT 1
1 INTRODUCTION 2
1.1 PURPOSE 2
1.2 BACKGROUND 2
1.2.1 National Nuclear Security Administration 2
1.2.2 Natural Language Processing 3
1.2.3 Artificial Intelligence, Machine Learning, and Deep Learning 3
1.3 ASSESSMENT OF THE STATUS QUO 7
1.4 SCOPE 7
2 APPROACH 8
2.1 DATA EXPORT AND LABELLING 8
3 DATA PRE-PROCESSING 10
3.1 TOKENIZATION 10
3.2 IDENTIFICATION OF ACRONYMS AND ABBREVIATIONS 10
3.3 REMOVAL OF STOP WORDS 10
3.4 STEMMING 11
3.5 RESULTS 11
4 CLUSTERING WORK BREAKDOWN STRUCTURE ELEMENTS 12
4.1 ASSUMPTIONS 12
4.2 METHODOLOGY 12
4.2.1 Non-negative Matrix Factorization 12
4.2.2 Latent Dirichlet Allocation 12
4.3 RESULTS 13
4.3.1 Non-negative Matrix Factorization with Kullback-Leibler Divergence 13
4.3.2 Non-negative Matrix Factorization with Frobenius Norm 13
4.3.3 Latent Dirichlet Allocation 14
5 CLASSIFICATION OF WORK BREAKDOWN STRUCTURE ELEMENTS 15
5.1 ASSUMPTIONS 15
5.2 METHODOLOGY 15
5.2.1 Duplicate Work Breakdown Structure Elements 15
5.2.2 Dense Neural Network 16
5.2.3 Convolutional Neural Network 16
5.2.4 Bidirectional Long Short-term Memory Neural Network 16
5.2.5 Hyperparameter Tuning 17
5.2.6 Ensemble Voting Classifier 18
5.3 RESULTS 18
5.3.1 Default Models 18

5.3.2 Hyperparameter Tuning 18
5.3.3 Ensemble Voting Classifier 19
6 MULTI-INPUT NEURAL NETWORK **21**
6.1 ASSUMPTIONS 21
6.2 METHODOLOGY 21
6.3 RESULTS 23
7 OBSERVATIONS **26**
7.1 UNSUPERVISED LEARNING 26
7.2 SUPERVISED LEARNING 26
7.3 LIMITATIONS 27
8 CONCLUSION **29**
APPENDIX A: REFERENCES 30

List of Figures

Figure 1: Deep learning is a type of machine learning 5
Figure 2: A basic MLP node *i* fires when $yi > 0$ using a step function 5
Figure 3: Rectified Linear Unit activation function 7
Figure 4: Natural language processing stages 8
Figure 5: Example of how Analyst labels raw WBS data based on classification scheme 9
Figure 6: Both the full dataset and the pre-labeled sample dataset undergo pre-processing 10
Figure 7: Clustering of WBS elements using NMF algorithm 13
Figure 8: Clustering of WBS elements using NMF algorithm with Frobenius norm and tf-idf vectorizer 14
Figure 9: LDA model shows slightly different clusters 14
Figure 10: Confusion matrix with class-level accuracies using the ensemble voting classifier model 20
Figure 11: Multi-input Bi-LSTM model shows best performance 25

List of Tables

Table 1: Example of how duplicate WBS element could exist in the dataset 16
Table 2: The softmax activation function predicts the probability of the WBS element belonging to each class 16
Table 3: Hyperparameter combinations used for model tuning 17
Table 4: Validation accuracy of unoptimized supervised machine learning algorithms with hyperparameters 18
Table 5: Validation accuracy for optimized supervised machine learning algorithms 19
using best hyperparameters from model tuning 19

Table 6: Validation accuracy on full training dataset using tuned models 19
Table 7: Validation accuracy on voting classifier 20
Table 8: The multi-input neural network uses a sample of distinct instances of WBS element and cost data 23
Table 9: Model tuning for multi-input neural network 24
Table 10: Model performance on full training dataset..... 24

This page intentionally left blank

ABSTRACT

The National Nuclear Security Administration collects work breakdown structure data for capital asset projects. Implementing the structure across projects grows exponentially in complexity because of varying scope, contextual changes, and vendor requirements. This paper will demonstrate how natural language processing can automate the process of identifying and classifying work breakdown structure elements used for capital asset projects. Natural language processing can improve the Cost Analyst's workflow by reducing time-consuming aspects of his or her work and enable better decision-making when managing capital asset projects.

This paper is nearly identical in scope to the paper entitled "Machine Learning: Classification and Clustering Strategies for Work Breakdown Structures" authored by Maura Lapoff and submitted to the Association for the Advancement of Cost Engineering (AACE) for presentation at the 2022 AACE International Conference and Expo (June 26-28, 2022 in San Antonio, TX).

1 INTRODUCTION

The National Nuclear Security Administration (NNSA) is a semi-autonomous agency within the U.S. Department of Energy (DOE) whose mission includes the following:

- to maintain the nuclear stockpile
- prevent the proliferation of nuclear weapons
- prevent, counter, and respond to nuclear and radiological threats, and
- to provide nuclear power to the U.S. Navy (Reference 9)

NNSA funds capital asset projects at management and operating (M&O) partner sites, which are national security laboratories, plants, and other facilities across the United States that support NNSA's mission. These sites are part of the nuclear security enterprise. M&O partners that run these sites must adhere to DOE policy and contractual agreements, but also have their own internal processes. Therefore, capital asset projects not only vary in scope and cost but also in how they are managed. Currently, capital asset project costs are not tracked using a standard work breakdown structure (WBS) template. This makes it difficult for the Cost Analyst to track and compare costs across different projects across the nuclear security enterprise.

1.1 PURPOSE

The purpose of this technical paper is to demonstrate how to apply natural language processing (NLP) methods to classify WBSs to a standard Level 2 format. These methods can expedite the process of cross-walking different WBS elements to a common WBS, while maintaining high accuracy. This will allow greater understanding and insight into capital asset project costs and enable better decisions in project and portfolio management of NNSA capital asset projects.

1.2 BACKGROUND

1.2.1 National Nuclear Security Administration

Within NNSA is the Office of Programming, Analysis, and Evaluation (PA&E, NA-MB-90), which provides programming services and leads analysis, evaluation, and cost estimating for the Associated Administrator for Management and Budget. The office develops cost estimates in support of the Planning, Programming, Budgeting, And Execution process, capital asset projects, and *Phase 6.x Process* programs[1]. It also leads Analysis of Alternatives for capital asset projects, conducts planning studies, and develops modeling tools to support enterprise decisions and foster a community of excellence within the Office of Management and Budget and across NNSA.

DOE Order 413.3B, Program and Project Management for the Acquisition of Capital Assets, Attachment 2 (Reference 10), defines capital asset projects as "land, structures, equipment and intellectual property…used by the Federal Government and have an estimated useful life of two years or more" and

[1] The *Phase 6.X Process* "provides the framework for nuclear weapons activities (including life extension programs), such as routine maintenance, stockpile evaluation, surveillance, baselining, and annual certification" (Reference 22).

"exclude items acquired for resale in the in the ordinary course of operations or held for the purpose of physical consumption such as operating materials and supplies."

1.2.2 Natural Language Processing

NLP is a type of machine learning (ML) that helps identify, classify, or analyze text and audio language data (Reference 18). For example, common applications include flagging spam emails, returning search engine results, and determining whether social media posts are positive or negative. NLP is inherently interdisciplinary, and its advancements can be attributed to research in computer science, linguistics, statistics, and various other fields.

1.2.3 Artificial Intelligence, Machine Learning, and Deep Learning

Artificial intelligence (AI) is a part of computer science concerned with "machines that respond to stimulation consistent with traditional responses from humans, given the human capacity for contemplation, judgment, and intention" (Reference 27).

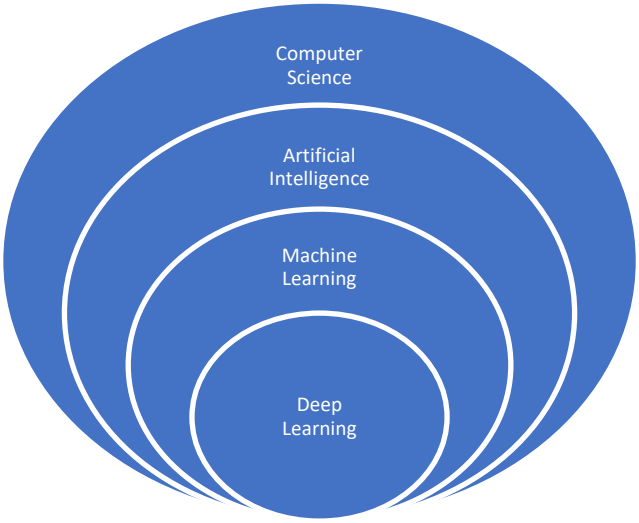
ML is a growing field within AI that broadly refers to any computer programming in which the computer "learns" from data and experience to improve its performance doing certain tasks (Reference 12). ML is useful for categorizing data and predicting outcomes when the mode of data collection is not tightly controlled and the relationships between variables are not necessarily linear (Reference 5).

Deep learning (DL) is a type of ML that uses multiple hidden layers in a neural network to identify complex patterns in data.

Figure 1 illustrates how DL fits under the larger AI umbrella. Rina Dechter coined the term to describe the process of backtracking so that whenever a search algorithm hit a dead end, the algorithm stored the reason for the dead end (Reference 8). The model could then jump back to whichever state had the "shallowest" variables that still satisfied the model's constraints and rerun the search. Today, DL has expanded into a wide field of complex ML algorithms that can be used to identify patterns in data that traditional ML techniques may fail to identify. Like traditional ML, DL can be unsupervised, supervised, or semi-supervised. With DL, the deeper the algorithm is, the more layers there are and the more abstract the features representing relationships in the data become (Reference 17).

Neural networks contain at least three layers of neurons, or nodes. The first layer is called the input layer and the last layer is called the output layer. All the layers between the input and output layers are called hidden layers (Reference 16). Neural networks are often called black boxes because the internal workings of their hidden layers are difficult to see, and their decisions are difficult to understand (Reference 25).

Figure 1: Deep learning is a type of machine learning



A multi-layer perceptron (MLP) is a simple neural network in which each layer in the network consists of the weighted contributions of its nodes (Reference 7). Each node, in turn, operates as a linear model with weights *w* and a bias *b* (Reference 16). Consider a node *i* that receives multiple inputs from the preceding layer. The linear equation in **Figure 2** demonstrates how the weighted sum of the inputs' weights and the bias threshold value determines whether the node fires. In this case, the activation function for the node firing is a step function (equation b). This means that if the solution y_i is greater than zero, the node fires (Reference 16).

Figure 2: A basic MLP node *i* fires when y_i > 0 using a step function

$$y_i = \sum_{i=1}^{k} w_j x_j + b \qquad \text{(a)}$$

$$f(x) = \begin{cases} 1 \mid y_i \geq 0 \\ 0 \mid y_i < 0 \end{cases} \qquad \text{(b)}$$

More advanced neural networks are non-linear, can propagate in both directions, and use different activation functions. For example, a more advanced MLP may replace the step function used for the hidden layers with a Rectified Linear Unit (ReLU) function (Reference 1and Reference 12). Unlike the step function, when y_i > 0, the output is y_i (

Figure 3) (Reference 12).

Figure 3: Rectified Linear Unit activation function

$$f(x) = \begin{cases} y_i \mid y_i \geq 0 \\ 0 \mid y_i < 0 \end{cases}$$

Selecting the appropriate type of neural network architecture (including its components) is essential to achieving accurate results.

1.3 Assessment of the Status Quo

Currently, M&O partner sites submit capital asset project data to a DOE database in compliance with DOE Order 413.3B (Reference 10). Project data includes various reporting requirements such as project estimates, Critical Decision milestone dates, project estimates, and WBSs; however, there is currently no standard WBS template for these projects so comparing costs proves difficult.

1.4 Scope

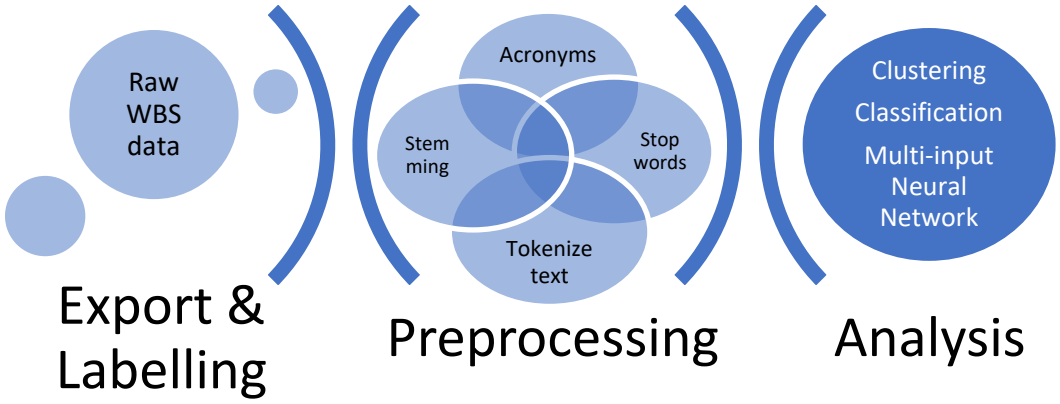
The scope of the analysis described in this paper includes all active, closed, and completed NNSA capital asset projects as defined in DOE Order 413.3B (Reference 10).

2 APPROACH

The NLP project occurs in three stages:

1) Data export and labeling
2) Data Pre-processing
3) Data Analysis (Error! Reference source not found.)

Figure 4: Natural language processing stages



2.1 DATA EXPORT AND LABELLING

A Technomics Analyst supporting PA&E exported 147 raw project files from a DOE project reporting database. The files consisted of WBS data for all active, completed, and cancelled NNSA capital asset projects. The project files were merged into one file with 90,682 WBS elements. Through this process, the Analyst determined that 110 of the files were either empty or only contained Level 1 (Project Name) data. Furthermore, four projects contained WBS elements but not their associated costs.

The Analyst selected ten projects from this export to serve as the sample dataset for the classification procedures. The Analyst consulted with capital asset subject matter experts (SMEs) to identify an appropriate high-level classification scheme for labeling the WBS element data. The final scheme consisted of six labels: Project Engineering and Design (PED), Site Preparation, Procurement, Construction, Project Management, and Start-up. The Analyst then conducted a manual review of the sample data to assign each WBS element to the appropriate label.

Figure 5: Example of how Analyst labels raw WBS data based on classification scheme

WBS element	Label
Facility construction	Construction
Glove box	Procurement
Conceptual Design	Project Engineering and Design

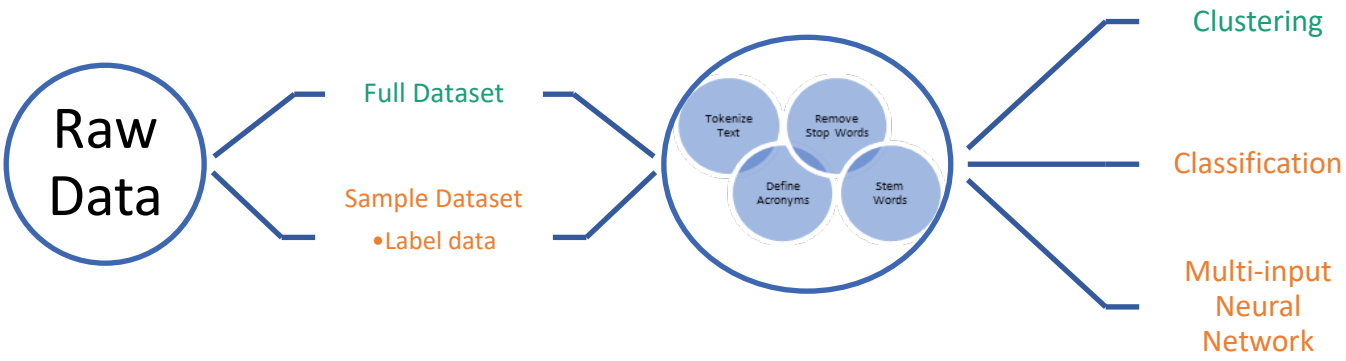
The Analyst searched for duplicate WBS element entries to ensure consistency of labeling within and across the sample projects. The Analyst then cleaned and organized the sample data and set it aside for the classification tasks. Meanwhile, a second Technomics Analyst cleaned and organized the full corpus of 147 project files into one dataset for the clustering tasks.

3 DATA PRE-PROCESSING

After collecting the data for the DOE database, the Analyst Team prepared the data for analysis. This pre-processing consisted of three steps (**Figure 6**):

1) Tokenization
2) Acronym and abbreviation identification
3) Stop word removal
4) Stemming

Figure 6: Both the full dataset and the pre-labeled sample dataset undergo pre-processing



3.1 TOKENIZATION

Tokenization is the process of separating each token from the rest of the WBS element's text (Reference 29). In this analysis, each token represents a word; however, depending on the analysis, tokens can consist of multiple words, phrases, or even sentences. The tokenization process unstitched the dataset into 314,791 tokens, or non-unique words.

3.2 IDENTIFICATION OF ACRONYMS AND ABBREVIATIONS

NLP requires that all acronyms and abbreviations in the text be fully defined prior to analysis. This ensures that the computer knows that phrases like "PM" and "Project Management" or "CON" and "Construction" are equivalent, improving accuracy of the results.

The Analyst identified 296 acronyms or abbreviations that required definition. Some of these shared the same meaning. For example, both "CONSTR" and "CON" referred to "Construction." Therefore, this process added 286 unique definitions to replace the acronyms and abbreviations identified.

3.3 REMOVAL OF STOP WORDS

Stop words are any words that the computer should ignore during the analysis. Generic stop words include commonly used words such as prepositions, while custom stop words are words unique to the analysis

that should be excluded. Here, the Analyst combined a generic list with a custom stop word list. The generic list included 1,149 words from (Reference 28) based on (Reference 19), (Reference 20), and (Reference 24). The custom list consisted of 95 project names, locations, and similar words or phrases selected for removal.

3.4 STEMMING

All words can then be stemmed using Porter's algorithm (Reference 4, Reference 21, and Reference 23). Stemming reduces words to a common form and may help model performance.

3.5 RESULTS

After tokenization, acronym definition, and stop word removal, the Analyst re-stitched the words together. This process removed 96 WBS elements, which were elements that entirely contained stop words marked for removal. Overall, the pre-processing process produced a dataset of 90,584 cleaned WBS elements.

4 CLUSTERING WORK BREAKDOWN STRUCTURE ELEMENTS

Clustering is an *unsupervised learning*[2] method to organize the full dataset into *k* clusters, where the Analyst chooses *k*.

4.1 ASSUMPTIONS

The Analyst Team made the following assumptions to develop the clustering models:

- Removing project names and locations from the text reduces bias and improves classification accuracy.
- Acronyms identified during pre-processing are accurate and comprehensive.

4.2 METHODOLOGY

The unsupervised learning process requires the full corpus for clustering analysis. The corpus consisted of 90,682 lines of text extracted from the cleaned WBS elements. The three clustering algorithms used were two Non-negative Matrix Factorization (NMF) variants and Latent Dirichlet Allocation (LDA). The Analysts chose to cluster the data into six groups ($k = 6$), representing six possible Level 2 buckets for the WBS data.

Clustering algorithms require the use of a vectorizer to identify the most influential words in the corpus. The NMF algorithms use a tf-idf vectorizer to determine the top words represented in each cluster. Alternatively, the LDA algorithm uses a term-frequency (tf) vectorizer, which only uses each word's tf rather than its tf-idf to determine top words within each cluster.

4.2.1 Non-negative Matrix Factorization

NMF uses linear algebra to identify the clusters in the corpus. There are several variations of NMF but this paper focuses on two: NMF with Frobenius norm (NMF-F) and NMF with Kullback-Leibler (NMF-KL) divergence. The two are quite similar but NMF-F assumes that any noise in the data is Gaussian while NMF-KL uses the Kullback-Leibler divergence as its error function (NMF-KL was theoretically proven to be equivalent to LDA's predecessor, pLSI) (Reference 11 and Reference 13).

4.2.2 Latent Dirichlet Allocation

LDA is a generative probabilistic model that uses word (or term) count vectors to identify topics in a corpus (Reference 3). LDA was proposed to address some of the limitations of predecessors Latent Semantic Indexing (LSI) and probabilistic LSI (pLSI). All three methods assume that the corpus is a so-called bag of words, meaning that neither word order nor document order contain inherent value (Reference 3).

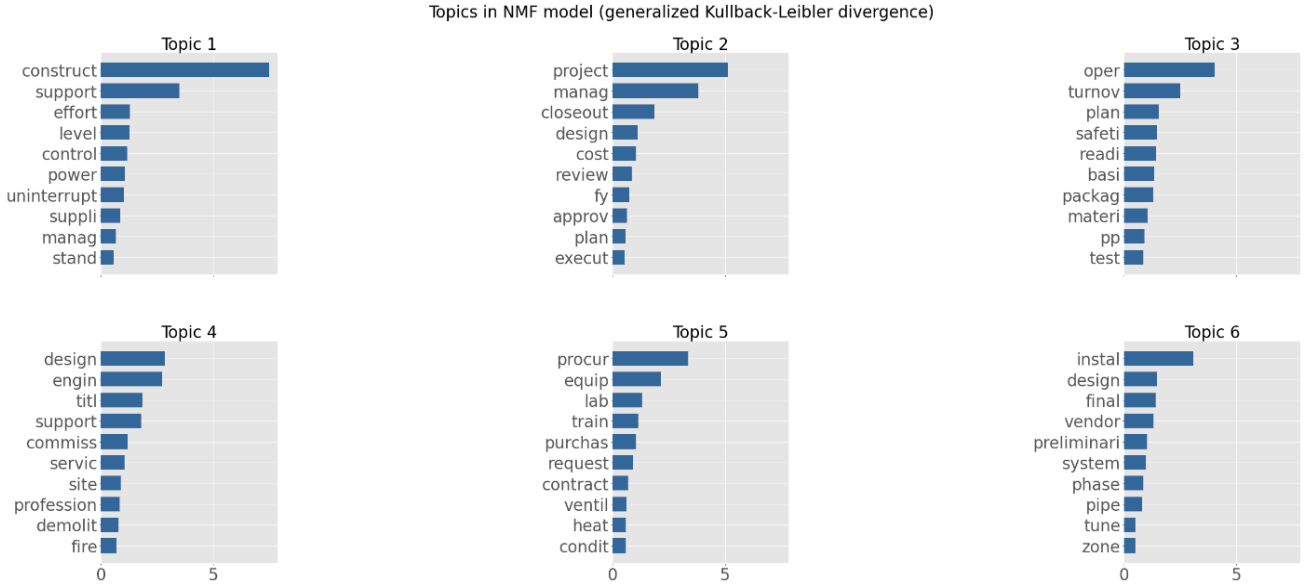
[2] Unsupervised learning is a type of machine learning in which the computer identifies patterns without human input (Reference 12).

4.3 RESULTS

4.3.1 Non-negative Matrix Factorization with Kullback-Leibler Divergence

The unsupervised clustering of text data, or topic modelling, uses the entire corpus of WBS data elements for analysis. Each model obtained results in a matter of seconds. Results for the two NMF clustering algorithms are shown in **Figure 7** and **Figure 8**. Each figure depicts the six clusters identified in the corpus and the top 10 (stemmed) words found in each cluster. Results of the NMF-KL model (Figure 7) suggest that "construction" is the most important word in Topic 1, followed by "support." The second cluster highlights "project" and "management" while the third cluster highlights "operations" and "turnover." Furthermore, the fourth cluster identifies "design" and "engineering" as top words while the fifth cluster finds "procurement," and "equipment" to be of similar importance. Finally, the sixth cluster lists "installation" and "design" as its top two words. This model took 6.4 seconds to run.

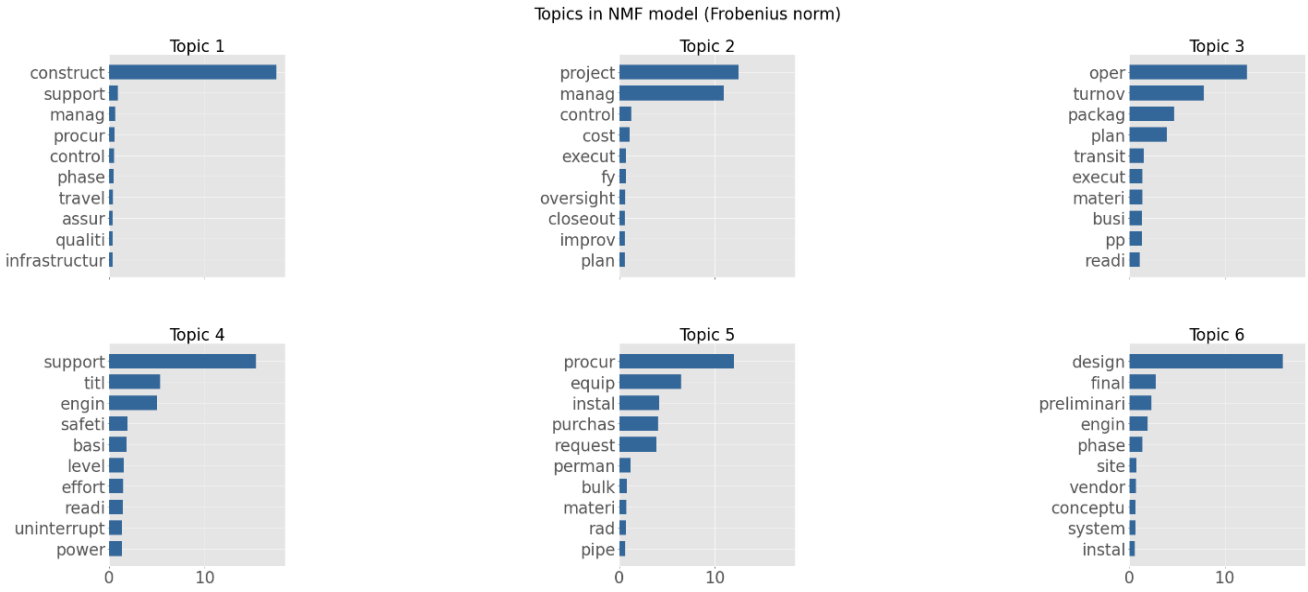
Figure 7: Clustering of WBS elements using NMF algorithm with generalized Kullback-Leibler divergence and tf-idf vectorizer



4.3.2 Non-negative Matrix Factorization with Frobenius Norm

Figure 8 depicts the results from the NMF-F model, which took 3.6 seconds to run. Like the previous NMF algorithm, the first cluster identifies "construction" as the top word, the second cluster identifies "project" and "management," the third cluster identifies "operations" and "turnover," the fourth cluster identifies "support," and the fifth cluster identifies "procurement," and "equipment." Finally, the sixth cluster lists "design" as its top word.

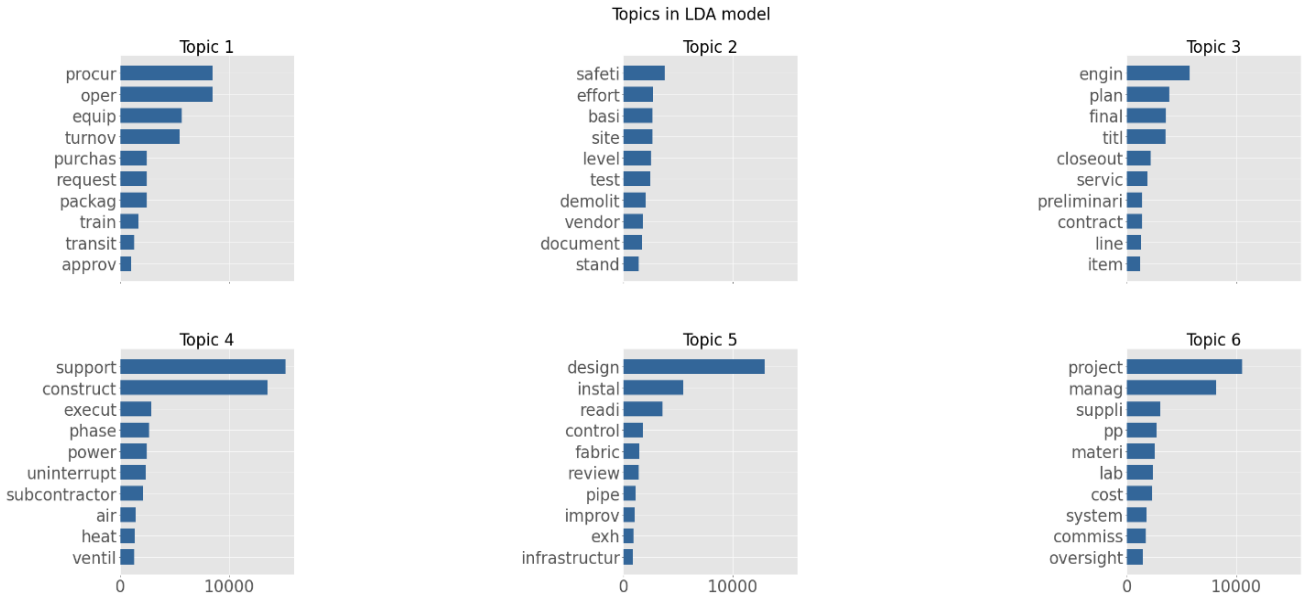
Figure 8: Clustering of WBS elements using NMF algorithm with Frobenius norm and tf-idf vectorizer



4.3.3 Latent Dirichlet Allocation

Figure 9 depicts the results for the LDA algorithm. This model took approximately nine minutes to run. Once again, the six clusters can be summarized by their top words: (1) “procurement,” “equipment,” “turnover” and “operations,” (2) “safety,” (3) “engineering,” (4) “support” and “construction,” (5) “design, and (6) “project” and “management.”

Figure 9: LDA model shows slightly different clusters



5 CLASSIFICATION OF WORK BREAKDOWN STRUCTURE ELEMENTS

5.1 ASSUMPTIONS

The Analyst Team made the following assumptions to develop the classification models:

- All WBS elements in the dataset belong under one of these six labels:
    1) Construction
    2) PED
    3) Procurement
    4) Project Management
    5) Site Preparation
    6) Start-up
- The ten-project sample dataset used for training is representative of the full WBS dataset.
- Removing project names and locations from the sample dataset's text will mitigate bias during training.
- Acronyms identified during pre-processing are accurate and comprehensive.

5.2 METHODOLOGY

The classification portion of the analysis used the pre-labeled sample data from Section 2.1 and can be broken down into the following steps:

1) Removal of duplicate WBS elements
2) Classification using default hyperparameters
3) Hyperparameter tuning
4) Ensemble voting classifier

5.2.1 Duplicate Work Breakdown Structure Elements

Classification is a type of supervised learning, which means that the algorithm requires a pre-labeled sample to learn how to classify the remaining, unlabeled data (Reference 12). In this project, the ten-project sample data labeled in Section 2.1 consisted of 1,500 WBS elements; however, many of these instances were duplicates. This was usually because each project file was actually a compilation of all Integrated Program Management Reports to date for that project. Therefore, the same WBS element could appear for each month that it incurred a cost. For example, for a WBS in **Table 1**, the demolition of Building 1a takes one month but demolishing Building 2a spans two months. The current classification tasks will only use the text data for analysis and do not need the date or Actual Cost of Work Performed (ACWP). Removing the duplicate elements leaves 761 instances of WBS element lines of text. This reduced sample then underwent an 80:20 split to create a training set and testing set.

Table 1: Example of how duplicate WBS element could exist in the dataset

Label	WBS Element	Date (DDMMYYYY)	ACWP ($K)
Site Preparation	Building 1a demolition	05012019	100
Site Preparation	Building 2a demolition	05012019	110
Site Preparation	Building 2a demolition	06012019	50

The dense neural network (DNN), convolutional neural network (CNN), and bidirectional long short-term memory (Bi-LSTM) algorithms contain different model architectures. The first implementation of each algorithm uses the default structures and hyperparameter values. Then, the models undergo tuning to further improve results.

5.2.2 Dense Neural Network

The only hidden layers in DNNs are known as dense layers. Dense layers are considered dense because each of their inputs is connected to each of the outputs in the next layer. Dropout layers typically follow each dense layer to help prevent model overfitting; a dropout layer with a 50 percent dropout rate randomly ignores half the input neurons during each training step (Reference 12).

The DNN used here consisted of an input object, an embedding layer, a one-dimensional global max pooling layer, a dense hidden layer using a ReLU activation function, and a dropout layer with a 50 percent dropout rate. The classification or prediction layer consists of a softmax activation function, which predicts the probability that each WBS element belongs to one of the six labels (Reference 12).

Table 2: The softmax activation function predicts the probability of the WBS element belonging to each class

WBS element	Prediction					
	PED	Construction	Project Management	Site Preparation	Procurement	Start-up
Glovebox purchase 1	.07	.10	.03	.10	.70	.05

5.2.3 Convolutional Neural Network

CNNs are particularly useful for recognizing complex patterns. Originally designed for image processing, their architecture loosely mimics how the brain processes images using receptive fields; convolutional layers differ from dense layers because they only receive inputs within their receptive fields rather than from all inputs in the previous layers (Reference 12).

The CNN used here included an input object and embedding layer, a dropout layer, two one-dimensional convolutional layers, a global max pooling layer, a dense layer, and another dropout layer. Similar to the DNN, CNN's hidden layers use a ReLU activation function while the output layer uses a softmax function.

5.2.4 Bidirectional Long Short-term Memory Neural Network

Recurrent Neural Networks (RNNs) are known for prediction, especially of time series data but also any of sequential data that vary in length, including text. This makes RNNs great for predicting the rest of a

sentence, for example (Reference 12). An RNN contains layers that have inputs from both the previous layer in the current time step (t) and the current layer from the previous time step (t-1). Therefore, recurrent neurons are also called “memory cells”; however, basic RNNs take a long time to train, and struggle with memory the further along in the time step the training goes. LSTM cells are an updated type of RNN that do not share the same memory issues as basic RNNs and also can forge” unimportant information, which improves efficiency (Reference 12 and Reference 15). Bidirectional LSTM cells (Bi-LSTM) are even better, allowing information to be fed into the layer from both directions, increasing the information the model can use during training (Reference 14).

The Bi-LSTM model consists of an input object, an embedding layer, two Bi-LSTM layers, and a dropout layer. Since this model consists of recurrent neurons, it uses separate activation functions for each direction; a hyperbolic tangent function (tanh) for the forward step and a softmax function for the recurrent step (Reference 6). Once again, the output layer uses a softmax function.

5.2.5 Hyperparameter Tuning

Model tuning is the process of testing different combinations of hyperparameter values to find the combination that results in the most accurate model. One method of model tuning is Grid Search cross-validation (CV), which searches across the entire hyperparameter space. For example, a Grid Search of a DNN model with two parameters with two hyperparameters each and three-fold CV results in four trials and 12 predictions; however, a model with four parameters with five hyperparameters each would have 1,024 trials and 3,072 model predictions. The Grid Search method produces accurate results, but quickly becomes computationally expensive as the number of hyperparameters increases. An alternative option, Random Search CV, randomly chooses which trials to run. This method improves efficiency, but at the expense of model accuracy. Ultimately, the analysis employs Grid Search CV model tuning since the sizes of the dataset and parameter space are both relatively small.

Table 3 lists the hyperparameters evaluated during the tuning process with Grid Search (three-fold CV).

Table 3: Hyperparameter combinations used for model tuning

Neural Network	Hyperparameter Values	Combinations
DNN	Embedding Dimension: [5, 10, 20] Hidden Layers: [1, 2] Neurons: [10, 20, 30, 40] Learning Rate: [.0003, .003, .03]	72
CNN	Embedding Dimension: [5, 10, 20] Hidden Layers: [1, 2] Filters: [80, 90, 100, 120, 130] K size: [5, 6, 7, 8, 9, 10] Neurons per Hidden Layer: [10, 20, 30, 40] Learning Rate: [.0003, .003, .03]	2160
Bi-LSTM	Embedding dimension: [5, 10, 15] Hidden Layers: [1, 2]	72

Neurons per Hidden Layer: [20, 25, 30, 35]
Learning Rate: [.0003, .003, .03]

5.2.6 Ensemble Voting Classifier

After the models are tuned and optimal hyperparameters are identified, a voting classifier combines the three models into an ensemble model. The idea behind this is that each classification algorithm has its own biases that lead to error so building a model that combines different algorithms may improve the performance over using one algorithm, alone (Reference 12). The analysis considers two ensemble models: the first equally weights all three models' contributions while the second leans more heavily on the better performing models.

5.3 RESULTS

5.3.1 Default Models

Table 4 provides the results of the three unoptimized models. The DNN model achieves the highest accuracy (73 percent) and also runs much faster than the CNN and Bi-LSTM models.

Table 4: Validation accuracy of unoptimized supervised machine learning algorithms with hyperparameters

Model	Default Hyperparameters	Accuracy	Time (s)
DNN	Embedding Dimension: 100 1 Dense Layer 30 neurons	73.0%	4.4
CNN	Embedding Dimension: 10 2 Convolutional Layers Filter: 128 Kernel size: 7 1 Dense Layer 128 neurons	57.2%	13.2
Bi-LSTM	Embedding Dimension: 128 2 Bi-LSTM Layers 64 neurons per layer	69.7%	98.6

5.3.2 Hyperparameter Tuning

Table 5 shows that tuning improves the performance for all three models; DNN accuracy increases to 74.2 percent, CNN accuracy increases to 72.6 percent, and the Bi-LSTM accuracy increases to 74.9 percent. Rerunning the tuned models on the full training dataset confirms that tuning improves performance across all three models (**Table 6**). Model tuning requires more time as the process requires

exploring all combinations of hyperparameters. Tuning the DNN model takes about three minutes while the Bi-LSTM model requires almost an hour, and the CNN model requires several hours.

Table 5: Validation accuracy for optimized supervised machine learning algorithms using best hyperparameters from model tuning

Model	Best Hyperparameters	Accuracy
DNN	Embedding Layer Dimension: 20 1 Dense Layer 40 neurons Learning Rate: 0.003	74.2%
CNN	Embedding Layer Dimension: 20 1 Convolutional Layer: Filter: 130 Kernel size: 7 1 Dense Layer 40 neurons Learning Rate: 0.003	72.6%
Bi-LSTM	Embedding Layer Dimension: 15 1 LSTM Layer 20 neurons Learning Rate: 0.03	74.9%

Table 6: Validation accuracy on full training dataset using tuned models

Model	Accuracy
DNN	77.6%
CNN	78.3%
Bi-LSTM	81.6%

5.3.3 Ensemble Voting Classifier

Table 7 shows the results of the ensemble voting classifiers, which combine the three models. A classifier that weights the three models equally increases accuracy to 82.2 percent; adjusting the relative weights to emphasize the higher-performing CNN and Bi-LSTM models further increases accuracy to 83.6 percent. The voting classifier process finished in 48 seconds.

A confusion matrix displays the predicted versus actual classifications across the six Level 2 buckets (

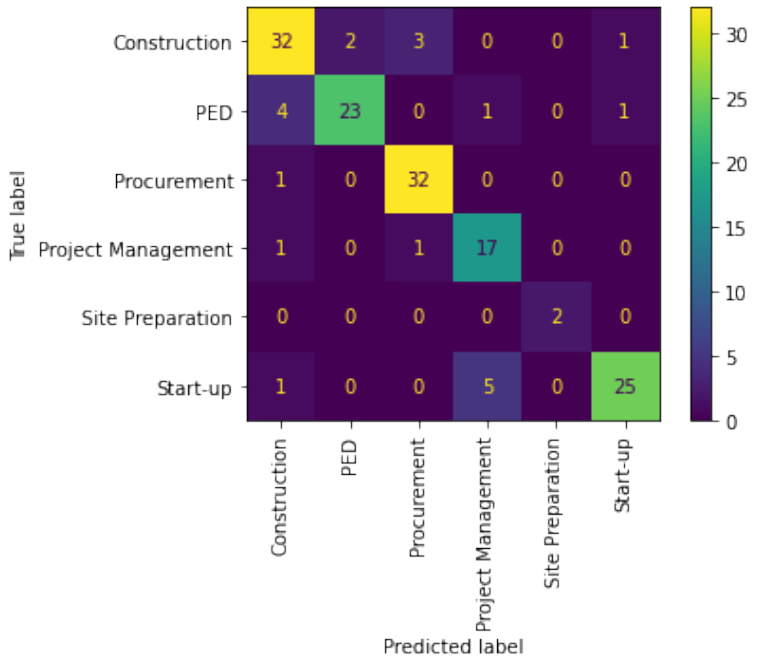
Figure 10). Although the classifier's macro-average is 83.4 percent, there is variation at the class-level. Most misclassifications occur while predicting PED and Start-up, so their accuracies are the lowest at 79.3 percent and 80.6 percent, respectively. Project Management and Construction perform better with

accuracies of 89.5 percent and 84.9 percent, respectively. Procurement and Site Preparation show the greatest performance with accuracies of 97 percent and 100 percent, respectively.

Table 7: Validation accuracy on voting classifier

Model	Weights			Accuracy	Time (s)
	DNN	CNN	Bi-LSTM		
Voting Classifier	0.33	0.33	0.33	82.2%	48.0
	0.24	0.38	0.38	83.6%	

Figure 10: Confusion matrix with class-level accuracies using the ensemble voting classifier model



Class	Accuracy
Construction	84.2%
PED	79.3%
Procurement	97.0%
Project Management	89.5%
Site Preparation	100.0%
Start-up	80.6%

6 MULTI-INPUT NEURAL NETWORK

A multi-input neural network is a neural network with more than one input layer. This means that multi-input neural networks can accept mixed data sources (e.g., image, categorical, and continuous data) as inputs to produce a prediction or classification output. This type of neural network can help determine whether adding cost data to the sample improves classification accuracy. The clustering and classification tasks conducted thus far ignored costs and any other numerical data as inputs. Here, the cumulative actuals associated with the sample WBS element data are reintroduced to the dataset. The two variables can then feed the models.

6.1 ASSUMPTIONS

Similar to the original classification tasks, the following are the assumptions for developing the multi-input neural networks:

- All WBS elements in the dataset belong under one of these six labels:
    7) Construction
    8) PED
    9) Procurement
    10) Project Management
    11) Site Preparation
    12) Start-up
- The ten-project sample dataset used for training is representative of the full WBS dataset.
- Removing project names and locations from the sample dataset's text will mitigate bias during training.
- Acronyms identified during pre-processing are accurate and comprehensive.

6.2 METHODOLOGY

The methodology for this task resembles that of the previous classification tasks but must now account for the newly added cost data.

The original sample containing 1,500 instances of WBS element data must now be cleaned so that only unique WBS text and cost combinations remain. This means that duplicate WBS text elements may stay in the dataset as long as their associated costs differ. For example,

Table 8 contains four instances containing both WBS element and cost data. The first instance is unique on both components. The second instance shares the same text data as the third and fourth instances but has a unique cost. The third and fourth instances, however, share both WBS element and cost data, and therefore the fourth instance is considered a duplicate and is dropped from the sample dataset.

Table 8: The multi-input neural network uses a sample of distinct instances of WBS element and cost data

	Label	WBS Element	Date (DDMMYYYY)	Cumulative ACWP ($K)
1	Site Preparation	Building 1a demolition	05012019	100
2	Site Preparation	Building 2a demolition	05012019	110
3	Site Preparation	Building 2a demolition	06012019	50
4	Site Preparation	Building 2a demolition	07012019	50

This process results in a dataset of 1,440 unique instances of WBS element and cost data. The cost data is normalized and the two inputs are fed into the DNN, CNN, and Bi-LSTM models.

Once again, the models will be tuned to find appropriate hyperparameter values; however, the process will use Random Search instead of Grid Search to account for the increased parameter space of the multi-input models.

6.3 RESULTS

The new models are retuned and evaluated for accuracy (**Table 9**). Rerunning the tuned models on the full dataset results in slightly lower performance, with the Bi-LSTM model still resulting in highest accuracy (**Table 10**).

Table 9: Model tuning for multi-input neural network

Model	Best Hyperparameters	Accuracy	Time (s)

DNN	Embedding Layer Dimension = 50
	• 1 Text Layer
	  ○ 30 neurons
	  ○ Dropout rate: 50%
	• 1 Cost Layer:
	  ○ 6 neurons
	  ○ Dropout rate: 80%
		96.4%	254

CNN	Embedding Layer Dimension: 10
	• 1 Text Layer:
	  ○ Filters: 30
	  ○ Kernels: 5
	  ○ Stride: 1
	  ○ Dropout rate: 50%
	• 1 Cost Layer:
	  ○ Neurons: 6
	  ○ Dropout rate: 80%
		96.8%	795

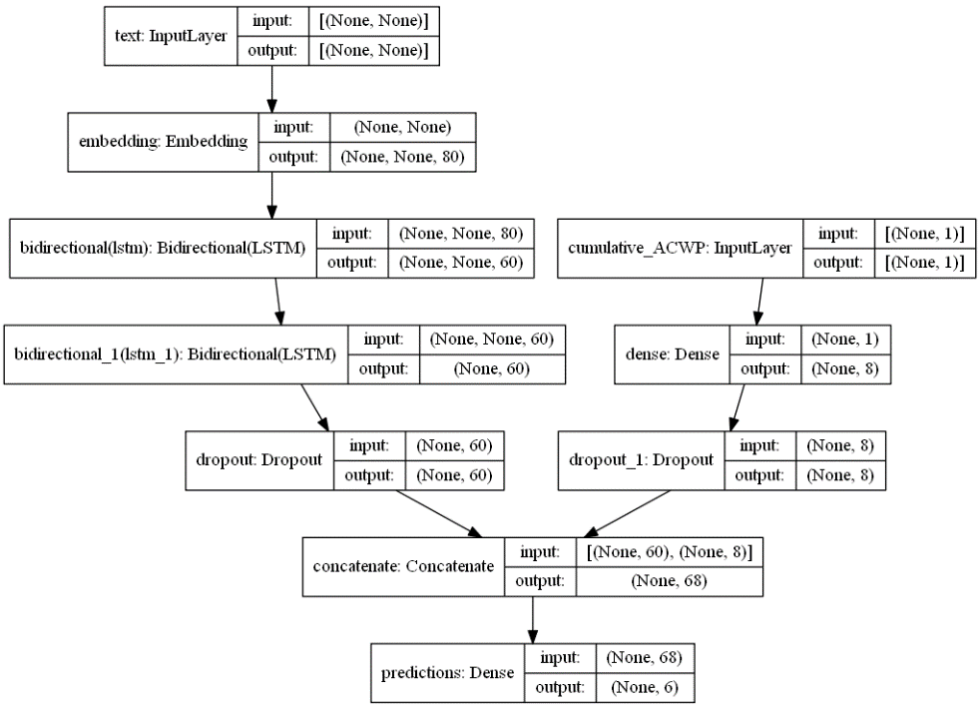
Bi-LSTM	Embedding Layer Dimension: 80
	• 1 Text Layer:
	  ○ Neurons: 30
	  ○ Dropout rate: 50%
	• 1 Cost Layer:
	  ○ Neurons: 8
	  ○ Dropout rate: 70%
		97.6%	5023

Table 10: Model performance on full training dataset

Model	Accuracy	Time (s)
Multi-input DNN	88.5%	20.9
Multi-input CNN	91.0%	1.34
Multi-input Bi-LSTM	93.1%	27.6

Figure 11 depicts the architecture of the multi-input Bi-LSTM model, which has an accuracy of 93.1 percent. All three multi-input models perform better than the weighted ensemble voting classifier, which was 83.6 percent accurate.

Figure 11: Multi-input Bi-LSTM model shows best performance



7 OBSERVATIONS

7.1 UNSUPERVISED LEARNING

The clustering models had somewhat similar results, especially between the NMF models. Each model identified six clusters, and there was agreement on topics focusing on construction, project management, and design. The NMF models identified a cluster focusing on turnover to operations while the LDA model identified a cluster that seemed to refer to safety basis and level of effort. Within-cluster variation also occurred. For example, the models did not agree what the design cluster contained; the NMF-F model associated design with its preliminary and final stages whereas the NMF-KL model associated it with engineering and installation. The LDA model only associated it with installation and not engineering. The Analyst who labeled the ten-project sample data also considered whether design and engineering should be separate categories, ultimately combining them into PED.

Similar differences emerged regarding support, and it appeared that the model struggled to determine whether support would serve as its own cluster or belonged within other clusters, such as construction. The Analyst who labeled the ten-project sample for classification similarly struggled with whether to create a separate support category. Ultimately, the final six categories used for data labeling were Construction, PED, Project Management, Procurement, Site Preparation, and Start-up. These are very similar to what was identified in the clustering results. In addition to the equivalently named categories, site preparation may refer to LDA's Topic 2 (consisting of safety basis, demolition, testing, etc.), and start-up is thematically synonymous with the turnover to operations topic. Ultimately, the clustering results validated the Analyst's classification scheme, especially the NMF models, which also ran much faster than the LDA model.

Still, both the Analyst and the clustering algorithm used six groups as Level 2 categories. Future work should consider whether changing the number of buckets improves or inhibits consistency between the computer and Analyst's schemes.

7.2 SUPERVISED LEARNING

The classification of WBS elements using a sample dataset proved successful once tuned. The original models showed a gap in performance between the worst (CNN) and best (DNN) performers; however, model tuning narrowed this gap, ultimately resulting in the Bi-LSTM model with the highest accuracy at approximately 82 percent. The model tuning process also identified that the simpler models (fewer hidden layers and neurons) tended to perform better; still the best model was the weighted ensemble model, which leveraged all three classification models. Although the tuning process took time, it was a necessary step to identify the optimal model hyperparameters values. Once identified, running the three tuned models and the two voting classifiers together for comparison took less than a minute.

Importantly, the addition of cumulative cost data further improved model performance. The three models even performed better than the ensemble models, demonstrating the utility of using both text and quantitative data to classify WBS elements to a Level 2 bucketing scheme.

7.3 LIMITATIONS

Both the supervised and unsupervised learning processes had limitations driven by the models' assumptions. For instance, it is possible that removing certain stop words during pre-processing hindered results, especially the custom list consisting of project names, locations, and other words the Analyst deemed likely to bias results toward the sample projects. It is possible that not all stop words were identified, since that would require exploring the entire dataset. Similarly, acronyms may have been improperly defined or ignored entirely; definitions may have varied between projects and the sample dataset may have not included all acronyms in the full dataset.

Selection bias is a common theme here and increasing the sample size used may improve the ability to generalize the models' results. Still, the time needed to label more of the dataset may offset the time savings of using NLP, though perhaps not the accuracy.

The use of unique WBS instances further reduced sample size and, as the confusion matrix demonstrates, only two instances of site preparation existed in the WBS sample's validation dataset. This suggests that increasing the sample size may be necessary to provide enough instances to train on the smaller buckets. Of course, the Analyst may have incorrectly labeled the instances of the sample. Input from SMEs likely mitigated this, but human error is always possible.

Certain assumptions also warrant further consideration. The clustering analysis assumes that stemming the words improves the clustering results by reducing complexity in the dataset and enabling the computer to better recognize word patterns. Future work can explore whether keeping the full version of words in the dataset changes the topic outcomes.

In fact, all words in the sample dataset were kept in their original (unstemmed) forms for classification. In this case, the Analyst assumed that the DL classification algorithms employed would be sophisticated enough to recognize patterns in the data even without stemming; therefore, future work can explore whether stemming the sample data improves classification performance.

As a final note, the ACWP values employed in the multi-input neural networks were cumulative to-date totals instead of current reporting period costs due to the lack of sufficient instances of current costs. If more data were available, current costs would have been used.

There are several other NLP techniques that this analysis did not employ but could be considered in future work. For example, an alternative to word stemming is lemmatization, where a word is reduced to its base rather than its stem (Reference 2). For comparison, lemmatization reduces both "is" and "are" to "be" while stemming reduces them to "i" and "ar," respectively.

In the classification analysis, the training process included training the vectorizers based on the words included in the sample. An alternative option would have been to use pre-trained word embeddings, which come from other NLP tasks. This can help expedite model training but risks inhibiting results when the corpus of interest includes vocabulary that are not well-represented in the pre-trained word embeddings (Reference 26) Future work can compare the current results to those when using pre-trained word embeddings.

There are many more model architectures and hyperparameter combinations that could be explored as part of this analysis. For example, variations in dropout rates and activation functions are opportunities

for further exploration, though any increase to the parameter space will increase the time needed for tuning.

The general theme of these models was the shallower, the better. Models performed best with only one hidden layer, likely due to the basic syntax contained in the WBS elements. Text data containing more nuanced or abstract language may require deeper models.

8 CONCLUSION

The NNSA supports various capital acquisition projects managed by M&O partners. These projects can vary widely in size, cost, scope, requirements, and management practice. This makes tracking costs difficult since WBS element data can vary greatly across projects, especially between projects managed by different M&O partners.

This paper explored the use of unsupervised and supervised learning strategies to identify a classification schema to categorize WBS data into six Level 2 WBS buckets. The unsupervised learning process successfully employed three clustering algorithms (NMF-F, NFM-KL, and LDA) to identify possible Level 2 buckets by analyzing the entire available WBS data through topic modeling. The results generally complemented the six-class schema previously selected to train the supervised classification models. The three supervised learning models (DNN, CNN, and Bi-LSTM) performed well once the models were tuned to find optimal hyperparameters; however, the best model performance came from a weighted ensemble model that leveraged all three tuned classification models for optimal performance.

Although there are many opportunities for future work, the clustering models helped validate the choice of Construction, PED, Procurement, Project Management, Site Preparation, and Start-up as possible Level 2 WBS categories. The classification models successfully used these categories to classify the remaining unlabeled WBS element text data. Now, capital acquisition projects can be efficiently and effectively mapped to a standard Level 2 WBS template through automation. This advance means that NNSA can better understand capital asset project costs, enabling more informed project and portfolio management decision-making.

APPENDIX A: REFERENCES

1. Agarap, Abien Fred. 2018. "Deep Learning using Rectified Linear Units (ReLU)." *arXiv preprint.* doi:arXiv:1803.08375.
2. Balakrishnan, Vimala, and Ethel Lloyd-Yemoh. 2014. "Stemming and lemmatization: A comparison of retrieval performances." *Proceedings of SCEI Seoul Conferences.* Seoul.
3. Blei, David M, Andrew Y Ng, and Michael I Jordan. 2003. "Latent Dirichlet Allocation." *Journal of Machine Learning Research* 3: 993-1022.
4. Bouchet-Valat, Milan. 2020. "SnowballC: Snowball Stemmers Based on the C 'libstemmer' UTF-8 Library." https://CRAN.R-project.org/package=SnowballC.
5. Bzdok, D, N Altman, and M Krywinski. 2018. "Statistics versus Machine Learning." *Nature Methods* 233-234.
6. Chollet, François. 2015. *LSTM layer.* Accessed January 31, 2022. https://keras.io/api/layers/recurrent_layers/lstm/.
7. Dayhoff, Judith E, and James M DeLeo. 2001. "Artificial neural networks: Opening the black box." *Supplement: Conference on Prognostic Factors and Staging in Cancer Management: Contributions to Artificial Neural Networks and Other Statistical Methods* 91 (8): 1615-1635.
8. Dechter, Rina. 1986. "Learning while searching in constraint-satisfaction problems." *Proceedings of the 5th National Conference on Artificial Intelligence.* Philadelphia. 178-183.
9. Department of Energy. 2022. *Missions.* https://www.energy.gov/nnsa/missions.
10. Department of Energy Office of Project Management. 2021. "DOE Order 413.3B: Program and Project Management for the Acquisition of Capital Assets." *DOE Directives.* Washington, DC: Department of Energy, January 12. https://www.directives.doe.gov/directives-documents/400-series/0413.3-BOrder-b-chg6-ltdchg/@@images/file.
11. Ding, Chris, Tao Li, and Wei Peng. 2008. "On the equivalence between non-negative matrix factorization and probabilistic latent semantic indexing." *Computational Statistics & Data Analysis* 52 (8): 3913-3927.
12. Géron, Aurélien. 2017. *Hands-on machine learning with Scikit-Learn and TensorFlow: Concepts, tools, and techniques to build intelligent systems.* O'Reilly Media.
13. Gillis, Nicolas. 2014. "The Why and How of Nonnegative Matrix Factorization." In *Regularization, Optimization, Kernels, and Support Vector Machines*, 257-291. New York: Chapman and Hall/CRC.
14. Graves, Alex, and Jurgen Schmidhuber. 2005. "Framewise phoneme classification with bidirectional LSTM and other neural network architectures." *Neural Networks* 18 (5-6): 602-610.
15. Hochreiter, Sepp, and Jürgen Schmidhuber. 1997. "Long Short-term Memory." *Neural Computation* 9 (8): 1735-1780.
16. IBM Cloud Education. 2020. *Neural Networks.* IBM. August 17. Accessed January 31, 2021. https://www.ibm.com/cloud/learn/neural-networks#toc-how-do-neu-vMq6OP-P.
17. Ilin, Roman, Thomas Watson, and Robert Kozma. 2017. "Abstraction Hierarchy in Deep Learning Neural Networks." *Proceedings of INNS/IEEE Int. Conference on Neural Networks.* Anchorage.
18. Joshi, Aravind K. 1991. "Natural Language Processing." *Nature* 253 (5025): 1242-1249.
19. Lewis, David D, Yiming Yang, Tony G Rose, and Fan Li. 2004. "RCV1: A New Benchmark Collection for Text Categorization Research." *Journal of Machine Learning Research* 5: 361-397. https://www.jmlr.org/papers/volume5/lewis04a/lewis04a.pdf.

20. Lex-tek International. n.d. *Onix.* http://www.lextek.com/manuals/onix/stopwords1.html.
21. Lovins, Julie Beth. 1968. "Development of a Stemming Algorithm." *Mechanical Translation and Computational Linguistics* 11 (1-2): 22-31.
22. NNSA Office of Cost Estimating and Program Evaluation. 2017. *Responsibilities for Indepedent Cost Estimates.* January 10.
23. Porter, M F. 1980. "An Algorithm for Suffix Stripping." *Program* 14 (3): 130-137.
24. Porter, Martin, and Richard Boulton. n.d. *Snowball.* Accessed December 21, 2021. http://snowball.tartarus.org/algorithms/english/stop.txt.
25. Qiu, F, and J R Jensen. 2004. "Opening the black box of neural networks for remote sensing classification." *International Journal of Remote Sensing* 25 (9): 1749-1768.
26. Rios, Anthony, and Brandon Lwowski. 2020. "An Empirical Study of the Downstream Reliability of Pre-Trained Word Embeddings." *Proceedings of the 28th International Conference on Computational Linguistics.* Barcelona.
27. Shukla, Shubhendu S, and J Vijay. 2013. "Applicability of artificial intelligence in different fields of life." *International Journal of Scientific Engineering Research* 1 (1): 28-35.
28. Silge, Julia, and David Robinson. 2016. "tidytext: Text Mining and Analysis Using Tidy Data Principles in R." *JOSS* (The Open Journal) 1 (3). doi:10.21105/joss.00037.
29. Webster, Jonathan J, and Chunya Kit. 1992. "Tokenization as the Initial Phase in NLP." *COLING* 4: 1106-1110.