Draft Deliverable Report

Predictive Analytics to Improve Part Forecasting for Key Fleets

A Case Study for Armor Companies

Aisha Najera Chesler, Michael Chaykowsky and Ken Girardini

RAND Arroyo Center

Forces and Logistics Program

DRR-A962-1
November 2020
Prepared for the United States Army

NOT CLEARED FOR PUBLIC RELEASE

RAND
CORPORATION

(This page is intentionally left blank.)

Preface

This report documents research and analysis conducted as part of a project entitled: *Predictive Analytics to Improve Equipment Readiness* sponsored by Deputy Chief of Staff, G-4, U.S. Army. The purpose of the project was to develop analytical tools to identify and address the drivers of equipment readiness over time and across multiple dimensions and to use tools to improve Army-wide fleet management and better sustain equipment readiness in high stress environments.

This research was conducted within RAND Arroyo Center's Force and Logistics Program. RAND Arroyo Center, part of the RAND Corporation, is a federally funded research and development center (FFRDC) sponsored by the United States Army.

RAND operates under a "Federal-Wide Assurance" (FWA00003425) and complies with the *Code of Federal Regulations for the Protection of Human Subjects Under United States Law* (45 CFR 46), also known as "the Common Rule," as well as with the implementation guidance set forth in DoD Instruction 3216.02. As applicable, this compliance includes reviews and approvals by RAND's Institutional Review Board (the Human Subjects Protection Committee) and by the U.S. Army. The views of sources utilized in this study are solely their own and do not represent the official policy or position of DoD or the U.S. Government.

Executive Summary

In this document we explore the use of Artificial Intelligence (AI) algorithms in forecasting demand for armor companies (CO) of an armor Brigade Combat Team. Demand forecasting is the process by which historical data is used to estimate the expected future demand. The demand we are trying to predict for the Army are the parts that will be needed to repair equipment, which is particularly desirable to improve equipment readiness during high operational tempo in deployed operations.

We chose this type of CO because most of the maintenance jobs, and the associated parts, are opened on M1A2 tanks that each CO employs. Along with the maintenance history for each specific tank in the armor CO, AI allows us to leverage serial number specific data like age and usage. The AI algorithm can be trained on data specific to the tanks in each armor CO to make more precise predictions.

We used a graph deep learning (DL) model to generate part predictions. While the graph DL model was more precise than current methods used to set inventory levels, more analysis needs to be done before AI algorithms are ready to replace the current methods used to calculate inventory levels for armor CO. Currently, we are only predicting parts for the key fleets, but the armor COs must provide some parts support a for a wide variety of other equipment (e.g., machine guns, radios, etc.). Another challenge is the graph DL model doesn't enforce mobility, value or inventory churn restrictions.

Our recommendation is to integrate the AI algorithm into current process for computing inventory levels for specific types of COs and use in the following two scenarios. The first scenario conceptualizes the algorithm as an allocation tool and to help COs cope with Army operations and maintenance (OMA) funding shortages or national supply availability issues by predicting which COs are most likely to need a specific part. The second scenario is to improve equipment readiness by using the model to supplement the inventories of COs prior to deployment or major training events.

Table of Contents

(This page is intentionally left blank.)

Figures

(This page is intentionally left blank.)

Tables

(This page is intentionally left blank.)

Summary

In this document we explore the use of Artificial Intelligence (AI) algorithms in the problem of demand forecasting for armor companies (CO) of an Armor Brigade Combat Team (ABCT). Demand forecasting is the process by which historical data is used to estimate the expected future demand. The demand we are trying to predict for the Army are the parts that will be needed to repair equipment, which is particularly desirable to improve equipment readiness during high stress environments. We chose this type of CO because most of the maintenance jobs, and the associated parts, are opened on M1A2 tanks that each CO employs. A few potential advantages are listed below:

- AI algorithms learn from individual serial number's maintenance history and usage to make "personalized" parts predictions for individual equipment
- AI algorithms allow the user to specify operational tempo (OPTEMPO), equipment usage and location. This allows logisticians to plan for future scenarios (versus a purely historical analysis)
- AI algorithms learn to recognize specific patterns that are established over time based on unit training cycles and potential differences in equipment failures across theaters, climate, or terrain.
- Some AI models will infer dependencies between parts that are often ordered together.
- With the addition of features to the training data set, AI algorithms can leverage correlations that are not anticipated but that can be used for prediction.

Because the model learns from the data, the building of the data set to train the model plays an important role, hence we have tried to aggregate various sources of information, among them we have included, maintenance, age, usage, location and property book data. These are referred to as the features of the model and function as "independent variables." The features were hand-picked by subject matter experts (SMEs) and organized in a time-series manner (e.g., columns of the training data set).

We used a graph deep learning (DL) model to generate part predictions. While the graph DL model was more precise than current methods used to set inventory levels, more analysis is required before AI algorithms are ready to replace the current methods used to calculate inventory levels for armor CO. Currently, we are only predicting parts for the key fleets, but the armor CO must provide some parts support a for a wide variety of other equipment (e.g., machine guns, radios, etc.). Another challenge is the graph DL model doesn't enforce mobility, value or inventory churn restrictions.

Our recommendation is to integrate the AI algorithm into current process for computing inventory levels for specific types of COs and use in the following two scenarios. The first scenario conceptualizes the algorithm as an allocation tool and to help COs cope with Army operations and maintenance (OMA) funding shortages or national supply availability issues by

predicting which COs are most likely to need a specific part. The second scenario is to improve equipment readiness by using the model to supplement the inventories of COs prior to deployment or major training events

Acknowledgments

We would like to thank Ms. Justine Barr and Mr. Ken Beam from Logistics Enterprise Support Agency (LESA), Robert Thurston Director of Army G-46, and Mr. William Moore who was the deputy G-4 for most of the time this project was ongoing. At RAND we would like to thank Bryan Hallmark and Caolionn O’Connell for edits, feedback, and encouragement.

(This page is intentionally left blank.)

Abbreviations

ABCT	Armor Brigade Combat Team
AI	Artificial Intelligence
ASL	Authorized Stockage List
BCT	Brigade Combat Team
BDE	Brigade
BN	Battalion
CO	Company
CSS	Common Shop Stock
DL	Deep Learning
ERP	Enterprise resource planning
GCSS-A	Global Combat Support System-Army
MIP	Mixed Integer Program
ML	Machine Learning
NIIN	National Item Identification Number
NTC	National Training Center
OPTEMPO	Operational Tempo (equipment usage)
SPA	Supply Performance Analyzer
SSA	Supply Support Activities

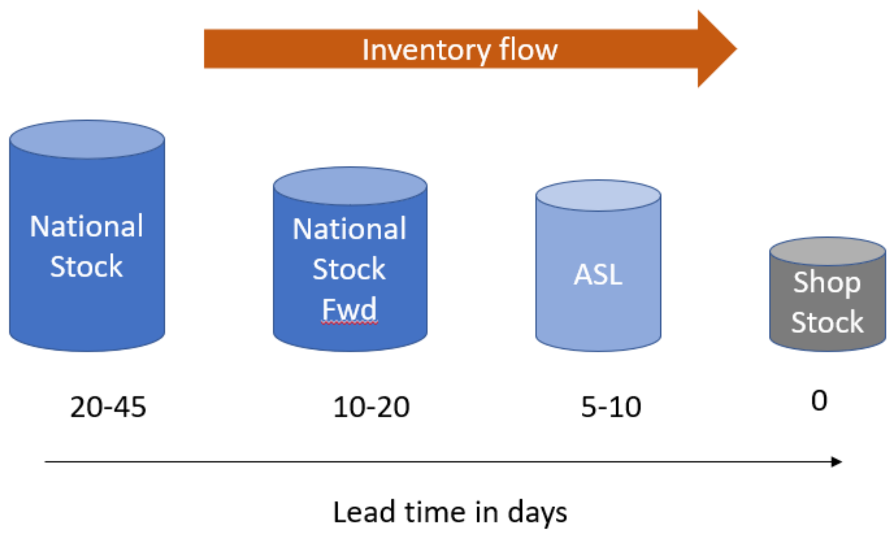
(This page is intentionally left blank.)

1. Introduction and background

Demand forecasting is the process by which historical data is used to estimate the expected future demand. Some consider forecasting to be an art (Chambers, Mullick, & Smith, 1971), as the method varies depending on many factors like availability of historical data, the time period to be forecast, desired accuracy, the cost, and benefit among others. It impacts various aspects of the supply chain management (Arkieva, n.d.), for example: optimization of inventory levels, improved planning and logistics, increase customer service levels, better capacity utilization and allocations of resources to name some. For over 20 years, as the Army has replaced or upgraded their supply and maintenance information systems, the Arroyo Center has collaborated with the Army to improve the supply chain with emphasis on setting inventory levels at different echelons of Army's supply chain (RAND Corporation , September 2020). This report documents a continuation of that work as the Army looks to leverage data from recently fielded Enterprise Resource Planning (ERP) software, called Global Combat Support System-Army (GCSS-A), that provides increased visibility of the supply, maintenance, and operational activities.

The Army's supply chain is divided into three main echelons: national stock, Authorized Stockage List (ASL), and shop stock. Figure 1.1 provides a sense for the inventory flow between echelons as well as the lead time ranges in days in a deployed environment to get parts from one echelon to the next. As we can see from the picture, having the required part at the shop level is an opportunity to save time and increase equipment readiness. Due to the increased number of stockage points and increasingly stringent storage and mobility constraints, at each echelon the list of parts stocked is typically a subset of the previous echelon. The discussion that follows will focus on ground Brigade Combat Teams (BCT).

Figure 1.1. Army's inventory flow for deployed operations



SOURCE: RAND Arroyo Center

Echelons of Army's supply chain

- **National**: At the national level, 86 percent of unique spare parts, consisting of most of the common consumables, are supplied by the Defense Logistics Agency (DLA). Repairable items are managed by the Army Material Command (AMC). Each organization has their own methods to set inventory levels. The inventories are typically held in large fixed distribution centers in the continental United States (CONUS). Both AMC and DLA set inventory levels to forward stock a subset of parts at forward distribution centers in some theaters.

- **ASL**: Each BCT includes a supply support activity (SSA) that operates a mobile warehouse of repair parts. The inventory levels in the SSA are referred to as its ASL. Different types of BCTs have different ASLs. Because the SSA must be mobile and the ASL must support a variety of equipment, the repair parts on the ASL are focused on repairing deadlined equipment in a deployed environment. When equipment fails and needs a part for repair, if the part is stocked in the ASL, then the equipment can be fixed without having to wait for the part to be shipped to theater (or from the theater rear area to the SSA if the part is stocked in a forward distribution center).

- **Shop**: At a lower organizational level BCTs are composed of battalions (BNs) and companies (COs). These units carry a small amount of inventory referred to as shop and bench stocks.[1] Shop stocks must be extremely mobile to move with the units in the field

[1] We will use the shorthand of shop stock for both as the differentiation between shop and bench stock can vary by unit.

(even more mobile than the SSA which is typically in the brigade rear area). Due to the limited mobility and storage the shop stock focuses on fast moving, small and less expensive items. When equipment fails, it is either fixed forward in the field or if it cannot be diagnosed and brought up quickly it will be evacuated for repair. Shop stock is most valuable for equipment readiness when the COs are deployed in the field away from the SSA. The recommended listings for shop stock are dominated by parts for the unit's mission essential equipment, as there is typically not sufficient storage space/mobility to support all the equipment in the unit.

Lead time demand is the key uncertainty[2] in the context of inventory decisions. In the Army context, the lead time demand we are trying to predict are the parts that will be needed to repair the equipment (particularly during high operational tempo for deployed operations) before replenishment can occur. There is uncertainty in the equipment failures and the parts required to repair the equipment and uncertainty with respect to replenishment times, particularly in deployed operations.

Machine learning (ML) and artificial neural networks have been able to predict demand for applications as varied as video games (Bersimas, Kallus, & Hussain, 2006) and spare parts of an aviation company (Şahin & Kızılaslan, 2013). In this document we explore the potential benefits of applying new Artificial Intelligence (AI) algorithms and tools to the problem of predicting future demands and setting inventory levels.

Potential benefits to using an AI algorithm

Below are some potential advantages to using an AI algorithm:

- An AI algorithm learns from individual serial number's maintenance history and usage to make "personalized" parts predictions for individual equipment.
- An AI algorithm allows the user to specify the anticipated rate of future equipment usage, also known as operational tempo (OPTEMPO), and when and where the equipment will be operating. [3] This allows logisticians to plan for future scenarios (versus a purely historical analysis).
- An AI algorithm learns to recognize specific patterns that are established over time based on unit training cycles and potential differences in equipment failures across theaters, climate, or terrain.
- Some AI models will infer dependencies between parts that are often ordered together.
- With the addition of features to the training data set, an AI algorithm can leverage correlations that are not anticipated but that can be used for prediction.

This report is focused on applying AI for parts demand forecasts and the setting of inventory levels for the shop stock in the lowest echelon of Figure 1.1. Specifically, this report will focus

[2] The uncertainty in lead time demand combines the uncertainty of the demand rate with the uncertainty of the replenishment lead time.

[3] In Chapter 3, we will discuss how logisticians can specify the future OPTEMPO and location as a means of planning for future deployments.

on setting the inventory levels for the shop stock in the armor COs of an ABCT. We selected the armor CO because most of the maintenance jobs, and the associated repair parts, are opened on the 14 M1A2 tanks that each CO employs. Hence, most parts in the shop stock for an armor CO are for the M1A2 tanks. The AI algorithm is trained on all the work orders and parts associated with M1A2 tanks in active component armor COs, providing a natural match for the shop stock needed in an armor CO.[4]

The document is organized in the following way. Chapter 2 gives an overview of the current way to set inventory levels for armor COs, and a brief discussion on some of the main differences between the current method and the AI algorithm. Chapter 3 provides details about the data used to build the features and the features themselves that the model trains on as well as an introduction to machine learning, particularly deep learning models, the specific type of model we developed, and a general background on AI. Chapter 4 discusses the performance of the new model in terms of accuracy and accommodation rates. Chapter 5 dives deeper into how the integration of both models is used to refine inventory recommendations. Lastly, Chapter 6 gives recommendations and future directions of work.

[4] We have identified several other applications including the M109A6 Paladin in ABCT fires batteries, M2A3 Bradley in ABCT mechanized infantry COs, and M113A3 in the headquarters CO of maneuver battalions and cavalry squadron.

2. Overview of the current inventory approach for an armor company

Because we will compare the performance of the AI algorithms described in this report to the common shop stock (CSS), in this section we provide some background on the CSS. We also provide some information on the data used in the CSS, as the same data is used to develop the training data sets used in the AI algorithms.

Common shop stock model

The Army is in the process of implementing common shop stocks (CSS). The CSS is computed using all the work order and part reservation across like units using a Mixed Integer Program (MIP) algorithm, leveraging data available since the fielding of GCSS-A increment 1 wave 2, which gives transactional data on all maintenance jobs opened and the associated part reservations. A MIP is an optimization algorithm where some variables are constricted to be integers. The output of the MIP is a list of parts, or National Item Identification Numbers (NIINs), and the associated safety stock levels, i.e. it prescribes what to stock and how much to stock. To compute the shop stock recommendations the MIP uses the maintenance jobs and associated part reservations across all like units, increasing the sample size of the work orders and associated part reservations, versus the historical practice of setting the shop stock using just the data for each unit individually. The MIP allows for several constraints: 1) the number of lines/storage locations, which can be broken out be storage category (e.g., bin, shelf, bulk); 2) the extended cube[5] (again, this can be by storage category), 3) extended dollar value, and 4) lower and upper bounds for each part to enforce individual part restrictions or storage location capacities. The constraints can also be broken by shop stock and bench stock, but for this analysis, we do not differentiate. The total dollar value, the total volume and the number of different parts to store are typically set by type of unit. The types and level of units that are authorized shop stocks are determined by policy, tactics and doctrine. Armor BCTs (ABCTs) typically carry shop stock at the CO level. In this work we will focus on the armor CO of the ABCTs. While each armor CO supports a large variety of equipment, the vast majority of the maintenance work orders and part reservations that are associated with the 14 M1A2 tanks.

[5] The extended cube or dollar value is calculated by multiplying the maximum number of each part expected to be on hand, referred to as the requirements objective (RO), times the unit cube or price of each part and summing across all parts in the storage category.

The type and number of each part to stock is determined by optimizing a readiness benefit function subject to the constraints listed above. Each part has a readiness benefit which is a convex piecewise linear function determined by:

- The number of x-day periods with demand quantity >0
- The value and variability of the lead time demand quantities for each period[6]
- Whether the part is a maintenance significant part (MSP) and the percentage of demands that are deadlining A equipment weighting factor applied by equipment type to place emphasis on parts support for mission essential equipment (e.g., in an ABCT the equipment weighting factor is sued to stress parts support for tanks over HMMWVs)
- The percent of units that had at least one demand that gets to how representative/robust the demands are expected to be (e.g., 10 BCTs with one demand each for a part would be favored over a part for which only one BCT received 10 demands).

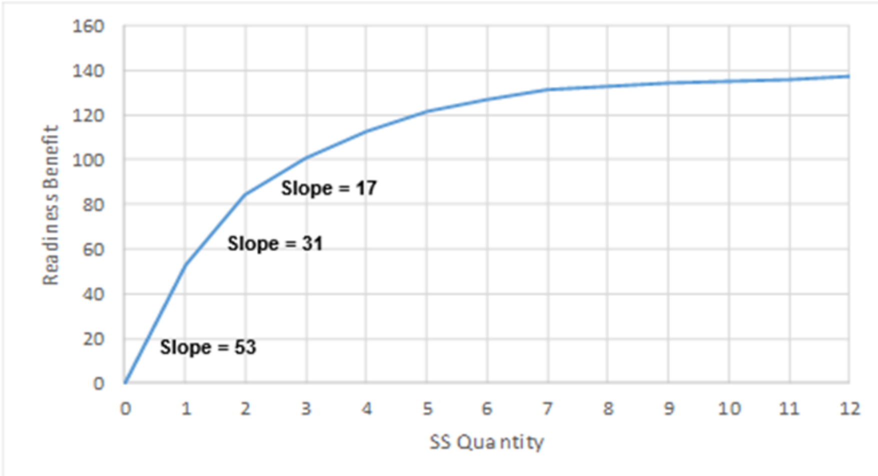
When the common shop stock is updated, churn factors[7] also weight the readiness benefit function of all parts to allow the user to rapidly analyze the tradeoff in performance versus cost/workload of moving to new inventory levels.

The readiness benefit function is derived from the lead time demand quantities for each part and weighted (which changes the slope) as described above. The result is a convex piecewise linear curve with decreasing slopes as safety stock (SS) is increased (see Figure 2.1). This implies that as more parts are added to the shelf (increasing the SS) the marginal readiness benefit for each additional part goes down.

[6] The positive led time demand quantities from each unit for each period are sorted into an array of decreasing values. For example, assume we have part with 6 positive lead time demand quantities given by the vector: [10, 8, 4, 1, 1, 1]. The initial SS =1 could then be issued 6 times to fill customer requests. However, adding a 2nd part to the shelf (SS=2) would fill only three part requests as the 2nd part would not be issued in the periods with a demand quantity of 1.

[7] The churn factors are just scalar weights that drive the solution back towards the current inventory levels.

Figure 2.1. Readiness benefit function by safety stock quantity



SOURCE: RAND Arroyo Center's analysis of GCSS-A data

Maintenance data

The data elements that drive the readiness benefit function maximized by the MIP are taken from maintenance and part reservation tables maintained in GCSS-A. The maintenance HEADER table is derived from three plant maintenance tables, AFIH, AFKO, and AUFK that are standard to the underlying data structure used in SAP (the ERP software of GCSS-A). [8]. All three of these tables contain data related to maintenance actions and provides one record for each work order (data element: *order_num*) across the Army. Using the maintenance HEADER records we join the RESB, which is the standard SAP data table documenting all the parts requested/reserved to complete the work order. There is one record for each part on a work order. This gives a link between the work order and the parts that were ordered. Since the HEADER table also contains information about the serial number, we can now link part reservations to the specific end items they were reserved for. That is, we can see when a tank went to the shop and what parts were reserved to complete the job. This visibility into the shop's activities allows us to have a complete picture of the maintenance history and associated part reservations at the serial number level. For the ABCT armor CO, the common shop stock MIP includes the maintenance history from all Active Component (AC) armor COs, approximately 60 of them. This maintenance history will typically include work orders for all the equipment on the armor COs property book that require maintenance (e.g., M1A2 tank, M113A3 personnel carrier, Light Medium Tactical Vehicle [LMTV], High Mobility Multipurpose Wheeled Vehicle [HMMWV],

[8] Order tables are standard SAP production tables for maintenance orders. Just as the RESB (reservation) table, they are common to SAP implementations, the ERP software of GCSS-A.

machine, guns, night vision goggles, etc.). The part reservations are aggregated according to the replenishment lead time (providing the lead time demand quantities described above).

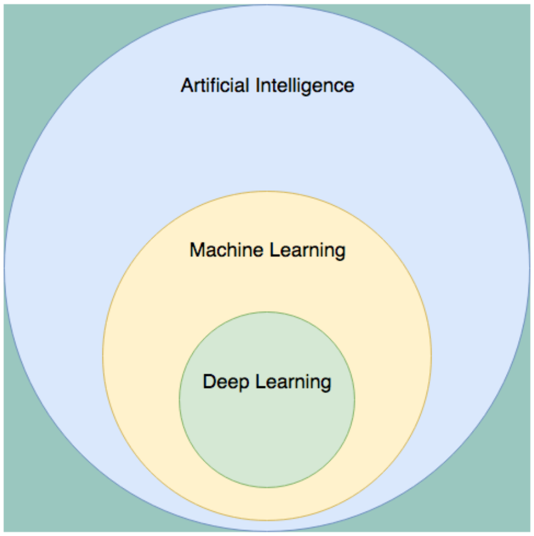
Approximately 75 percent of the part reservations in the maintenance history of armor COs are parts for the M1A2 tanks. This proportion is reflected in the CSS for the armor CO, as most of the parts on the recommended shop stock are to support M1A2 tanks.

3. Demand forecasting with AI algorithms

This section begins with a general introduction to AI algorithms, it will walk us through the details of the model developed to forecast the armor CO demands and will dive deep into the building of the datasets used by the models for the predictions.

The set of algorithms, that range from simple to complex, which solve problems in ways that are considered "smart" or intelligent are called Artificial Intelligence (Ceron, 2019). Within that set, there is a subset of algorithms that improve automatically though "experience", or without being explicitly programmed. The study of such subset is called Machine Learning (ML).[9] Moving through another layer of the Venn diagram of Figure 3.1 is Deep Learning (DL) models. DL models are inspired by the way the human brain works, by filtering information.

Figure 3.1. AI, ML and DL



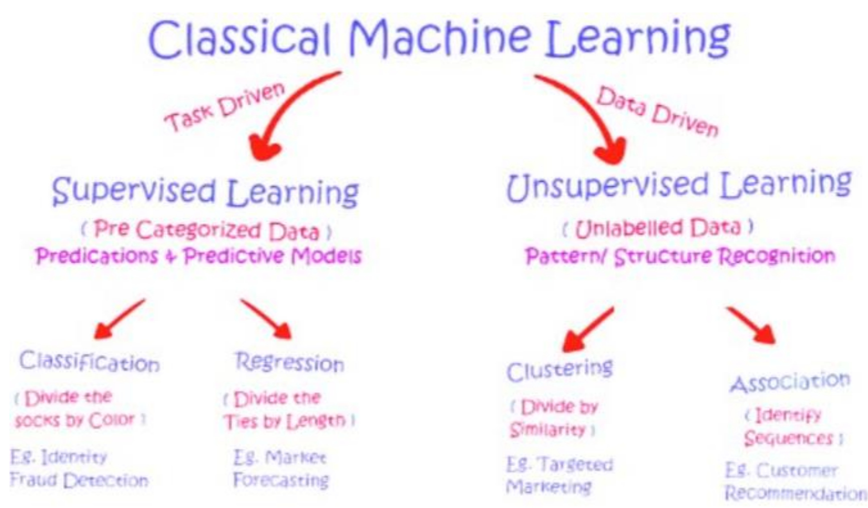
SOURCE: Medium.com Introduction to Deep Learning

[9] There are a couple of relevant differences between ML and DL models. One is that ML approaches do not capture some of the non-linearities as well as DL models and the other is that DL approaches extract information from large data sets that generalize better to unseen data

Deep learning models

Deep learning is a machine learning technique[10] that replicates on computers what humans do naturally, learn by example (Mathworks , n.d.). In this technique, the computer learns to classify tasks directly from images, text or sound. For example, DL is the key technology behind driverless cars. Driverless cars use labeled image databases for navigation and movement planning by detecting correspondence between images that the car “sees” with objects that it knows (Haydin, n.d.). Through repetitive training, the models can uncover hidden patterns and insights and recognize objects based on their inherent features. Models that are trained from data sets that have been classified or labeled are referred to as *supervised.* The model we developed is in the lower left hand corner of Figure 3.2 (supervised for classification) which provides an overview of the classical machine learning framework.

Figure 3.2. Supervised versus Unsupervised ML learning



SOURCE: Medium.com Supervised vs Unsupervised Machine Learning

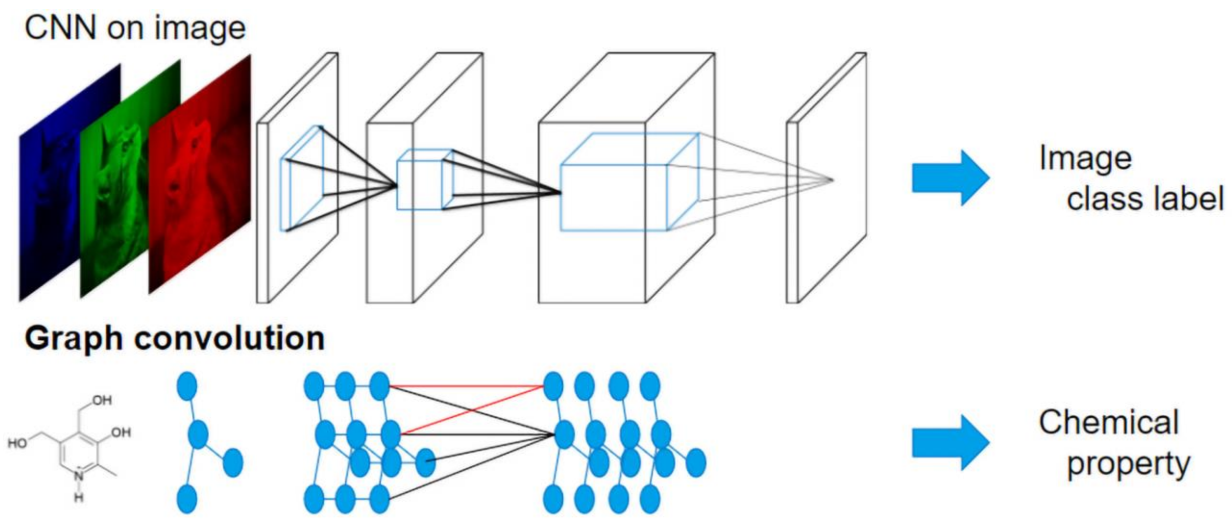
Graph Deep Learning

A burgeoning field within deep learning is performing predictive operations on graph data structures. A graph data structure is a finite set of nodes (person in a social network, atoms in a molecule, part in tank), and edges that connect them. The edges represent a relationship or an interaction between the nodes; hence they hold extra information about the structure of the data. For example, when two parts are ordered together in a maintenance job, a way to capture the

[10] More specifically supervised learning. With supervised learning you use labeled data, which is a data set that has been classified, to infer a learning algorithm. https://www.sciencedirect.com/topics/computer-science/supervised-learning

relationship is by representing each part as a node, and an edge joining them would indicate that they *go* together.

Figure 3.3. Graph convolutions



SOURCE: Towardsdatascience.com: Practical Graph Neural Networks for Molecular Machine Learning

Examples of the type of problems that a graph deep learning can solve are making predictions for the whole structure of the graph (edge predictions), like predicting the solubility of a molecule (Bronstein, 2020), or the classification of individual nodes, like identifying parts that will fail for a tank in a given month. In a similar way to deep learning models convolving[11] over data to extract information and make a prediction, a graph deep learning model convolves over nodes and edges in a graph, extracts its relevant features and makes predictions based on the topological structure, or state, and a feature matrix. In image processing, the convolution works as a filter that can sharpen, blur or enhance the edges of an image. Similarly, the general mechanism shared by graph deep learning methods is to aggregate its node representations, by aggregating features of neighboring nodes together. Figure 3.3 shows a pictorial representation of the convolutional network process on an image and the convolution on a graph, where we can observe the structure of the graph maintained through the layers.

Later, we will go into further detail on the model we built to personalize the recommendations for any armor CO, but for now it helps to situate it in the context of the definitions presented. We constructed a graph deep learning approach on structured data, and we will refer to it as *DL model*.

[11] Convolution is a linear operator that helps to abstract features from the data. This process is like mimicking the response of a neuron in the primary visual cortex to a specific stimulus.

Figure 3.4. Training data sample

UIC5	month	sloc_resb	pniin	n_fails	PRC0	age	parts_last_Q	PRC_last_Q	ric1	ric2	location	mnths_since_fail	prevorder	Difference_Distance_count	Difference_Distance_sum
WAD8A	1/31/2017 0:00	076N	99201	0	0.78	2217	0	0	WJS1			4	4	0	0
WAD8A	2/28/2017 0:00	0SQR	99201	0	0.78	2231	0	0				4	5	0	0
WAD8A	3/31/2017 0:00	18XB	99201	0	0.78	2245	0	0	A921		SWA	4	6	0	0
WAD8A	4/30/2017 0:00	18XB	99201	1	0.78	2259	0	0	A921		SWA	4	0	18	76.8798
WAD8A	5/31/2017 0:00	18XB	99201	0	0.78	2273	2	1.56	A921		SWA	4	1	478	12698.0623
WAD8A	6/30/2017 0:00	18XB	99201	0	0.78	2287	0	0	A921		SWA	4	2	0	0
WAD8A	7/31/2017 0:00	18XB	99201	0	0.78	2301	0	0	A921		SWA	4	3	200	3417.8796
WAD8A	8/31/2017 0:00	18XB	99201	1	0.78	2315	0	0	A921		SWA	4	0	65	2805.15015
WAD8A	9/30/2017 0:00	076N	99201	0	0.78	2329	2	1.56	WJS1			4	1	398	4121.2044
WAD8A	10/31/2017 0:00	076N	99201	0	0.78	2343	0	0	WJS1			4	2	38	561.6324
WAD8A	11/30/2017 0:00	076N	99201	0	0.78	2357	0	0	WJS1			4	3	0	0
WAD8A	12/31/2017 0:00	076N	99201	0	0.78	2371	0	0	WJS1			4	4	0	0

SOURCE: RAND Arroyo Center

Graph Deep Learning for Armor Companies

Previously, we mentioned that in the context of inventory decisions there are many unknowns, and during deployed operations, there are two key ones: 1) future demand and 2) replenishment times. It is important to keep this in mind, since both are the drivers to the integration of the CSS and DL models which we will discuss in the next chapter.
While it is possible to personalize inventory decisions at different levels of the supply chain (individual serial number, shop, ASL, or national), for different time periods (month, quarter), the data set that we use and present here is aggregated at the shop level. The training data set, which is used by the model to learn, is constructed by month, UIC5[12] (or CO level data) and part (or prime NIIN: pniin). We focus on this level of aggregation because we have found that the DL aggregated at the CO level is more accurate than aggregating 14 individual forecasts, one for each tank. The *features* of the model are how the data is organized to learn. They are the columns of the data set, see Figure 3.4 These features are referred to as the training data and, function as "independent variables". The features were hand-picked by subject matter experts (SMEs) and organized in a time-series manner. Furthermore, some of the other features, beyond the maintenance history are the same for all the tanks in the CO. The model solves a *classification problem* where the "dependent" or target variable is a binary outcome that indicates whether a part will be needed for a given month for a given CO. In other words, the model does not provide the levels for the parts, it forecasts whether a part will be needed for a given month for a given CO. The lists of parts provided by the model are also referred to as predictions. In contrast to the CSS, which gives the same parts for all the COs, the predictions are different for each CO because each CO learns from a unique history of maintenance, usage, age, etc. The DL model makes these predictions by learning from multiple sources of data on a single equipment type, the M1A2 tank, and it trains on those sources.

In the next section we will detail the data sources used and how they are joined to construct the training data set, but roughly speaking the model learns from maintenance, OPTEMPO, location, age and property book data.

[12] UIC5 are the first 5 characters of the Unit Identifier Code

Building the training and validation data set

Table 3.1 contains all the variable names and a quick description of each of them. These variables constitute the training data. In this section, we describe each of them, what data sources were used and how we aggregated them.

Table 3.1. Training data variables

Variable	Description
UIC5	A five-digit alphanumeric character used to identify a CO in GCSS-A
month	Month
age	Sum of the ages of all serial number at that CO
sloc_resb	Storage Location of the maintenance unit the reservation was made under
pniin	Prime NIIN
n_fails	A boolean variable that indicates whether the part failed. If it is equal to 1, it means the part failed, 0 means the part did not fail. This is the dependent variable we are going to predict.
PRC0	Price of the pniin
parts_last_Q	Number of given pniin that were ordered in the prior month
PRC_last_Q	total money spent on the given pniin during the prior month
ric1	routing identifier code for the SSA to which the request was sent to
ric2	if the CO changed rics midway through the month
location	Location of unit deployment. Options include: Europe, Korea or South West Asia or National Training Center (NTC). If the entry is empty, the unit was at home station
prevorder	Number of months since last time the part was ordered for this company
Difference_Distance_count	Number of dispatches that the CO accrued by month. That is the total number of times the tanks assigned to the company left the motor pool.
Difference_Ditance_sum	Distance in (km) that tanks assigned to the CO accrued by month
Ewma5	exponential weighted moving average of the target variable with alpha = 0.05
Ewma24	exponential weighted moving average of the target variable, n_fails, with span = 16

Maintenance data in the DL model

Similar to the MIP model used to compute the CSS discussed in the prior section, the AI algorithm uses maintenance data for all AC armor COs, but we subset the data by equipment type. For the armor COs, the AI algorithm utilizes only the data of the M1A2 tanks, since they dominate the maintenance work orders and part reservations, but also because some of the data used to train the AI is not available for certain items that require maintenance at the CO level, like machine guns, or night vision goggles. The way the data are treated and aggregated is also different, each of them resting on different yet complementary ideas. On one hand, the rationale behind the common shop stock list is that end-items that are of the same type behave similarly. In the context of inventories, this translates to the idea that if one part has failed a lot across the armor COs, we anticipate that it will likely fail again and for other tanks, and thus are the parts that get stocked. In a sense we use information from the population to predict individual behavior. By comparison, in the AI algorithm we assume that each end-item is an individual with different needs. For example, the individual maintenance history of a tank is what will ultimately determine its future fails, for example, an older tank that has seen a lot of usage will not have the same needs as a newer tank with lower usage.

Data for the DL model is aggregated at the *time frame-unit-pniin* level. Where the time frame is given at the month level and the *unit* is the CO.[13] This means that if we select *month-company-engine* the training data will have a row entry for the engine for each month that indicates whether a given CO (or UIC5) demanded an engine. The level of aggregation determines the level of fidelity of the predictions, in other words, the prediction will forecast if the CO will need an engine the next month or not. Remember that unlike the CSS model, the AI output doesn't provide levels, though it could, it just has not been implemented yet.

The set of parts, or pniins, that the DL uses was built from readiness drivers parts which account for 65 percent of deadlining reservations for the M1A2. Unlike the CSS, this list does not include the many constraints to enforce mobility of the shop stock or deal with inventory churn. In some sense this is on purpose, as we want to have the ability to add forecast parts needed for deployments even if they are not usually stocked.

Variables that are included as part of the maintenance data are:

- Fails, a boolean variable that indicates whether the part (pniin) failed
- Number of months since the pniin last failed (aggregated at CO)
- For each pniin, we include its unit price which is invariable across all COs and time
- SLOC, storage location of the maintenance unit the reservation was made under
- RIC1, the SSA that supports the CO
- RIC2[14], alternate SSA

We discussed that unlike the CSS model, the training data set for the DL included various sources of information which are used to build features. Those alternate sources will be described next, with each of the features used in the DL model and that are not used in the current CSS approach.

Property Book

The maintenance data is linked to the property book data via the serial number. This linkage is used to identify the CO that each tank is assigned to, as well as the SSA that supports the CO. This assignment is what allows us to do the grouping of the data to the UIC5 level. The variable derived from this data source is *UIC5*

[13] While it is possible to build the training data set at different levels, the focus of our analysis is done at the at the *month-company* level. Data can also be aggregated at the quarter level, and by serial number, battalion or brigades.

[14] Occasionally the SSA that supports the CO changes in the same month; hence we might see another RIC.

Location

The unit is linked to location data via the supporting SSA derived from the reservation. Location of the SSA is derived via the DoD automatic address file. Given that the activity of the armor CO typically varies depending on their training schedule or deployment status, we want to make sure that the training data set captures this feature. If the unit was deployed, it gives the location of deployment, Europe, Korea or South West Asia, when it is null, it means that the unit is at home station.[15] If the unit is during a training event at NTC it will indicate NTC. This variable acts like a Boolean variable for each location.

Age

The age data is linked to the maintenance data via the serials number and rolled to the CO level. The age of a serial number is estimated based on the first time the serial number shows up on 026 reports.[16] The model's feature derived from this data is sum of the serial number ages.

OPTEMPO

OPTEMPO[17] is given at the serial number level and joined to the maintenance data too. We implemented vehicle usage in two modalities. 1) The *real OPTEMPO* assumes that we know that a unit is going to be in a period of high activity, as in a training rotation or deployment. We provide the model with data that is representative of historical values for the number of dispatches and distance driven during such periods. In other words, real OPTEMPO allows us to simulate a deployment scenario because the model learns from these stress conditions. 2) The other modality uses a *lagged OPTEMPO*, in this scenario the model only sees what happened in the past but makes no assumptions about future deployment status. The analysis presented below is for real OPTEMPO unless stated otherwise. The two variables listed in the file are: 1) number of dispatches that the CO accrued by month (*difference distance count*) and 2) distance in (km) that the CO accrued by month (*difference distance sum*)

EWMA

The last feature included in the data set is the exponential weighted moving average (EWMA) of the fails over a period of 24 months, which implies that the smoothing constant is

---

[15] We can also specify the home station location (e.g., Ft Hood or Ft Stewart).

[16] 026 reports are maintenance reports from the Army's legacy maintenance management information system. RAND has that data source going online going back to 2010 and archived back to about 2006. From this data the age is estimated based on the first time there is a record for a given tank serial number.

[17] Previous RAND work (Shawn McKay, 2020) observed errors in the dispatch data coming from GCSS-A and they developed three methods to clean the identified data issues: business rules, machine learning, and usage proxies. The data used in this analysis has gone through the cleansing process.

0.08. The EWMA calculation is given by = (current predicted fail – previous month's EWMA) × 0.08 + current month's EWMA.

Figure 3.3 is a sample of the training data set. Each of the columns is one of the variables and the rows provide and instance in time (by month) of the level of aggregation, CO-pniin. In this snapshot we can see that the pniin 99201 failed in April and August for CO WAD8A.

Test and Validation sets

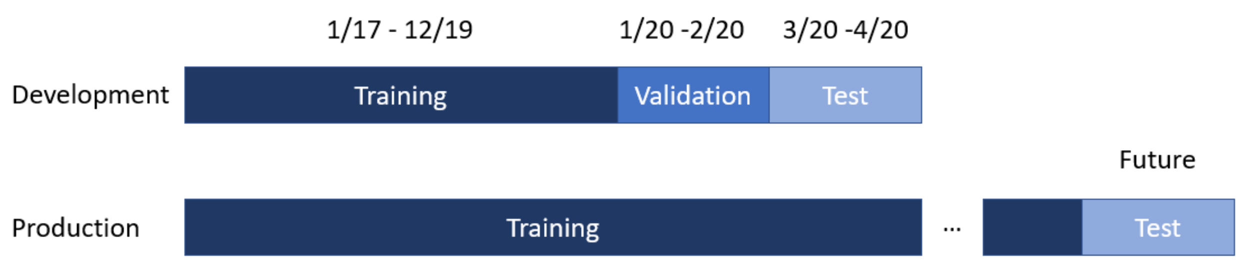
We used monthly data starting in January 2017 through December 2019 as our training data set. Validation dataset is a sample of data used to evaluate the model in an unbiased way (Shah, 2017). It is a part of the data that the model has not seen and hasn't been used for training. It helps to tune the model hyperparameters. We used January 2020 and February 2020 as our validation data set.

In this report, the test dataset is used to evaluate the performance of the predictions made by the model. The model does not "see" the test dataset which is used for evaluation only. For this report we used March 2020 and April 2020 as the test data set, as shown in Figure 3.4.

When choosing validation and test sets it is important to choose them so that they are representative of the future data you will see in production. Sometimes the sets are chosen as random subsets of the data, however, because we are building a time series forecasting model, we chose the periods for the training and validation data sets sequentially. The validation set is chosen carefully to reflect that of the test set in production. We use the validation set to drive the model's performance and this performance should generalize to production. In cases where we want to predict one period out from current, the validation set would be the most recent period with known data.

Once the model is in production there is no validation dataset as the model has been tuned properly during development, and the test dataset becomes the future time periods (in our case next two future months). In this production case, we need to estimate the values of the features for the future period set. For example, we need to be able to set or estimate the age of the tanks (just one or two months older), location of each armor CO, and the OPTEMPO (e.g., number of dispatches and miles).

Figure 3.5. Test and validation sets



SOURCE: RAND Arroyo Center

4. Deep Learning model performance

In the previous chapter, we provided a brief introduction on deep leaning models, how the training dataset was built, and the training, validation and test data sets were defined to build and evaluate the model. In this section we examine the performance of the DL model in terms of precision and accommodation rates. We will present different instances of the model and compare it to the performance of the CSS. Then we will discuss a case study that will lead into the next chapter, the integration of the models.

To compare the DL model against the CSS, we match the number of true positives of the two approaches, and we look at the following three metrics:

- Precision: how correct are the positive classifications.
    - True positives / (True positives + False positives)
- Recall: How complete is the positive classification coverage
    - True positives / (True positives + False negatives)
- False Positive rate: the probability that a false alarm will be raised
    - False positives / (False positives + True negatives)

The matching of the true positives between models is done by adjusting a "knob" in the DL model, the loss function value. We can think of this knob as a confidence threshold we have in our prediction that a failure will occur. If we choose a lower confidence threshold, we get more true positives (and increased accommodation rate) but at the expense of more false positives. So, lowering the confidence threshold reduces the precision of the model, as defined above.

As we can see in Table 4.1, CSS performs well but at the cost of a high false positive rate (units carry a lot of items that do not get a demand in two months). When both models perform at similar precision rates, the DL gives less false positive predictions than the CSS, i.e. it recommends fewer items that do not get demands. The DL model is well suited to identifying a subset of inventory for a unit to take if it needs to be self-sustaining and temporarily detached from the supply chain. In Table 4.1 we can see that, increasing the knob value (the threshold on our confidence in the predictions) increases the precision. Note that the table also includes other instances of the model, where the "knob" is at 0.5 and at 0.7. The reason we have included multiple thresholds for the model is because in the next chapter, when we refine the inventory recommendations, we will describe different uses for the model instances, but the idea to keep in mind is that with a higher value for knob we have more certainty in the predictions that will fail, and with a smaller value for the knob we have more certainty in the predictions that will not fail

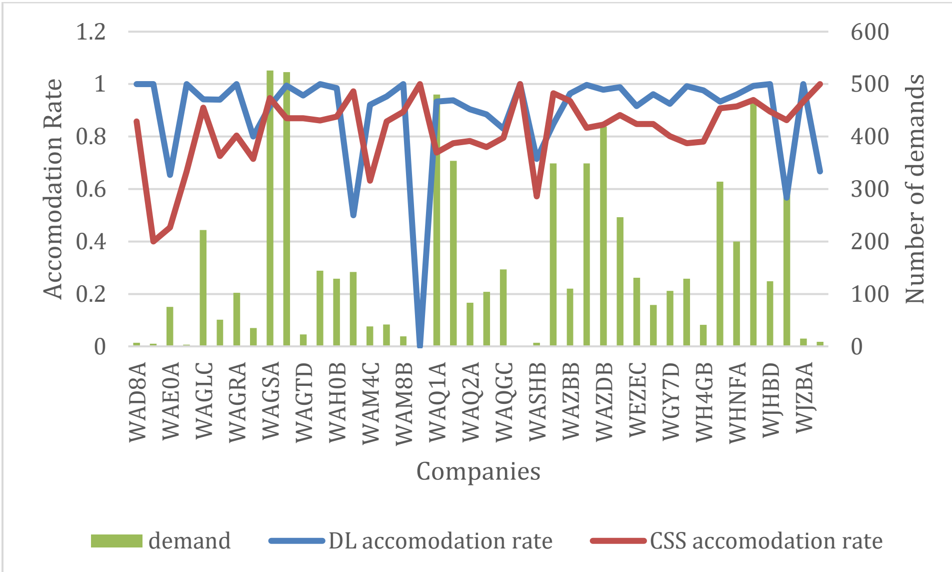
Table 4.1. Models' performance

	Accommodation	Precision	Recall	False Positive Rate (accuracy)
CSS	86%	11%	87%	72%
DL model (.25)	92%	10%	76%	39%
DL model (.5)	56%	14%	40%	15%
DL model (.70)	21%	15%	15%	5%

When we compare the accommodation of the shop stock against the DL model with 0.25 confidence threshold, we find that for most COs the DL model outperforms the CSS. In Figure 4.1, we can depict the variation in accommodation rates by CO. The red line is the accommodation rate for the CSS and the blue line is the rate for the DL model. There are a couple of COs for which the CSS does better, our conjecture is that since these are COs with little activity, the DL model predicts few parts, and the change in accommodation rate for being wrong on a smaller set of demands is higher. However, this higher accommodation rate can be achieved with the DL model comes at a price. In the 0.25 version, for the 53 companies for which there were predictions different than zero the DL model has a median value of 2.5 million and about 1.1 thousand cubic feet. To obtain the depth of the parts predicted by the model and calculate the value and cube of the solution, we used a variable safety level model on item unit price and unit cube.[18] If we compare this with the value $712,912 and 282 cubic feet for the CSS, we see that the CSS, which was developed with constraints in place, is less expensive and more mobile.

[18] The model calculates the level for each part based on characteristics of the NIIN required to satisfy given service level goals.

Figure 4.1. Accommodation rates by CO, DL (.25) vs CSS



SOURCE: RAND Arroyo Center's analysis of DL model's predictions and demand

Case study

To understand how this translates into actual predictions we will look at one case study, CO WHNFA. We chose this CO because it is the CO that ordered the greatest number of parts that were not in the CSS during March and April. This is a CO that also happens to be deployed in Europe during these months.

The purpose of this case study is to determine how well the DL model predicts parts that are not already in the CSS. To supplement the CSS we would focus on part forecasts where we had greater confidence (0.70 version of the model). In Table 4.2 we have provided a list of the parts that the 0.70 version of the DL model predicted would fail. It gives the part number, the nomenclature, how big and expensive each part is, the actual fails and the predicted to fail. The table shows that of the five parts the DL model predicted would fail that are not regularly stocked in the CSS, four parts failed during the months of March and April. The investment for the predictions adds to a total dollar value of $1,065,661.79 and 477.53 cubic feet. To obtain the depth of the parts predicted by the model and calculate the value and cube of the solution, we used a variable safety level model on item unit price and unit cube. Without the engine and the track shoe the investment is relatively minor and mobile.

Table 4.2. Case study predictions

pniin	NOMEN	Depth	cube	PRCss	Failed	Predicted
12014816	WHEEL, SOLID RUBBERTIR	10	22.393	4,452	1	1
14964092	RACK, M1 TANK	1	8.337	1,081.07	1	1
13904980	PERISCOPE, ARMORED VEH	3	5.001	12,420.7	1	1
14355175	TRACK SHOE ASSEMBLY	304	324.615	144,210	1	1
15482910	ENGINE	1	117.188	903,498	0	1

The main take away in this analysis is that while the DL model performs better than the CSS both in terms of accuracy and accommodation rate, it is currently focused on short term forecasts. The DL model is not yet ready to replace the current CSS, which provides a stable inventory that performs well and can be maintained over time. Also, currently the DL model lacks constraints on total inventory value and mobility. In the next chapter we will discuss how the DL model can be used to refine the CSS in specific scenarios.

5. Refining Inventory recommendations at the shop level

Similar to how current medical practices are determined by averaging responses across large cohorts to produce “standards of care”, current shop stock lists at the CO level are produced under the theory that equipment should get the same “care” or “maintenance”. The Army computes the shop stock levels with a method that uses work order and part demand data from all like COs.

The concept of making medical care more personalized has been gaining traction recently under the term, precision medicine. According National Library of Medicine, precision medicine is “an emerging approach for disease treatment and prevention that takes into account individual variability in genes, environment, and lifestyle for each person” (Medicine). If we apply this idea to the management of equipment maintenance, then we arrive to the goal of developing a data-driven “treatment” for individual equipment, or smaller groups of tanks that behave similarly, for example tanks in the same CO go through the same training or deployment schedule. Taking the notion to inventory theory, personalizing the shop stock means to tailor the parts recommended based on the specific characteristics of the equipment and future usage. More specifically, the DL approach learns from data like age, usage, environment, previous maintenance history, and anticipated future operation of the armor CO. While there are many areas of medicine where precision medicine is applied, its role in day-to-day healthcare is relatively limited. Likewise, our analysis in the previous chapter suggests that a purely DL driven algorithm is not ready to fully replace the current method used to set inventory at the shop level. There are several reasons for this. First, DL doesn’t have any restrictions on mobility and value, and while it outperforms the traditional approach in precision and accuracy, the recommended lists are often too expensive and not mobile.[19] Second, there are still data quality and signal to noise (sample size) issues to making predictions at the individual tank or CO level We recommend an approach that incorporates both algorithms via a heuristic.

Integration of the models

In cancer therapy, one of the most well-known examples of precision medicine is used *in combination* with chemotherapy to provide a significant benefit for women diagnosed with breast cancer (Butts, et al., 2013). Similarly, our approach is that of combining “treatments”. The intention of the amalgamation of approaches is to first satisfy the general needs of deployed

[19] These limitations could be alleviated by not including or disregarding forecasts for specific parts like very expensive repairable items (e.g., the tank engine) and large bulky items (e.g., tank track and road wheels – for which onboard spares are carried on each tank). However, additional development would be required to enforce the kind of detailed mobility and storage configuration constraints possible in the MIP used to compute the CSS.

operations for an armored CO and then add or remove materials to the inventory list based on two things: 1) how certain we are that a specific part will be needed for a given armor CO, and 2) what we know about what the CO will be doing, specifically, whether the armor CO will be at home station, on a National Training Center (NTC) rotation, on a rotational deployment or on a contingency deployment and the OPTEMPO indicative of those activities. We can use the DL model to refine/complement the existing inventory recommendations. The DL output provides a list of parts that it believes will fail and that list can be varied by adjusting the desired threshold level of confidence (loss value). Using this information and some level of accepted risk, the heuristic business rules can be designed to add parts that are not currently in the CSS that believes are likely to fail and removes, or does not replenish, parts that it thinks will not be needed. This approach will allow the DL model to play a larger role in setting inventory levels as we get more historical data, higher quality data, and a richer feature set.

This section will delve into the two scenarios: 1) using the tool to turn replenishment *off* for high dollar value parts on the CSS that and not expected to get demands with high probability and 2) to order high dollar value parts <u>not</u> in the CSS that are expected to get demands with high probability. In the first scenario, the DL model acts as an allocation tool and to help COs cope with OMA funding shortages. The unit could conserve obligation authority by not stocking or replenishing parts that the DL model predicts it will not need (even if that part is on the common shop stock). In the second scenario, the DL model is used to supplement and could be used prior to deployment, major training events, or when a CO may be detached from the supply chain. In this latter case, the DL model is being used to push parts to COs anticipated to face high OPTEMPO operations in anticipation of demand.

To explore the first scenario, we only focus on removing parts that are over $1,000 from the current CSS list. The parts that will be removed are those that the model predicts with high confidence that will not be needed. They correspond to the 0's (not failing) of the 0.25 DL model (0.25 loss function).

We start with the parts in the CSS and remove the 0's from the 0.25 DL model. After the removal of the parts,[20] we see that the overall accommodation rate only dropped by 1 percentage point, from 86 percent to 85 percent. If we were to blindly remove all the parts in the shop that are over a $ 1000 then the drop-in accommodation would go down to 78 percent. In other words, the model is pretty good at predicting which parts are not going to be needed. This translates into an overall saving of almost 13 million dollars across all 60 COs. Turning off replenishment in GCSS-A can be done by NIIN by setting a parameter called "auto replenishment off". Turning auto replenishment off by NIIN is particularly beneficial for those COs with more scarce resources as it allows them to more freely allocate their budget into parts that will be needed. Another opportunity is to review the list of parts that are not expected to get a demand and assess

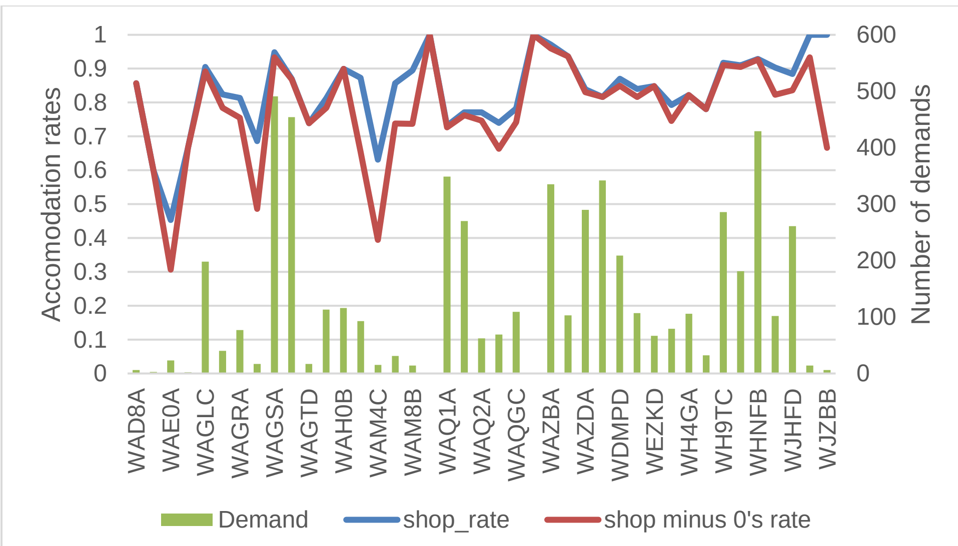
---

[20] Each company starts with a shop of 525 parts. The number of parts that each company would remove is variable, but on average each CO could remove around 21 NIINS, or a total of 1,269 parts among all COs.

if any of the parts are in short supply, regardless of unit price, as this could result in misallocation of scarce inventory. To illustrate the scenarios, imagine that you have a bike shop and that whenever there are less than 10 wheels, Amazon automatically ships new ones. If the DL model is saying that it does not expect a demand of wheels and the wheels are in short supply, turning automatic replenishment with Amazon off helps in two ways, the credit card[21] is no longer tied to the wheel, and hence can be used to buy another item, and when the suppliers do get the wheels, they are more likely to be shipped to the shops that need them.

An important thing to point out is that the 1 percent accommodation drop is not uniform overall COs. The units that have a high number of demands show higher accuracy than the units with low number of demands. This is because if the model predicts a false positive in a smaller sample of demands it weighs more heavily than in a larger sample of demand.

Figure 5.1. Accommodation rates by CO



SOURCE: RAND Arroyo Center's analysis of DL model's predictions and demand

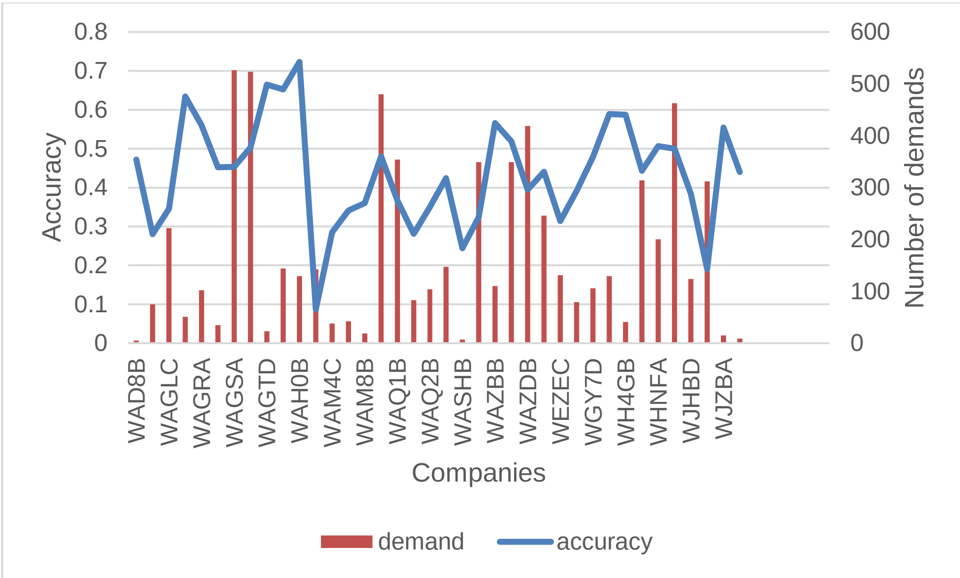
For the second scenario, we look at parts that are not on the CSS that are expected to get a demand. These parts correspond to the 1s (failing) of the 0.75 instance of the DL model not on the CSS. They are usually expensive or bulky parts. Recall that, in the case study of the previous section we see that the model predicted that the periscope and track shoes would have a demand during that period (Table 4.2), we then used the depth calculations which is 1 for the periscope, at $12,420.70, and 304 track shoes that are 325 cu ft total. . As we can see in Figure 5.2, the

---

[21] In this case, assume the credit card is "obligated" or available credit is reduced when the wheel is ordered. Whereas on Amazon the credit card is not billed until shipment occurs.

model performs well for some COs and not for others, where accuracy is defined as the probability that a false alarm will be raised, or the false positive rate.

Figure 5.2. Accuracy by CO



SOURCE: RAND Arroyo Center's analysis of DL model's predictions and demand

Hence, if we can reasonably predict the parts that will be needed to repair equipment (particularly during deployed or detached operations) then units should consider taking the recommended parts along.

6. Recommendations and Future directions

Our analysis shows that the DL model is best suited to be used in tandem with the MIP optimization algorithm that the CSS utilizes. The CSS provides the same list of parts and levels for all the armor CO, but we argue that there is room for improvement by personalizing the recommendations for any given armor CO. Personalizing the shop stock means to tailor the recommended parts at the CO level based on the specific characteristics of the equipment and future usage. This approach is similar to emerging medical treatments for disease and prevention that take into account individual variability in genes, environment, and lifestyle for each person to personalize treatments.

In Chapter 4, we demonstrated that personalized recommendations improve the performance of the COs and that the DL model is more accurate than the CSS. However, we see that the recommended "treatment" for individual COs sometimes falls outside of current shop restrictions policy. Either because the list is too expensive or is not mobile. This led us to develop two approaches for the integration of both methods: 1) using the DL model as a tool to turn replenishment *off* for high dollar value parts on the CSS that and not expected to get demands with high probability and 2) to order high dollar value parts not on the CSS that are expected to get demands with high probability.[22] The first scenario means we are conceptualizing the DL-model as an allocation tool and to help COs cope with OMA funding shortages. The second scenario uses the model to supplement COs prior to deployment, during major training events, or during high intensity deployments.

The DL model can be applied to the training data sets for other COs whose demand is dominated by mission critical equipment, for example the M2A3 in a mechanized infantry CO of an ABCT, M109A6/7 of an ABCT artillery battery, the M113A3 for the headquarters CO of maneuver BNs, and the Stryker for all shops held in the SBCT . Creating training data sets for different equipment types is relatively straight forward, as long as we have access to the necessary data to create features like their age and OPTEMPO.

The research team has also done analysis at the ASL and national levels. The advantage at these echelons is the larger sample size driving the forecasts. However, there are additional challenges as well. Unlike at the shop level in the examples given above, the ASL must support numerous types of mission critical equipment while enforcing tradeoffs between ASL performance versus mobility and inventory churn considerations. As a result, a hybrid approach (the Army uses a similar MIP-based approach to compute common ASLs across like BCT types)

[22] Ordering parts not currently stocked could be used to develop recommendations should the Army choose to integrate elements of push within its existing pull-based supply chain. The push system could be expanded in high intensity multi domain operations that severely hamper the information flows required in the pull-based system. See RAND PAF-1P-586 9/20 for more discussion of conditions requiring push versus pull logistics.

likely makes the most sense, complicated by the need to integrate the forecasts across several DL models. The challenge at the national level is the long lead times associated with replenishment, procurement or repair, which requires forecasts across longer horizons.

There are two areas that hold the promise of improving the DL models. The first is improved data quality. In the current training data sets, this will likely come from improved quality of the equipment usage data (McKay, Girardini, Broyles, & Kim, 2020) documents many of the problems with that data and improvements in the quality of the maintenance workorder and part reservation data. The second is expanding the features in the training data sets. The features in the data sets could leverage the Army's efforts to develop a data ecosystem that gathers information from different domains (e.g. logistics, training, installations, financial). We know, for example, that even when BCTs support similar equipment types, the demand can vary from mission to mission, brigade to brigade, or different time periods. Hence, we anticipate that the DL model would learn and extract hidden correlations that can be used for the predictions. Similarly, at the national level the DL model predictions would likely benefit from features that give a better sense of the state of unit training budgets and training calendars. We hypothesize that the DL model would recognize patterns and infer dependencies.

"Personalization" of inventory decisions at different levels of the supply chain (individual serial number, shop, ASL, or national), for different time periods (month, quarter), is possible, and DL models learn to recognize patters in the data. Building the data structures to train the models at the different levels of Army's supply chain provides an opportunity, with a lot of potential, to advance Army analytics and use data as a strategic asset to enable agile and resilient logistics.

Appendix A: Model

With the plethora of tabular data in business intelligence and supply chain analysis it makes sense that much of the analysis is performed on tabular style data sets. However, with recent progress at the intersection of graph network analysis and deep learning there is an opening for the application of these methods to supply chain recommendations. A graph network, G, is made up of two parts, an adjacency matrix, A, and a feature matrix, X.

$$G(X, A)$$

The application of using graph networks to model tabular data is clear when we break down the graph into these component parts. For example, say we want to build a recommendation system that suggests parts for tanks before they fail. The traditional tabular method would be to pass the machine learning algorithm the feature matrix X directly and predict the target column y based on a learned latent feature space determined by the columns of X. With the introduction of graph networks, we can now construct an adjacency matrix A where the neural network model can learn representations of the network's nodes based on the feature matrix *and* the topology of the graph network.

With a table of data X where each row vector x_i in the data represents a tank – month – part, each column vector x_j represents a feature in our model (i.e. price of part, miles driven, etc.), and the independent variable y that can be 0 or 1 based on whether or not that part was ordered by that tank in that month. A simple graphical representation of this data set would be to create a graph where nodes represent possible parts that can be ordered and the graph represents a given tank – month combination (i.e. tank "A1" and month "April 2020"). On each node (part) we have a class of 0 or 1 whether or not tank A1 required that part in April 2020. Finally, we simply create as many graphs as tank – month combinations in our data set, so graph G_k represents the k-th tank – month combination. This data structure decision creates an additional benefit for the algorithm if there exists a topological framework that the model can learn more efficiently. Now we have to make a decision on how to create the edges in the graph. We can think of this model as a recommendation system. If parts are frequently ordered together it would make sense for the model to think of these parts as 'tied together' by an edge. So, for our example undirected weighted graph G_k we will create an edge between two nodes if those parts have been ordered together and optionally add an edge weight equal to the number of times these two parts have been ordered together. Graph G_k has n x m feature matrix X and n x n adjacency matrix A.

We can formalize our problem as predicting the probability that node i is in class c using a graph neural network as a 2-layer graph convolutional network (GCN) that learns representations of nodes to use in the node classification problem by aggregating feature vectors within the neighborhood of that node in the graph. The final node representation embedding is a d-dimensional vector for each node forming a n x d matrix Z,

$$Z = f(X, A) = softmax(\hat{A}\ ReLU(\hat{A}XW^0)W^1)$$

where $\hat{A}$ is the normalized adjacency matrix and weights $W^0 \in \mathbb{R}^{Fxd}$ an input-to-hidden weight matrix and $W^0 \in \mathbb{R}^{dxC}$ a hidden-to-output weight matrix[23]. The loss function for the classification problem is evaluated as the cross-entropy error with $C = 2$ classes and Y_{train} the labeled example nodes in the dataset.

$$L = -\sum_{l \in Y_{train}} \sum_{c=1}^{C} Y_{lc} \ln{(Z_{lc})}$$

Weights of the neural network W^0 and W^1 are trained with gradient descent.

Implementation relies heavily on how the graph data is structured from the raw tabular data. In the example above we describe one of the simplest versions of representing this data set using graph networks. However, one can imagine a more dynamic graph setting where there is a single changing graph that can add or remove nodes based on orders at each time step t. There are various tradeoffs for different graph deep learning implementations and these considerations must be weighed when developing your training data set. For our implementation we empirically saw an improvement in performance when using static graphs over dynamic graphs, but this can vary depending on the question asked of the model.

[23] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. https://arxiv.org/pdf/1609.02907.pdf, 2016.

(This page is intentionally left blank.)

References

Arkieva. (n.d.). *Arkieva* . Retrieved from Why is Demand Forecasting important for effective Supply Chain Management?: https://blog.arkieva.com/demand-forecasting-for-supply-chain-management/

Bersimas, D., Kallus, N., & Hussain, A. (2006). Inventory Management In the Era of Big Data . *Production and Operations Management* .

Butts, C., Kamel–Reid, S., Batist, G., Chia, S., Blanke, C., Moore, M., . . . Ashbury, F. (2013). Benefits, issues and recommendations for peronalized medicine oncology in Canada . *Current Oncology*.

Ceron, R. (2019, December 9). *IBM.com*. Retrieved from AI, machine learning and deep learning: What's the difference?: https://www.ibm.com/blogs/systems/ai-machine-learning-and-deep-learning-whats-the-difference/

Chambers, J. C., Mullick, S. K., & Smith, D. D. (1971). How to Choose the Right Forecasting Technique. *Harvard Business Review*.

Haydin, V. (n.d.). *Intelligent Sorftware Engineering*. Retrieved from How Machine Learning Algorithms Make Self-Driving Cars a Reality: https://www.intellias.com

Mathworks . (n.d.). *Mathworks*. Retrieved from What Is Deep Learning? 3 things you need to know : https://www.mathworks.com/discovery/deep-learning.html

McKay, S., Girardini, K., Broyles, J., & Kim, D. (2020). *Improving Ground Vehicle Usage Data.* RAND Arroyo Center.

Medicine, N. L. (n.d.). *What is precision medicine?* Retrieved from https://ghr.nlm.nih.gov/primer/precisionmedicine/definition

RAND Corporation . (September 2020). *Strengthening the Army's Data Science Capabilities: A Brief History* . RAND CP-A561 .

Şahin, M., & Kızılaslan, R. D. (2013). Forecasting Aviation Spare Parts Demand Using Croston Based Methods and Artificial Neural Networks . *Journal of Economic and Social Research*, 1-21 .

Shah, T. (2017, December 6). *Towards Data Science* . Retrieved from About Train, Validation and Test Sets in Machine Learning: https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7