

Improving After Action Review (AAR) Applications of Natural Language Processing and Machine Learning

Kim Cates and Marc Banghart

KBR Incorporated

Alexander Plant

Abstract

After action reviews (AARs) are used within the military and organizations to assess events and their corresponding training outcomes. These team discussions provide a learning-focused method to assess performance and analyze failures or possible improvements to future events. Useful information is frequently embedded within these AARs in the form of unstructured text and speech. This article proposes a solution to analyze and trend AARs digitally. We discuss solutions to capture data using hand-held devices. Such devices allow for audio ingested into a data pipeline where speech-to-text processing occurs. Audio processing operates by identifying primitive language components such as phonemes paired with contextual modeling of their relationships to identify the most likely textual output. We then discuss the conversion of the speech to text and the application of Natural Language Processing (NLP) to enable analytics. NLP techniques uncover semantic patterns in unstructured text which then are correlated with team performance measures. Such trends allow for optimization of military training courses through revealing success-promoting factors between AAR and team performance.

After action reviews (AAR) have been the foundation of the U.S. Army training life cycle for decades. These reviews function as collaborative posttraining meetings to allow the team to engage in self-learning and self-correction

(Morrison & Meliza, 1999). The meeting, typically led by a unit leader or facilitator, focuses on asking the group (a) what was planned during the training, (b) what occurred, (c) why the events unfolded the way they did, and (d) what should be modified before the next training. These retrospective sessions provide an opportunity for soldiers to evaluate previous performance. Analysis of multiple AARs over time may further result in identification of training deficiencies and team improvement opportunities. Aggregated results and analysis of AARs within a specific organization may further yield organizational insights. In order to enable analysis of multiple AARs over a longer time frame, both data collection and analysis must be addressed.

Capturing AAR data digitally poses several challenges. AARs are performed in a highly verbal manner. AAR processes utilize open-ended questions, involve the entire team, and may be formal or informal. The AAR discussion and outcome may be manually documented once complete. Although there are standards that should be followed during an AAR, inconsistencies will undoubtedly exist between facilitators and across units. The verbal nature and varying structure of AARs together make them difficult to document.

Electronic data directly captured during the AAR (using voice-to-text technologies) can be analyzed utilizing algorithms such as Natural Language Processing (NLP). For example, an algorithm could automatically identify that a shortage of training equipment could be of concern based on certain phrases and words utilized across multiple AARs. Repeating words or phrases can be visualized in electronic dashboards to provide insight into AARs and underlying training successes or failures.

NLP emerged out of the 1950s from the intersection between linguistics and artificial intelligence (AI). It is utilized to extract information from text sources such as documents (Nadkarni et al., 2011). Searching the internet for content is an everyday example of NLP. NLP broadly works by creating a mathematical representation of text, which can then be analyzed. However, to the end user, these mathematical details are hidden. There are multiple techniques within NLP to include sentiment analysis, topic modeling, text classification, and text clustering. The most common NLP approach is classifying text.

Text classification occurs through the calculation of word frequencies in a text field. These word frequencies can be used for word combinations linked to each class label to be captured. For instance, a model could classify a written review as positive if the review consists of words such as “good,” “very,” “happy,” “liked,” “again,” or “enjoyed.” In contrast, reviews containing words such as “poor,” “never,” “boring,” “unsatisfied,” “little,” or “not” would be classified as negative. More advanced NLP techniques can analyze the inferred context and meaning of words by utilizing mathematical vectors. These vectors are calculated by learning each word’s conditional probability of occurrence given all other words in a text field, thus quantifying each word’s context. Textual clustering, or the groupings among relevant semantic words or phrases, can be captured through grouping textual vectors by proximal distances from other clusters. Additionally, an embedded vector can be compared against another for similarity through applying cosine distance to both vectors.



Although the Pentagon has recently invested \$2 billion into AI capabilities, there have been few documented applications of NLP within the Department of Defense (Millman, 2018). In the early 2000s, the military's main advancement in NLP was shown through a voice interactive device project which focused on voice-to-text translation. The primary purpose was to free up soldiers' hands while accessing or storing data in a computer database for such tasks as vehicle troubleshooting or paperless documentation of diagnostic information (Rodger et al., 2001). In May 2012, the Defense Advanced Research Projects Agency launched the Deep Exploration and Filtering of Text (DEFT) program to enable defense analysts to discover implicit patterns in language (Onyshkevych, 2012). More recently, the Department of Defense introduced the Joint Artificial Intelligence Center to standardize AI practices, tools, data sharing, and technology across the military. In the Joint Artificial Intelligence Center, the Operations Center Cognitive Assistant project intends to increase accessibility and detection of troops' urgent calls using NLP approaches to efficiently label verbal communications by the degree of urgency (Freedberg, 2019).

This article will demonstrate the value of NLP approaches when applied to course surveys and narratives. Multiple techniques will be utilized to include topic mod-

Kim Cates completed an MS in data analytics from Seton Hall University in 2019. She has an extensive background in research, machine learning, statistical modeling, and neuroscience. While working as a data scientist at KBR Incorporated, Cates has focused extensively on capturing trends from unstructured text using natural language processing (NLP) centered on building predictive maintenance models across various aircraft platforms. Kim intends to apply her expertise in NLP to build optimized workflow systems for defense in logistics applications.

Marc Banghart, PhD, has over twenty years of experience within the defense and intelligence sector and is a chief engineer for KBR Incorporated. He earned his PhD in industrial engineering in 2017 and holds graduate degrees in both computer engineering and systems engineering. In his current technical role, he applies artificial intelligence, machine learning, and cognitive capabilities to complex systems within the defense sector. He has served on various boards to include the Institute of Electrical and Electronics Engineers Accelerated Stress and Reliability Conference and is a past chair of the American Society for Quality Reliability and Risk Division. He currently serves on a PhD committee at Mississippi State University researching the application of model-based systems engineering to complex systems.

Alexander Plant is a diversified technical problem solver with experience in software engineering, cloud engineering, Agile practices, and distributed systems architecture. He has worked on projects large and small ranging from training delivery and data engineering for bespoke defense platforms to financial transaction systems and payroll integrations for Fortune 500s. He is passionate about open-source software and loves finding simple, composable solutions to complicated problems.



eling. Topic modeling will provide insight into the trends of word distributions as they vary from each topic. Topics can be used for identifying word-groups affiliated with different courses and outcomes. Furthermore, effective modeling of course success through the application of machine-learning models and neural networks to quantified text values will be summarized. Lastly, this article will include a proposed infrastructure for AAR data storage and management using scalable, cloud-native solutions, and discuss how these relationships will inform efficient training design.

Methods

Data Collection

The analysis utilized a dataset available from Coursera to demonstrate the capabilities of NLP. The Coursera data set was selected due to COVID-19 restrictions that prevented data capture as originally planned. The data set included course review text, the rating of course, and the type of course. The course review was an unstructured text field while the course rating was on an ascending Likert scale of 1 through 5. With minor modifications, the techniques applied in this article can easily apply to an AAR data set.

The course review's text was preprocessed before applying any text analysis. Unnecessary characters were removed along with any meaningless words. Irrelevant characters involve any single characters or special characters such as "!", "a," "\$," or "-." Removed words are most often characterized by words in prepositional phrases such as "to," "the," "in," and "from." Additionally, words were transformed to their base word or "stem" through a process called stemming. For instance, the words "repairing" and "repaired" would be reduced to "repair." In effect, noise is filtered from the model by removing redundant words while power is increased by adding to words' semantic value.

Data Engineering

Storage. The use of cloud-native storage and accessible computational resources can provide a cost-efficient and convenient method of storing unstructured AAR data. Recent developments have simplified the creation of a big data solution for the aggregation and processing of large sums of data. Decades ago, it often required excess hardware like mainframe computers and massive parallel storage. Even with innovative frameworks like Hadoop, maintenance of a fleet of commodifying hardware and specialized configurations was still needed. The advent of web-based object stores and managed analytics offerings through vendors like Azure and Amazon Web Services (AWS) have had a pivotal impact on big data efforts.



The cornerstone of such an approach is an HTTP-based object store such as AWS Simple Storage Service (S3) or Azure Blob storage. Such services provide a highly durable and available storage facility for files of varying magnitudes. These providers bill on a discrete storage and transfer basis as opposed to paying for possible usage as in typical capital expenditure scenarios, which reduces administrative and financial burden. Since files are accessible over the internet via a typical HTTPS connection, these services enjoy wide support across many development platforms. In this scenario, AWS S3 would function to store individual AARs in an access-controlled bucket for later analysis. The AAR would be recorded by a bespoke application and transmitted to AWS S3 by the laptop or tablet-based field device in an asynchronous manner based upon the availability of a WAN connection.

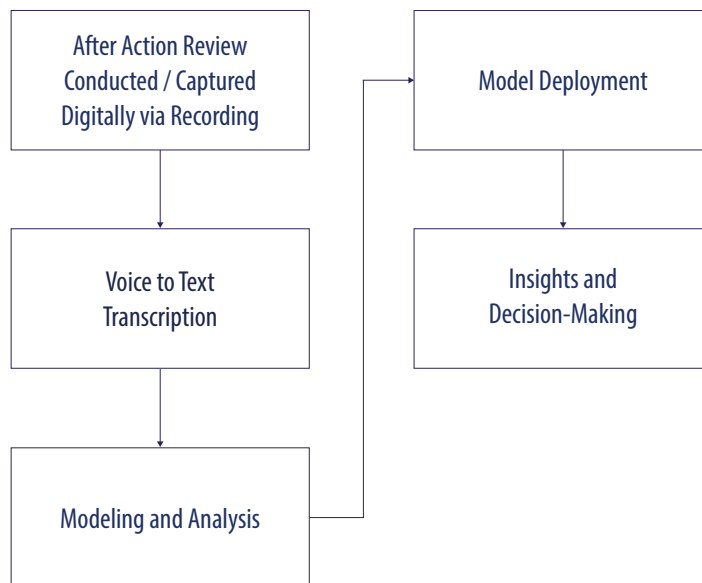
Analysis. Once an AAR recording is in an object store format, a cloud provider can pair its extract, transform, and load (ETL) data integration process with streaming transformation tools to cleanse the data. Services such as AWS Glue and Azure Data Factory permit the creation of automatic data extraction, cleaning, and movement jobs. Other preanalytic tasks are integrated into the ETL process. Speech-to-text, for instance, can be accomplished using AWS Transcribe in place of commercial offerings that might not meet Department of Defense-mandated security standards. This opens the possibility of easing the data capture process by allowing voice recording capability instead of cumbersome typing on touchscreen keyboards. Tags can also be applied based on the source and content to further categorize and enrich the data. AAR data would notionally be converted to text, tagged appropriately, and then stored in an adjacent AWS S3 bucket for processing.

A cloud provider's managed analytics service offerings can subsequently be used to perform a wide range of analyses ranging from typical statistical methods to machine learning. Data scientists typically develop NLP solutions in interactive Python environments known as notebooks using machine learning frameworks such as Keras, Scikit-Learn, and TensorFlow. AWS SageMaker and Azure Machine Learning provide managed notebook packages for the development of these solutions. Our typical machine learning workflow for productionizing these types of solutions involves the use of SageMaker for not only development but also compilation and deployment. SageMaker's additional components like Neo and Hosting Services facilitate this by providing cross-compilation and managed API endpoints for machine learning models respectively. These deployment capabilities enable not only programmatic access over a network-accessible API but also model deployment to edge devices like low-power servers and mobile devices. SageMaker is ideal in that it would allow for AAR data to be read into its analytics platform. Within this platform, NLP approaches are developed for uncovering semantic patterns in text. Once robust NLP models are established, SageMaker provides automated analytics jobs to be run and stored into their respective data warehouses. NLP results may then be read into a business intelligence (BI) tool to generate visualization. For example, data



Figure 1

The Notional After Action Review (AAR) Process



could be read from AWS S3 bucket into SageMaker where the NLP preprocessing will be executed. The NLP output would be stored back into AWS S3, which would then populate a prebuilt AAR dashboard within Kibana. Within this infrastructure, the dashboard would be updated automatically as the AAR data is stored.

Consumption. Techniques used to consume processed data and ML models can vary dramatically depending upon user needs and operational agility. Models' exposure via remotely deployable units and RESTful APIs means that a variety of solutions can consume the models and data, not the least of which are the usual crop of BI applications like Tableau and Power BI. Custom web and mobile apps can be developed to leverage more advanced charting, integration, and formatting capabilities afforded by full-featured development environments. For this application, a simple web application would be developed to query AWS Lambda APIs exposed by AWS API Gateway to facilitate search and sieve capabilities on transcribed AAR data and the actionable insights they yield. Periodic reporting and stakeholder dashboards would be developed using Tableau linked to these same APIs (or directly to the object store in cases where additional capabilities are required). So long as access to the AAR artifacts is highly available and governed, the sky is the limit—augmented reality, geographic information systems, and other exotic applications all integrate with this notional system as depicted in Figure 1.



NLP Approaches

Modeling Course Satisfaction

Course satisfaction was modeled using both machine learning models and a convolutional neural network (CNN). Both approaches transform the processed text into numerical values but differ in their transformation operations. A dictionary, also referred to as a corpus, is created before applying these operations to the text field. A corpus lists all the keywords to be analyzed in the text field of interest. These keywords are usually determined by some minimum word count threshold. For this specific application, the words had to occur at least a hundred times to be considered part of the corpus. Words not meeting this threshold are removed from the text field to reduce variance. The models are trained using these text values as an input and course rating as the output. During the training process, the models' weights are optimized to generate predictions consistent with the actual course rating. The model iteratively aims to reduce the error between predicted course rating and actual course rating, or loss, through backward propagation.

Before training the machine learning models, the text is transformed into numerical values through term-frequency inverse-document frequency (TF-IDF). TF-IDF captures the term frequency (TF), or the total times a term occurs in a document. Inverse-document frequency (IDF) resembles how many documents consist of the same term. The TF-IDF value emerges through the product of TF and IDF. This value increases by the number of times a word occurs in a document and is stabilized by the number of documents that contain that word. TF-IDF allows not only for word frequencies to be captured but also controls for the words that commonly occur.

The outcome of the TF-IDF transformation is a matrix in which each column represents a word in the corpus and the rows correspond to individual surveys. The values within each cell are the word frequencies, or the product of TF and IDF, for each word. The machine learning models are then trained on this word frequency matrix as the input with course rating as the output. Logistic Regression, Naïve Bayes Classifier, Support Vector Machine Classifier with a Linear Kernel, and Random Forest Classifier were trained on the word frequency matrix extracted from the review's text field.

In contrast, the neural network model maps each word in the corpus to a vector of continuous numbers through an embedding layer. The embedding layer functions to identify similarities among discrete variables, or in this case, words. Before words are fed into the embedding layer, each word is transformed into a unique integer, or a "token," that will function as its index in the embedding layer. The word vectors are generated by a superficial densely connected layer. The embedding layers work to iteratively generate the conditional probabilities that other words will occur given the presence of the input word. Therefore, the output word vector is representative of the condi-



tional probability a word occurs given all the other words in the corpus. Because of this, embedding layers are often used to visualize semantic similarities among words. Consequently, calculating the cosine distance between two-word vectors functions as a measure of similarity between two words. The t-Distributed Stochastic Neighbor Embedding (t-SNE) technique was applied for reducing the words vector from 50 to two. The words, or two-dimensional vectors, can be visualized as a scatterplot in which semantically similar words are illustrated by their proximity to each other. Each word vector in the review string forms the input matrix to the neural network. All word vectors are zero-padded to 50 to make all input matrices the same size. Following this, these matrices become the input to the convolutional layer. Within the convolutional layer, several nodes function as a “filter.” These filters are 3x3 matrices that convolve the matrix. The dot product of the convolving filter and the input word matrix form the output of the convolutional layer. A max-pooling layer and drop-out layer are subsequently applied to control for increased bias or overfitting. The output of these layers is then fed into a densely connected layer where the prediction output is generated.

The ordinal variable, the course rating, was preprocessed using a dummy coding approach. In this approach, each rating was binary encoded resulting in five columns where a “1” was used to represent the ranking. The columns would then be “0” where the ranking was not present. The output layer consisted of a densely connected layer with an output shape of five to represent the five-point rating scale. A sigmoid activation function was used to reduce the predicted probability to either a 0 or 1. The binary cross-entropy loss function was applied to this output to effectively update model parameters from these predictions.

A scatterplot represents the reduced two-dimensional output of the word vectors from the embedding layer. The word vectors are reduced from 50 dimensions to two dimensions (x and y) using the t-SNE technique. The outcome scatterplot conveys how the embedding layer in the neural network captures semantic similarity among words. For instance, words such as “research” and “study” are close to one another just as “suggest” and “recommend” are overlapping another.

Results

Machine Learning Models

Overall, the machine-learning models demonstrated above-average predictability of course satisfaction. That is, the models averaged 75% accuracy in predicting course satisfaction. The Logistic Regression model performed superior at 77% accuracy while the Random Forest Classifier exhibited the lowest predictability at 73%. When looking more closely at each rating’s accuracy in the Logistic Regression mod-



el, it is apparent that middle rankings did not perform as well. The more evident rankings such as 1 or 5 yielded more robust results and is most likely due to definitive co-occurring words associated with a poor rating or a great rating. In contrast, ratings 2 through 4 do not consist of distinguished language that is unique to their rank. When it comes to AARs, it is important that the language captured consists of enough variance for machine-learning models to effectively capture differences in training performance. This is contingent on the scale of training performance along with the data size captured.

Convolutional Neural Network

The CNN performed overall better than the machine-learning models with an average prediction accuracy of 95%. This increase in predictability of the CNN compared to other models is most likely due to the differences in processing text. Contrary to traditional machine-learning models, the embedding layer captures semantic similarity among words while the TF-IDF matrix only allows for patterns in co-occurring words to be modeled. In a more defined sense, the CNN can detect a range of similar words occurring together due to their similarity and therefore, considers synonyms to have the same impact on the model's output course rating. In contrast, the TF-IDF does not reflect the interdependence of synonyms. However, the CNN class-by-class results are consistent with machine-learning models' outcomes revealing that the course ranking of 5 performing superior to other rankings.

Linear Discriminant Analysis Interpretation

Topic modeling through Linear Discriminant Analysis (LDA) allows for underlying contexts in which language may be used in the unstructured text analyzed. While machine-learning models aim to extract patterns in the text linked to preexisting groupings of course rating, LDA discovers naturally occurring groupings. These hidden contexts can then provide more insight into model performance. By referring to topics generated as shown in Figure 2, five general themes emerge from the reviews. Topic #0 represents overall extremely positive reviews of the course. Topic #1 consists of extremely positive reviews that are associated with a machine-learning course. Topic #2 captures positive reviews that are linked to an introductory Python course. Topic #3 does the same but focuses on an overall data course. Lastly, Topic #4 contains good reviews, but the good reviews are less positive than those captured in Topic #0. Overall, these core groupings of words extracted from the LDA algorithm shed light on the key text patterns in the course reviews. Considering the inconsistent accuracies from the models' predictions per course rating, it is not surprising



Figure 2

Core Topic Groupings of Words Extracted through LDA Algorithm

Topic #0:

course learn lot thank good love learn lot help amaze awesome really teach good course nice way new excellent course learn informative fun things wonderful love course experience teacher want course help excellent course amaze course understand

Topic #1:

course great great course learn machine machine learn best course great specialization project look helpful best course forward introductory look forward thank andrew start introductory course ng courser cover complete far algorithms knowledge content work ml

Topic #2:

good course easy really like understand introduction time program make great follow python start learn basic bite easy understand good course think assignments little use way context know feel lecture course good explain

Topic #3:

useful recommend course data class highly great recommend program assignments clear science dr use recommend course practical lecture learn understand chuck concepts dr chuck information tool design excellent python intro overview challenge

Topic #4:

course thank excellent enjoy learn make really material lecture videos content information like work provide understand think read time quiz excellent course life present help way professor students use question study

that trends are revealed to be related to course content and positivity rather than course rating. Moreover, the top rating of five occupied most of the course ratings taking up 74% of the data with a sample size of 79,173. In other words, the unbalanced output most likely caused lower accuracies in the less represented rating levels along with inherent topics captured by the LDA algorithm.

Future Directions


NLP approaches demonstrate modeling feasibility of Army training performance through textual analysis of AARs. Devices capable of AAR speech capture to be processed for subsequent NLP analysis could provide a capability to improve training outcomes. Overall data architecture and approach as described in this paper can be adapted for military environments and tailored for integration with processes such as the Army Lessons Learned program.

Both the machine-learning model and the CNN model revealed predictability of course rating by analysis of course reviews. This same process can be applied to AAR transcribed data as the input and training performance measures as the output. A separate embedding layer could be created for each of the text fields in



terms of (a) what was planned, (b) what went wrong, and (c) how improvements could be made. Doing so will increase variance to the model input and capture distinct patterns linked to differences in training performance. These AARs can be associated with training requirements via systems such as the Army Training Information System that contains metadata surrounding training requirements and skill decay rates.

The predicted output, the course rating, was extremely imbalanced and the review itself consisted of a limited narrative due to limited input data. Therefore, the models all performed relatively poorly on less distinguishable classes. In addition, the models only used one text field as the input. The discussion nature of AARs allows for multiple text inputs to be entered into the model. Application of similar models on AARs would benefit Army training practices allowing the detection of key elements in positive or negative training sessions to be identified. That is, the underlying patterns in the language used when training is successful or goes poorly can be identified. Furthermore, characterizations of AARs linked to training performance may be used to track downward or upward trends in improvements or lack thereof from AARs over time. Extracted lessons from past training can serve as critical guidelines for future improvement. Using this paradigm which stores, aggregates, and analyzes training data, Army leaders can better forecast and understand historical training dynamics, lessons learned, and future planning.

The use of deep learning to yield actionable AAR insight opens the door to myriad possibilities about the iterative process of improvement. Automation tooling and industry-standard methodologies permit the approach outlined in this article to be adapted to a variety of problem domains. Logistics, personnel management, and medicine are examples of other fields of interest with large amounts of free-text records that can be analyzed via novel ML techniques. Many commercial off-the-shelf platforms in use by the Department of Defense's service branches also incorporate such functionality. As the use of ML to analyze free-text data proliferates the industry, applications tailored specifically to military needs will be critical. This work, exploring the specific application of ML to the Army AAR process, can help inform future Army efforts to develop innovative, specialized machine-learning applications to better serve the warfighter. 

References

- Freedberg, S. J. (2019, November 13). *Exclusive Pentagon's A1 problem is 'dirty' data: Lt. Gen. Shanahan*. Breaking Defense. <https://breakingdefense.com/2019/11/exclusive-pentagons-ai-problem-is-dirty-data-lt-gen-shanahan/>
- Millman, R. (2018, September 10). *US military to spend \$2 billion on developing artificial intelligence*. Internet of Business. <https://internetofbusiness.com/us-military-to-spend-2-billion-on-developing-artificial-intelligence/>



- Morrison, J. E., & Meliza, L. L. (1999). *Foundations of the after action review process: Special report 42*. United States Army Research Institute for the Behavioral and Social Sciences. <https://apps.dtic.mil/sti/pdfs/ADA368651.pdf>
- Nadkarni, P. M., Ohno-Machado, L., & Chapman, W. W. (2011). Natural language processing: An introduction. *Journal of the American Medical Informatics Association*, 18(5), 544–551. <https://doi.org/10.1136/amiajnl-2011-000464>
- Onyshkevych, Boyan. (2012). *Deep exploration and filtering of text (DEFT) (Archived)*. Defense Advanced Research Projects Agency. <https://www.darpa.mil/program/deep-exploration-and-filtering-of-text>
- Rodger, J., Pendharkar, P., Paper, D., & Trank, T. (2001). Military applications of natural language processing and software. *AMCIS 2001 Proceedings*, 233. <https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1672&context=amcis2001>

