Presented at the ICEAA 2023 Professional Development & Training Workshop - www.iceaaonline.com/sat2023

February 2023

U.S. DEPARTMENT OF ENERGY

National Nuclear Security Administration

All in the Hierarchy: Meta-Estimators to Standardize Work Breakdown Structures

Technical Paper

Maura Lapoff
Patrick Shelton
Technomics, Inc.

Table of Contents

List of Figures

List of Tables

This page intentionally left blank

ABSTRACT

The National Nuclear Security Administration has demonstrated how to use Machine Learning and Natural Language Processing to map disparate cost data to a standard, high-level Work Breakdown Structure. However, mapping data to deeper WBS levels becomes increasingly complex due to the hierarchical relationship between levels, rendering common machine learning models inadequate. Here we demonstrate how to implement a Hierarchical Classification Machine Learning scheme to map multi-level, hierarchical cost data to a common WBS.

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
- to prevent the proliferation of nuclear weapons
- to prevent, counter, and respond to nuclear and radiological threats
- to provide nuclear power to the U.S. Navy (Reference 5).

1.1 Mission

This mission takes place within the Nuclear Security Enterprise (NSE)–a network of national laboratories, production and testing plants, and other national security facilities across the United States, which are collectively known as the NSE. These sites are run by Managing and Operating (M&O) contractors, each responsible for executing capital asset projects that enhance and expand NNSA's ability to achieve its mission. While M&Os must adhere to DOE policy and contractual agreements, each has its own internal processes, so the management and tracking of capital asset projects varies between sites. This variation is encouraged by a gap in DOE policy that does not mandate a standard work breakdown structure (WBS) for cost accounting and earned value management (EVM). This environment makes it more difficult for NNSA to track, compare, and benchmark costs across different projects, sites, and over time.

1.2 Purpose

The purpose of this technical paper is to demonstrate how to apply hierarchical classification and natural language processing (NLP) methods to classify WBSs to a standard, multi-level format. A previous ICEAA paper (Reference 12) demonstrated how NNSA successfully used Machine Learning (ML) and NLP to automate the classification of disparate cost data to a common, high-level, WBS. However, moving deeper into the WBS becomes increasingly difficult due to the path-dependent, hierarchical nature of the data. This paper expands upon the previous work by demonstrating how hierarchical classification can benefit the cost community by standardizing inconsistent cost data to a common, multi-level WBS.

1.3 Background

1.3.1 National Nuclear Security Administration and the Department of Energy

Two offices within NNSA conduct cost estimates for capital asset projects. The Office of Cost Estimation and Program Evaluation (CEPE, NA-1.3) is responsible for independent cost estimation and analysis in support of the NSE (References 6 and 17). The Office of Programming, Analysis, and Evaluation, (PA&E, NA-MB-90), meanwhile, conducts programmatic cost estimates in support of the Planning, Programming, Budgeting, and Execution (PPBE) process—primarily early-stage estimates in the form of planning studies and Analyses of Alternatives (AoAs). These offices use similar data sources to perform their analyses, including the cost data employed in this paper, which is submitted to the Project

Assessment and Reporting System (PARS) maintained by the DOE Office of Project Management (DOE-PM) (References 7and 8).

NNSA does not currently have a mandated standard WBS for capital asset projects. However, two recent PA&E efforts have contributed to developing such a WBS,

1. The ongoing NLP work described here and in Reference 12, which resulted in a high-level classification model, hereafter referred to as Model 1.
2. The Capital Acquisition Estimating Framework (CAEF), developed in partnership with the Cost Estimating and Analysis Group (CEAG), which is a guidance document for cost estimating for NNSA capital asset projects (Reference 4).

Each effort proposes a different standard WBS. To better align and promote consistency between the efforts, the NLP analysts incorporated the CAEF's recommended scheme as the new starting point for Model 2, the subject of this paper.

Although NNSA is a semi-autonomous agency within the DOE, its capital projects still fall under the purview of DOE-PM, which, in addition to managing PARS, develops "policy, requirements and guidance for the planning and management of capital asset projects" (Reference 7). Therefore, the implications of Model 2 extend beyond NNSA and may serve as an opportunity for further alignment with offices outside PA&E.

1.3.2 Assessment of the Status Quo

Per DOE Order 413.3b (Reference 7), every project with a Total Project Cost (TPC) greater than $10 million must report performance, including planned costs and categories, at the approval of mission need (CD-0), to DOE PM's PARS database. Projects with a TPC greater than $20 million then must conduct monthly earned value reporting to PARS after the approval of the performance baseline and preliminary design (CD-2). While it is not a policy requirement, most projects conduct some degree of earned value reporting to PARS prior to CD-2. With the requirement to report planned costs and categories often several years before actually starting earned value reporting, cost categories in PARS reports can become confusing, bloated, and misleading for several reasons: (1) a project's scope can change between CD-0 and CD-2; (2) staff turnover can cause knowledge loss; and (3) individual M&O earned value (EV) systems can update and change. These issues are exacerbated by the lack of a standard WBS format, as it affords EV managers the flexibility to map costs to the categories they see fit, rather than a pre-determined structure and despite the fact that every capital project has the same overarching process.

This system of developing a WBS and element names unique to every project creates interpretation issues for any analyst, manager, or oversight body looking to evaluate a project's cost and schedule performance. PARS reporting rarely includes a WBS dictionary, so anyone without access to the EV manager and project team or without pre-existing knowledge of the project often will have to make assumptions about how to interpret WBS elements. The English language is complex, so how to interpret certain words can vary depending on context. The same word can have different meaning to different project offices, project types, and individual people, and even within the same projects and teams. For example, the word "instruments." A project that is purchasing tools to conduct lab experiments may classify these costs as instruments. But when that same project (or another project) is constructing the building that houses the lab, the lab's environmental controls and other interfaces with building utilities can also be filed as instruments. And of course, someone without any context may think the project is outfitting a marching band. The point being, without a standard WBS format or dictionary to characterize the meaning and context of a word in a WBS element, that word is open to interpretation. On top of that, EV reporting is often very concise. For example, an EV manager could report costs under a category called "Labs." With minimal outside information and context, one analyst could consider those lines to be the cost to build or renovate a lab. Another analyst could interpret "Labs" as the cost to outfit labs with scientific instruments, and another analyst could see that cost as the cost for one of the national laboratories to conduct management of the project. If attempting to determine how much a project spent on facility construction vs. project management, each analyst would reach a different conclusion.

Different conclusions between analysts can and have resulted in grossly different conclusions about the execution of existing projects, which in turn translates to grossly different cost estimates and executability analysis for future projects. With an organizational structure that requires multiple offices to produce cost estimates at various stages of a project's lifecycle, it is important that each office has the same foundation of good data with which to make independent estimates.

Lack of a standard WBS necessarily opens the door for different cost estimating offices to interpret the same historic data differently. Unfortunately, this situation cannot be rectified without substantial reconciliation efforts that add cost, schedule, and risk to capital projects. This paper seeks to provide a remedy to this status quo via NLP and machine learning.

1.4 Classification Overview

The following section provides an overview of the various classification paradigms available and how these paradigms can be implemented to standardize cost data in the form of a WBS. The section serves as background informing the approach taken in Section 2.

1.4.1 Binary and Multiclass Classification

The simplest classification task is binary classification, the mapping of data into two groups or categories. For example, a binary classifier would be useful if NNSA only wanted to map capital cost data into either Total Estimated Costs (TEC) and Other Project Costs (OPCs).

The model implemented in Reference 12, hereafter referred to as Model 1, is a *multiclass* classifier that classified project cost data into six, Level 2 classes: Site Preparation, Project Engineering and Design (PED), Construction, Procurement, Project Management, and Start-up (Table 1). Multiclass classifiers map data into more than two groups or categories, but with still only one output class or target (Table B2).

Table 1: Model 1 classified WBS elements to one of six Level 2 classes

Level	WBS Code	Title
1	**1**	**Project Name**
2	1.1	Site Preparation
2	1.2	PED
2	1.3	Construction
2	1.4	Procurement

2	1.5	Project Management
2	1.6	Start-up

1.4.2 Single-label and Multi-label Classification

Model 1 is also an example of single-label classification because there is only one answer for a given target. That is, the goal is to assign only one Level 2 label per WBS element. This differs from other classification tasks where more than one label within a specific category may be appropriate, meaning that classes are not mutually exclusive. Table B2 demonstrates how the logic behind single-label multi-class classification (in the form of binary and multi-class classification) and multi-label classification differ and lead to different results even with similar inputs.

1.4.3 Multioutput Classification

Model 2 aims to classify WBS elements beyond Level 2, thereby creating a multi-level classification scheme that better reflects an actual WBS. This means that we are adding to the number of targets in our task, leading to what is known as multioutput classification (or multitask classification, since there are multiple classes possible for each output or target). Two approaches to multioutput classification are traditional multioutput classification and classifier chains. In the case of standardizing WBS elements to a common, seven-level WBS (comprising Levels 2-8), a traditional multitask classifier treats each level of the hierarchy as an independent model and there is no assumed correlation between the targets. For simplicity, these will be referred to as Multioutput (MO) classifiers.

The classifier chain (CC) applies multiple binary classifiers to the multi-target problem. Unlike the tradition multitask classifiers, classifier chains assume there is a correlation between the targets. As the name implies, the classifiers are linked so that the prediction results from one classifier are used as feature inputs in the next. The default CC randomizes the order of the chain since most multioutput problems do not know the optimal order of the chain.

Both MOs and CCs are “meta-estimators” that determine how to apply the actual classifiers (such as logistic regression or random forest), which are typically known as “base estimators” or, as described in 1.3.6, “local classifiers.” To ensure alignment with the modeling (and computer code) nomenclature, “base estimator” will refer to the classifiers implemented under MO and CC classification paradigms, whereas “local classifier” will refer to the classifiers implemented under the hierarchical meta-estimator paradigms.

Table B2 in Appendix B provides an overview of the main types of classification. For example, consider a simplified sample of four WBS elements: “Site Prep,” “PM Support,” “Procure GB,” “Conceptual Design.” A binary classifier could try to determine whether each line of text belongs to a category, like “Construction” or not. In that case, “Site Prep” would be considered “Construction” while the other three lines of text would not. However, there would be no insight into where those remaining strings belonged in the WBS. A multiclass classifier similar to that seen in Model 1 solves that problem. Now, “Site Prep” is still classified as Construction but now “PM Support,” “Procure GB,” and “Conceptual Design,” are classified as “Program Management,” “Equipment,” and “Pre CD-2,” respectively. Multilabel classification is similar to multiclass classification except that the outputs are not necessarily mutually exclusive, meaning that a text string may or may not be classified as more than one label.

1.4.4 Multi-input Multioutput Classification

Table B2 provides an overview of classification types that take single inputs. However, many classification tasks involve multiple inputs. Recall that Model 1 involved a Multi-input classifier that used both WBS text and Actual Cost of Work Performed (ACWP) inputs to determine the Level 2 class in which the WBS element belonged. A Multi-input Multioutput (MIMO) classifier, by extension, uses multiple inputs to classify data to multiple outputs. Table B3 demonstrates the new setup, using a simplified version of the WBS used for Model 2, since the WBS used for Model 1 (Table 1) only includes one output (Level 2). A typical MIMO does not inherently understand whether the outputs are related to each other. There is nothing coercing the outputs to any format or hierarchy, meaning that a WBS element identified as belonging to one class in Level 2 may then be classified to a completely unrelated class in Level 3. Ideally, this would not be the case, but there is no way to ensure this without modifying the MIMO to account for the relationship between outputs.

1.4.5 Hierarchical Classification

Hierarchical classification is a growing field of research that aims to apply advanced machine learning classification techniques to hierarchically-related data (Reference 22). A limited but growing number of analysts have successfully applied hierarchical classifiers to real-world applications, but these are largely limited to biology or music (e.g., animal species taxonomy, gene classification, genre classification). In hierarchical classification, the model can be thought of as consisting of a meta-estimator and a local classifier (or base estimator). The meta-estimator is the overarching paradigm determining how the local classifiers are implemented across the hierarchy. There are three main types of meta-estimators, shown in Table 2. Table B4 demonstrates how the hierarchical model can result in multiple outputs that are hierarchically related. The following sections will the three hierarchical meta-estimators in more detail using the sample WBS in Figure 1[1] (the full WBS used for Model 2 is attached in Appendix D).

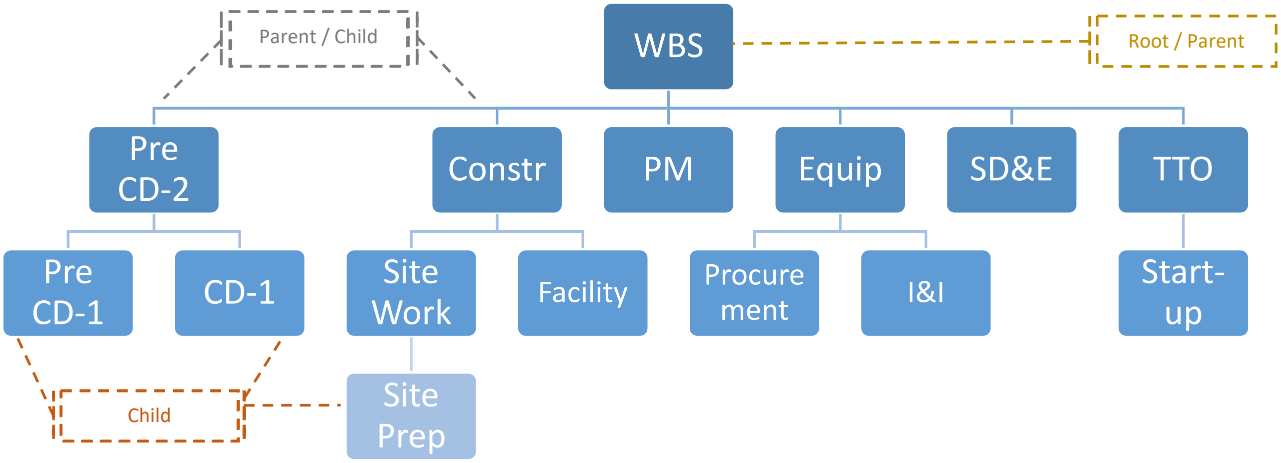
Table 2: Hierarchical Meta-Estimators[2]

Hierarchical Meta-Estimator	Description
Local Classifier Per Level	Multiclass Classifier for each Level
Local Classifier Per Node	Binary Classifier for each non-Root Node
Local Classifier Per Parent Node	Multiclass Classifier for each Parent Node

[1] In the WBS, the following abbreviations are used: Construction (Constr), Program Management (PM), Equipment (Equip), Systems Design and Engineering (SD&E), Transition to Operations (TTO), Critical Decision (CD), and Installation and Integration (I&I).

[2] Additional information on hierarchical estimators can be found in Reference 11.

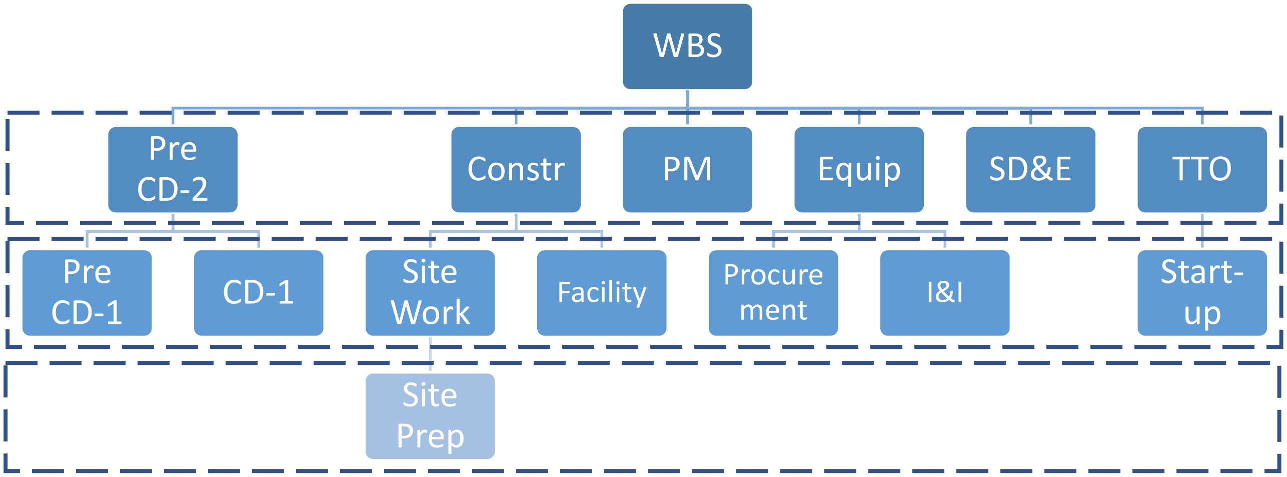
Figure 1: A hierarchical structure contains classes that operate like nodes in a network.



1.4.5.1 Local Classifier Per Level (LCPL)

The simplest hierarchical model is the LCPL implementation. Figure 2 demonstrates the LCPL paradigm, which applies a multiclass classifier to each level of the hierarchy. In the case of our WBS example, there are three levels of nodes under the root node. A single-label, multiclass classifier is applied to each level, meaning that a WBS element can only be classified into one class per level. LCPL models require the fewest classifiers and are therefore considered more efficient than the other options. Yet, the model does not coerce results to the hierarchy, undermining its designation as a hierarchical classifier. That is, the classification results obtained at one level do not inform that of the next level. Therefore, a WBS element could be classified as Program Management (PM) at one level, but Installation and Integration (I&I) at the next.

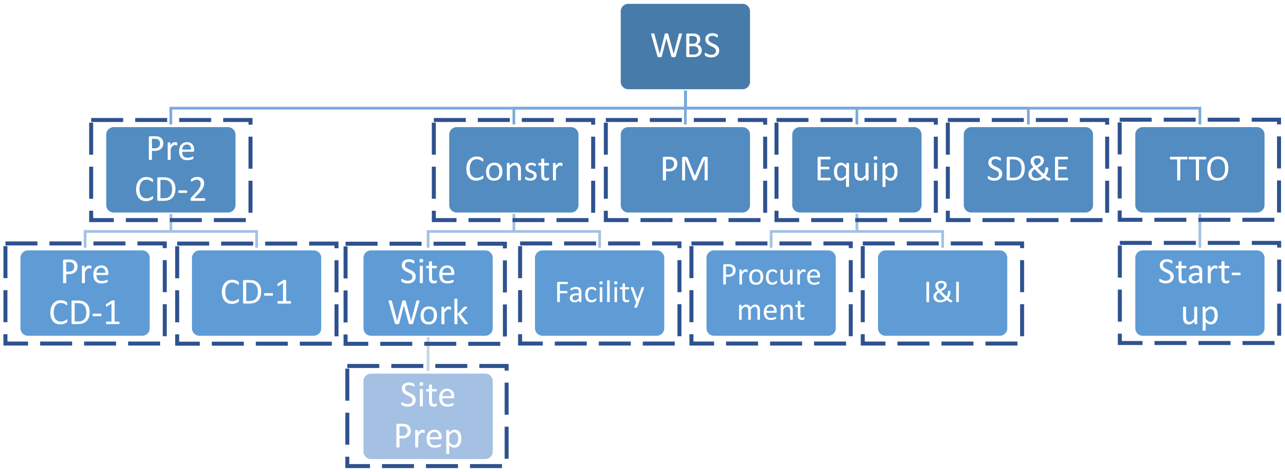
Figure 2: A Local Classifier Per Level is a multiclass classifier trained on each level of the hierarchy.



1.4.5.2 Local Classifier Per Node (LCPN)

In contrast to the LCPL meta-estimator, the LCPN paradigm applies binary classifiers to all non-root nodes. For example, for the given WBS, a classifier at the Construction (Constr) node would predict whether a WBS element is construction. Similarly, a classifier applied at the PM node will predict whether the element is PM. This means that a given WBS element could be classified both a Construction and Program Management, even though those nodes exist at the same level. Thus, classification at each node is not mutually exclusive, lending the LCPN paradigm to multi-label classification problems, where more than one label may be correct, such as in movie genres, but not to single-label problems like a WBS.

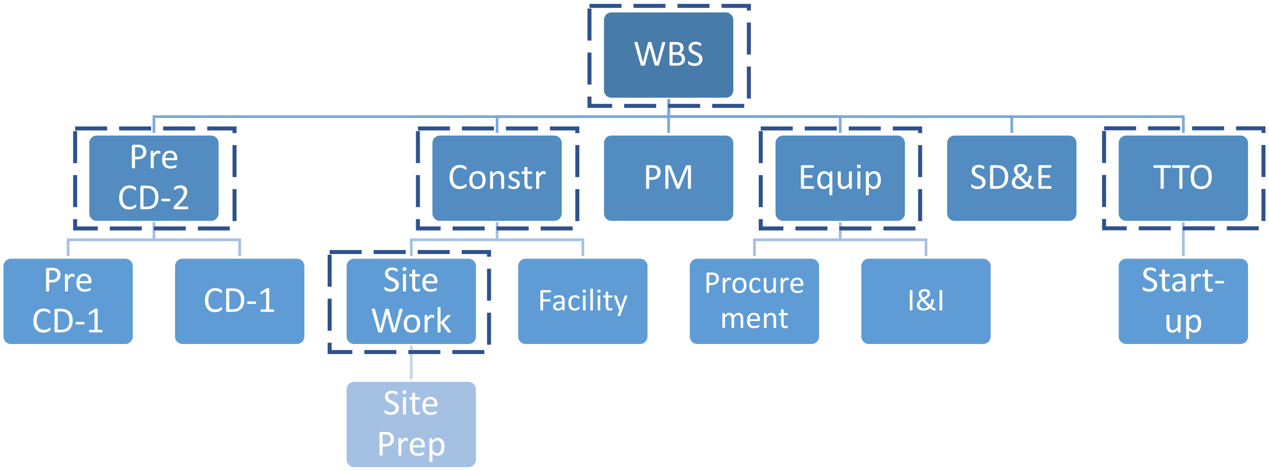
Figure 3: A Local Classifier Per Node is a binary classifier trained on each non-root node of the hierarchy.



1.4.5.3 Local Classifier Per Parent Node (LCPPN)

The final meta-estimator considered is the LCPPN, a multiclass classifier applied at each parent node to predict its child nodes. This is the preferred classifier for single-label multiclass classification, where there is only one “correct” label per level.

Figure 4: A Local Classifier Per Parent Node is a multiclass classifier trained on each parent node of the hierarchy.



1.4.5.4 Evaluating Hierarchical Models

Hierarchical classification models are evaluated using modified versions of the usual classification statistical metrics. In Model 1, (flat) classification model performance was evaluated using accuracy, which is the proportion of correctly classified WBS elements to total WBS elements. While useful as an aggregate metric, accuracy is problematic when there is class imbalance in the data. In this case, precision, recall, and f1-score are often better metrics to use.[3] Precision (P) refers to the ratio of True Positives (TP) to the sum of TP and False Positives (FP) (Equation 1). Recall (R) is the ratio of TP to the sum of TP and False Negatives (FN) (Equation 2). F1-score is the harmonic mean of these two metrics (Equation 3) (Reference 3).

Equation 1: Precision

$$\boxed{P = \frac{TP}{TP + FP}}$$

Equation 2: Recall

$$\boxed{R = \frac{TP}{TP + FN}}$$

Equation 3: F1-score

$$\boxed{F1 = 2\,x\,\frac{P * R}{P + R}}$$

While useful for classifying WBS elements to a high-level (and flat) classification scheme, these metrics are not recommended for hierarchically organized data. Instead, Reference 10 recommended modified versions of these metrics, depicted in Equation 4, Equation 5, and Equation 6 respectively.

[3] Refer to Reference 23 for an overview of classification metrics.

Equation 4: Hierarchical Precision

$$\boxed{hP = \frac{\sum_i |a_i \cap B_i|}{\sum_i |a_i|}}$$

Equation 5: Hierarchical Recall

$$\boxed{hR = \frac{\sum_i |a_i \cap B_i|}{\sum_i |B_i|}}$$

Equation 6: Hierarchical F1-score

$$\boxed{hF \ = \ 2 \ x \ \frac{hP * hR}{hP + hR}}$$

$$\boldsymbol{a_i = The\ set\ of\ predicted\ classes, including\ their\ parent\ classes\ for\ WBS\ element\ i}$$

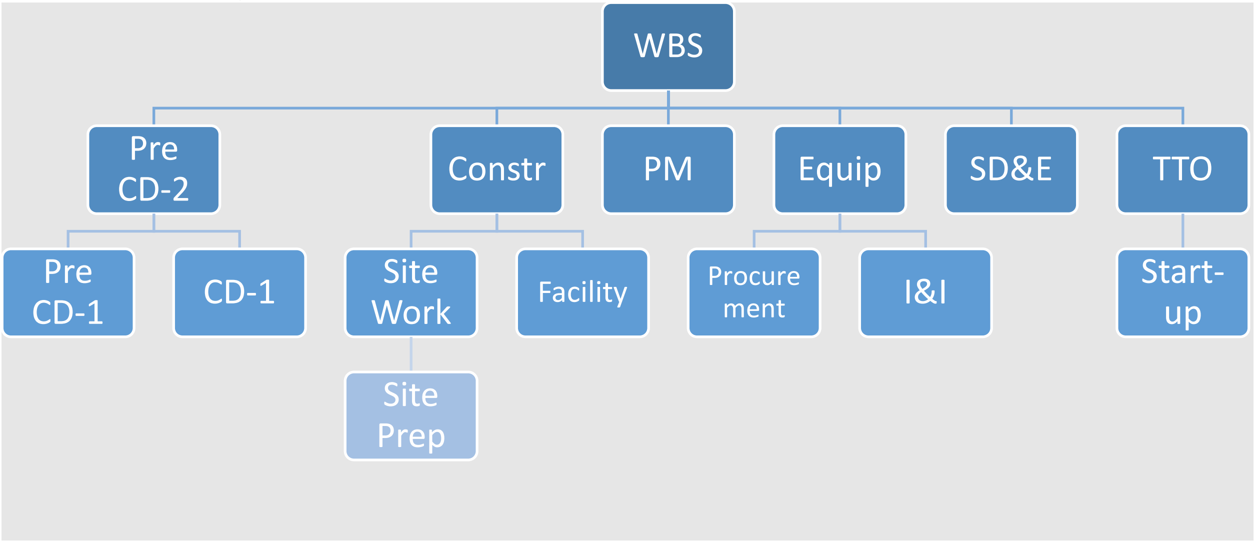
$$B_i = The\ set\ of\ true\ classes, including\ their\ parent\ classes, for\ WBS\ element\ i$$

To understand how these hierarchical metrics work, let us first consider what success looks like for a hierarchical classification model. The goal for applying a hierarchical classification model to WBS cost data is to correctly classify a WBS element to the exact class and level it belongs. However, the deeper the hierarchy, the fewer training samples per class exist, so correct classification becomes increasingly difficult. Consider the WBS element listed in Table 3 and the WBS in Figure 5. The information in the text indicates that this WBS element belongs in CD-1, which is a child node of Pre CD-2. If the model predicts that this text actual belongs in Pre CD-1, that means that the model correctly predicted that the class belonged in Pre CD-2, but then incorrectly chose the child node Pre CD-1 instead of CD-1. Here, that means that $\boldsymbol{a_1}$ = 2 because the predicted class is two levels down (Pre CD-2 → Pre CD-1) and B_1 = 2 because the true class is also two levels down (Pre CD-2 → CD-1). Yet the number of predicted classes contained in the true classes ($a_1 \cap B_1$) is only 1 because Pre CD-2 is contained in both the predicted set and the true set. Therefore, hP = 0.5, hR = 1.0, and hF = .67. For additional examples, see Figure 5 and Reference 23.

Table 3: Sample WBS element after preprocessing and concatenation.

i	Concatenated WBS Element
1	Project conceptual dsgn opc

Figure 5: Modified metrics measure performance of hierarchical classifiers



$\boldsymbol{i}$	**Predicted**	**True**	$\boldsymbol{a_i}$	$\boldsymbol{B_i}$	$\boldsymbol{a_i \bigcap B_i}$	$\boldsymbol{hP}$	$\boldsymbol{hR}$	$\boldsymbol{hF}$

1	Pre CD-1	CD-1	2	2	1	1/2 = 0.5	1/1 = 1.0	.67
	Site Work	CD-1	2	2	0	0/2 = 0	0/0 = NA	NA
	CD-1	CD-1	2	2	2	2/2 = 1.0	2/2 = 1.0	1.0
	TOTAL		6	6	3	3/6 = 0.5	3/3 = 1.0	.67

2 Approach

2.1 Data Export and Labeling

The results obtained from Model 1 (Reference 12) indicated considerable class imbalance. For Model 2, introduction of the CAEF WBS provided the team an opportunity to redo the sample selection and labeling process so that a more representative sample of projects would be labelled to the new scheme. Two analysts selected ten historical and ongoing projects and labelled them using the CAEF WBS as the new baseline. This was an iterative process in which the analysts adjusted the scheme to accommodate the data, as needed.

Three deviations from the initial data sampling occurred after the labeling process was completed.

The first was that the team discovered two archived capital projects that were completed before the establishment of DOE-PM's data repository. The analysts labeled these projects and added them to the sample.

The second was that the team, during the analysis process, discovered that several WBS elements in the ongoing projects included in the sample for labeling were changed by their Management and Operating (M&O) contractors. Some were minor changes, but one project changed most of their text and hierarchy, meaning that many of the WBS elements initially labelled would not end up in the true dataset. Therefore, the analysts updated the labeled sample set to incorporate the changes.

The third deviation occurred when the team learned that DOE-PM was undergoing a similar task of benchmarking the capital project data to a common format. This provided an opportunity to cross-check the decision-making employed.

The final sample dataset consisted of 12 capital projects; 10 of which were obtained from PARS and two from archival records that existed prior to the creation of PARS. Four of the 12 projects included two versions of the data files submitted by the M&Os mid-analysis. Therefore, the raw data consisted of 16 project files that were labelled to the new, multi-level WBS (Appendix C) adapted from the CAEF (Reference 4). Once the files were combined and modified to include their higher-level elements, the combined list of WBS elements included 123,547 lines of text data. After removing duplicate lines and the higher-level elements (which were already added to their corresponding lower-level elements), this sample file consisted of 3822 unique lines of labeled cost data. Figure 6 depicts the breakdown of these 3822 lines of sample data to Level 2, which consists of nine classes. Note that the only Level 2 labels shown are those that were identified in the sample dataset. Furthermore, General also includes DOE PM Support - Federal Oversight Cost due to their small sample sizes.

Figure 6: Level 2 classes used for Model 2 are adapted from the CAEF WBS.

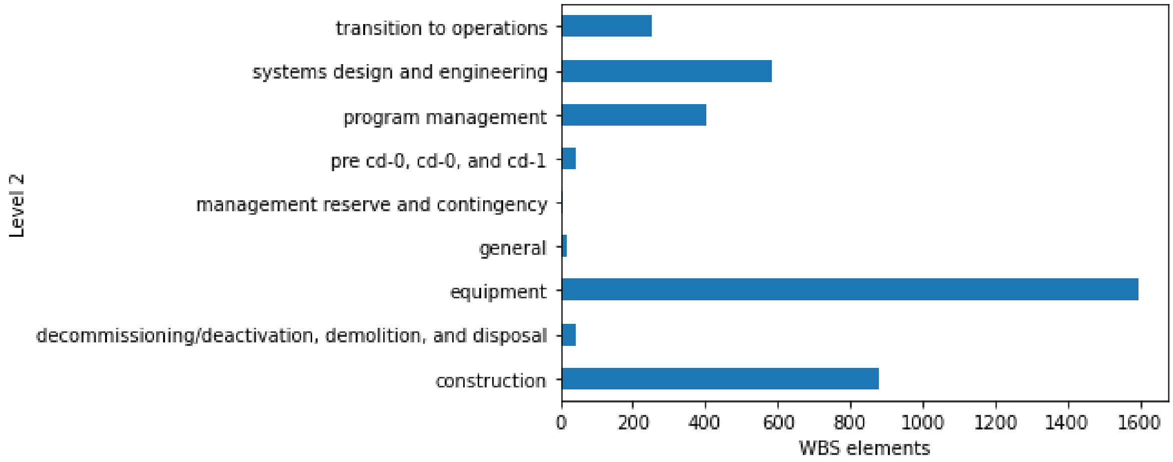


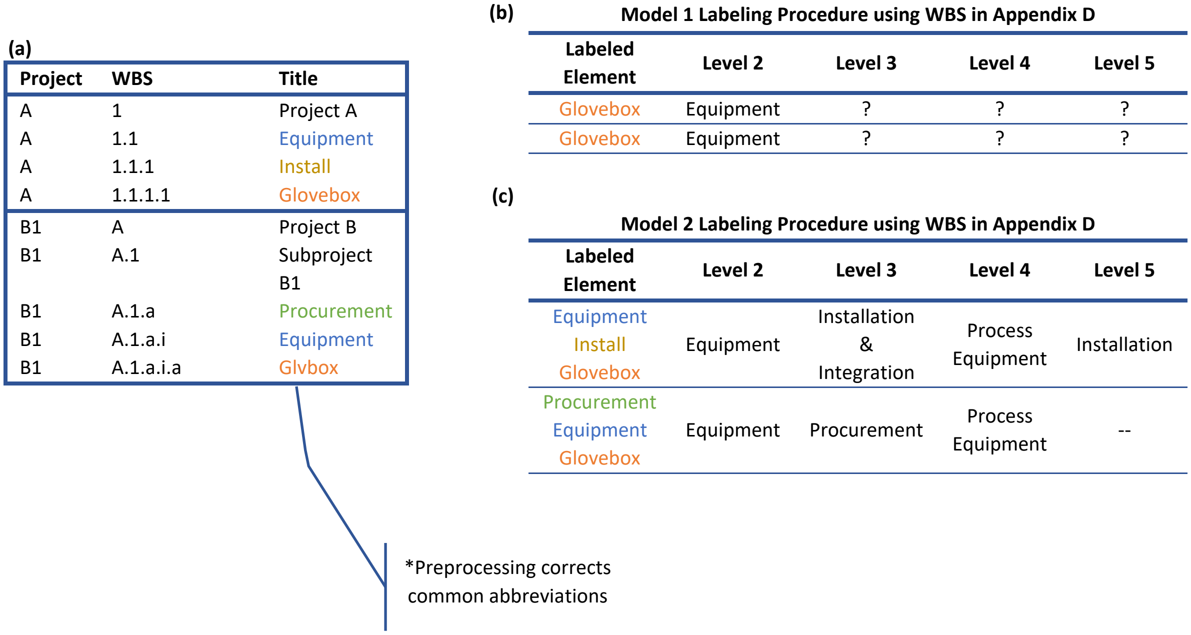
Figure 7: Example of how analyst labels raw WBS data based on classification scheme

WBS element	Model 1	Model 2		
	Level 2	Level 2	Level 3	Level 4
Facility construction	Construction	Construction	Facility	Miscellaneous Facility Construction
Glove box procure	Procurement	Equipment	Procurement	Process Equipment
Conceptual Design	Project Engineering and Design	Pre-CD-0, CD-0, CD-1	CD-1/Conceptual Design	Conceptual Design

A limitation identified in Model 1 was the omission of the higher-level elements associated with a WBS element when classifying these elements to a common Level 2 WBS. This poses an issue when the higher-level elements provide information crucial to determining where the lower-level element belongs. For example, consider cost data for two projects, A and B1. The projects are to be labelled and included in the sample dataset for model training and tuning (Figure 8a). The data indicate that Project A involves *installing* a glovebox while Project B1 involves *procuring* a glovebox. The labeling procedure employed in Model 1 only considers the lowest-level elements when labeling the data to a common scheme (Figure 8b). This means that for both projects, “Glovebox” is the only text used to inform how to manually label the element to the multi-level WBS scheme (Appendix D). According to this scheme, both instances of “Glovebox” would fall under Level 2: “Equipment.”

Model 2, however, considers the higher-level elements when manually labeling the data (Figure 8c). This means that now Project A’s “Glovebox” element becomes “Equipment Install Glovebox” and Project B1’s “Glovebox” element becomes “Procurement Equipment Glovebox.” This provides a much clearer picture of what the projects are doing and allows for labeling the elements to a lower level in the WBS scheme.

Figure 8: Model 1 ignores higher-level element text when classifying WBS elements to a common Level 2 WBS. Model 2 incorporates higher-level text into the text string, ignoring project names and other stop words.



(a)

Project	WBS	Title
A	1	Project A
A	1.1	Equipment
A	1.1.1	Install
A	1.1.1.1	Glovebox
B1	A	Project B
B1	A.1	Subproject B1
B1	A.1.a	Procurement
B1	A.1.a.i	Equipment
B1	A.1.a.i.a	Glvbox

*Preprocessing corrects common abbreviations

(b) **Model 1 Labeling Procedure using WBS in Appendix D**

Labeled Element	Level 2	Level 3	Level 4	Level 5
Glovebox	Equipment	?	?	?
Glovebox	Equipment	?	?	?

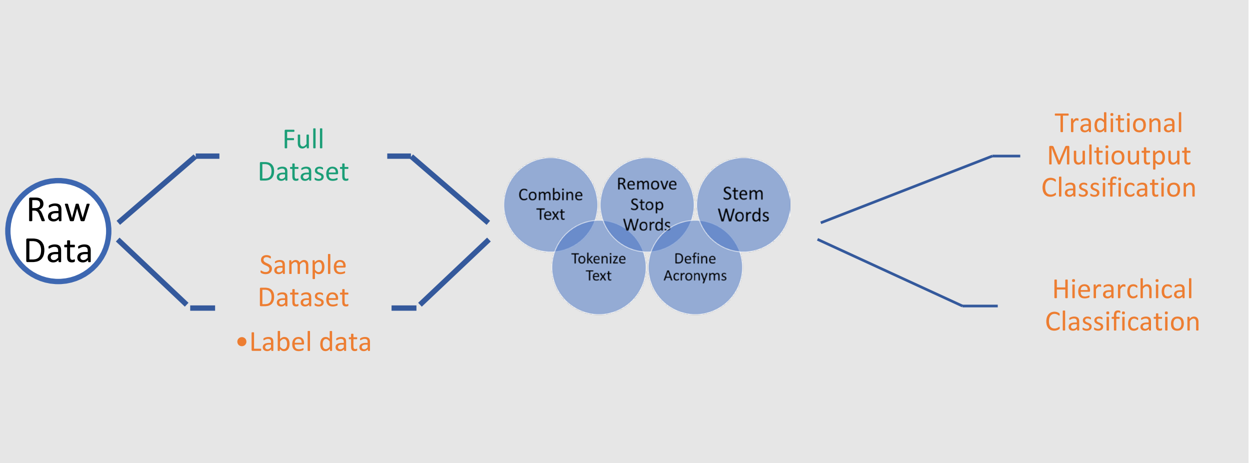
(c) **Model 2 Labeling Procedure using WBS in Appendix D**

Labeled Element	Level 2	Level 3	Level 4	Level 5
Equipment Install Glovebox	Equipment	Installation & Integration	Process Equipment	Installation
Procurement Equipment Glovebox	Equipment	Procurement	Process Equipment	--

2.2 Data Pre-processing and Preparation

Unlike Model 1, the first step in the data cleaning process concatenated all higher-level elements to their corresponding lower-level elements so that the full context of each element would be included in the model. The proceeding steps included similar preprocessing to that for Model 1: tokenization, acronym and abbreviation identification, stop word removal, and stemming (Figure 9). The following sections overview each of these concepts.

Figure 9: The process for preparing the data now includes combining higher-level element text to their corresponding lower-level elements.



2.2.1 Tokenization

Tokenization is the process of dividing the text data into usable chunks of information, known as tokens (Reference 26). In Reference 12, WBS elements were tokenized at the word-level. For Model 2.0, WBS elements were tokenized at varying levels depending on vectorization type. See Section 2.2.5 for more details.

2.2.2 Acronyms and Abbreviations

A total of 310 acronyms and abbreviations were used. This included those identified during the pre-processing for Model 1, as well as additional acronyms and abbreviations discovered in labeling the current sample.

2.2.3 Stop Words

The initial stop words list included 728 unique words (References 13, 14, and 20). However, the analysts determined that 22 words in this generic stop list would potentially negatively influence the analysis: *i, in, one, ones, other, others, up, general, three, threes, two, twos, turn, turned, turning, turns, well, wells, work, worked, working, works*. For example, *i*, though referring to the pronoun in the stop word list, often appears in the sample text in the context of "Title I Design," a text string that we are trying to distinguish from "Title II Design" and "Title III Engineering" in our multi-level WBS. Removing these words from the generic stop word list resulted in a final list consisting of 706 unique words.

The stop word list also includes an update of to the custom stop word list used for Model 1. In addition to the 706 generic stop words, the analysts updated the custom stop word list to now include 719 words (or phrases) and 171 that were in the form of abbreviations. Many of these custom words were the project names existing in PARS (not just those in the sample dataset), as well as locations, company names, and variations of these words and phrases identified while consolidating the data.

2.2.4 Stemming

Reference 12 introduced stemming as an optional part of preprocessing that reduces words to a common stem (References 1, 2, 14, and 19). In Model 1, stemming was applied to the traditional machine learning algorithms but not the deep learning algorithms. Initial exploratory analysis for this paper indicated that stemming did not improve the model performance (in fact, stemming slightly reduced performance across most conditions) (Table B1) and was therefore dropped from the analysis.

2.2.5 Other Considerations

Splitting separates the data into a "train" and "test" set so that a portion of the sample can be used for model tuning and training. The most common approach is to randomly split of the data to reduce the risk of bias. In this case, there is notable class imbalance, which worsens as the labeling progresses down the hierarchy. To attenuate this issue, a better option is to use iterative stratification, which takes the class representation into consideration when splitting the data. This ensures that the relative ratios of class occurrence are more proportional between the training and test set (Reference 21).

Reference 12 explores the use of count vectorization and term frequency-inverse document frequency (tf-idf) vectorization to turn the tokens into a usable (numeric) representation that the classification models can understand. For a review of these methods, consult Reference 18.

Model 2 also uses an alternate method, creating word2vec (w2v) word embeddings to vectorize the text based on a word's similarity to other words in the corpus (References 15, 16, and 24). These embeddings are either trained using the current sample text data or pretrained using external data. Due to the sparse nature of WBS text in this analysis, the word embeddings were pretrained using a corpus of construction-related text obtained from Reference 25.

3 Hierarchical Classification of Work Breakdown Structure Elements

3.1 Modeling Assumptions

Model 2 defines similar assumptions to those used in Model 1:

- The 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 and abbreviations identified during pre-processing are accurate and comprehensive.

Furthermore, the following assumptions are unique to Model 2:

- All WBS elements in the dataset belong under one of the labels identified in Appendix D.
- The iterative splitting methods employed occurred as intended.

3.2 Methodology

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

1) Removal of high-level WBS elements and duplicate low-level, WBS elements
2) Hierarchical classification using default hyperparameters
3) Hyperparameter tuning
4) Hierarchical classification with tuned parameters

3.2.1 Duplicate Work Breakdown Structure Elements

Similar to Model 1, duplicate WBS elements exist due to the timeseries characteristic of EV data. However, the duplicates in Model 2 are now determined based on the concatenated text string rather than just the WBS element used in Model 1. Table 4 provides an example of this change. Two buildings are undergoing demolition across two months in 2019. In Model 1, it would appear that "Building 1a demolition" appears twice and can be reduced to one instance; similarly, "Building 2a demolition" appears twice and can also be reduced to one instance. Yet when the full context is considered, it becomes clear that although the "Building 1a demolition" elements are true duplicates, the two "Building 2a demolition" elements are not.

Table 4: Duplicate WBS elements are determined by the fully combined text string.

Higher-Level Elements	Original WBS Element	Date (DDMMYYYY)	Concatenated WBS Element
Site Preparation	Building 1a demolition	05012019	Site Preparation Building 1a demolition
Site Preparation	Building 1a demolition	06012019	Site Preparation Building 1a demolition
Site Preparation	Building 2a demolition	05012019	Site Preparation Building 2a demolition
D&D	Building 2a demolition	06012019	D&D Building 2a demolition

3.2.2 Default Multitask Models – Machine Learning Base Estimators

In the context of typical multitask classifiers (Multioutput and Classifier Chains), the underlying classifier is known as the "base estimator." In hierarchical classification, the nomenclature is slightly different, and the underlying classifier is instead referred to as the "local classifier."

The default multioutput models consisted of 36 model combinations using two meta estimators (MO and CC) with either count, tf-idf, or w2v vectorization and one of six local classifiers (Table 5).

Table 5: Default base estimators used for multioutput classifiers.

Meta-Estimator	Base Estimator	Default Hyperparameters
Multioutput Classifier Chain	support vector classifier	Gamma: Scale Kernel: RBF Class Weight: None Degree: 3 C: 1
	stochastic gradient descent Classifier	Loss: Hinge Penalty: L2 Alpha: 1e4 Tolerance: 1e3 Learning Rate: Optimal Class Weight: None
	decision tree	Criterion: Gini Maximum Features: None
	Multinomial Naïve Bayes	Alpha: 1
	logistic regression	Tolerance: 0.0001 Class Weight: None Solver: lbfgs Penalty: L2 C: 1
	random forest	Number of Estimators: 100 Class Weight: None Criterion: Gini

3.2.3 Default Hierarchical Models – Machine Learning Local Classifiers

The default hierarchical models consisted of 54 model combinations using three meta estimators (LCPL, LCPN, and LCPPN) with either count, tf-idf, or w2v vectorization and one of six local classifiers (Table 6).

Table 6: Default hyperparameter values for local classifiers.

Meta-Estimator	Local Classifier	Default Hyperparameters
Local Classifier Per Level Local Classifier Per Node Local Classifier Per Parent Node	support vector classifier	Gamma: Scale Kernel: RBF Class Weight: None Degree: 3 C: 1
	stochastic gradient descent Classifier	Loss: Hinge Penalty: L2 Alpha: 1e4 Tolerance: 1e3 Learning Rate: Optimal Class Weight: None
	decision tree	Criterion: Gini Maximum Features: None
	Multinomial Naïve Bayes	Alpha: 1
	logistic regression	Tolerance: 0.0001 Class Weight: None Solver: lbfgs Penalty: L2 C: 1
	random forest	Number of Estimators: 100 Class Weight: None Criterion: Gini

3.2.4 Hyperparameter Tuning

Hyperparameter tuning is the process of comparing combinations of hyperparameter values to improve, and hopefully optimize, performance. Tuning proved essential to achieving Model 1's classification results and is a necessary stop when implementing machine learning models. Models require retuning each and every time data are updated, but this is generally a minor task assuming the nature of the data stays the same. Due to the inherent structural changes between Model 1 and Model 2, the hyperparameter values chosen for Model 1 are no longer relevant. Table 7 lists the hyperparameters evaluated during the tuning of the traditional machine learning models with Grid Search using three-fold cross-validation (cv).

Table 7: Hyperparameter combinations used for model tuning

Base Estimator / Local Classifier	Hyperparameters and Values
support vector classifier	Gamma: Scale, Auto Kernel: RBF, Poly, Linear Class Weight: None, Balanced Degree: 3, 4, 5 C: 1e-1, 1, 10
stochastic gradient descent	Loss: Modified Huber, Log Loss Penalty: L2, L1, elastic net Alpha: 1e5, 1e4, 1e3 Tolerance: 1e4, 1e3 Learning Rate: Adaptive, Optimal Class Weight: None, Balanced
decision tree	Criterion: Gini, Entropy Maximum Features: Auto, Sqrt, Log2, None
Multinomial Naïve Bayes	Alpha: 1e-2, 1e-1, 1
random forest	Number of Estimators: 100, 200, 300 Class Weight: Balanced, Balanced Subsample, None Criterion: Gini, Entropy, Log Loss
logistic regression	Tolerance: 1e4, 1e3 Class Weight: Balanced, None Solver: netwon-cg, saga, lbfgs Penalty: L1, L2, elasticnet, None C: 1e-1, 10, 100

3.2.5 Deep Learning Models

Model 1 ultimately employed a deep learning algorithm to classify cost data to a Level 2 WBS. So far in this analysis, the only algorithms employed as base estimators / local classifiers have been traditional machine learning models (Table 5). However, deep learning algorithms can also serve as the local classifiers in a hierarchical classification model. Table 8 shows the hyperparameter values used for tuning the LCPPN models with Dense Neural Networks (DNNs) and Convolutional Neural Networks (CNNs) as the local classifiers.

Table 8: Hyperparameter combinations used for model tuning neural networks

Base Estimator / Local Classifier	Hyperparameters and Values
DNN	Embedding Dimension: 100,200 Hidden Layers: 1, 2 Dropout: .3, .5, .7 Neurons per Layer: 50, 100, 200
CNN	Embedding Dimension: 100,200 Hidden Layers: 1,2 Number of Filters: 32, 64 Number of Kernels: 5, 10, 15, 20 Dropout: .3, .5, .7

3.3 Results

3.3.1 Default Multitask Models – Machine Learning Base Estimators

Table 9 provides the initial results of the non-hierarchical MO models using default hyperparameter values. Across all base estimators, the best performing vectorizer was count vectorization with an average hF score of 0.619; across all vectorization methods, the best overall base estimator was logistic regression, with an average hF score of 0.657.

Table 9: Average hF scores of MO using machine learning base estimators with default values on iteratively split data (cv = 3).

Base Estimator	hF			
	Count vectorizer	**Tf-idf Vectorizer**	**W2v vectorizer**	**Average**
decision tree	0.491636 +/- 0.018175	0.476184 +/- 0.027004	0.517459 +/- 0.067366	**0.495**
logistic regression	0.67936 +/- 0.145861	0.657558 +/- 0.149226	0.634407 +/- 0.124991	**0.657**
Multinomial Naïve Bayes	0.631319 +/- 0.124602	0.616618 +/- 0.117396	0.540504 +/- 0.130607	**0.596**
random forest	0.58508 +/- 0.077778	0.488256 +/- 0.019319	0.616216 +/- 0.102235	**0.563**
stochastic gradient descent	0.667149 +/- 0.108162	0.647868 +/- 0.121252	0.592118 +/- 0.109405	**0.635**
support vector classifier	0.657934 +/- 0.096872	0.617552 +/- 0.099562	0.656076 +/- 0.096352	**0.644**
Average	**0.619**	**.584**	**0.593**	**0.599**

Table 10 provides the same results but for the default CC models. The logistic regression base estimators' hF score averaged at 0.655 across all vectorization methods and the count vectorization usually outperformed the other vectorizers regardless of base estimator with an average hF score of 0.602.

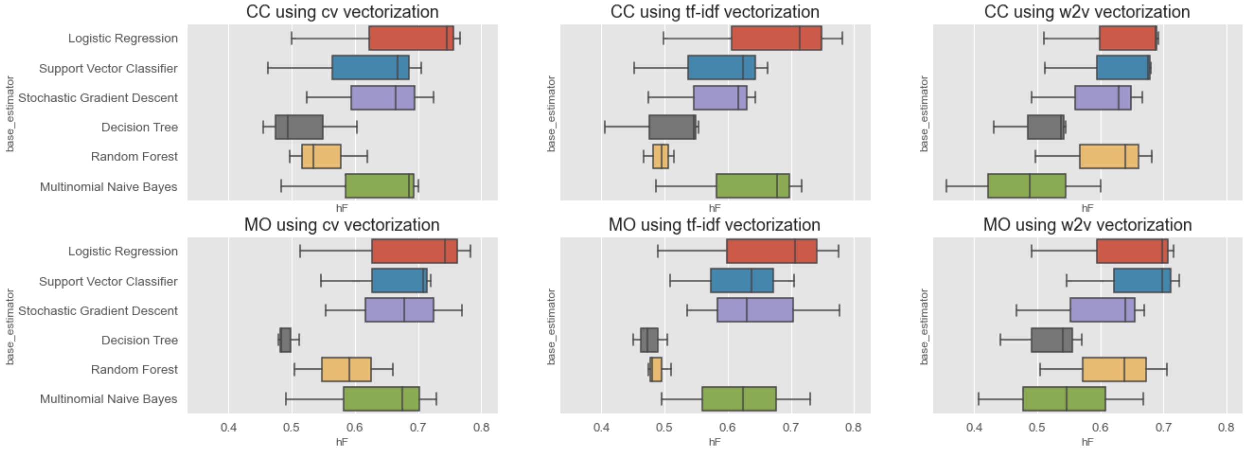
Across the untuned MO and CC classification models, the best untuned model was the MO using a logistic regression base estimator and count vectorization (average hF = 0.679) (Table 9). Figure 10 illustrates how each of the CC and MO models' hF scores compare (cv = 3).

Table 10: Average hF scores of CC using machine learning base estimators with default values on iteratively split data (cv = 3).

Base Estimator	hF			
	Count vectorizer	**Tf-idf Vectorizer**	**W2v vectorizer**	**Average**
decision tree	0.517141 +/- 0.076969	0.502025 +/- 0.083037	0.504853 +/- 0.063473	**0.508**
logistic regression	0.670171 +/- 0.14832	0.664808 +/- 0.14838	0.630073 +/- 0.10298	**0.655**

multinomial naive Bayes	0.623214 +/- 0.120838	0.62707 +/- 0.123213	0.481782 +/- 0.122177	**0.577**
random forest	0.550608 +/- 0.062974	0.492732 +/- 0.024265	0.6061 +/- 0.096933	**0.550**
stochastic gradient descent	0.638021 +/- 0.102911	0.578175 +/- 0.091017	0.596006 +/- 0.092413	**0.604**
support vector classifier	0.611307 +/- 0.130792	0.579974 +/- 0.112575	0.622433 +/- 0.095785	**0.605**
Average	**0.602**	**0.574**	**0.574**	**0.583**

Figure 10: Performance of default multioutput models using cv, tf-idf, and w2v vectorization (cv = 3).



3.3.2 Default Hierarchical Models – Machine Learning Local Classifiers

Figure 11 depicts the results of the unoptimized hierarchical models. Across all meta-estimator and vectorization combinations, the logistic regression resulted in the highest average hF score using three-fold cv. The support vector classifier and stochastic gradient descent models failed under the LCPL and LCPN paradigms but performed relatively well with LCPPN. LCPPN was also the meta-estimator that most aligned with the nature of a hierarchical, multi-level WBS, so proceeding with the LCPPN meta-estimator made sense.

Figure 11: Performance of default hierarchical models using count, tf-idf, and w2v vectorization (cv = 3).

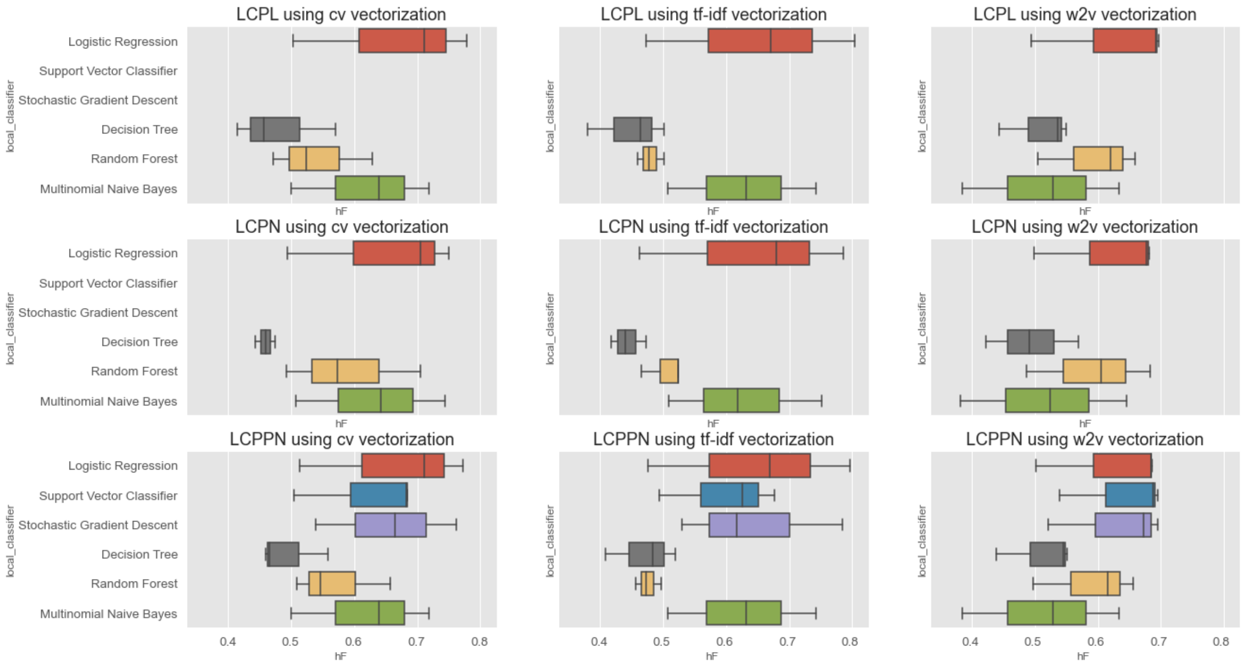


Table 11 shows how the untuned performance of the LCPL models. Like the CC and MO models, on average, the logistic regression performed better than the other local classifiers (average hF score = 0.647) and the count vectorizer performed better than the other vectorizers (average hF score = 0.576). Importantly, both the stochastic gradient descent and support vector classifier classifiers failed under the LCPL paradigm. This failure also occurred under the LCPN paradigm (Table 12). Once again, the LCPN paradigm generally preferred the logistic regression and count vectorization over the other options. Across all vectorizers, the logistic regression showed an average hF score of 0.637 and across all local classifiers, the count vectorizer had the highest average hF score of 0.582.

Table 11: Average hF scores of LCPL using machine learning base estimators with default values on iteratively split data (cv = 3).

Local Classifiers	hF			
	Count vectorizer	**Tf-idf Vectorizer**	**W2v vectorizer**	**Average**
decision tree	0.480409 +/- 0.080819	0.448159 +/- 0.062432	0.509413 +/- 0.058219	**0.479**
logistic regression	0.6644 +/- 0.143628	0.648978 +/- 0.166346	0.627026 +/- 0.115721	**0.647**
multinomial naive Bayes	0.619415 +/- 0.11058	0.627505 +/- 0.118139	0.515241 +/- 0.12502	**0.587**
random forest	0.540946 +/- 0.080791	0.479609 +/- 0.021115	0.594641 +/- 0.080529	**0.538**
stochastic gradient descent	--	--	--	--
support vector classifier	--	--	--	--

Average	**0.576**	**0.551**	**0.562**	**0.563**

Table 12: Average hF scores of LCPN using machine learning base estimators with default values on iteratively split data (cv=3).

Local Classifier	hF			
	Count vectorizer	Tf-idf Vectorizer	W2v vectorizer	Average
decision tree	0.458833 +/- 0.015605	0.443174 +/- 0.027661	0.494198 +/- 0.073243	**0.465**
logistic regression	0.649707 +/- 0.137499	0.642329 +/- 0.164931	0.618888 +/- 0.104537	**0.637**
multinomial naive Bayes	0.630868 +/- 0.118383	0.626156 +/- 0.121258	0.517346 +/- 0.132089	**0.591**
random forest	0.590302 +/- 0.106981	0.504552 +/- 0.033291	0.591596 +/- 0.098591	**0.562**
stochastic gradient descent	--	--	--	--
support vector classifier	--	--	--	--
Average	**0.582**	**0.554**	**0.556**	**0.564**

Table 13 shows how the untuned performance of the LCPPN models. On average the logistic regression performed better than the other local classifiers (average hF score = 0.646) and the count vectorizer performed better than the other vectorizers (average hF score = 0.605).

Across all hierarchical models using default machine learning local classifiers, the best average hF score occurred with the LCPPN meta-estimator using logistic regression and count vectorization (average hF score = 0.665).

This time, the stochastic gradient descent and support vector classifier classifiers did not fail, and the stochastic gradient descent performance approached that of the logistic regression's (average hF score = 0.655).

Table 13: Average hF scores of LCPPN using machine learning base estimators with default values on iteratively split data (cv=3).

Base Estimator	hF			
	Count vectorizer	Tf-idf Vectorizer	W2v vectorizer	Average
decision tree	0.494105 +/- 0.055132	0.470522 +/- 0.056631	0.511514 +/- 0.063155	**0.492**
logistic regression	0.665483 +/- 0.135932	0.647491 +/- 0.16152	0.623957 +/- 0.105361	**0.646**
multinomial naive Bayes	0.619415 +/- 0.11058	0.627505 +/- 0.118139	0.515241 +/- 0.12502	**0.587**
random forest	0.57098 +/- 0.07679	0.475833 +/- 0.020211	0.58969 +/- 0.082162	**0.546**

stochastic gradient descent	0.6554 +/- 0.112225	0.643723 +/- 0.12947	0.629257 +/- 0.094976	**0.643**
support vector classifier	0.623723 +/- 0.10381	0.598654 +/- 0.094605	0.640223 +/- 0.088026	**0.621**
Average	**0.605**	**0.578**	**0.585**	**0.589**

3.3.3 Hyperparameter Tuning

After comparing the initial set of default models, LCPL and LCPN meta-estimators were dropped from the analysis to simplify the tuning process going forward, leaving the MO, CC, and LCPPN paradigms as the meta-estimators of interest.

Under each condition, the tuning process identified the best base estimators / local classifiers and corresponding hyperparameter values using count, tf-idf, and w2v vectorization. Table 14 depicts the best base estimators and their corresponding parameter values identified through tuning the MO and CC models. Both the MO with count vectorization and tf-idf vectorization identified the stochastic gradient descent classifier as the best estimator while the using w2v vectorization, the support vector classifier became the preferred estimator. Tuning the CC models resulted in the same vectorizer – base estimator pairs with minor changes in the hyperparameter values.

Table 14: Tuning the MO and CC models identified the best local classifier and parameters (cv=3).

Meta-Estimator	Vectorizer	Best Base Estimator	Best params

MO	cv	stochastic gradient descent	Loss = Log
			Alpha = 0.0001
			Penalty = L2
			Tol: 0.0001

	tf-idf	stochastic gradient descent	Loss = Log
			Alpha = 0.0001
			Penalty = elasticnet
			Tol: 0.0001

	w2v	support vector classifier	Gamma = 0.4
			Kernel = rbf
			Class Weight: None
			Degree = 3
			C = 1

CC	cv	stochastic gradient descent	Loss = Log
			Alpha = 0.0001
			Penalty = L2
			Tol: 0.001

tf-idf	stochastic gradient descent	Loss = Log
		Alpha = 0.0001
		Penalty = elasticnet
		Tol: 0.001

w2v	support vector classifier	Gamma = 0.3
		Kernel = rbf
		Class Weight: None
		Degree = 3
		C = 1

When tuning the LCPPN models, the stochastic gradient descent estimator appeared as the best estimator regardless of vectorization method though the optimal hyperparameter values varied slightly across models (Table 15).

Table 15: Tuning the LCPPN models identified the best local classifier and parameters (cv=3).

Meta-Estimator	Vectorizer	Best Local Classifier	Best params

LCPPN	cv	stochastic gradient descent	Alpha = 0.0001
			Class weight = None
			Loss = Log_loss
			Penalty = L2
			Tol = 0.0001
			Learning Rate = Optimal

	tf-idf	stochastic gradient descent	Alpha = 0.001
			Class weight = Balanced
			Loss = Log_loss
			Penalty = elasticnet
			Tol = 0.001
			Learning Rate = Optimal

	w2v	stochastic gradient descent	Alpha = 0.001
			Class weight = None
			Loss = modified huber
			Penalty = elasticnet
			Tol = 0.0001
			Learning Rate = Optimal

Refitting the MO and CC models on the full training set using the best estimator and parameters identified in Table 14 resulted in the hF results depicted in Table 16. Among both the MO models and CC models, the highest hF score was the MO model using stochastic gradient descent with cv vectorization (hF = .875).

Table 16: Tuned MO and CC model performance on test set.

Meta Estimator	Vectorizer	Best Base Classifier	hF

MO	cv	**stochastic gradient descent**
		Loss = Log
		Alpha = 0.0001
		Penalty = L2
		Tol: 0.0001
			0.875

	tf-idf	**stochastic gradient descent**
		Loss = Log
		Alpha = 0.0001
		Penalty = elasticnet
		Tol: 0.0001
			0.874

	w2v	**support vector classifier**
		Gamma = 0.4
		Kernel = rbf
		Class Weight: None
		Degree = 3
		C = 1
			0.831

CC	cv	**stochastic gradient descent**
		Loss = Log
		Alpha = 0.0001
		Penalty = L2
		Tol: 0.001
			0.777

	tf-idf	**stochastic gradient descent**
		Loss = Log
		Alpha = 0.0001
		Penalty = elasticnet
		Tol: 0.001
			0.782

	w2v	**support vector classifier**
		Gamma = 0.3
		Kernel = rbf
		Class Weight: None
		Degree = 3
		C = 1
			0.811

The LCPPN with stochastic gradient descent and count vectorization outperformed these results (hF = .889) but the three tuned MO models outperformed the remaining tuned LCPPN models. Still, all LCPPN models performed better than their untuned predecessors as well as all other untuned, default model combinations considered (Table 17).

Table 17: Tuned LCPPN model performance on test set.

Meta Estimator	Vectorizer	Best Local Classifier	hF

LCPPN	cv	**stochastic gradient descent**
		Alpha = 0.0001
		Class weight = None
		Loss = Log_loss
		Penalty = L2
		Tol = 0.0001
		Learning Rate = Optimal
			0.889

	tf-idf	**stochastic gradient descent**
		Alpha = 0.001
		Class weight = Balanced
		Loss = Log_loss
		Penalty = elasticnet
		Tol = 0.001
		Learning Rate = Optimal
			0.809

	w2v	**stochastic gradient descent**
		Alpha = 0.001
		Class weight = None
		Loss = modified huber
		Penalty = elasticnet
		Tol = 0.0001
		Learning Rate = Optimal
			0.792

3.3.4 Deep Learning Models

Model 1 ultimately employed a deep learning algorithm to classify cost data to a Level 2 WBS. So far in this analysis, A major question that arose when scoping the current analysis was whether deep learning models were compatible with the hierarchical classification model architecture, which used a different software package. With the addition of a few custom functions, deep learning algorithms such as Dense Neural Networks (DNN) and Convolutional Neural Networks (CNNs) can serve as local classifiers in a hierarchical model. Table 18 demonstrates the results of the MO, CC, and LCPPN model tuning. Although the neural networks worked under the MO and LCPPN paradigms, they failed to yield results under the CC. Therefore, only the MO and LCPPN models were successfully tuned.

Table 18: Tuning the models with Randomized Search CV (cv=3).

Meta-Estimator	Base Estimator / Local Classifier	Param grid	Best hF score
MO	DNN	Embedding Dimension: 100,200 Hidden Layers: 1, 2 Dropout: .3, .5, .7 Neurons per Layer: 50, 100, 200	0.563
	CNN	Embedding Dimension: 200 Hidden Layers: 2 Number of Filters: 64 Number of Kernels: 15 Dropout: .5	0.648
CC	DNN	Embedding Dimension: 100, 200 Hidden Layers: 1, 2 Dropout: .3, .5, .7 Neurons per Layer: 50, 100, 200	––
	CNN	Embedding Dimension: 200 Hidden Layers: 2 Number of Filters: 64 Number of Kernels: 15 Dropout: .5	––
LCPPN	DNN	Embedding Dimension: 100,200 Hidden Layers: 1, 2 Dropout: .3, .5,.7 Neurons per Layer: 50, 100, 200	0.643
	CNN	Embedding Dimension: 200 Hidden Layers: 2 Number of Filters: 64 Number of Kernels: 15 Dropout: .5	0.639

Table 19 depicts the results of applying the tuned neural nets on the test set. The tuned MO models failed to produce results, but the LCPPN models resulted in an hF of 0.854 and 0.896 with the DNN and CNN, respectively. Therefore, across all tuned model combinations, the LCPPN using a CNN local classifier performed the best.

Table 19: Tuned neural net model performance on test set (cv = 3).

Meta Estimator	Base Estimator / Local Classifier	Best Hyperparameter Values	Best hF
MO	DNN	Embedding Dimension: 100 Hidden Layers: 2 Dropout: .5 Neurons per Layer: 200	--
	CNN	Embedding Dimension: 200 Hidden Layers: 1 Number of Filters: 32 Number of Kernels: 10 Dropout: .3	--
CC	DNN	--	--
	CNN	--	--
LCPPN	DNN	Embedding Dimension: 200 Hidden Layers: 1 Dropout: .3 Neurons per Layer: 200	0.854
	CNN	Embedding Dimension: 200 Hidden Layers: 2 Number of Filters: 64 Number of Kernels: 15 Dropout: .5	0.896

4 Observations

4.1 Hierarchical Classification

To the authors' knowledge, this analysis is the first implementation of hierarchical classification to the fields of cost estimating and cost analysis and presented at the ICEAA Professional Development and Training Workshop. The results indicated that not only was the LCPPN meta-estimator the most relevant paradigm to standardize WBS cost data to a common format, but also the model that yielded the best results. The LCPL and LCPN models failed when using two of the ML algorithms as local classifiers and the CC models failed when using the neural networks as base estimators. It is likely that the LCPL, LCPN, and CC tuning failures were due to inherent incompatibility between the underlying base estimator / local classifier with the meta-estimator. In the case of the failure of applying the tuned MO model to make a prediction on the test set, there appeared to be a computer code issue that could be resolved through addition of custom additions to the software packages used to ensure interoperability between them. In fact, much of this analysis required integrating and customizing code from different software packages and so this work also serves as a proof-of-concept that hierarchical classification can be implemented using neural networks as the underlying local classifiers with the appropriate customizations.

4.2 Areas of Improvement

There are limitations to the approach taken in this analysis to classify disparate cost data to a common, multi-level WBS. They fall into two main camps: (1) limitations due to underlying model assumptions and (2) limitations due to model design.

4.2.1 Data and Model Assumptions

The model assumptions listed in 3.1 are worthy of review. The first assumption was that the sample dataset was representative of the full WBS dataset. Ideally the projects sampled for this analysis represent the capital projects not included in the sample. It is possible that selection bias occurred. In fact, there were changes to the projects sampled for Model 2 because we later determined that those selected for Model 1 were not representative enough. This is a difficult task as the sample should be represent various M&Os, project types, complexity, status, and other factors that are sometimes to ensure when there is such a small dataset of projects available from which to choose. There are some projects that are quite distinct. It is difficult to predict whether future projects will deviate from those currently existing in the dataset. In fact, without a DOE requirement for standardizing WBS's for capital projects, there is also no way to prevent the M&Os from changing how they collect and label their data for ongoing projects while in process. This is exactly what happened when we were working on Model 2, meaning that additional time was spent to relabel several sample projects just because their submitted data changed drastically.

4.2.2 Model Design

The model design is another consideration. For one, the hierarchical classification models employed in this analysis are "top-down" models, meaning that the classification always begins at the top of the hierarchy. This may limit the information gleaned from the text, since in some cases it may be easier to start with more specific, lower-levels and work upwards. The analysts are exploring ways to address this limitation through the implementation of other types of multioutput models.

Another limitation is that currently, the tuning process is applied across the meta-estimator, meaning that the best estimators are a result of tuning that maximizes the hF score across various hyperparameter combinations. Model performance may further improve if the classifiers were tuned uniquely at each node. This is something the analysts are also exploring, though it is not clear that such a modification to the packages implemented would be possible without additional customization. If such a customization is not possibly, an alternative option would be to build a traditional multioutput neural network such as that seen in Reference 9 that allows more manual manipulation of the model tuning process but must be designed to ensure that the hierarchy of the multi-level WBS is enforced when tuning and predicting the results.

As we continue to work on this analysis, we are also exploring ways to add additional inputs to the model, as was done in Model 1, thereby creating a hierarchical. Model 1 incorporated both WBS text and cost as inputs, but the time-phased nature of the data also provides additional opportunities, as well. Importantly, the interpretation of text is sometimes predicated on the associated date of the WBS element, and therefore is an important piece that should be incorporated into the model, if possible.

5 Conclusion

The findings in this paper demonstrate how NNSA used hierarchical classification to – with a high degree of accuracy – standardize disparate capital asset cost data to a common, hierarchically-organized WBS. The analysis compared multiple non-hierarchical and hierarchical multioutput model candidates for Model 2. Ultimately, the selected Model 2 was a hierarchical model (LCPPN) using a tuned neural net (CNN) as its base estimator. This work leveraged and expanded upon the findings in Model 1 (Reference 11), which successfully used NLP to map cost data to a standard Level 2 WBS. Model 2 goes several steps beyond Level 2, meaning that it can support both high-level benchmarking efforts as well as lower-level standardization and analysis at NNSA (for a review of potential use cases, see Appendix C).

This approach represents one of three possible high-level approaches that can be taken towards mapping cost data to a common WBS.

1) Manually labeling each and every WBS to the preferred standard (i.e., analysts by hand).
2) Automatic labeling using a keyword search or a similarly deterministic algorithm.
3) Automatic labeling using a prediction algorithm (e.g., the machine learning approach presented here).

The first option is the status quo on how the cost estimating and analysis community operates. Analysts spend many hours manually re-labeling all WBS elements across all projects to a preferred standard. This option is fine for small datasets where there is not enough data available to train a model. It may also be a preferable option when the data are too variable for historical data to be able to properly classify the current/future data.

However, this approach has two major drawbacks. First, even experienced, qualified analysts will map WBS elements differently. This leads to some degree of subjectivity, which varies from analyst to analyst. Second, for any reasonably sized data set, this approach requires a phenomenal amount of time. In summary, manual labeling is simple and easy to understand, but may be inconsistent and highly time consuming.

The second option is a deterministic approach that attempts to map WBS elements according to some prescribed set of rules. For example, using a list of keywords to search within the WBS elements to perform the mapping. This approach is best suited for (1) portfolios that contain similar enough projects so that the keyword search provides adequate coverage, and (2) when the goal of the standardization is a fairly high-level WBS. The deterministic nature of this approach allows for expediency but at the expense of flexibility. Even small changes in word order or other semantic nuances could cause error in the results and there is no way to identify when a miss occurred without further manual intervention. To put it simply, there are no measures of performance for missed opportunities. Mapping data to a WBS is a single-label problem but using a keyword search can turn it into a multi-label problem when more than one label is returned for the same WBS element. Therefore, this type of algorithm requires additional rules to address conflicting results and prioritize certain labels over others. The deterministic nature of these decision rules means that there is little opportunity for nuanced interpretations of text and, once again, suffers from the same limitations when changes in word order may lead to missing results.

The third option is a predictive approach such as the one discussed in detail in this paper. There is a heavy research and development component up front, but once developed, it can be updated and maintained with new data at a far lesser effort. This option is able to handle large, complex datasets that would be cost-prohibitive to label by hand. Further, it provides a consistent approach with measurable performance. For example, we may not know how many errors a manually mapped dataset has – it is rarely possible to eliminate all human error. But we *can* estimate the error that our algorithm may have.

These automated methods – regardless of the level of sophistication – do have a few drawbacks as well. First, the methodologies are less clear to most analysts, even in the simple settings, and especially in the complex ones. They may be considered “black box” to some, even though the mathematics are fairly well understood. Second, the methodologies will have error. Much in the same way that self-driving cars may ultimately prove safer than human-driven ones, the individual tends to believe that they are less prone to error than the general population. As such, there is a tendency for an analyst to trust their manual mapping (even if there is human error) but distrust the automation – especially if developed by someone else. In summary, automated labeling is more complex to implement and understand, but is highly consistent and substantially more time/cost effective than a manual approach.

The third option is one that is very common to use during the development and refinement of automation. Self-driving features are not yet in consumer vehicles, but semi-automated driving features such as smart cruise controls that can break, accelerate, and change lanes are becoming more accepted by consumers. A semi-automated methodology includes an “analyst-in-the-loop” step that verifies and cleans up errors in the automation. There are several possibilities here. For one, the automated models can flag WBS records that were mapped with a low degree of confidence. As another option, information from the training process may be used to notify analysts of historically “tricky” data for the model to correctly classify. Either way, information is provided such that a manual step is layered in to check and clean the results before being made final. As an added bonus, this cleanup step can be cycled back into the model training data, further improving the performance of future iterations. This approach blends the best of both worlds. Analysts have a high degree of confidence in the mappings, conducted a in consistent, time effective, and repeatable way.

Many models are narrow in their applicability or are best applied to a specific niche. However, this model presents multiple avenues for implementation that could improve some of the most foundational aspects of cost estimating and project management. Our goal in presenting this work to the ICEAA community is threefold: (1) to share the know-how, lessons learned, and framework of this model so that it may be adapted to other data sources, (2) to break the chain of perpetual development of narrow use-case models, and (3) to minimize duplication of effort within the industry. This model represents a working demonstration that could hypothetically be applied to any hierarchically structured dataset, not just work breakdown structures.

In an era where organizations are increasingly interested in more efficiently understanding their data, this model presents an excellent leverage opportunity. With the necessary approval, the model authors will consider publishing the model (e.g., on GitHub or other open-source platforms) which would allow other agencies to utilize it in its “as is” state. The end user would still be required to provide a labelled sample dataset for training and tuning purposes unique to their intended use-case.

The cost estimating and analysis community continues to receive more and more data each year. As volumes of information grow, the methodologies, techniques, and toolsets must grow with it. Manual processing of data, including WBS’s, is becoming increasingly less viable at scale. The community *must* adapt methodologies as presented in this paper, or risk being left being as others encroach into the space, bringing innovation and technology into an area that is lacking. Hierarchical classification performed by utilizing NLP and machine learning is an emerging field, even outside of our community. This work is ongoing and continues to evolve with the cutting-edge research on the topic area. The authors look forward to continuing be on the forefront of this, sharing results with both the ICEAA community and the greater machine learning communities at large.

APPENDIX A: REFERENCES

1. Balakrishnan, Vimala, and Ethel Lloyd-Yemoh. 2014. "Stemming and lemmatization: A comparison of retrieval performances." Proceedings of SCEI Seoul Conferences. Seoul.
2. Bouchet-Valat, Milan. 2020. "SnowballC: Snowball Stemmers Based on the C 'libstemmer' UTF-8 Library." https://CRAN.R-project.org/package=SnowballC.
3. Chinchor, Nancy. 1992. "MUC-4 evaluation metrics." MUC4 '92: Proceedings of the 4th conference on Message Understanding. McLean. 22-29. https://doi.org/10.3115/1072064.1072067.
4. Department of Energy. 2023. "Capital Acquisition Estimating Framework." https://www.energy.gov/sites/default/files/2022-09/UUR_SAND2022-9243%20C_CapitalAcquisitionEstimatingFramework.pdf.
5. Department of Energy. 2023. Missions. https://www.energy.gov/nnsa/missions.
6. Department of Energy. 2023. Supporting NNSA's missions. https://www.energy.gov/nnsa/supporting-nnsas-missions.
7. 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.
8. Department of Energy. 2023. Mission: Office of Project Management. https://www.energy.gov/projectmanagement/mission.
9. Ferreira, Beatriz Q, Luís Baía, João Faria, and Ricardo Gamelas Sousa. 2018. "A Unified Model with Structured Output for Fashion Images." arXiv. https://arxiv.org/pdf/1806.09445.pdf.
10. Kiritchenko, Svetlana, Stan Matwin, Richard Nock, and A. Fazel Famili. 2006. Learning and Evaluation in the Presence of Class Hierarchies: Application to Text Categorization. Advances in Artificial Intelligence. Canadian AI 2006: Lecture Notes in Computer Science 4013: 395-406.
11. Miranda, F.M., N. Köehnecke, and B.Y. Renard. 2023. "HiClass: a Python Library for Local Hierarchical Classification Compatible with Scikit-learn." Journal of Machine Learning Research, 24 (29): 1–17. https://jmlr.org/papers/v24/21-1518.html.
12. Lapoff, Maura, NNSA Office of Programming, Analysis, and Evaluation. "Advanced Natural Language Processing for Work Breakdown Structures." 2022. ICEAA Professional Development and Training Workshop.
13. Lex-tek International. n.d. Onix. http://www.lextek.com/manuals/onix/stopwords1.html.
14. Lovins, Julie Beth. 1968. "Development of a Stemming Algorithm." Mechanical Translation and Computational Linguistics 11 (1-2): 22-31.
15. Mikolov, Tomas, 2013. "Efficient Estimation of Word Representations in Vector Space." arXiv. https://arxiv.org/pdf/1301.3781.pdf.
16. Mikolov, Tomas Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeffrey Dean. 2013. "Distributed Representations of Words and Phrases and their Compositionality." Advances in Neural Information Processing Systems. https://arxiv.org/pdf/1310.4546.pdf.
17. NNSA Office of Cost Estimating and Program Evaluation. 2017. Responsibilities for Indepedent Cost Estimates. January 10.

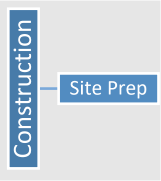
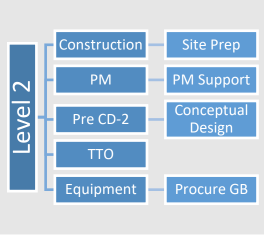
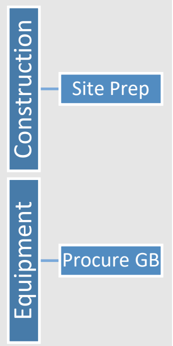
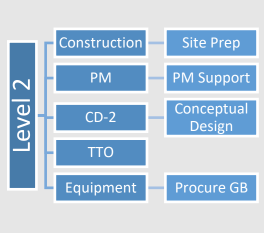
18. Pedregosa, Gaël V, et al. 2011. "Scikit-learn: Machine Learning in Python." JMLR 12 (1): 2825-2830. https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html.
19. Porter, M F. 1980. "An Algorithm for Suffix Stripping." Program 14 (3): 130-137.
20. Porter, Martin, and Richard Boulton. n.d. Snowball. Accessed December 21, 2021. http://snowball.tartarus.org/algorithms/english/stop.txt.
21. Sechidis, Konstantinos, Grigorios Tsoumakas, and Ioannis Vlahavas. 2011. "On the Stratification of Multi-Label Data." Department of Informatics Aristotle University of Thessaloniki. lpis.csd.auth.gr/publications/sechidis-ecmlpkdd-2011.pdf
22. Silla, Carlos N, and Alex A. Freitas. 2011. "A survey of hierarchical classification across different application domains." Data Mining and Knowledge Discovery 22: 31-72. https://doi.org/10.1007/s10618-010-0175-9
23. Tharwat, Alaa. 2018. "Classification assessment methods." Applied Computing and Informatics 17(1): 168-192.
24. 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.
25. Tixier, Antoine JP, Michalis Vazirgiannis, and Matthew R. Hallowell. (2016). "Word Embeddings for the Construction Domain." arXiv. https://arxiv.org/abs/1610.09333
26. Webster, Jonathan J, and Chunya Kit. 1992. "Tokenization as the Initial Phase in NLP." COLING 4: 1106-1110.

APPENDIX B: ADDITIONAL TABLES

Table B1: Stemming does not significantly improve performance.

Model Combination	Average hF	
	Not Stemmed	Stemmed
CC	**0.5831385**	**0.583040611**
decision tree	0.508006333	0.492458667
logistic regression	0.655017333	0.656744667
multinomial naive Bayes	0.577355333	0.575243667
random forest	0.549813333	0.549678
stochastic gradient descent	0.604067333	0.619944333
support vector classifier	0.604571333	0.604174333
LCPL	**0.5629785**	**0.561043**
decision tree	0.479327	0.469834667
logistic regression	0.646801333	0.648961667
multinomial naive Bayes	0.587387	0.586214
random forest	0.538398667	0.539161667
stochastic gradient descent	--	--
support vector classifier	--	--
LCPN	**0.56399575**	**0.568464**
decision tree	0.465401667	0.467151667
logistic regression	0.636974667	0.637989667
multinomial naive Bayes	0.591456667	0.589690667
random forest	0.56215	0.579024
stochastic gradient descent	#DIV/0!	#DIV/0!
support vector classifier	#DIV/0!	#DIV/0!
LCPPN	**0.589039778**	**0.589948111**
decision tree	0.492047	0.494807
logistic regression	0.645643667	0.647830333
multinomial naive Bayes	0.587387	0.586214
random forest	0.545501	0.555630333
stochastic gradient descent	0.642793333	0.63418
support vector classifier	0.620866667	0.621027
MO	**0.598516333**	**0.600608611**
decision tree	0.495093	0.491640667
logistic regression	0.657108333	0.660222667
multinomial naive Bayes	0.596147	0.595580333
random forest	0.563184	0.566582667
stochastic gradient descent	0.635711667	0.644705333
support vector classifier	0.643854	0.64492
Grand Total	**0.582002487**	**0.583062**

Table B2: Classification types differ by number of targets, target cardinality, and result.[4]

Classifier	Targets	Target Cardinality	Labels	Inputs	Outputs		Result
Binary	1	2	**Construction** 1. Yes 2. No	**Text** Site Prep PM Support Procure GB Conceptual Design	**Construction** Yes No No No		
Multiclass	1	>2	**Level 2** 1. Pre CD-2 2. Construction 3. PM 4. SD&E 5. TTO Equipment	**Text** Site Prep PM Support Procure GB Conceptual Design	**Level 2** Construction PM Equipment Pre CD-2		 Level 2 Construction Site Prep PM PM Support Pre CD-2 Conceptual Design TTO Equipment Procure GB
Multilabel	>1	2	**Site Work** 1. Yes 2. No **Equipment** 1. Yes 2. No	**Text** Site Prep PM Support Procure GB Conceptual Design	**Construction** Yes No No No	**Equipment** No No No Yes	
Multiclass-Multioutput	>1	>2	**Level 2** 1. Pre CD-2 2. Construction 3. PM 4. SD&E 5. TTO 6. Equipment	**Text** Site Prep PM Support Procure GB Conceptual Design	**Level 2** Construction PM Equipment Pre CD-2 ss	**Equipment** -- -- Procurement --	 Level 2 Construction Site Prep PM PM Support CD-2 Conceptual Design TTO Equipment Procure GB

[4] Table B2 adapted from Scikit-learn's API (Reference 18).

Equipment

1. Procurement
2. Installation & Integration

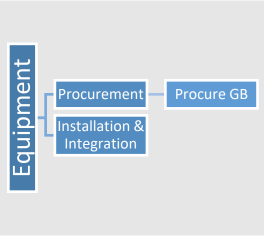
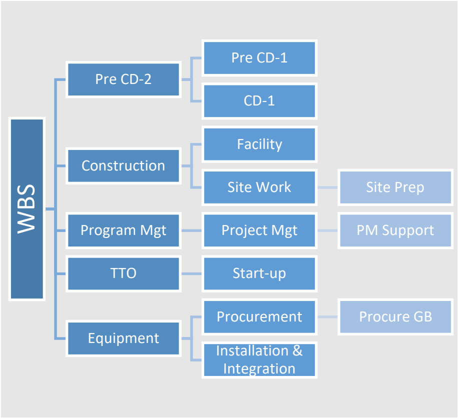


Table B3: A MIMO classifier is a classifier that uses more than one input to classify data into more than one output.[5]

Classifier	Targets	Target Cardinality	Labels	Input		Output	
				Text	**Project Type**	**Level 2**	**Level 3**
MIMO	>1	>1	**Level 2** 1. Pre CD-2 2. Construction 3. PM 4. SD&E 5. TTO 6. Equipment **Level 3** 1. Pre CD-1 2. CD-1 3. Site Work 4. Facility 5. Procurement 6. Installation & Integration 7. Project Management 8. Conceptual Design 9. Start-up	Site Prep	NC	Construction	Site Work
				PM Support	NC	PM	Project Management
				Procure GB	Equip	Equipment	Procurement
				Conceptual Design	Equip	Pre CD-2	Conceptual Design

[5] Table B3 adapted from Scikit-learn’s API (Reference 24).

Table B4: Hierarchical classification classifies data when the multiple output targets are hierarchically related.[6]

Classifier	Targets	Target Cardinality	Labels	Input	Output		Result
Hierarchical Classifier	>1	>1	**Level 2** 1. Pre CD-2 2. Construction 3. PM 4. SD&E 5. TTO 6. Equipment **Level 3** 1. Pre CD-1 2. CD-1 3. Site Work 4. Facility 5. Procurement 6. Installation & Integration 7. Project Management 8. Conceptual Design 9. Start-up	**Text** Site Prep PM Support Procure GB Conceptual Design	**Level 2** Construction PM Equipment Pre CD-2	**Level 3** Site Work Project Management Procurement Conceptual Design	 WBS Pre CD-2 Pre CD-1 CD-1 Construction Facility Site Work Site Prep Program Mgt Project Mgt PM Support TTO Start-up Equipment Procurement Procure GB Installation & Integration

[6] Table B4 adapted from Scikit-learn's API (Reference 24).

APPENDIX C: NNSA USE CASES

C.1. Immediate Benefit – Cost Estimating and Project Benchmarking

The most obvious and easiest application of this model is for cost estimating and project benchmarking. Cost estimates often rely on historic data from analogous projects to inform the anticipated cost of a new project. Analogies are typically applied as proportional factors to account for differences in total cost. For example, an analyst tasked to estimate the cost to design a new particle accelerator could calculate the ratio of the total design cost to the total construction cost for previously constructed particle accelerators. Analogies can also be in real dollar amounts – "the gloveboxes purchased for Projects X, Y, and Z are analogous to the gloveboxes needed in Project A." This same logic applies in project benchmarking, where an analyst will seek to validate an existing cost estimate, especially ones based on subject matter expert (SME) judgment, with historic costs to determine the relative accuracy of the existing estimate.

Previously, to accomplish any of this foundational estimating work, an analyst would have to research the database of historic projects, determine which ones were most analogous, find the relevant cost data, and manually categorize it to the best of their knowledge. Depending on the experience level of the analyst and knowledge of historic projects, this process can be extremely labor-intensive (as the authors of this paper know first-hand), especially when the data is ambiguous and open to interpretation. The results of this work greatly reduce this arduous process down to selecting the relevant analogous projects and verifying the model results via post-processing. This allows analysts to focus more time on problem solving and addressing stakeholder needs vs. cleaning and categorizing data. In some cases, this could also result in reducing the amount of time needed to complete a cost estimate, which allows for more efficient project execution and use of taxpayer funds.

In short, these results provide cleaned and categorized historic cost data from every NNSA capital asset project in PARS, allowing a cost estimator to immediately use them in their work.

C.2. Near Term – Parametric Modeling

Analogies are often the most accessible solution for estimating when a large body of historic data is not readily available, but since these results provide an organized database of every PARS project's EV data, parametric modeling becomes much more practical. While it was not impossible prior to this model, it would have been a prohibitively expensive time and effort commitment to fund a group of analysts to manually categorize enough of the projects to consider a parametric solution. With the scope of work that PA&E, CEPE, and DOE-PM are responsible for, relative to the number of analysts and funding available to them, this undertaking was never seriously considered. Now with the data readily available, analysts just have to quickly verify the model's results and conduct any needed post-processing, which allows them to focus more effort developing cost estimating relationships between the various WBS buckets. This work allows longstanding questions, such as the correlation between project type (nuclear vs. non-nuclear, experimental vs. production, etc.) and project management, for example, to be explored and answered, which then directly feeds into providing more accurate, efficient, and defensible estimates.

C.3. Medium Term – PARS Integration

The immediate and near-term use cases are listed as such since they are only limited by the ability of analysts to synthesize and interpret the results provided by the model. The remaining use cases require inter-organizational discussions and thus are expected to take longer to implement, be more aspirational, or both.

Arguably the simplest of these use cases is to have the model directly integrate into the PARS database. DOE PM is already considering asking EV managers to manually reclassify data into a high level standard WBS, so this model represents a time-saving solution to that problem. Since the data that feeds into the model is PARS EV data, it makes sense to simply integrate the model with PARS. This would mean that every time a project EV manager uploads new data to PARS, the model automatically categorizes it, offering PARS users instant access to cleaned data in a standard format. Without PARS integration, the model would have to pull the PARS EV data and run in a separate platform, and likely be more difficult to automate. This would also force interested analysts to have to request access to another platform and increase the burden of spreading awareness of the model to the NNSA community. Integrating the model with the PARS database both improves the operational efficiency of the model and enhances the capability offerings of PARS. Considering that PARS is looking to add more monthly users, this use case is highly feasible if all necessary stakeholders agree on the value of the model.

C.4. Longer Term – Standard WBS Format

This is possibly the most impactful use case, but also the most difficult to implement. As discussed in the assessment of the status quo, the impetus for creating this model was lack of a standard WBS format and accompanying WBS dictionary. Because the various M&O sites track EV data according to their own WBS's and this data is then interpreted in different ways by various NNSA offices, there was a need to create a complex NLP model to synthesize the huge body of incongruous data into a standard format to facilitate accurate cost estimates and informed project management, tracking, and benchmarking. Now that this model has created its own WBS format and dictionary, the challenge becomes getting stakeholders to agree to use it. This challenge can be addressed in several ways.

One option, the path of least potential resistance and effort, involves the model becoming an intermediary for use in reconciling multiple WBS formats. For example, if M&Os want to continue to produce unique WBS's for each project, the model enables a side-by-side comparison of the WBS's. Then, the M&Os could be made aware that their EV data is being reformatted before being analyzed, and they may see the value in eliminating that step and formatting their WBS's in the same format as the model's format. In the authors' opinion, the ideal scenario for the NNSA would be that every future project uses a standard WBS, thereby negating the need for the model.

A second option, the path of greatest potential resistance and effort, entails updating the CAEF created by CEAG The CEAG Council created the CAEF as an attempt create a standard WBS format with M&O buy-in, and the CAEF WBS format was the original WBS framework for this model. However, during model development the analysts applying the CAEF WBS to historical data identified areas for improvement. This spurred the creation of the WBS and dictionary utilized for this model. Updating the CAEF would involve presenting this paper and its findings to the CEAG Council to discuss and approve these changes. Even if there is less interest in updating the CAEF, or if the revisions do not align with the structure presented here, the goal is for all stakeholders to agree on a standard and exhaustive WBS, regardless of the structure's similarity to the NLP model. Once a WBS is agreed upon, the model can be updated to categorize that data into that WBS. Despite the potential value of this option, there sheer number of stakeholders whose buy-in would be required could mean this option is more aspirational than feasible.

A third option that is similar in concept to and likely to have a higher probability of success than the second option encompasses collaborating with the three main NNSA offices that conduct cost estimates with the PARS data – PA&E, CEPE, and DOE-PM. To some degree, all three offices have conducted manual categorization of the PARS data, each with different results and assumptions. If it is not possible to persuade the M&Os to adopt a standard format, it would still be a huge efficiency improvement if the cost estimating offices agreed on how to best format and define WBS elements that underlie cost estimates. On top of that, it would be even more beneficial if the offices aligned their assumptions about the historic cost data, since having the same WBS format only goes so far if there are different interpretations of how existing data should be categorized into that format. For example, should "construction management" be an indirect Project Management cost or a direct Construction cost? Is "30% Design" part of Conceptual Design or Preliminary Design? As stated previously, there are a myriad of words and terms that can be and are defined differently between offices, projects, and contexts, creating differing assumptions and definitions.

Each office is tasked with conducting cost estimates independent of the other offices, but agreement on foundational data assumptions and a WBS does not infringe upon that independence. Rather, it means that all cost estimating organizations have access to the same historical data and there is agreement that the data is authoritative. This necessarily means that discussions regarding the differences between organizations' cost estimates can focus on what's important, including but not limited to differences in estimating assumptions, cost estimating methodologies, consideration of uncertainty/risk, etc. The benefit is all estimating organizations and stakeholders, decisionmakers included, having a clear understanding of differences and solid basis for determining the most realistic cost estimate and budget. With this path forward, it is again more important to agree on a WBS and definitions vs. advocating for the structure presented in this paper. Being able to run a cost estimating office agreed-upon format through the model would strengthen the NNSA cost estimating community, present leadership with more accessible and consistent results, and maintain the independent strengths of each cost estimating office.

APPENDIX D: STANDARD WORK BREAKDOWN STRUCTURE

WBS	Title	Level
0	Project	1
1	Pre CD-0, CD-0, and CD-1	2
1.1	Pre CD-0	3
1.1.1	Pre CD-0 Design	4
1.1.2	Miscellaneous Pre-CD-0	4
1.2	CD-0/Pre-Conceptual Design	3
1.2.1	CD-0 Estimate	4
1.2.2	Program Requirements Development	4
1.2.3	ICE/ICR During CD-0 Design	4
1.2.4	Pre-Conceptual Design	4
1.2.5	Miscellaneous CD-0/Pre-Conceptual Design	4
1.3	CD-1/Conceptual Design	3
1.3.1	Analysis of Alternatives (AoA)	4
1.3.2	ICE/ICR during CD-1 Design / IPR	4
1.3.3	Conceptual Design	4
1.3.4	Miscellaneous CD-1/Conceptual Design	4
1.4	Miscellaneous Pre-CD-0, CD-0 - CD-1 Activities	3
2	Construction	2
2.1	Site Work	3
2.1.1	Information Technology	4
2.1.2	Site Preparation	4
2.1.2.1	Site Clearing	5
2.1.2.2	Site Demolition & Relocations	5
2.1.2.3	Site Earthwork	5
2.1.2.4	Temporary Construction	5
2.1.2.5	Miscellaneous Site Preparation	5
2.1.2.6	General Conditions	5

2.1.3	Site Improvement	4
2.1.3.1	Roadways & Parking Lots	5
2.1.3.2	Pedestrian Paving	5
2.1.3.3	Site Development	5
2.1.3.4	Landscaping	5
2.1.3.5	Tunnel	5
2.1.3.6	Warehouse/Locker Room/Storage	5
2.1.3.7	Miscellaneous Site Improvement	5
2.1.4	Site Utilities	4
2.1.4.1	Site Civil/Mechanical Utilities	5
2.1.4.1.1	**Domestic Water & Fire Protection Water**	**6**
2.1.4.1.2	**Sanitary Sewer**	**6**
2.1.4.1.3	**Storm Sewer**	**6**
2.1.4.1.4	**Natural Gas Supply**	**6**
2.1.4.1.5	**Miscellaneous Site Civil / Mechanical Utilities**	**6**
2.1.4.2	Site Electrical Utilities	5
2.1.4.2.1	**Electrical Service/Distribution**	**6**
2.1.4.2.2	**Site Lighting**	**6**
2.1.4.2.3	**Site Communications & Security**	**6**
2.1.4.2.4	**Site Information Technology**	
2.1.4.2.5	**Miscellaneous Site Electrical Utilities**	**6**
2.1.4.3	Miscellaneous Site Utilities	5
2.1.5	Miscellaneous Site Work	4
2.2	**Facility**	**3**
2.2.1	Facility Structure	4
2.2.1.1	New Construction	5
2.2.1.1.1	**Foundation & Substructure**	**6**
2.2.1.1.2	**Shell (Superstructure)**	**6**
2.2.1.1.2.1	Roofing	
2.2.1.1.2.2	Miscellaneous Shell	

2.2.1.1.3	**Interiors**	**6**
2.2.1.1.3.1	Doors	7
2.2.1.1.3.2	Stairwells	7
2.2.1.1.3.3	Miscellaneous Interiors	7
2.2.1.1.4	**Miscellaneous New Construction Structure**	**6**
2.2.2.2	Refurbishment	5
2.2.2.2.1	**Foundation & Substructure**	**6**
2.2.2.2.2	**Shell (Superstructure)**	**6**
2.2.2.2.2.1	Roofing	
2.2.2.2.2.2	Miscellaneous Shell	
2.2.2.2.3	**Interiors**	**6**
2.2.2.2.3.1	Doors	7
2.2.2.2.3.2	Stairwells	7
2.2.2.2.3.3	Miscellaneous Interiors	7
2.2.2.2.4	**Miscellaneous Refurbished Structure**	**6**
2.2.2.3	Miscellaneous Facility Structure	5
2.2.2	Facility Utilities	4
2.2.2.1	New Construction	5
2.2.2.1.1	**Conveying System**	**6**
2.2.2.1.2	**Plumbing System**	**6**
2.2.2.1.2.1	Plumbing Fixtures	7
2.2.2.1.2.2	Domestic Water Distribution	7
2.2.2.1.2.3	Sanitary Waste	7
2.2.2.1.2.4	Rain Water Drainage	7
2.2.2.1.2.5	Miscellaneous Plumbing System	7
2.2.2.1.3	**Heating Ventilation and Air Conditioning (HVAC) System/Mechanical System**	**6**
2.2.2.1.3.1	Heating	7
2.2.2.1.3.2	Ventilation	7
2.2.2.1.3.3	Air Conditioning	7

2.2.2.1.3.4	Miscellaneous HVAC Systems and Mechanical System	7
2.2.2.1.4	**Fire Protection**	**6**
2.2.2.1.4.1	Water and gas-based systems	7
2.2.2.1.4.2	Alarm systems	7
2.2.2.1.4.3	Structural-based systems	7
2.2.2.1.4.4	Miscellaneous Fire Protection Systems	7
2.2.2.1.5	**Electrical System**	**6**
2.2.2.1.5.1	Electrical Service and Distribution	7
2.2.2.1.5.2	Lighting and Branch Wiring	7
2.2.2.1.5.3	Communications and Security	7
2.2.2.1.5.4	Information Technology	7
2.2.2.1.5.5	Miscellaneous Electrical Systems	7
2.2.2.1.6	**Miscellaneous New Construction Utilities**	**6**
2.2.2.2	Refurbishment	5
2.2.2.2.1	**Conveying System**	**6**
2.2.2.2.2	**Plumbing System**	**6**
2.2.2.2.2.1	Plumbing Fixtures	7
2.2.2.2.2.2	Domestic Water Distribution	7
2.2.2.2.2.3	Sanitary Waste	7
2.2.2.2.2.4	Rain Water Drainage	7
2.2.2.2.2.5	Miscellaneous Plumbing	7
2.2.2.2.3	**Heating Ventilation and Air Conditioning (HVAC) System/Mechanical System**	**6**
2.2.2.2.3.1	Heating	7
2.2.2.2.3.2	Ventilation	7
2.2.2.2.3.3	Air Conditioning	7
2.2.2.2.3.4	Miscellaneous HVAC Systems and Mechanical System	7
2.2.2.2.4	**Fire Protection**	**6**
2.2.2.2.4.1	Water and gas-based systems	7
2.2.2.2.4.2	Alarm systems	7

2.2.2.2.4.3	Structural-based systems	7
2.2.2.2.4.4	Miscellaneous Fire Protection Systems	7
2.2.2.2.5	**Electrical System**	**6**
2.2.2.2.5.1	Electrical Service and Distribution	7
2.2.2.2.5.2	Lighting and Branch Wiring	7
2.2.2.2.5.3	Communications and Security	7
2.2.2.2.5.4	Information Technology	7
2.2.2.2.5.5	Miscellaneous Electrical Systems	7
2.2.2.2.6	**Miscellaneous Refurbishment Utilities**	**6**
2.2.3	Support Facilities	4
2.2.4	Special Construction	4
2.2.5	Miscellaneous Facility Construction	4
2.3	**Construction Management**	**3**
2.3.1	Site Work	4
2.3.2	Facility	4
2.3.2.1	Facility Structure	5
2.3.2.2	Facility Utilities	5
2.3.3	Equipment	4
2.3.4	Decommissioning/Deactivation, Demolition, and Disposal	4
2.3.5	Transition to Operations	4
2.3.6	Miscellaneous Construction Management	
3	**Systems Design and Engineering**	**2**
3.1	**System Definition and Design**	**3**
3.1.1	Site Work	4
3.1.2	Facility Construction	4
3.1.2.1	Facility Structure	5
3.1.2.2	Facility Utilities	5
3.1.3	Decommissioning/Deactivation, Demolition, and Disposal	4
3.1.4	Equipment	4
3.1.5	Transition to Operations	4

3.1.6	Miscellaneous System Definition and Design	4
3.2	**Requirements Management**	**3**
3.2.1	Site Work	4
3.2.2	Facility Construction	4
3.2.2.1	Facility Structure	5
3.2.2.2	Facility Utilities	5
3.2.3	Decommissioning/Deactivation, Demolition, and Disposal	4
3.2.4	Equipment	4
3.2.5	Transition to Operations	4
3.2.6	Miscellaneous Requirements Management	4
3.3	**Risk Management**	**3**
3.3.1	Site Work	4
3.3.2	Facility Construction	4
3.3.2.1	Facility Structure	5
3.3.2.2	Facility Utilities	5
3.3.3	Decommissioning/Deactivation, Demolition, and Disposal	4
3.3.4	Equipment	4
3.3.5	Transition to Operations	4
3.3.6	Miscellaneous Risk Management	4
3.4	**Configuration Management**	**3**
3.4.1	Site Work	4
3.4.2	Facility Construction	4
3.4.2.1	Facility Structure	5
3.4.2.2	Facility Utilities	5
3.4.3	Decommissioning/Deactivation, Demolition, and Disposal	4
3.4.4	Equipment	4
3.4.5	Transition to Operations	4
3.4.6	Miscellaneous Configuration Management	4
3.5	**Specialty Engineering and Analysis / Design Management / Engineering Management**	**3**

3.5.1	Site Work	4
3.5.2	Facility Construction	4
3.5.2.1	Facility Structure	5
3.5.2.2	Facility Utilities	5
3.5.3	Decommissioning/Deactivation, Demolition, and Disposal	4
3.5.4	Equipment	4
3.5.5	Transition to Operations	4
3.5.6	Miscellaneous Specialty Engineering & Analysis/Design Management/Engineering Management	4
3.6	**Title I/CD-2/Preliminary Design and Title II/CD-3 Final Design**	**3**
3.6.1	Title I / CD-2 Preliminary Design	4
3.6.1.1	Site Work	5
3.6.1.2	Facility Construction	5
3.6.1.2.1	Facility Structure	6
3.6.1.2.2	Facility Utilities	6
3.6.1.3	Decommissioning/Deactivation, Demolition, and Disposal	5
3.6.1.4	Equipment	5
3.6.1.5	Transition to Operations	5
3.6.1.6	Miscellaneous Title I / CD-2 Preliminary Design	5
3.6.2	Title II / CD-3 Final Design	4
3.6.2.1	Site Work	5
3.6.2.2	Facility Construction	5
3.6.2.2.1	Facility Structure	6
3.6.2.2.2	Facility Utilities	6
3.6.2.3	Decommissioning/Deactivation, Demolition, and Disposal	5
3.6.2.4	Equipment	5
3.6.2.5	Transition to Operations	5
3.6.2.6	Miscellaneous Title II / CD-3 Final Design	5
3.6.3	Miscellaneous Title I/CD-2/Preliminary Design and Title II/CD-3 Final Design	4

3.6.3.1	Site Work	5
3.6.3.2	Facility Construction	5
3.6.3.2.1	Facility Structure	6
3.6.3.2.2	Facility Utilities	6
3.6.3.3	Decommissioning/Deactivation, Demolition, and Disposal	5
3.6.3.4	Equipment	5
3.6.3.5	Transition to Operations	5
3.6.3.6	Other Miscellaneous Title I/CD-2/Preliminary Design and Title II/CD-3 Final Design	5
3.7	**Title III Design / Design Closeout**	**3**
3.7.1	Site Work	4
3.7.2	Facility Construction	4
3.7.2.1	Facility Structure	5
3.7.2.2	Facility Utilities	5
3.7.3	Decommissioning/Deactivation, Demolition, and Disposal	4
3.7.4	Equipment	4
3.7.5	Transition to Operations	4
3.7.6	Miscellaneous Title III Design	4
3.8	**Safety Basis**	**3**
3.8.1	Site Work	4
3.8.2	Facility Construction	4
3.8.2.1	Facility Structure	5
3.8.2.2	Facility Utilities	5
3.8.3	Decommissioning/Deactivation, Demolition, and Disposal	4
3.8.4	Equipment	4
3.8.5	Transition to Operations	4
3.8.6	Miscellaneous Safety Basis	4
3.9	**Miscellaneous Systems Design and Engineering**	**3**
4	**Program Management**	**2**
4.1	**Project Management, Project Controls, Operations**	**3**

4.1.1	Site Work	4
4.1.2	Facility Construction	4
4.1.2.1	Facility Structure	5
4.1.2.2	Facility Utilities	5
4.1.3	Systems Design and Engineering	4
4.1.4	Decommissioning/Deactivation, Demolition, and Disposal	4
4.1.5	Equipment	4
4.1.6	Transition to Operations	4
4.1.7	Miscellaneous Project Management, Project Controls, Operations	4
4.2	**Information and Communication Management**	**3**
4.2.1	Site Work	4
4.2.2	Facility Construction	4
4.2.2.1	Facility Structure	5
4.2.2.2	Facility Utilities	5
4.2.3	Systems Design and Engineering	4
4.2.4	Decommissioning/Deactivation, Demolition, and Disposal	4
4.2.5	Equipment	4
4.2.6	Transition to Operations	4
4.2.7	Miscellaneous Information and Communication Management	4
4.3	**Training and Travel**	**3**
4.3.1	Site Work	4
4.3.2	Facility Construction	4
4.3.2.1	Facility Structure	5
4.3.2.2	Facility Utilities	5
4.3.3	Systems Design and Engineering	4
4.3.4	Decommissioning/Deactivation, Demolition, and Disposal	4
4.3.5	Equipment	4
4.3.6	Transition to Operations	4
4.3.7	Miscellaneous Training and Travel	4
4.4	**Office and Administrative Support (O&AS)**	**3**

4.4.1	Site Work	4
4.4.2	Facility Construction	4
4.4.2.1	Facility Structure	5
4.4.2.2	Facility Utilities	5
4.4.3	Systems Design and Engineering	4
4.4.4	Decommissioning/Deactivation, Demolition, and Disposal	4
4.4.5	Equipment	4
4.4.6	Transition to Operations	4
4.4.7	Miscellaneous O&AS	4
4.5	**Environment, Safety, and Health / Quality Assurance (ES&H/QA)**	**3**
4.5.1	Site Work	4
4.5.2	Facility Construction	4
4.5.2.1	Facility Structure	5
4.5.2.2	Facility Utilities	5
4.5.3	Systems Design and Engineering	4
4.5.4	Decommissioning/Deactivation, Demolition, and Disposal	4
4.5.5	Equipment	4
4.5.6	Transition to Operations	4
4.5.7	Miscellaneous ES&H/QA	4
4.6	**Safety Basis**	**3**
4.6.1	Site Work	4
4.6.2	Facility Construction	4
4.6.2.1	Facility Structure	5
4.6.2.2	Facility Utilities	5
4.6.3	Systems Design and Engineering	4
4.6.4	Decommissioning/Deactivation, Demolition, and Disposal	4
4.6.5	Equipment	4
4.6.6	Transition to Operations	4
4.6.7	Miscellaneous Safety Basis	4

4.7	**Material and Supply**	**3**
4.7.1	Site Work	4
4.7.2	Facility Construction	4
4.7.2.1	Facility Structure	5
4.7.2.2	Facility Utilities	5
4.7.3	Systems Design and Engineering	4
4.7.4	Decommissioning/Deactivation, Demolition, and Disposal	4
4.7.5	Equipment	4
4.7.6	Transition to Operations	4
4.7.7	Miscellaneous Material and Supply	4
4.8	**General Requirements**	**3**
4.8.1	Site Work	4
4.8.2	Facility Construction	4
4.8.2.1	Facility Structure	5
4.8.2.2	Facility Utilities	5
4.8.3	Systems Design and Engineering	4
4.8.4	Deactivation and Decommissioning	4
4.8.5	Equipment	4
4.8.6	Transition to Operations	4
4.8.7	Miscellaneous General Requirements	4
4.9	**Miscellaneous Program Management**	**3**
4.9.1	Site Work	4
4.9.2	Facility Construction	4
4.9.2.1	Facility Structure	5
4.9.2.2	Facility Utilities	5
4.9.3	Systems Design and Engineering	4
4.9.4	Deactivation and Decommissioning	4
4.9.5	Equipment	4
4.9.6	Transition to Operations	4
4.9.7	Other Miscellaneous Program Management	4

5	**Equipment**	**2**
5.1	**Procurement**	**3**
5.1.1	Furniture, Fixtures & Office Equipment	4
5.1.2	Process Equipment	4
5.1.3	Support Equipment	4
5.1.4	Miscellaneous Procurement	4
5.2	**Installation and Integration**	**3**
5.2.1	Furniture, Fixtures & Office Equipment	4
5.2.1.1	Installation	5
5.2.1.2	Commissioning	5
5.2.1.3	Construction Materials	5
5.2.1.4	Miscellaneous Furniture, Fixtures & Office Equipment	5
5.2.2	Process Equipment	4
5.2.2.1	Installation	5
5.2.2.1.1	**Equipment**	**6**
5.2.2.1.2	**Utilities**	**6**
5.2.2.1.2.1	Conveying System	7
5.2.2.1.2.2	Plumbing System	7
5.2.2.1.2.2.1	Plumbing Fixtures	8
5.2.2.1.2.2.2	Domestic Water Distribution	8
5.2.2.1.2.2.3	Sanitary Waste	8
5.2.2.1.2.2.4	Rain Water Drainage	8
5.2.2.1.2.2.5	Miscellaneous Plumbing System	8
5.2.2.1.2.3	Heating Ventilation and Air Conditioning (HVAC) System/Mechanical System	7
5.2.2.1.2.3.1	Heating	8
5.2.2.1.2.3.2	Ventilation	8
5.2.2.1.2.3.3	Air Conditioning	8
5.2.2.1.2.3.4	Miscellaneous HVAC Systems and Mechanical System	8
5.2.2.1.2.4	Fire Protection	7

5.2.2.1.2.4.1	Sprinklers	8
5.2.2.1.2.4.2	Standpipes	8
5.2.2.1.2.4.3	Fire Protection Specialties	8
5.2.2.1.2.4.4	Miscellaneous Fire Protection System	8
5.2.2.1.2.5	Electrical System	7
5.2.2.1.2.5.1	Electrical Service and Distribution	8
5.2.2.1.2.5.2	Lighting and Branch Wiring	8
5.2.2.1.2.5.3	Communications and Security	8
5.2.2.1.2.5.4	Miscellaneous Electrical System	8
5.2.2.1.2.6	Miscellaneous Equipment Utilities	7
5.2.2.2	Commissioning	5
5.2.2.3	Construction Materials	5
5.2.2.4	Miscellaneous Process Equipment	5
5.2.3	Support Equipment	4
5.2.3.1	Installation	5
5.2.3.2	Commissioning	5
5.2.3.3	Construction Materials	5
5.2.3.4	Miscellaneous Support Equipment	5
5.2.4	Miscellaneous Installation and Integration	4
5.2.4.1	Installation	5
5.2.4.2	Commissioning	5
5.2.4.3	Construction Materials	5
5.2.4.4	Other Miscellaneous Installation and Integration	5
5.4	**Miscellaneous Equipment**	**3**
5.4.1	Furniture, Fixtures & Office Equipment	4
5.4.2	Process Equipment	4
5.4.3	Support Equipment	4
5.4.4	Other Miscellaneous Equipment	4
6	**Transition to Operations**	**2**
6.1	**Management Self-Assessment (MSA)**	**3**

6.1.1	Site Work	4
6.1.1.1	MSA Planning	5
6.1.1.2	MSA Execution	5
6.1.1.3	Miscellaneous MSA Site Work	5
6.1.2	Facility Construction	4
6.1.2.1	MSA Planning	5
6.1.2.2	MSA Execution	5
6.1.2.3	Miscellaneous MSA Facility	5
6.1.3	Equipment	4
6.1.3.1	MSA Planning	5
6.1.3.2	MSA Execution	5
6.1.3.3	Miscellaneous MSA Equipment	5
6.1.4	Other Miscellaneous MSA	4
6.2	**Start-up**	**3**
6.2.1	Facility Construction	4
6.2.1.1	Facility Structure	5
6.2.1.2	Facility Utilities	5
6.2.2	Site Work	4
6.2.3	Equipment	4
6.2.4	Miscellaneous Start-up	4
6.3	**Miscellaneous Transition to Operations**	**3**
6.3.1	Facility Construction	4
6.3.1.1	Facility Structure	5
6.3.1.2	Facility Utilities	5
6.3.2	Site Work	4
6.3.3	Equipment	4
6.3.4	Other Miscellaneous Transition to Operations	4
7	**Decommissioning/Deactivation, Demolition, and Disposal**	**2**
7.1	**Facility**	**3**

7.1.1	Decommissioning/Deactivation	4
7.1.2	Demolition	4
7.1.3	Disposal	4
7.1.4	Miscellaneous Facility Decommissioning/Deactivation, Demolition, and Disposal	4
7.2	**Equipment**	**3**
7.2.1	Decommissioning/Deactivation	4
7.2.2	Demolition	4
7.2.3	Disposal	4
7.2.4	Miscellaneous Equipment Decommissioning/Deactivation, Demolition, and Disposal	4
7.3	**Miscellaneous Decommissioning, Deactivation, Demolition, and Disposal**	**3**
7.3.1	Decommissioning/Deactivation	4
7.3.2	Demolition	4
7.3.3	Disposal	4
7.3.4	Other Miscellaneous Decommissioning, Deactivation, Demolition, and Disposal	4
8	**Management Reserve and Contingency**	**2**
8.1	**Management Reserve and Contingency (M&O)**	**3**
8.2	**Contingency (DOE)**	**3**
9	**Escalation**	**2**
10	**General**	**2**
10.1	**Taxes, Permits, Insurance, and Bonds**	**3**
10.2	**Fees and Contingencies**	**3**
10.3	**Miscellaneous General**	**3**
11	**DOE PM Support - Federal Oversight Cost**	**2**