2022

Expeditionary Domain Awareness - Intelligence Support to NECC & NECC Support to Intelligence Analysis (NECC focus)

Das, Arijit

Monterey, California: Naval Postgraduate School

https://hdl.handle.net/10945/71925

NAVAL RESEARCH PROGRAM
NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

EXPEDITIONARY DOMAIN AWARENESS - INTELLIGENCE SUPPORT TO NECC & NECC SUPPORT TO INTELLIGENCE ANALYSIS (NECC FOCUS)

EXECUTIVE SUMMARY

Principal Investigator (PI): Mr. Arijit Das, Computer Science (CS)

Additional Researcher(s): Dr. Neil Rowe, CS; Mr. Peter Ateshian, CS; and Mr. Tony Kendall, Information Science

Student Participation: Ms. Aroshi Ghosh, CIV INT, CS

Prepared for:
Topic Sponsor Lead Organization: N2/N6 - Information Warfare
Topic Sponsor Organization(s): Office of Naval Intelligence (ONI)
Topic Sponsor Name(s): Mr. Robert Inscore, Asst. Director for Programs and Budget/ONI-N5
Topic Sponsor Contact Information: Robert.inscore@navy.mil/301-669-3301

This research is supported by funding from the
Naval Postgraduate School, Naval Research Program (PE 0605853N/2098).

Project Summary

The Navy Expeditionary Combat Command (NECC) community gathers information and intelligence documents that accumulate over time on file stores. The intelligence consumers needed a method to search all prior knowledge documents, preferably based on common language keywords and phrases. This challenge could be solved by working with an existing vendor product with the associated licensing, support, and maintenance. The Naval Postgraduate School (NPS) team took a computer science (CS) approach to identify the various workings of a document store/search portal (system). An evaluation of each technology step involved was conducted, and potential solutions and their associated costs were considered. The team found that given user specifications, a tech-savvy team, and combined with open-source and Department of Defense (DOD)–licensed software, one can build and maintain a system that meets the requirements of the Department of Navy (DON) community.

Keywords: *Naval Expeditionary Combat Command, NECC, expeditionary domain awareness, intelligence, operations, collaboration, portal, information stream, Naval Expeditionary Combat Forces, tribes, database, Hadoop, artificial intelligence, machine learning*

Background

The initial understanding was that the NPS team would be given raw datasets to evaluate and analyze. After several meetings with NECC, it became clear that data is preprocessed and summarized in the form of reports. Reports can be in any format, namely Adobe Acrobat PDF, Microsoft Word and PowerPoint, images and plain text. These reports are distributed to the community via email/file systems and need to be searched later. There is no centralized system to store, analyze, and generate analytics (based on the documents) for the community.

One option was to evaluate vendor content management systems (CMS) applications, but instead the NPS team looked at the challenge with a computer science mindset. The team had prior background with processing large datasets and extracting analytics using the Hadoop Distributed File System (HDFS) and a relational database. Common algorithms/code focus on plaintext; since the NPS team was familiar with processing binary files and extracting needed information in plaintext, this could be applied to the non-plaintext documents. Data growth is an important consideration that should be handled with technology seamlessly. For this the NPS team used its background in Big Data technologies with HDFS.

After documents were loaded into the system, algorithms had to be researched that could extract the keywords in plaintext and create metrics. These metrics will help in intelligent results when the user community searches historical information with keywords and phrases. The NPS team used its background in document classification to evaluate algorithms that worked.

For any such system to be viable, the user community needs a friendly user interface. There is also the challenge of multiple devices like a laptop, desktop, handheld devices, and phones. The NPS team looked at openly available technologies like HTML5, JavaScript, open-source webserver, and Python programming libraries. The frontend (browser on laptop/phone) needs to send data over the internet to a webserver (middleware) that is subsequently sent to the database (Oracle) backend. For all of this to work, the three parts need to be compatible.

Overall, technologies need to be available via DOD licensing and be cost effective. The plan was not to recommend any esoteric or custom software that might be a financial challenge and face lack of developer community support. Instead of total reliance on vendor consulting teams, these technologies must be supported by DON in-house technology teams with training and minimal vendor support.

Findings and Conclusions

The NPS team started with building a sandbox to evaluate the possible technologies. For the document store that can scale up, it considered the Oracle vendor database product. Using the Oracle XE laptop version, the team loaded documents and wrote SQL to query them. In the sandbox, NPS used the laptop version with the assumption that a production Oracle database will work with the same codebase as the laptop version.

In the studied system, keywords need to be extracted from the documents. While it is easy to achieve this with plaintext, with binary format documents, the solution is to use optical character recognition (OCR) technology. The first step is to convert the documents to image format and then use the OCR application to extract the keywords. Extracted keywords need to be cleaned of punctuation marks and stop-words (words used for grammatical sense) and lemmatized (variations of a word need to be made one). All final extracted words are stored along with the original documents, thus the database handles binary and plaintext datatypes.

For each document loaded, the keywords are used to create a matrix using term frequency-inverse document frequency (TF-IDF) vector algorithms. To calculate distance metrics, the cosine similarity algorithms are run on the matrix. Distance metrics are critical when the end users search for documents using keywords and phrases, as they will help generate a list of documents that are closest to the search string.

The team evaluated the frontend on a laptop using a browser. The middleware is from Flask (open source application server), which is a Python programming language product. Flask lets one build the full software application in Python, so all the algorithms are Python packages that can be deployed to the Flask webserver, and use the database as a store. When the frontend was deployed on an Android phone, it uses the Java programming language while Flask uses Python. A workaround is to use a Java to Python connector Jython, which lets Java applications to use Python libraries/code. This is an extra layer of software that can increase execution time and will degrade speed of execution with data growth.

The system studied by the team requires that each time a document is loaded, all the calculations have to be redone; this can be a challenge when the number of documents start to grow. The sandbox did not fully test data growth using a HDFS system.

Initial reports were sent to the topic sponsor, and their results were encouraging. The study has raised awareness of the problem, and the full report will be sent next. More research needs to be done before this idea can be implemented into production.

Recommendations for Further Research

The Naval Postgraduate School team studied the document store/search system on a laptop sandbox, so a next step will be to be scale up the evaluation. A server-based system can be used with a Hadoop Distributed File System backend to understand the challenges of large-volume execution. A repository of datasets in the terabyte range will be a more realistic test of the system. The middleware technology needs to work on all platforms—if it works on Python and not on Java means a more generic middleware architecture needs to be researched and evaluated. Frontend technologies need to be looked at on a wide range of devices with large user community involvement. The middleware architecture needs to handle user growth for loading/searching of documents, thus more options beyond Flask need to be evaluated.

Additionally, there are many Department of Defense (DOD) content management system vendors who can be reached to present their solutions and evaluated. More studies need to be done with other DOD entities that may have already solved this problem.

Acronyms

CMS	content management system
DOD	Department of Defense
DON	Department of Navy
HDFS	Hadoop Distributed File System
NECC	Naval Expeditionary Combat Command
NRP	Naval Research Program
NPS	Naval Postgraduate School
OCR	optical character recognition