Winter 2018

DOD HPC INSIGHTS

A publication of the Department of Defense High Performance Computing Modernization Program

Workforce Development:

HPCMP's Collaborative Relationship with DoD Military Academies

HPCMP "Hero" Awards

Contents

Cover Photo:
United States Naval Academy commencement ceremony, Navy-Marine Stadium, Annapolis, Maryland. DoD photo by Mass Communication Specialist 1st Class Chad J. McNeeley (Released)

Inside Cover Photo:
F-16 and the US Air Force Academy Chapel, Colorado US Air Force photo: Mike Kaplan (Released)

First word

Ms. Sandy Landsberg
Acting Director,
High Performance Computing
Modernization Program

Welcome back to HPC Insights. There have been some new and exciting changes at the High Performance Computing Modernization Program Office since our last publication. Our previous Director, Dr. David Horner, was selected to be the Director of the Engineer Research and Development Center (ERDC) Information Technology Laboratory. I am pleased to share with you that Dr. Will McMahon is the new HPCMP Director, effective 21 January 2019. I have enjoyed the opportunity to serve as the HPCMP's Acting Director since May 2018, and will return to my former position as Deputy Director.

This issue highlights the HPCMP's efforts in the area of Workforce Development. Our cover story highlights work from young talent at the US Military Academies and their use of high performance computing from detecting cyber threats, to US security via social media, to design and analysis of flight vehicles via computational aerodynamics—some innovative approaches to combat uniquely contemporary problems. Through our collaborations with the US Military Academies and universities across the nation, our goal is to actively educate the next generation of scientists and engineers in high performance computing before they even enter the workforce.

Several new HPC systems have been installed and are now operating at the DoD Supercomputing Resource Centers (DSRCs). In this this issue, you'll find an illustrated guide detailing current HPC systems and their specifications (cores, memory, speed, etc.). For our users, we have an informative article on the use of modules, and an enhanced version called "bcmodule", to increase user productivity. Also of great importance to our user community, there is an article announcing the roll-out of the next-generation Secret Defense Research and Engineering Network (SDREN), a classified wide-area network that supports the HPCMP's classified supercomputing centers and users at the Secret-level.

I congratulate our 2017 and 2018 Hero Award Winners who have been recognized for a variety of professional accomplishments such as long-term performance, innovation, and technical excellence. We welcome and congratulate our up-and-coming award winners, may they be a part of our community for years to come; and we wish our lifetime achievement award winners good luck in their future endeavors.

I hope that the articles included in this publication will provide the reader a small glimpse into the wide-ranging efforts of the High Performance Computing Modernization Program as it continues to support and directly impact the Department of Defense and our warfighters.

Workforce Development:

DoD Military Academies Use HPC

Over the years, the HPCMP has enjoyed a collaborative working relationship with the DoD Military Academies. The HPCMP provides resources in an effort to increase the interest in high performance computing, help educate both faculty and students about ways HPC can improve research and engineering efforts, and to prepare future Military leaders to excel in a technologically advanced environment.

The following four articles are the result of research conducted during Honors Projects at the United States Naval Academy and the United States Military Academy at West Point, and document the use of HPC to tackle cyber warfare threats via social media. Their work shows that the Military Academies are preparing their students to understand the technological complexities of modern warfare in a cyber world and providing them with the knowledge and resources needed to defend against attack

Using Machine Learning to Predict the Popularity of Reddit Comments

Sean Deaton,
Scott Hutchison,
Suzanne J. Matthews
United States Military Academy, West Point

Introduction
The perceived popularity of electronic media profoundly impacts user activity on-line. Popularity can imply that content is vetted by other users and worthy of consumption. Many companies devote a vast amount of resources to predict the future popularity of on-line material, especially those on social media platforms. Reddit is a popular social media website where users share news, pictures, video, and other types of media. While the Pew Research Center estimates that only 4% of Americans use Reddit[1], the site's self-aggregated statistics indicate that there were 234 million unique visitors in December 2015, and approximately 8 billion unique page views[2]. Reddit attracts an audience that is distributed between male and female (53% vs. 47%), and enjoys an approximate parity of US versus international users (54% vs. 46%)[2].

In addition to sharing media, users who create accounts on the website can subscribe to communities known as "subreddits", which enable them to keep track of, and interact with, content of personal interest. Each subreddit has its own community page. Users vote to move particular submissions (or posts) to the top of a subreddit. Posts with a high number of positive votes ("up-votes") rise to the top of a community's front page, where they have a high chance of being viewed by other visitors to the subreddit. Negative votes ("down-votes") reduce visibility. Users can interact with a post by making comments and replying to each other.

Increasingly, visitors use Reddit as a news source. In a recent survey, approximately 70% of surveyed Reddit users indicated they use the site as their primary news source[1]. For the 2016 Presidential campaign, 45% of Reddit users used the site to track the election. This is on-par with other social media platforms such as Facebook and Twitter, which were estimated at 52% and 43% respectively[1]. Politicians and celebrities have taken notice; a number of them have done Q&As (known as IAMAs) with Reddit users on the website, notably Presidents Barack Obama and Donald Trump, and former presidential candidate Bernie Sanders[3].

However, the site does have a history of inspiring vigilantism, with victims frequently having their personal information posted on the Internet (a practice known as doxing). One notable example includes Sunil Tripathi, a missing student who was erroneously marked as a leading suspect in the Boston Marathon bombings before the true culprits were identified. Reddit later issued an apology in which they condemned the "witch-hunts and dangerous speculation"[4].

Reddit has made efforts to curb the spread of misinformation. In November 2016, Reddit banned the PizzaGate5 subreddit from the site, stating that "we don't want witch-hunts on our site"[5]. This resulted in a backlash from users on `r/theDonald` subreddit, who stated the move amounted to censorship. Reddit changed the way its front-page algorithm works after users of the `r/theDonald` were allegedly exploiting the algorithm to spam the front page with pro-Trump messaging. Reddit's CEO is hesitant to ban the subreddit due to the controversy it could cause. However, posts stickied (i.e., highlighted) from `r/theDonald` are currently banned from Reddit's front page[6].

During the course of this project, we seek to determine if the popularity of comments on particular Reddit posts can be predicted using popular machine learning techniques. In the context of Reddit, comments that are rated as popular tend to show near the top of a post's comment page. Theoretically, a reader can be influenced by the top comments on a post's page. Actors with the ability to predict the popularity of Reddit comments can create more successful marketing campaigns or targeted advertising. More concerning, however, would be its use by a malicious actor. If an individual knew what features led to popularity in a subreddit, s/he could manipulate comments in the community to spread propaganda. Individuals using Reddit as a news source could then be presented with false information that is seemingly vetted by the community.

We hypothesize that the same features commonly marking a post as being popular are also applicable to comments. To test our hypothesis, we surveyed the literature for features and machine learning techniques that have been used to predict Reddit popularity in the past. We tested our hypothesis through obtaining, fitting, and testing two million Reddit comments through three different supervised machine learning classification algorithms.

Our results showed accuracy worse-than-random, with extremely low kappa statistics. Therefore, we concluded there is no evidence to suggest that the set of identified features allow a user to predict the popularity of a comment. Tailoring a comment to achieve popularity in any given subreddit appears to be more nuanced than focusing on any sole feature, making it difficult for a malicious actor to unfairly manipulate the perceived popularity of their comment. We hope our results will motivate researchers to identify new sets of features for Reddit comment popularity analysis.

Background
Reddit's popularity algorithms are open-source. A post[7] on Hacking and Gonzo explores these algorithms. Submission time and the post score are important factors in a post's popularity. A post's score is calculated by subtracting the number of down-votes from the number of up-votes.

Popularity
One way Reddit classifies popularity is by describing a post as *hot*. When classified as hot, a post becomes prominently featured on the subreddit. Hotness is determined by the score of the post on a logarithmic scale, such that the first 10 up-votes produce the same weight as the next 100. This number is then added to the number of seconds since epoch divided by 45,000. This shows that time has a large impact on how prominently a post is featured[7].

Unlike posts, the time a comment is created does not affect its popularity. Rather, comment popularity is determined using a confidence sort[7] based on the Wilson score interval, a commonly used confidence interval estimate for binomial distributions. The algorithm for the confidence sort is shown in Figure 1. Each comment receives a tentative ranking that the algorithm believes it will get to with 85% confidence[7]. A comment with 10 up-votes and 1 down-vote will have a higher confidence score than a post with 40 up-votes and 20 down-votes and, thus, be ranked higher and appear closer to the original post[8].

The algorithm looks at the proportion of up-votes a comment has received compared to the total votes and the sample size. However, this research focused on identifying *what makes a user up-vote a comment*. To try and answer this question, our research goal was to try and identify the set of features that make users choose to up-vote comments.

```
def _confidence(ups, downs):
  n = ups + downs
  if n == 0:
      return 0
  z = 1.281551565545
  p = float(ups) / n
  left = p + 1/(2*n)*z*z
  right = z*sqrt(p*(1-p)/n + z*z/(4*n*n))
  under = 1+1/n*z*z
  return (left - right) / under

def confidence(ups, downs):
  if ups + downs == 0:
      return 0
  else:
      return _confidence(ups, downs)
```

Figure 1. Source Code for Reddit's Confidence Sort[7]

Related Work
Several researchers have attempted to study what makes Reddit posts popular, with varying levels of success. For example, two independent studies[9,10] have pointed to the time of day as being an important indicator for a post making it to the front page, particularly 0900 PST. Lakkaraju *et al*. studied[11] the importance of submission titles in predicting the popularity of posts containing images. The authors concluded that sentiment is a strong predictor and is specifically niche to particular communities. Furthermore, polarizing posts, determined by some sentiment analysis, fair better than neutral ones[11]. Certain subreddits inherently get more up-votes than others, to include `r/funny`, which ties to the fact that images tend to be more highly up-voted. The authors also found that the title of the post should be similar to the wording used within the community, but at the same time be novel enough to introduce dissimilarity[11].

A student paper at Stanford[12] attempted to classify posts to subreddit based on post title alone. In pre-processing, they removed each common word and counted number of instances. Every tenth post was used to construct the actual model, while the other nine were used for training. In classifying, they applied three main algorithms to classify post titles: linear classification, Multinomial Naïve Bayes, and support vector machines (SVM). They concluded that an SVM classifier performed best on the test set, with an accuracy of 96.46% on 2 subreddits, and 84.91% on 10 subreddits.

In a blog post published in 2016, DataStories analyzed the trends found in the top 100 posts that occur on the Reddit front page[9]. For their analysis, they collected the top 100 posts for every 2 minutes for 22 days. They then deleted any posts that were on the front page for less than 2 minutes. This yielded a total of 2,344 unique posts. Their findings support the work of earlier researchers, verifying the importance of posting time, the relative success of polarizing posts compared to neutral ones, and the relative success of positive headlines to stay on the front page for a long time[9]. While image posts tend to get more up-votes than textual posts, the latter tend to stay on the front page longer and receive more comments. They also found that certain subreddits dominate the front page, such as `r/funny`, `r/pics`, and `r/gifs`. Post titles containing numbers have a higher likelihood of reaching the front page. Lastly, the authors note that the average lifetime for a post on the front page is 4 hours and 15 minutes; over 85% of the posts get to the front page in less than 3 hours[9].

In 2016, Tracy Rohlin studied Reddit post popularity as the subject of his Masters' thesis[13]. He gathered data from six subreddits. Only the first 1,000 posts in a subreddit were considered. Each post was first pre-processed and represented as a feature vector using the bag of words model, term frequency-inverse document frequency, or Linear Dirichlet Allocation. Each post in a subreddit was labeled "popular" or "unpopular" by comparing its voting score to a threshold.

Figure 2. Overview of Methodology.
Pre-processing reduces the set of data from 58 million to 239,000 comments

The resulting feature vector was fed into either a Naïve Bayes classifier or a Support Vector Machine. The accuracy of each classification was above 50% for each subreddit. The author notes that lowering the text length cutoff (i.e., increasing the amount of text considered by the algorithms) would have increased the dataset size, and possibly increased model accuracy.

We note that most research in this area has analyzed the popularity of Reddit posts. Posts are the primary mechanisms for content delivery on Reddit. Many users are dubbed lurkers, or users who only consume content, rather than produce it. These users may look at the original post and never make it to the underlying comments, or look at the comments rather than visiting the article links associated with the post. As a result, comments are a possible mechanism to perpetuate false information or perform targeted attacks. If a malicious actor knew what would be popular within the subreddit, s/he could more easily gain support from the community to achieve the objectives. The main contribution of our work is providing an initial analysis of how feasible it would be for a malicious actor to manipulate features of a comment to make them more popular.

Methodology

We study the performance of the Naïve Bayes and Support Vector Machine classifiers, as explored by various studies[9,11,13] in the literature. We also examined the efficacy of a Decision Tree classifier, hypothesizing that this classifier would yield high accuracy. Figure 2 shows an overview of our methodology, which is covered in greater detail in the subsections that follow.

For experimentation, we leveraged the Department of Defense's High Performance Computing supercomputer, Topaz, which is hosted by the US Army Engineer Research and Development Center (ERDC). The cluster has 3,456 nodes with 36 cores of Intel's Xeon E5-2699v3, each at 2.3 GHz and running SuSE Linux.

Data and Feature Set

The data was derived from a post15 on `r/datasets`. Uncompressed, the size of the comments is 30 GB and includes 53,851,542 comments. The dataset contains comments created in reply to a post, and does not include information on the original post. This limits the extent of our analysis. Each line of the file was in JSON. We used several fields in the file for analysis (Table 1).

The `distinguished` field was used to determine moderator status (i.e., if the comment was distinguished, the poster is either an administrator, moderator, or other individual with special rank in the community). Moderators are usually vetted members on Reddit and frequently post in the subreddits they moderate. Given this, we hypothesized that their status as an authority figure within a subreddit may improve their chances of getting up-votes on a particular comment.

The time of creation (`created_utc`) was also used as a feature, though we chose to focus on the hour. In the DataStories blog post, it was found that 0900 PST was the best time to obtain upvotes9. While the study focused on posts, we hypothesized that creation time may also be an important indicator for comments.

We also used the controversiality field as a feature in our analysis. The DataStories blog post found that comments with polarizing sentiments (i.e., either overly positive or negative) yielded the most upvotes[9]. We hypothesized that this polarity in sentiment implies that those comments are extremely controversial within a subreddit.

The comment text (`body`), was used for our bag of words and sentiment analyses. The bag of words was selected, as it was the sole feature analyzed in Rohlin's thesis. Rohlin obtained better-than-random results using only the bag of words, suggesting it might also a good feature for comment popularity[13]. For sentiment analysis, we used an 8,000 word subjectivity lexicon[16] that is publicly available from the University of Pittsburgh. Each word in the sentiment lexicon was labeled as being either positive, negative, or neutral. We used the Python TextBlob[17] library's sentiment analysis feature on each comment to predict its sentiment label, which was included as part of each comment's feature vector.

Lastly, the number of up-votes (`ups`) was the field used primarily for determining popularity. We also used the subreddit field to separate comments by `subreddit` in our pre-processing step, as we discuss in greater detail below.

Moderator Status	Hour of Creation
Sentiment of comment	Controversiality
Bag of words	

Table 1: Selected Features

Pre-processing

To label popularity, we first instantiated a Python dictionary such that each subreddit is represented as a separate key, and the value is initially an empty list. During the first pass over the data, we populated the list associated with each subreddit with the number of up-votes each comment in that subreddit received. At the conclusion of this step, each subreddit had a list of all the up-vote counts associated with it.
As in Rohlin's thesis, we defined popularity as the top 25% of posts within a subreddit[13]. Thus, during the next step, each list was sorted and then the value at the index of the top quartile was selected. This value indicates the threshold for popularity for that subreddit. Note that this value will be different between subredddits. Since popularity varies wildly between subreddits, this is an important distinction. Next, we performed a second pass over the data. Any comment with a number of up-votes above the specified threshold for a particular subreddit was labeled popular, while anything below the threshold was labeled as unpopular. This represents the "true" popularity label.

Finally, we removed a number of unpopular comments from each subreddit's dataset, such that the number of unpopular comments equaled the number of popular comments in that dataset. This ensured our training and testing datasets each had a 50/50 mixture of popular and unpopular comments. This mimics the process outlined in Rohlin's thesis[13]. Ensuring that we had the same number of unpopular and popular comments also established a baseline for classification. After pre-processing, 239,099 comments were left spanning 8,259 subreddits. Each subreddit's comments were written to separate CSV files. Each row of the CSV file was a separate comment, with the columns representing the features shown in Table 1.

Classification

The top five subreddits with the most comments after pre-processing were examined for classification: `r/news`, `r/worldnews`, `r/todayilearned`, `r/destinythegame`, and `r/leagueoflegends`. We noted that the first three subreddits are "fact"-based subreddits, and are commonly used by Reddit users for knowledge acquisition. The last two are "gamer"-based subreddits, and are discussion forums for two very popular on-line games.

The CSV file for each subreddit was processed separately. Each field containing text was label- encoded, producing an integer that the classifier can read. For example, if a comment is popular, the popular field is set to 1; otherwise, it is set to 0. The body of the text is turned into a vector where each index represents a distinct word. The number at that index represents the frequency that word appears in the comment.

Following Rohlin's procedure, we reserved 20% of the data for testing. The remaining 80% was used for training the classifiers13. Each classifier was fitted with both the features and popularity label from the training data. Each classifier then predicted the popularity of the testing data without knowing the true popularity label.

Performance Metrics

To measure the efficacy of each classifier, the accuracy, precision, recall, F1, and Cohen's kappa statistic was computed using the predicted values in comparison to the true popularity label for each prediction. The average value across each prediction within that subreddit was used for the computation of the accuracy, precision, and recall values. Accuracy is defined as the ratio of correctly identified popular predictions to total predictions. Each prediction can be classified by its relation to its "true" value. Predictions correctly marked as popular are referred to as true positives (tp). Likewise, predictions correctly marked as unpopular are true negatives (tn). A prediction that is marked as popular when it actually is not is referred to as a false-negative (fn). Lastly, a prediction that was marked as unpopular when it truly was popular is considered a false-positive (fp). The recall and precision statistics gauge the sensitivity and usefulness of our results. Recall is defined by the ratio of the number of true positives to the sum of true positives and false-negatives (Equation 1).

$$(1)\ recall = \frac{tp}{tp+fn}$$

$$(2)\ precision = \frac{tp}{tp+fp}$$

Precision (Equation 2) is the ratio of the true positives to the sum of true positives and false- positives (fp) combined. It measures the positive predictive value of the classifier. For our purposes, it was the ratio of the comments correctly identified as popular to the total number of truly popular comments. As a supplement to accuracy, we used the F1 measure (equation 3), which is the weighted average of both precision and recall.

$$(3)\ F1 = 2 \times \frac{precision \times recall}{precision + recall}$$

Lastly, we used Cohen's kappa statistic[18] to measure the classifier's ability to use fitted data to properly label comments. Since half of our testing dataset was composed of unpopular comments, a naïve classifier can easily achieve 50% accuracy simply by guessing that every comment is unpopular. Thus, the kappa statistic determined that predictions are due to chance (closer to zero) and not due to accurate fitting (closer to one). Likewise, the F1 measure is also expected be low.

Results

Table 2 and Table 3 show the results of our experimentation. Across classifiers and subreddits, accuracy ranges from 42.0% to 52.7%. The accuracy numbers are lower than expected, and no classifier achieves better-than-random results. Given that the testing dataset contained an even mixture of popular and unpopular comments, successful classification should yield an accuracy score of greater than 50% with a high kappa statistic.

The Cohen's kappa statistics for the subreddits listed in Table 2 and Table 3 range from -0.160 to 0.056. Again, a kappa statistic score close to zero indicates that our classifier's popularity prediction agrees with the true popularity label primarily by chance. The negative value indicates that certain features hurt the Naïve Bayes classifier's ability to make random guesses. These two scores indicate that every classifier yielded worse-than-random results. We found no pattern in determining which classifier performed the best. The subreddit *r/destinythegame*, however, produced the highest kappa statistic with the Decision Tree classifier. However, the kappa statistic score for the Decision Tree classifier is only 5% (0.056), which is still close to zero.

The F1 score in Table 2 and Table 3 only show the balance between recall and precision. The low score indicates a rather significant imbalance. In every case, precision was higher than recall. This indicates that predictions include more false-negatives than false-positives. In the context of our experiments, false-negatives correspond to labeling comments as popular when they truly are unpopular. Our results indicate that our classifiers did no better than a naïve classifier that randomly predicted whether comments were popular or not. We hypothesized that we would achieve better-than-random results with a much higher kappa statistic than what our experiment yielded. We based this hypothesis on Rohlin's previous work classifying the popularity of Reddit posts using solely a bag of words[13]. While the goal of this paper was not to validate Rohlin's results, we were attempting to apply his work to Reddit comments. However, this was not a perfect re-creation. We did not use the same subreddits, and we incorporated other features not used in Rohlin's thesis, mainly metadata. For certain subreddits, we included two to three times as many data points. We also examined each classifier's ability to utilize fitted data, rather than make random guesses, to classify popularity with Cohen's kappa statistic.

Conclusions & Future Work

Most prior work studying Reddit popularity has focused on posts. In this paper, we used a dataset consisting of two million Reddit comments to predict Reddit comment popularity. Through the fitting and testing of these comments on three different supervised machine learning classifiers, we determined each classifier's ability to predict Reddit popularity. Our chosen mixture of features contain both metadata and content, and were largely derived from the literature.

Our results indicate that there is no evidence suggesting that Reddit comment popularity is determined by our chosen feature set. Each classifier produced worse-than-random accuracy scores and extremely low kappa statistics. We were unable to validate the results produced by other authors conducting similar experiments. We believe this discrepancy is attributable to a difference in context between posts and comments.

However, the results do not indicate that it is impossible to tailor comments to garner more up-votes from the community. Rather, our results imply that the particular features that make posts popular are not necessarily the same features that make comments popular. Comments, we now hypothesize, are more sensitive to the original post and other comments in reply to the original post. Future work should attempt to quantify this context for further classification of comment popularity. Given the nature of comments, we predict that the time relative to the original post, the number of comments previously posted, and the similarity in sentiment between the original post and the comments are all features worthy of future exploration.

The subjectivity lexicon used for determining sentiment needs expansion. The University of Pittsburgh's lexicon was chosen because it was freely available under the GNU Public License, and originally seemed like a good fit for our experiments. While an 8,000 word lexicon may initially appear to be a comprehensive choice, it is insufficient for a platform like Reddit. Many users intentionally misspell or alter words. For example, in the popular subreddit *r/rarepuppers*, the word 'puppy' is intentionally spelled 'pupper.' The sentiment of 'pupper' would not be classified by the lexicon even though it should likely have the same sentiment as 'puppy.'

Despite our results, we believe Reddit users should not let their guard down and assume a comment represents a consensus of people's views on a particular subject based solely on the comment's popularity. Especially for fact-based subreddits, a myriad of different sources should be analyzed before trusting any particular comment. We hope our work will motivate others to study Reddit comment popularity. To assist others with training future classifiers, all of the work and results of these experiments are available freely on GitHub[19]. We also believe the time-consuming pre-processing step is a potential avenue for future parallelization. This will enable a higher volume of comments to use in future studies, which we predict will increase classifier accuracy.

Acknowledgements

We are grateful to the DoD High Performance Computing Modernization Program (HPCMP) for providing us access to the Topaz cluster, on which we ran all of our experiments. The opinions expressed in this paper are solely of the authors, and do not necessarily reflect those of the Department of Defense, the US Army, or the United States Military Academy.

	r/worldnews (3,031)			*r/todayilearned (2,242)*			*r/news (2,058)*		
Classifier	Accuracy	Kappa	F1	Accuracy	Kappa	F1	Accuracy	Kappa	F1
Linear SVM	0.471	-0.057	0.471	0.520	0.037	0.495	0.521	0.042	0.498
Decision Tree	0.525	0.051	0.524	0.520	0.045	0.538	0.457	-0.087	0.462
Naïve Bayes	0.503	0.007	0.506	0.511	0.021	0.495	0.516	0.032	0.514

Table 2. Statistics on "fact" based subreddits

	r/destinythegame (1,917)			*r/leagueoflegends (1,862)*		
Classifier	Accuracy	Kappa	F1	Accuracy	Kappa	F1
Linear SVM	0.507	0.009	0.469	0.527	0.045	0.470
Decision Tree	0.524	0.056	0.556	0.469	-0.071	0.408
Naïve Bayes	0.507	0.015	0.507	0.420	-0.160	0.409

Table 3. Statistics on "gamer" subreddits

References/Sources

1. *M. Barthel, G. Stocking, J. Holcomb, and A. Mitchell. Seven-in-ten Reddit users get news on the site. Pew Research Center, 2016.*
2. *Reddit Help. Audience and demographics. 2015. https://reddit.zendesk.com/hc/en-us/articles/205183225-Audience-and-Demographics.*
3. *B. Tedeschi. The best politics Reddit IAMA's. Website, 2017.*
4. *Reddit. Reflections on the recent Boston crisis. 2013. https://redditblog.com/2013/04/22/reflections-on-the- recent-boston-crisis/.*
5. *A. Ohlheiser. Fearing yet another witch hunt, Reddit bans 'pizzagate'. The Washington Post, 2016.*
6. *S. Huffman. TIFU by editing some comments and creating an unnecessary controversy. Reddit Announcements, 2016.*
7. *A. Salihefendic. How Reddit ranking algorithms work. Medium.com, 2016. https://medium.com/hacking-and-gonzo/how-reddit-ranking-algorithms-work-ef111e33d0d9#.a8l1ow50b*
8. *R. Munroe. Reddit's new comment sorting system. Reddit Originals (RedditBlog). 2009. https://redditblog.com/2009/10/15/reddits-new-commet-sorting-system*
9. *Data Stories. We analyzed 4 million data points to see what makes it to the front page of Reddit. Here's what we learned. 2016. https://blog.datastories.com/blog/reddit-front-page*
10. *E. Gilbert. Widespread underprovision on Reddit. In Proceedings of the 2013 conference on Computer supported cooperative work, pages 803–808. ACM, 2013.*
11. *H. Lakkaraju, J. J. McAuley, and J. Leskovec. What's in a name? Understanding the interplay between titles, content, and communities in social media. CWSM, 1(2):3, 2013.*
12. *R. Talreja, M. Fang, and J. Baena. Classifying Reddit post titles to subreddit. 2016. http://robots.stanford.edu/cs221/2016/restricted/projects/rtalreja/final.pdf T*
13. *. Rohlin. Popularity prediction of Reddit texts. Master's thesis, San Jose State University, SJSU ScholarWorks, 2016.*
14. *Pedregosa, Fabian, et al. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research pp. 2825-2830. 2011.*
15. *Stuck_In_the_Matrix. I have every publicly available Reddit comment for research. 1.7 billion comments @ 250 GB compressed. Reddit Post, 2016. https://www.reddit.com/r/datasets/comments/3bxlg7/i_have_every_publicly_available_reddit_comment/.*
16. *T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. Proceedings of HLT-EMNLP-2005. 2005.*
17. *S. Loria et al. TextBlob: Simplified Text Processing. 2013. https://textblob.readthedocs.io/en/dev/index.html*
18. *A.J. Viera, J.M. Garret. Understanding interobserver agreement: the kappa statistic. Family Medicine 37(5). pages 360-363. 2005.*
19. *S. Deaton. redditpopularity. 2017. https://github.com/FourMoreCups/redditpopularity*

Live Detection of Network Attacks Using Social Media

Ben Fry
Supervised by
Associate Professor
Nate Chambers
United States Naval Academy

Introduction
Detecting cybersecurity attacks on networks is a central challenge faced by the Navy, DoD, and cybersecurity community at large. Recognizing an attack in its early stages allows victims to respond quickly, place defenses and mitigate the issue. For example, in the case of a distributed denial-of-service (DDoS) attack, the affected party is often pinged at high rates in order to overwhelm the party's bandwidth resources and take its site down. Clearly, getting ahead of this kind of attack would allow the victim to respond before too much damage has been incurred. However, due to the growing size and complexity of networks, standard methods of threat detection have become challenging. Additionally, standard approaches are only possible when administrators between networks are willing to work together, and public and private domains openly report the details of ongoing attacks, which is rarely the case due to monetary, political, or security-related factors. Nevertheless, the users of such networks often turn to social media without delay to express their complaints when things go awry.

There is a small community of hackers, system administrators, and security vendors on Twitter who discuss security vulnerabilities regularly. Altogether, previous work has suggested that there are about 32,000 of these users [1]. Though this small community is responsible for reporting network attacks, their posts can be overwhelmed through adversaries injecting fake tweets. Our approach focuses on the everyday users affected by the attacks. These users are far more plentiful, and much less prone to be affected by these fake, injected tweets. Through natural language processing (NLP), these tweets can be interpreted and used to improve cybersecurity and awareness. The goal is that these users naturally use similar language when talking about attacks, regardless of the type or entity under attack.

The model that was started during this project uses NLP and machine learning on millions of posts on Twitter to map nationwide network attacks. Several NLP techniques were used to create this system. The goal is to make this model as reliable as possible. The first step was to identify the type of language contained in Twitter posts that might indicate a potential DDoS attack. Fig. 1 provides a few candidate posts from September, 2014 that reflect the type of information we tried to interpret [2]. Obviously, processing this information required machine learning and advanced NLP algorithms to extract useful and relevant data

Figure 1: Twitter posts related to cyber-threats

A benchmark for our model is to show that cyber-threats do occur where the model detects them, and prove that the model is able to detect them in a timely manner. With this capability, incidents like the one in which CENTCOM's Twitter and YouTube accounts were hacked in mid-January, 2015 could be much less embarrassing. However, this would be the simplest of cases. Ideally, more serious threats such as full-scale DDoS attacks will be detected, and the victims would be notified in time to respond.

NLP algorithms have proved effective on Twitter in detecting events in real-time. For example, these algorithms have succeeded in detecting major flu out-breaks, and have even outpaced the CDC tracking of the flu (Ritterman et al. 2009, Lamb et al. 2013). Thus, we were confident that, if approached correctly, Twitter could be used to indicate when and where certain cyber-attacks are taking place.

The purpose of this research project was to continue building a near-real-time model that utilizes Twitter and natural language processing (NLP) technique to detect nationwide cyber-threats. The goal was to detect cyber-attacks, and to gain insight into the language people used when mentioning these attacks. In this article, we first describe the method by which data on previous cyber-attacks was collected. We then show how machine learning algorithms, support vector machines (SVMs), and maximum entropy (MaxEnt) were utilized to make classifications of when entities were attacked. Finally, we present the results and describe how the quality of the data used in these algorithms greatly determined the accuracy of the model.

Figure 2:
Example of Twitter Advance Search Query

Data Collection
To begin collecting data for our work, we had to gather a list of dates when companies were attacked. An initial list of 35 attacks was compiled by searching various news outlets. From this list, we removed all attacks that exploited or stole user credentials instead of denying service to users, resulting in a final list of 24 attack dates. From this list we then collected data utilizing two separate approaches.

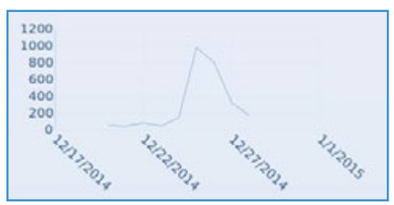
Tweepy
The first approach was to use Twitter's API with Tweepy, a Python library, to download Twitter data. Though this method generates several gigabytes of daily data, the data only represents 1 percent of all daily tweets. These tweets are randomly sampled throughout the day which presents several issues, such as having an insufficient number of tweets mention the attacked entity. If an attack was committed on a smaller company, much less people would tweet about it. This, paired with the 1 percent sampling, resulted in an unbalanced dataset. When building our dataset, the attack tweets contained all tweets that mentioned the attacked entity on that day, whereas non-attack tweets were built from all tweets on a day where an attack did not occur. The combination of having few attack tweets and an excess of non-attack tweets led to the imbalance and extremely poor results. In order to solve these issues, we had to gather more attack data, which led to our next approach.

Phantom Browser
The second approach we took involved building a phantom browser in order to scrape specific data from Twitter's Advanced Search, which is only available on their website. The phantom browser would query Twitter's Advanced Search with a date range and entity keyword. An example of a query searching for Planned Parenthood on July 7, 2016 can be seen in Figure 2. The browser would then repeat the process of downloading all tweets on the current page, then scrolling down and downloading the next section of available tweets. The data down-loaded per entity included: four days before and following the attack, and the day of the attack. This resulted in a much more balanced dataset, with each entity having eight days of non-attack tweets and one day of attack tweets.

One issue we ran into with this approach was that the phantom browser would stop refreshing into the next section of tweets. We believe this is due to the browser downloading too much data too quickly, and having Twitter block the browser for a time. In order to add more data to the entities that did not get the full amount of data, tweets gathered from the previous section's method were added. This resulted in a dataset that was much more balanced and complete. The majority of the days that this dataset represented contained every tweet that mentioned the desired entity that day. The distribution of tweets mentioning an entity during the attack day and within the surrounding days was fairly normal; with the frequency of tweets spiking during an attack and leveling off after the attack. This can be seen in Figure 3.

Features
Language is complex, based on complex syntax and human intuition; something that is difficult to program into computers. Since the establishment of NLP, research has sought methods to model languages within the paradigm of a computer. In order to represent a language effectively, features of a given language need to be tailored to the project. For our purposes, we were interested in how people spoke about network attacks on a social media platform, resulting in our extracted features.

Pre-processing
In order to generate useful features, a tweet's language first had to be cleaned and normalized. Examples of this process include replacing all URL's with a 'URL' tag removing symbols, non-English letters, white space, and various punctuation. This process is essential in feature generation, because our features are sensitive to the slightest differences in a sentence. For example, if two users mention Xbox and one user's tweet contains a typo, an apostrophe after the Xbox, without pre-processing these would be two separate features. Through thorough pre-processing, these kinds of inconsistencies are removed from the text.



"X is offline", "DDOSing X", "X is down"
"hackers attack X", "LizardSquad attacked X"

Figure 3 (left): Frequency of Tweets Mentioning Entity
Figure 4 (above): Example of Victim Context Feature

Figure 5:
Constructed Features

Attacked Entity	Date
Universal Music	2012-01-19
RIAA	2012-01-19
MPAA	2012-01-19
DOJ	2012-01-19
Hadopi	2012-01-19
FBI	2012-01-19
Pirate Bay	2012-05-16
Bank of America	2012-09-19
Chase	2012-09-19
Wells Fargo	2012-09-19
PNC	2012-09-19
Wells Fargo	2012-09-25
PNC	2012-09-26
PNC	2012-09-27
Pirate Bay	2012-11-13
Xbox Live	2012-12-25
Reddit	2013-04-19
Feedly	2014-06-11
Evernote	2014-06-10
Playstation	2014-12-25
Xbox	2014-12-25
Tor	2014-12-26
Github	2015-03-27
Planned Parenthood	2015-07-30

Figure 6:
List of all attacked Entities

N-Grams and Victim Context Tagging
The features we extracted from the tweets had to encompass how people talked about network attacks. The first features we chose were uni-, bi-, and tri-grams. These features were selected due to their ability to retain the original sentences' semantics.

We then constructed a victim context feature, which created an N-gram containing the entity's name, but replaced the name with an 'X' tag. This feature was able to represent some of the words users use in order to describe a website that is offline. Examples of this feature can be seen in Figure 4. Following an analysis of the learned feature weights for an SVM, these features were usually weighted the most, meaning they were the most influential. Some of these example features can be seen in Figure 5.

Training the Classifiers
While we were testing both of the classifiers, maximum entropy, and support vector machines, it quickly became obvious that the MaxEnt performed much worse than the SVM. We believe that this is due to the nature of the MaxEnt decision algorithm, which is biased towards an equal, balanced dataset. Our dataset, which contained an eight-to-one ratio of non-attack days to an attack day, did not meet this assumption. Regardless, we still include the results in this section. Within the SVM, we were able to optimize a portion of the algorithm using cross-validation in order to obtain respectable results.

Training Dataset
As mentioned above, the training dataset did not have an equal balance of non-attack and attack days. The chosen non-attack days were four days before and after the attack, which could potentially make the decision for the classifier more difficult due to the overlap of users talking about the attack. Out of the 24 total attack dates, spanning 2012-2015, we chose all attacks occurring between 2012-2013 (17 total attacks), to build our dataset. The complete list of attack dates can be seen in Figure 6.

Support Vector Machines
After the SVM was identified to be the better of the two classifiers, one cross-validation was used to determine the best initialization for the slack parameter. The cross-validation technique included looping over six different slack parameters. Within each loop of the 17 total training dates, 16 were chosen to train the classifier, leaving one out to test the classifier. This was repeated 17 times, such that all 17 attack dates were tested for each slack parameter. The averaged accuracy for each slack variable initialization can be seen in Figure 7. After the optimal slack parameter was found, a classifier was trained on all 17 attack dates using this optimal value and was saved for testing.

Maximum Entropy
For MaxEnt, the 'out of the box' classifier was used. Though there are various ways to initialize this classifier, the baseline results for this classifier were much worse than SVMs, so no further work was put into determining the optimal initializations.

Results
The results of our work are much better than the previous semester. The comparison of these results can be seen in Figure 7.

Classifier Comparison
The F1 scores are constructed with recall and precision metrics. This score not only takes into account how many times the classifier predicted the correct labels, but also how many times the classifier predicted false-positives. Thus, this score is indicative of overall classifier performance by taking into account the possibility of over-classification. For each of the 8 testing attack days, the same method of building non-attack days via the surrounding eight days was used. The highest total accuracy achieved on this test set was 90 percent.

Influence of Data
The classification results are largely dependent on the dataset used for training and testing. When we trained and tested the classifiers on data collected via Twitter's API, the results were inconclusive. The in- complete data resulted in many features which had little correlation to the task at hand. When we looked into some of the features the classifiers weighted the most, the majority of them were often uninformative. These features ultimately led to accuracies that were worse than random. The dataset collected with the web scraper provided much better results. Features we expected to be highly correlated (examples in Figures 4 and 5), typically were weighted the most by the classifiers. The classifiers were then able to make informed decisions, leading to respectable accuracies.

Discussion
As discussed above, the results are largely dependent on the quality of the dataset. The results of the classifications were essentially dependent on how frequent users mentioned the attack. If a smaller company was attacked, the chances of the classifier detecting an attack was greatly increased. The frequency of tweets for such a company were very low during a non-attack day, but would spike during an attack. These types of attacks were much easier to detect because of the disparity of extracted features between the attack and non-attack days. For larger entities, such as Xbox Live, where users are continuously complaining about service, classifications were often more difficult.

Future Work
For future work, it is recommended that different types of classifiers and features be explored. Our features only included those listed above and could be greatly expanded upon. If other features are explored, various methods for feature selection could be used in order to determine the types of features most influential on the classifier's decisions. This could potentially result in a greater overall accuracy of the classifiers.

Acknowledgements
This work was created using "Riptide" at Maui High-Performance Computing Center and was published in full at the 2018 Annual Conference of the North American Chapter of the Association for Computational Linguistics.

To read the complete paper with continued research, visit:
https://www.usna.edu/Users/cs/nchamber/pubs/naacl2018-ddos-chambers.pdf.

References
A. Lamb, MJ. Paul, M. Fredze Separating Fact from Fear: Tracking Flu Infections on Twitter. In North American Chapter of the Association for Computational Linguistics (NAACL). 2013.

A. Ritter, E. Wright, W. Casey, and T. Mitchell Weakly Supervised Extraction of Computer Security Events from Twitter. In WWW, 2015.

J. Ritterman, M. Osborne, E. Klein Using Prediction Markets and Twitter to Predict a Swine Flu Pandemic. In 1st International Workshop on Mining Social, 2009.

N. Chambers and T. Oates. Broad-Scale Situational Awareness: Monitoring the State of the Nations Net- works through Social Media. UMBC-USNA Cyber Innovation Grants, 2015.

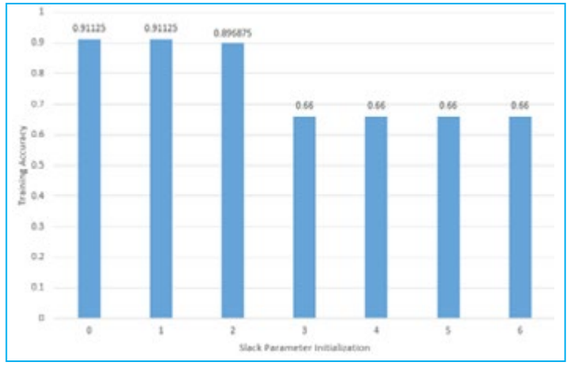


Figure 7:
Training Accuracy with Different Slack Initializations

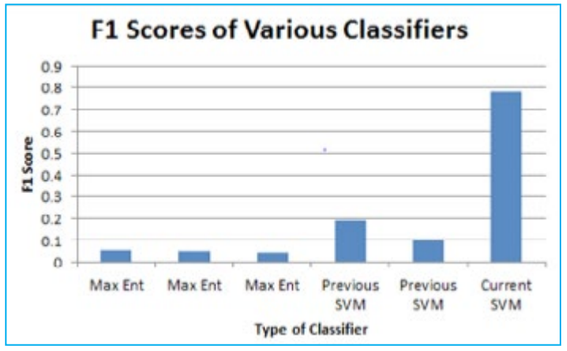


Figure 8:
F1 Scores of Various Classifiers

Creating Intelligent Users of CFD using HPCMP CREATE™-AV Kestrel

Major Matthew Stachell
United States Air Force Academy

Cynthia Dahl
High Performance Computing Modernization Program Office

The United States Air Force Academy's (USAFA) Aeronautical Engineering program is one of the leading undergraduate programs in the field, and offers some of the most effective instruction in computational fluid dynamics (CFD)—an increasingly significant component of aerodynamic design and analysis. When designing their AE 342 *Computational Aerodynamics* course, Dr. Russell Cummings and Dr. Scott Morton understood the need for cadets to have a working knowledge of CFD in order to leave the Academy as skilled aeronautical engineers; they also understood that the average cadet would probably never become a CFD expert. Therefore, Dr. Cummings says, the goal in *Computational Aerodynamics* became to "create intelligent users of CFD," and the course is scaled to this objective. For most cadets, this is their first experience with supercomputing or using CFD software. The *Computational Aerodynamics* course introduces cadets to both the theory and application of the CFD process, covering everything from grid generation to CFD solvers and post-processing.

In 2010, Dr. Cummings decided to use the DoD High Performance Computing Modernization Program's CREATE™-AV Kestrel software—a computational analysis tool for fixed-wing aircraft—instead of the commercial codes he had previously been using in his *Computational Aerodynamics* course. Although he admits to being motivated, at least in part, by Kestrel's affordable price tag (gratis), Dr. Cummings is quick to mention the additional benefits to using Kestrel. He explains that Kestrel's user-friendly GUI interface is intuitive and familiar to cadets who grew up using Windows or other GUI interfaces with similar buttons, tabs, and drop-down menus. Whereas previous codes depended on command line environments which cadets had to edit and submit manually, Kestrel allows jobs to be submitted within the GUI. All of this translates to a vastly decreased learning curve in the classroom; and in the intense educational environment of the Academy, where cadets are required to complete a demanding set of courses within four years, time is valuable. Dr. Cummings also benefits from Kestrel's intuitive interface, noting that another measure of Kestrel's success in the classroom is actually the absence of students in the instructors' offices. "The number of people coming to my office to complain or seek help is almost nonexistent now," he says. "The cadets run through the Kestrel tutorial and a simple job all within minutes, and then they are off and running. They are so excited about what they are doing that they rarely have any questions."

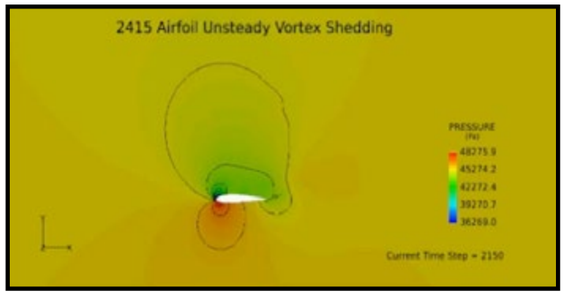


Figure 1:
Screen shot of students' CFD video on YouTube:
http://www.youtube.com/watch?v=rGZ6DfgcHsQ&feature=youtube

One testament to this "excitement" is a CFD video made with Kestrel that some of the *Computational Aerodynamics* students posted on YouTube in their down-time. The video shows a NACA2415 airfoil with unsteady vortex shedding, and is set to the hit song Sail by AWOLNATION. Dr. Cummings was first made aware of the YouTube post by another instructor who had forwarded him the link with the tagline: "CFD at its finest."

After completing *Computational Aerodynamics*, USAFA cadets continue to use Kestrel and develop their CFD skills in various other engineering courses. For example, in their aircraft design courses, students essentially design a full aircraft from scratch. USAFA instructor Major Matthew Satchell says, "It is common for students to run Kestrel on a design regularly to figure out stability derivatives." He continues, "It's exciting to see how CFD and supercomputers are integrated into the entire design process, from concept development through to building development."

In the senior-level *AE 472 Advanced Computational Aerodynamics* course, cadets work with a mentor using HPCMP resources and Kestrel software to research and investigate real problems of interest to the Air Force and the DoD. Major Satchell says that some of Kestrel's newer capabilities allow students to run multi-body simulations with relative ease, and thereby "puts a powerful tool into the hands of cadets." He explains, "The complexity that goes into the interactions of two bodies is immense. With a standard problem such as a fighter jet dropping a bomb, for example, Kestrel makes it much more apparent how to push an object away, determine when that object releases, or if it has an engine and thrust." Some of the 2017 research projects for *Advanced Computational Aerodynamics* included an investigation of flows involved in C-130 cargo drops and an analysis of flows around individual propeller blades of the C-130. "These are incredibly computationally expensive and complicated problems" says Satchell, "but Kestrel provides analysis that wouldn't otherwise be possible [in a classroom environment]."

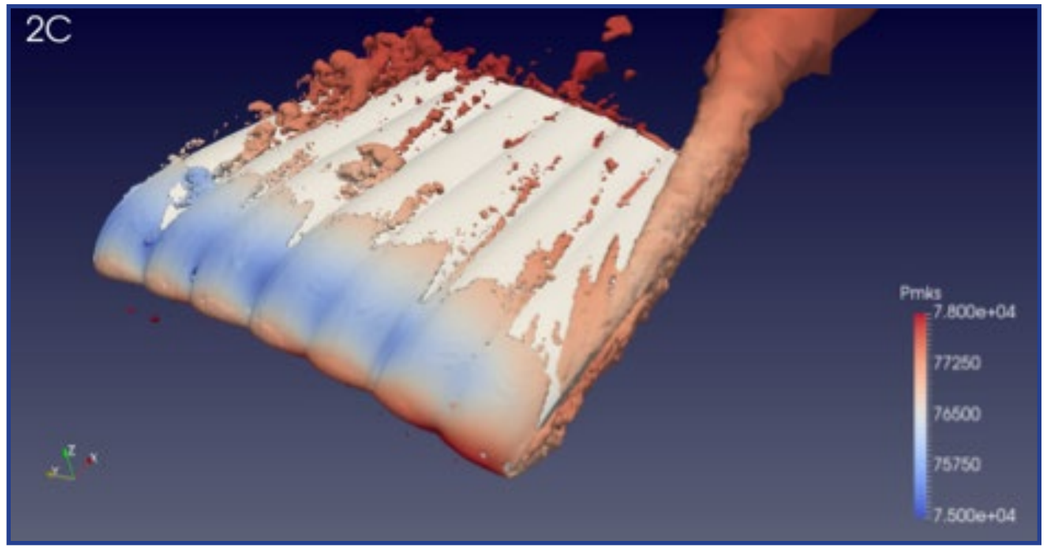


Figure 2:
Visualization of Kestrel CFD results from Cadet Weingast's research on JPADS parachutes

Cadet Jeffrey Weingast, currently a senior at the Academy, had no experience with supercomputing or using CFD software before taking *Computational Aerodynamics* as a junior last year. But after learning the basics of CFD and becoming familiar with Kestrel, Cadet Weingast decided to pursue his increasing interest in the field by taking the *Advanced Computational Aerodynamics* course with Major Satchell in the fall of 2017. Here he worked on an Army-sponsored project that used Kestrel to investigate the aerodynamics of Joint Precision Airdrop System (JPADS) parachutes. He explains, "We analyzed the parachute design used to drop cargo out of C-17s. The better we understand the aerodynamic characteristics of the parachutes, the more effectively we can steer those parachutes toward the drop zone." When he signed up for the course, Cadet Weingast never imagined he would be using CFD to research parachutes of all things, but says he found the research both interesting and exciting. Overall, he says, "working with Kestrel increased my understanding of how CFD works and the details of aerodynamics. It helped me visualize fluid flow in my mind and conceptualize the problems I was trying to solve."

Cadet Weingast graduated from the Air Force Academy with a B.S. in Aeronautical Engineering in May and plans on entering pilot training in October of 2018. Although he doesn't anticipate actively using Kestrel or supercomputers in the near future as a pilot-in-training, he says he definitely sees himself using the supercomputing skills he learned as a USAFA undergrad later on in his career. But that ability to "visualize fluid flow" and "conceptualize problems" he spoke of will undoubtedly be an invaluable asset to Cadet Weingast —not just as an aeronautical engineer sometime down the road—but as an Air Force pilot in the very near future, dealing with the physics first-hand, in real-time, inside a cockpit.

It has now been almost a decade since the Air Force Academy first introduced Kestrel into its classrooms. The result, according to Major Satchell, is that "Kestrel is becoming a DoD standard, or at least an Air Force standard." Hundreds of cadets have used Kestrel in their engineering courses and now have a working knowledge of the software, its capabilities, and how to apply it to an array of real-world problems. And many Academy graduates, like Cadet Weingast, may not go directly into the field directly after graduation to become "CFD experts"; but given the range and complexity of the work being done by these students, it is probably safe to say that the original objective to "create intelligent users of CFD" at the United States Air Force Academy has not only been met, but exceeded.

Using Vectors to Identify Entities in Online Attacks

Stephen da Cruz
Supervised by
Associate Professor
Nate Chambers
United States Naval Academy

Cyber-attacks are an everyday occurrence in the modern world. The threat has come to the attention to the public, and is of great concern to anyone who values their privacy or property. Countless companies have had their customers' private information and credit card information stolen by cyber criminals. The potential for companies to report breaches of security late, or not at all, in order to avoid negative consequences is a major concern. It is in the best interest of the general public to be aware of successful attacks so they can respond appropriately and in a timely manner.

The overall goal of this work is to identify cybersecurity attacks early. Ideally, this would mean detecting an attack on some victim before the media reports on it. We theorize that it is possible to identify at least some security events before they're public knowledge simply by analyzing what people say in social media; specifically, tweets from the freely available Twitter Streaming API. The strategy is to first identify what entities are trending in real-time. Then all tweets that mention the trending entity will be analyzed for key attributes that may indicate an attack. For example, an attack on Bank of America that is responsible for disabling ATMs may say "ATM not working #bofa (@Bank of America in Las Vegas, NV)".

However, this article does not address this entire process. Instead, it simply focuses on resolving Twitter handles to entities. By doing this, we may be able to improve accuracy when identifying what is trending. Hopefully, this will expand the amount of data available for determining whether or not an attack is occurring.

Previously, this issue was tackled by creating word vectors for known entities and words commonly seen with these entities. These vectors were then compared via cosine similarity with the hypothesis that alternate names or acronyms of these entities would be the most similar. Although this work met with some partial success, the experiment was not designed in a quantitative fashion, so aside from some interesting results, it was not particularly useful. In summary, although this program could identify words that were important to an entity and sometimes a true acronym, it never accurately guessed the correct one.

During this project, the work done by Midshipman First Class Peter Goutzounis was built off of a supervised classifier he created that used multiple features to determine whether a username was the correct "handle" for a given entity. On Twitter, people have a username which is usually their daily conversational name, and a handle with a "one-word" account name. In testing, the classifier was given a Twitter username and handle. The classifier would guess whether the handle and username were one and the same based off of such features as:

1. **Percent Substring:** The length of all the words in the username that are substrings of the Twitter handle divided by the length of all the words in the username.

2. **Contains Entire Name:** The username is a substring of the Twitter handle.

3. **Is Semi-Acronym:** The first letters of all the words in the username are combined into a string with no spaces. Prepositions in the user-name are included in full instead of just taking the first letter. For example, "Bank of America" will become "BofA." This new string matches the handle.

4. **First Word Substring:** The first word of the username is a sub-string of the Twitter handle.

5. **First N Letter Match:** The first N letters of the username and handle match.

6. **Last Word Sub-string:** The last word of the username is a sub-string of the handle.

7. **Capitalization:** The capital letters in the username and handle are the same and occur in the same order.

8. **Last N Letter Match:** The number of letters matching when starting from the back.

9. **Is Acronym:** The first letter of each word in the username combined is the Twitter handle.

10. **Edit Distance:** The number of edits that must be made for the username and entity to match.

This work added two features to Goutzounis' classifier in attempt to improve on his work and, overall, to determine the usefulness of using vectors as a means to determine word similarity.

The logic behind this was that while Goutzounis' classifier did an excellent job analyzing the purely aesthetic similarities between usernames and handles, word vectors were an attempt to bring context and meaning to usernames and handles that otherwise have none to a computer. For example when a human sees the handle @KingJames and the username LeBron James, if they know anything about basketball, they will know that LeBron James is also known as King James. A computer will not naturally "know" this; however, with the help of word vectors the hope is that words like king, basketball, NBA, and the like will be seen in conjunction with LeBron James often enough that the computer will be able to tell that the aforementioned username and handle are similar.

This is where the majority of this project's efforts were focused. They concentrated on the method for creating and saving the vectors for the entities and their handles to use as much information as possible (the most tweets). It was rather mundane work in the beginning, effectively just making parsers to go through tweets and check for entities contained in the tweet in constant time, while at the same time handling the fact that entities could be more than one word. The program read-in the entities from a previously existing file and added them to a hash set. The tweets would be read-in line-by-line. Each of these lines would be broken up by spaces so they could be checked to see if they were in the hash set. This process would be repeated with each word, combined with the one and two previous words with added spaces, to check to see if an entity containing more than one word existed in the line. If so, all the words of the sentence, minus the entity itself, were added to its vector. This vector was then saved to a separate file.

This worked relatively well for a couple hundred thousand or even a couple million tweets, but as the number of tweets increased, so did the size of the vector and string manipulations the program needed to handle, which would grind the process to a halt. The manner in which the data was stored was very similar to how another program could read it in and combine different vectors:

> *ENTITY v1 num v2 num v3 num ... To solve this issue, this process was distributed over a number of iterations and saved each run of the vector creation to a separate file. These files were then combined into one large file with could be read in by the supervised classifier. In all, the vectors were created with more than five hundred million tweets with the top most seen entities having counts above six hundred thousand appearances. Once all of the vectors had been combined into one file, the size of the file was 951 megabytes, nearly one gigabyte.*

Initially, all words were excluded from a vector that did not satisfy the prerequisites of a certain IDF score to remove common words; however, the program displaying the IDF would not always work accurately. To make matters worse the spelling of most tweets is horrendous or nontraditional, thus it proved to be ineffective, seeing as an incorrect spelling of "the" as "teh" would be marked as an uncommon word and saved. Optimally, common words should have been removed from the vectors.

To explain the work performed with neural networks, one must first understand what neural networks are. Neural networks take an input that is processed by an interconnected hidden layer of nodes which give an output. In this case, the inputs are the entities, and the outputs are predicted neighboring words.

The neural network we used was provided by Deep Learning for Java, otherwise known as DL4J. The neural network created by them works in a similar way to the word vectors used before. The network is fed a series of strings or sentences, and each word has its network built off of the surrounding words and features of the sentence. When words are compared, the respective vector matrices are analyzed with cosine similarity.

There are numerous features with DL4J's neural network such as iterations for each sentence, how many times a sentence is fed back into the neural network, and how many times a word must be seen to consider it valid.

Setting the environment up was the hardest part. It is not particularly easy setting up Java projects. DL4J, on the other hand, was created using Maven. Constant assistance in getting everything to work was needed, and once that was complete it was still difficult to understand the base code written for DL4J users. Eventually it worked, but not much of the pre-existing code was able to be edited, only applied it to the datasets. As such, even running the code on what seemed like perfectly reasonable short sentences resulted in hanging errors and failures for the programs to end smoothly. It is not known whether the results were affected (vector matrices), but there were clearly some internal errors that needed to be resolved. The more serious matter was the fact the DL4J framework could not be edited to accept bigrams. This posed an issue in that many of the usernames were multi-worded such as "justin bieber". This issue was not resolved.

Running the neural network was much slower than the previous method. Where the previous method could run through approximately fifty million tweets in about an hour, it took this neural network method almost four hours to run two million tweets. The iterations were set at a relatively low number (5), the number of units or nodes were set at (150), and the minimum required number of appearances was (5), seeing as common people's Twitter usernames and handles are obscure, and shouldn't be missed.

Although one might expect the experiment and result sections to be separate, so much of the work fed into further testing that experiments and results have to be explained in unison.

	Accuracy
Baseline	51.939%
Optimal	80.616%
Optimal (seen 50)	71.448%
Optimal (seen 500)	69.204%
Optimal (no Edit Distance)	52.194%

Table 1. Previous Performance

	Accuracy	Improvement
Word Vectors	80.636%	0.020%
Neural Network	80.616%	0.000%
Word Vectors (seen 50)	72.081%	0.633%
Word Vectors (seen 500)	70.088%	0.884%

Table 2. Features Added

The first thing that must be noted is that the supervised classifier already worked very well. Please refer to Table 1 for these results. It operated at an accuracy rate of 80.616%. The baseline operation (with no features turned on) of the supervised classifier was 51.939%. When one feature in particular (edit distance) was removed, the accuracy of the classifier fell to 52.194%, showing that it did most of the work. However, none of the features, including edit distance, performed better than the baseline on their own, and features started to be removed from the big picture, the accuracy would begin to fall.

When analyzing the results of method one, please refer to Table 2. With the addition of the word vector cosine comparison, the accuracy of the classifier in- creased to 80.636%. This was not as high hoped, seeing as this was an improvement of 0.020%. Seeing as it was already working with a well-oiled machine it was understood that it might not be able to be improved too much, and any increase is a good sign.

Another route of testing was used where words were only examined that were seen greater than (50) times. The logic behind this was that word vectors could only be accurately built if an entity was seen a few times, so that one reference out of context would not mess it up. When the classifier was tested as it was without word vectors on words seen more (50) times, it was 71.448% accurate. With word vectors its accuracy went up to 72.081%. This was an improvement of 0.633%.

Following this trend the threshold of appearances was raised to (500). The classifier was 69.204% accurate where it was 70.088% accurate, an increase of 0.884%. It is disappointing to say, but the neural networks did not improve the performance under any conditions.

Even though all of the improvements of method one, Word Vectors, were under a percent, a few things must be taken into consideration. First is that the supervised classifier was already working extraordinarily well. After all, the classifier already matched human performance, as noted in previous research. The fact that there was any improvement is a good sign.

It should also be noted that the decrease in accuracy as the threshold for required appearances increased should not be seen as a bad sign. More studies should be conducted into this but it is hypothesized the average user has a handle that is very similar to that person's true username, making it very easy for the classifier to identify seeing as it relies almost exclusively on different ways to examine string similarities. As entities are seen more often this means they are more popular, and it is believed to have a higher chance of having a handle completely unrelated to their username (ex: The Associated Press is @ap).

However out of all of this, perhaps the biggest area of improvement will be with the issue of common words and IDF scores. As words are seen over and over again, the common words begin to clog up the vectors and words that are seemingly unfamiliar share many of the same words far too often, like the common words: the, is, of, for to, and. Once these are removed it is hypothesized that there will be a drastic improvement of the Words Vectors, but overall the results from method one are pleasing.

The work done with neural networks was disappointing. However it was with tempered hopes that the work on it began. Only three full weeks were spent with DL4J, two of which were bogged down in simply the set-up of the coding environment. Whereas method one used word vectors from over (40000) entities, the neural networks picked up only (17000), with many of these being parts of a larger username (justin bieber became justin and bieber). The sample size was also two hundred and fifty times smaller.

The positives are that there is a foreseeable future for work being done with DL4J neural networks with distributed processes, and on supercomputers to speed-up the run-time so that it is feasible to work with hundreds of millions, and possibly billions, of tweets. If there was more time, the DL4J networks might have been able to run over a couple million tweets, then be combined, like the Word Vector method, time constraints were an issue. This work was completed using Riptide at the Maui High Performance Computing Center.

NEXT-GENERATION SECRET DREN ROLLS-OUT

Douglas E. Johnson,
Deputy Associate Director for Networking

The next-generation Secret Defense Research and Engineering Network (SDREN) is on a roll and coming to a site near you. After more than two years of planning for this SDREN architecture upgrade and technology refresh, the DREN team has partially completed the hardware roll-out to include new routers, new SDREN Joint Sensors (SJSs), and eventually new encryptors.

SDREN is a classified wide-area network operating at the Secret-level. It runs as an overlay on the DREN backbone using National Security Agency (NSA) Type-1 encryptors and common key material among all sites. This allows every SDREN site to communicate with every other SDREN site. SDREN supports the High Performance Computing Modernization Program's (HPCMP's) classified supercomputing centers and users, as well as the DoD Test and Evaluation (T&E) community, all at the Secret-level. The previous SDREN design and implementation occurred following the DREN II deployment in the mid-2000s. A new SDREN architecture and technology refresh is long overdue.

With the completion of the DREN III implementation in 2014, the DREN team began the effort to redesign SDREN, taking advantage of the new capabilities and increased bandwidth provided by DREN III. The redesign included new technologies and a complete equipment refresh including new SDREN routers, new Joint Sensors, and new NSA Type-1 encryptors. The goals for the next-generation SDREN redesign were to provide improved SDREN services and increased network performance by leveraging DREN III's Ethernet (Layer 2) transport and greater bandwidths, integrating IPv6 as a native service, improving support for multicast, and providing native support for 9,000 byte jumbo frames.

Beginning in early 2018, the DREN team began a three-phase upgrade of 78 SDREN sites, from old and outdated hardware to the new SDREN architecture, in the following phases: (1) roll-out of new routers and SJSs to all SDREN sites, (2) limited rollout of KG-350 encryptors at select SDREN sites, to operate in parallel with the current architecture to fully test the capabilities of the new encryptors, and (3) finish rollout of the KG-350 encryptors to the remaining SDREN sites. As of the date of this publication, Phase 1 is fully complete and Phase 2 is underway.

Users of the SDREN resources will see immediate benefits from the new network. It is well-known that SDREN has been bandwidth constrained for several years, based on the limited throughput of the existing SDREN encryptors. With the implementation of the redesigned SDREN architecture, network performance will increase by at least 10X. For most sites, future SDREN limitations will be based on the size of the DREN III service at the site, rather than the throughput of the SDREN encryptor. This will be a welcome change for many SDREN users.

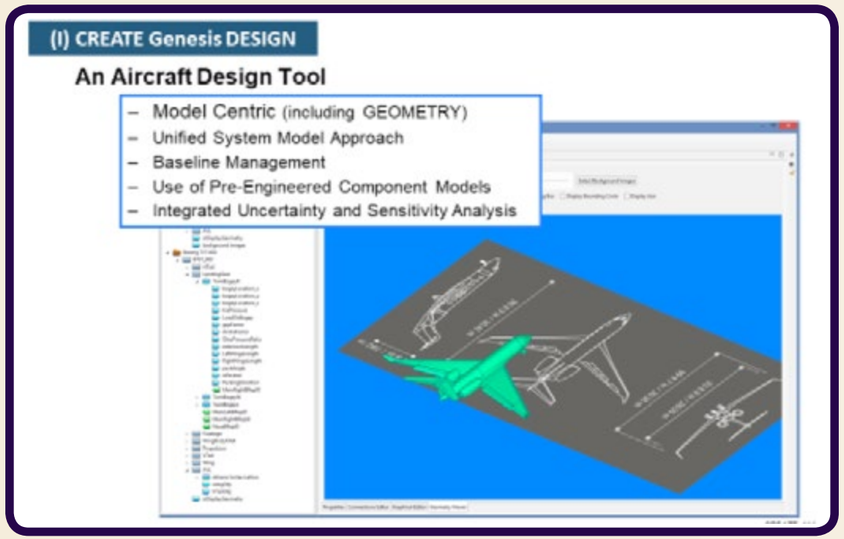
Although the increases in bandwidth capacity and, thus, network performance will be the most noticeable change to SDREN users, there are many new technical capabilities as well. Unfortunately, some of these new capabilities may go unnoticed by SDREN users. These new technical capabilities include Ethernet (Layer 2) transport, integrated IPv6, improved multicast support, and native 9,000 byte jumbo frames support.

In summary, roll-out of the new SDREN architecture is well underway, with full implementation expected in FY19. The next-generation SDREN technology refresh will bring not only new technical capabilities to SDREN users, but also huge increases in SDREN bandwidth and performance. These improvements will benefit the entire SDREN community.

HPCMP CREATE™ Genesis

A Catalyst for Change in Defense Acquisition

By Cynthia Dahl
High Performance Computing Modernization Program Office



Science historian Thomas Kuhn calls a paradigm shift "a scientific revolution," where an intellectual battle of sorts ensues between adherents to the traditional model of a scientific discipline and the followers of a new theory that challenges that paradigm. Based on this premise, Associate Director of the DoD HPCMP CREATE™ program, Dr. Robert Meakin currently finds himself at the head of a scientific revolution within the world of DoD acquisition engineering.

After all, the primary objective of the CREATE element of the HPCMP is to provide the physics-based engineering software tools necessary to fundamentally change the existing paradigm associated with Defense engineering design cycles and its associated workflows. Instead of the traditional "design, build, test, fix" paradigm, CREATE tools allow a virtual test-driven design cycle, enabling early detection of design faults and system performance anomalies. Dr. Meakin explains, "We are not suggesting that simulation replaces the need for physical testing. Mother Nature always gets the deciding vote. When we are surprised by how she votes, the consequences are usually very expensive – schedule, money, sub-optimal systems, and even loss of life. It is crucial to understand that we get to decide when she votes and how often. HPC and physics-informed analysis is the way for us to avoid, or at least to minimize, any surprises that arise when Mother Nature votes."

Dr. Meakin is confident that CREATE software provides essential capabilities needed to enable this paradigm change, but the intellectual battle has not yet been won; because change is hard—for both people and organizations. "We are naturally resistant to change," says Dr. Meakin. "We all have ways of doing things that we are comfortable with and understand; and technology unknown to us represents risk. The effort to fully evaluate new technology is not a simple thing." He also observes that "engineers, by nature, are innovative and don't shy away from difficult problems. They will use their best engineering judgement and best available trusted tools to provide decision makers the data needed in the time-frame allowed. Given a new tool, they won't just go on faith and believe that it's better than their current tool or methodology. Trust is essential, and it takes time. Even though we have extensive verification and validation data for all of the CREATE tools, engineers will have to test it independently against their own data until they develop trust in the new tool. Change generally takes either a crisis of some sort, or a desperate need for a certain capability for which no other source exists." Thomas Kuhn acknowledged this difficulty in overcoming people's natural resistance to change. He even went so far as to argue that sometimes the only thing that ultimately convinces people of a new paradigm is time itself and acceptance from the next generation of scientists who have grown up with it.

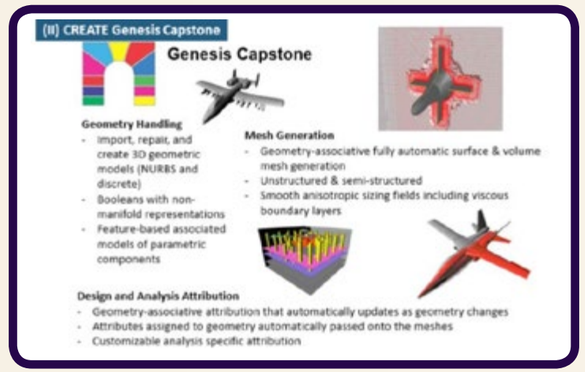
This is where CREATE Genesis comes into play. Genesis is a non-export controlled software suite tailored specifically for engineering educators. It is comprised of Genesis-DESIGN, a design tool with a user-friendly interface and a library of pre-existing models and components; Capstone, CREATE's geometry and mesh generation software; and Genesis-CFD, a reduced-capability version of Kestrel, CREATE's physics based simulation software for air vehicles. Genesis was designed for use in U.S. academic institutions' undergraduate and graduate courses with the intention of building a relationship of trust with future engineers even before they enter the workforce. "If they are familiar with the [HPCMP CREATE] brand and understand its value as they enter the workforce," says Dr. Meakin, "it will set their expectations for how engineering is done in the DoD." He continues, "If you have major universities producing hundreds of engineers each year that enter the workforce, we will be able to flood the market with engineers who understand the value of physics-based simulation and analysis. That's how we win. They will enter the workforce able to make immediate impacts. Over the course of their careers, they will become the next managers, decision makers, educators and innovators. That's how we change the paradigm, change the world. "

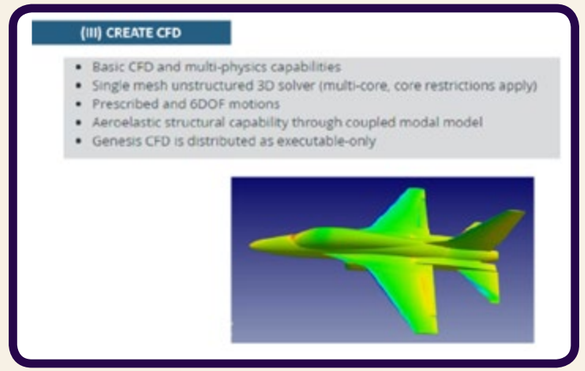
Before "world domination" occurs, however, Genesis has started out on a slightly smaller scale by launching their first pilot program at Georgia Tech during the 2018 winter semester. Through the pilot program, Genesis software is integrated into Georgia Tech's Aerospace Engineering curriculum which includes (but is not necessarily limited to) its Aircraft Design courses, Applied Aero, and Applied CFD courses. Along with access to the Genesis software suite of tools, the pilot program includes sample course outlines, comprehensive tutorials, and web-based technical support. Additionally, the pilot program funds two graduate students to tailor the material to the institution's specific educational goals and to help bridge the gap between students and practicing engineers. In return, explains Dr. Meakin, the CREATE program is looking for suggestions and feedback from both the faculty and students "so we can harden the tools and make sure we understand the use-cases in a classroom setting." He estimates it will take about 2-3 more of these pilot programs before the Genesis software suite is ready for use at other educational institutions, but this is the goal. In 2019, CREATE is planning a widespread marketing push to integrate Genesis software into universities and academic institutions throughout the United States.

Dr. Meakin hopes that by using this model, Genesis will provide an important catalyst for change in Defense acquisition. By proactively educating the next generation of engineers in an academic setting, the CREATE element of the HPCMP is arming them with the experience and the tools they need to implement a new paradigm as they enter the workforce—a scientific revolution that will help the United States and the DoD maintain a significant strategic advantage over other nations. As its name implies, Genesis may just be the beginning of a new world in engineering, originating directly from the next generation workforce.

Independent of the pilot deploys, Genesis is already available to any U.S. academic institution, organization, or individual for educational, research, or evaluation purposes. (It is not permitted for use in commercial applications).

For more information regarding Genesis or to download the Genesis software suite, visit: https://www.hpcmpcreategenesis.org/.





Delivering HPC Power to Researchers

Enhancing the F-35

By Dr. Juan Carlos Chaves
HPCMP PETTT SIP On-site Scientist,
Army Research Laboratory (ARL)

Problem:

The F-35 Joint Strike Fighter program, a critical program for the DoD, is DoD's largest cooperative program with eight Partner nations participating. Power and Thermal Management System (PTMS) and the Aircraft Fuel Thermal Management System (AFTMS) are integral to the successful execution of the F-35's mission set. Modern tactical aircraft subsystems face challenging weight and volume limitations, and future heat load increases will be added to the F-35 which will require improvements to the PTMS and AFTMS.

The DoD's Flight Systems Integration Branch is leveraging modeling and simulation to determine the most promising technologies that will improve these systems. To conduct the studies, researcher use models of all the major F-35 subsystems, leveraging a combination of MATLAB and Simulink serial-based processing frameworks. These complex multilayer coupled dynamic models are too computationally intensive to achieve enough throughput for significant studies in conventional PC or high-end Linux workstation architectures; therefore, researchers want to leverage HPC solutions to support these studies.

Solution:

Computational scientists at the Army Research Laboratory, working in DoD's High Performance Computing Modernization Program (HPCMP) under the User Productivity, Technology Transfer and Training (PETTT) Program, performed a comprehensive analysis of the F-35 model requirements to leverage scalable HPC capabilities. They developed an effective workflow that runs the F-35 models as standalone executables in one or more nodes of DoD HPC resources. The scientists enabled optimal HPC processing by (1) devising innovative techniques for inter-model communication, (2) enabling the passing of state information or parameters among models (including initial states) without recompilation, and (3) provided advanced custom-made PBS scripts, detailed documentation, and expert advice for researcher production runs on HPC resources—allowing for the computing power needed by researchers to improve the F-35.

Solution:

The tri-service F-35 is what is known as a fifth-generation aircraft, which means it integrates advanced stealth, sensor fusion, digitally-optimized flight performance, net-enabled operations and automated logistics in a multirole aircraft far superior to anything that has come before. Compared with a Cold War fighter, the F-35 will be six times more effective in air-to-air combat, six times more effective in suppression of enemy air defenses, five times more effective in destroying ground targets, and four times more effective in evading enemy air defenses.

It will need all of those gains to sustain U.S. global air dominance in the decades ahead, because potential enemies have not stood still in their own military investments. Whether the mission is defeating adversary fighters or striking heavily defended surface targets or providing close air support to ground troops or collecting tactical reconnaissance or delivering non-strategic nuclear weapons, F-35 is central to joint warfighting plans through mid-century. Modeling, simulation and HPC solutions as the one supported by HPCMP PETTT in this narrative play an increasingly important role in the F-35 Joint Strike Fighter program.

"This material is based upon work supported by, or in part by,
the Department of Defense High Performance Computing Modernization Program (HPCMP)
under User Productivity, Technology Transfer and Training (PETTT) contract number GS04T09DBC0017."

Hero Award Winners Announced

Denise O'Donnell,
High Performance Computing Modernization Program Office

It is with great pride and enthusiasm that the program celebrates all of our Hero Award winners. We encourage them to keep up the good work, and we wish our Lifetime Achievement Award winners luck in their future endeavors. Since their inception in 2003, the annual High Performance Computing Modernization Program Hero Awards recognize outstanding contributions by individuals who support the overall mission of the Department of Defense (DoD) and the HPCMP.

Lifetime Achievement Award:

Awarded to a retiring member of the Program who has made significant contributions to the HPCMP and the Department of Defense throughout their time with the Program.

2017
Dr. Douglass Post

2018
David Cole,
(Navy DSRC)

Long Term Sustained Performance:

Awarded to an individual whose support provides a superior contribution to the Research, Development, Technology and Evaluation (RDT&E) community for the last five years or longer.

2017

David Dumas,
(ERDC)

Stephen Finn,
(DTRA)

Ann Ware,
(AFRL)

2018

Joe Lalosa,
(NAS Patuxent River)

Technical Excellence:

Awarded to an individual whose support demonstrates scientific or engineering excellence using HPCMP resources to advance creative and effective technology.

2017

Nicholas Bisek,
(AFRL)

Mehdi Ghorevshi,
(US Air Force Academy)

Phil Dykstra,
(Parsons Corporation)

2018

Joseph Ambrico,
(NS, Newport)

Tim Yeager,
(AFRL)

HPCMP Team Achievement:

Awarded to a team whose support has resulted in specific exemplary performance of importance to the HPCMP and the Department of Defense over the past year.

NAVAIR - 4.3.2.1 Fixed-Wing CFD Team (2017)

Shawn Woodson, Cholwon Paek & Dan Prosser, (US Navy)

This computational effort was accepted by the NAVAIR CFD FW team at the beginning of May 2017 and initial results for the baseline and the first wing-mounted store configuration were provided to the requesting Program Office within four weeks of the initial receipt of the geometry. The initial 24 baseline aircraft calculations and the 24 first-configuration store load-out calculations were all performed on the Thunder cluster at AFRL during a three-week time frame this summer despite the one-week maintenance shutdown which occurred in late May.

DREN DJS-SJS Migration Team (2017)

Rob Scott, Stephen Bowman & Ron Broersma, (SPAWAR-Pacific)
Mark Heck, (SPAWAR-Atlantic)
Stephen Anthony, Bradford Bloyer, James Cook, Aaron Dymond, Edward Kosiba, Kyle Krejci, & Zachary Thompson, (ASRC Federal Inuteq, LLC)
Jonathan Belsan, James Bray, Edward Jackson, & Bryan Keeler, (Parsons Corporation)
Jody Thomas, (GeoWireless)
Heather Elliott, & Eric Peterson, (Breakpoint Labs)

Starting on 21 March 2017, the DREN DJS/SJS Migration Team worked diligently to migrate 88 DJSs from RedHat (RHEL) 5 and Gator to RHEL 7 and Bro. This migration supported the HPCMP Security team's initiative to in-source the Cyber Security Service Provider (CSSP) functions. By the first week of April 2017, all 88 DJS devices were migrated, greatly exceeding all schedule expectations. To migrate the sensors on SDREN, a complete hardware infrastructure and support processes needed to be developed for the classified environment before any migrations could begin. Development of this infrastructure began the first week of April 2017. Starting on 01 May 2017, the migration of 76 SJSs to RHEL 7 and Bro began, and was completed the following week. The DJS/SJS Migration Team did a remarkable job completing the SJS migrations in approximately the same time as it took for the DJS migrations and significantly exceeding all schedule expectations. The entire project was completed in record time.

CREATE Ships RSDE Team (2018)

Keawe Van Eseltine, Brian Giang, Todd Heidenreich, Matthew Knobloch, Cathy Ngo, Cullen Sarles, Vu Trinh, Jason Kingsley, Nick Mullican, Tyler Weisbeck, Heather Barden, Amanda Kowalski, & Leighton Carden, (NSWC, Carderock)

Delivery of the ASSET-Submarine Design tool software architecture has been vital to the Navy's efforts to design two new classes of submarines. This high visibility effort was completed on a compressed timeline to meet Navy acquisition priorities. During the submarine tool development effort, the team managed to transition the tool's dated software architecture to a current one that is also used by the surface ship RSDE development project. This allowed for a reduction to the total cost and manpower - by several full-time equivalents (FTEs) - needed to update the software. They delivered a highly capable automated design space exploration (DSE) capability for submarines to support SSN(X) analysis of alternatives in FY2020. This effort required significant dedication and personal commitment to success of a demanding team effort.

Up & Coming within the HPCMP:

Awarded to an individual who has been with the program for two years or less whose support provides a distinct contribution to the HPCMP community.

2017
Abdul Williams, (Phacil, Inc.)

Miles McGee, (AFRL)

2018
Travis Utz, (SAIC)

Emily Heisler, (NSWC, Carderock)

HPCMP Interconnector:

Awarded to an individual whose support brings a group of people together to effectively solve HPCMP challenges

2017
William Hayes, (AFRL)

Kevin Schoen, (AFRL)

2018
Kathy Hollyer, (Navy, Patuxent River)

Innovative Practices:

Awarded to an individual whose support demonstrates creative business practices to improve the overall HPCMP business model.

2017
Melissa Nelson, (ERDC)

2018
Bill Culp, (SAIC)

Congratulations to all of our winners for their hard work and contributions to our Program. Keep up the good work!

HPCMP CREATE™ TEAM
Awarded at NDIA Conference

On October 24th 2017, the HPCMP CREATE Team was awarded the Lt. Gen. Thomas R. Ferguson, Jr. Systems Engineering Excellence Award for 2017 at the NDIA Systems Engineering conference. The award was accepted by Dr. Douglass Post, former Associate Director for CREATE, on behalf of the 180 government personnel and support contractors in the CREATE program.

The award criteria include promotion of robust systems engineering principles, effective systems engineering process development, increased mission capability and substantially increased performance. The award recognized the CREATE team's determined pursuit of a vision to enable fundamental change in defense acquisition via high performance computing and multi-disciplinary, physics-based software. The team's accomplishments in software engineering across multiple military and technical domains greatly improves DoD's ability to design and develop the advanced weapons systems to defend the nation.

The team's efforts and rigorous software development practices have provided the DoD with government-owned tools that enable cost, schedule, and risk reduction in weapons system acquisition, and improved warfighting capability.

Modules and bcmodule

Steven R. Thompson
SAIC, Army Research Laboratory,
DoD Supercomputing Resource Center,
Aberdeen Proving Ground, MD

The purpose of a module command is to configure the user's shell for running an application. The information in the shell helps users in locating executables, data files, and shared libraries that are required to run an application. A correctly configured shell enables a user to run an application efficiently and effectively.

The Baseline Configuration Team provides "bcmodule," a command which executes like the standard "module" command, but has numerous improvements and new features. Users can think of bcmodule as a total replacement for module, though technically it is a "wrapper" which calls "module." The two commands can be used interchangeably in the same shell, except when one of the new features of bcmodule is needed.

First we give an explanation of modules in general and then we discuss bcmodule in particular.

What are modules and what do they do?

The purpose of the module command is to establish the proper environment in the user's shell for running an application. The shell itself contains information in two types of variables that are called "shell variables" and "environment variables." The information in these variables is used to locate executables, data files and shared libraries. Having the shell environment variables set correctly can make the difference between being able to run an application and not.

A module command for an application typically augments the user's PATH variable so that the executable for that application can be located. It may also set an environment variable, such as LD_LIBRARY_PATH, which tells the operating system where to look for libraries the executable requires to run. It may set other environment variables that contain the location of required data files. Without modules, users would need to know where an application resides, how to add that path to the PATH variable, which libraries are needed, where they are, how to tell the operating system how it can find them, etc. So using modules makes computing much easier, even for experts.

How does one use modules?

The basics one needs to know about modules include the module command, the module sub-commands, module directories, and modulefiles.

Let's start with the modulefiles. Each version of an application, compiler, MPI, etc. will have its own modulefile, which is a file containing the information required to initialize the shell environment to run that version of the application, compiler, or MPI. To configure a shell, a user executes a "module load" on the modulefile. This means that the "module" command reads the modulefile, and modifies the shell accordingly. Strictly speaking, "module" is the command, but instead of speaking of "loading a modulefile," one usually speaks of "loading a module."

Before a user can load a module, the module command has to know where the modulefile is. To find a modulefile, a user's shell must be aware of the directory that the modulefile resides in. The module command finds a modulefile using the same method the shell uses to find a command: it searches along a path, which is an ordered list of directories, until it finds the file it is looking for. This path is stored in the shell as the environment variable MODULEPATH. You can see its value by executing "echo $MODULEPATH." To set this path to include "directory_1," one executes "module use directory_1." In executing this, the module command reads the current value of $MODULEPATH and adds "directory_1" to it, thus modifying your shell, which is now aware of one more directory where modulefiles can be found.

Now that the module command can find the modulefile, we are ready to have the modulefile modify a shell (i.e., to have it "work"). This is done by executing a "module load," such as "module load compiler/gcc/4.9.3." The "load" in this command is called the sub-command, and the modulefile is "compiler/gcc/4.9.3." The portion of the full path to the modulefile that precedes the modulefile name is the directory that was added to MODULEPATH earlier by "module use." To remove a directory from your MODULEPATH, execute "module unuse directory_2." This will make the modulefiles in "directory_2" disappear from your shell's point of view.

Module commands always have the form " module switches sub-command args." To say a module is loaded means that it is in effect (i.e., your shell is configured according to that modulefile). If a user wishes to run a different version of the same application, then he/she must "unload" the current module, and then execute a module load on the desired version. A module "unload" removes the additions to the shell environment that the corresponding module load had made. Executing "module swap" can be used to unload one module and load another in one command. A module can load one or more other modules.

Users typically have multiple modules loaded at any one time (just not multiple versions of a particular application). To see which modules are currently loaded, execute "module list." To see which modules one can load, execute "module avail." Executing "module use directory" as described above will make more modules available to your shell. To see what a module does to your shell, execute "module show <module_name>." To see a list of all "module" sub-commands execute module with no arguments, or "module --help" (notice there are two dashes).

A summary of the more important "module" subcommands is given in Figure 1.

The staff at the DSRCs maintain modulefiles, some of which come from vendors and some are generated locally. Users who run their own codes can even write their own modulefile if they wish. To do so, you may copy and then amend an existing modulefile, or ask for help from the HPC Help Desk.

The BCT command "bcmodule"

The command "bcmodule" works in all six shells supported by the HPCMP: sh, bash, ksh, zsh, csh, and tcsh. Both module and bcmodule should be initialized by default upon login. Execute "bcmodule" to see the help menu of sub-commands. Since bcmodule can be thought of as a replacement for module, it can be used to list the modules that are currently loaded. Figure 2 illustrates an example, using the C-shell (csh or tcsh). Notice that sub-command "list" has been abbreviated to "li"; such abbreviations are optional.

Module commands that modify the shell:

module use mod_directory	add a directory to $MODULEPATH
module unuse mod_directory	remove a directory from $MODULEPATH
module load modulefile	modifies the shell to run an application
module unload modulefile	undoes what the module load did
module swap mod_a mod_b	unload module mod_a and load mod_b

Module commands that provide information without modifying the shell:

module show module_name	show what the module does to your shell
module list	list which modules are currently loaded
module avail	list which modules could be loaded

Figure 1:
The most commonly used module subcommands.

```
1 centennial01> bcmodule li
Currently Loaded Modulefiles:

1) compiler/intel/2017.1.132      2) mpi/sgimpt/2.15      3) Master
```

Figure 2:
The "list" subcommand, abbreviated "li," shows loaded modules.

```
2 excalibur05> bcmodule show Master
/usr/cta/modules/3.2.10.1/modulefiles/Master:

   setenv          BC_HOST excalibur
   setenv          BC_CORES_PER_NODE 32
   setenv          MODULE_VERSON 3.2.7
   setenv          CSI_HOME /usr/cta
   setenv          DAAC_HOME /usr/cta/DAAC
   setenv          PET_HOME /usr/cta/pet
   setenv          WORKDIR /p/work1/thompson
   setenv          SAMPLES_HOME /usr/cta/SCR
   setenv          CENTER /p/cwfs/thompson
   setenv          MPSCP_CONFIG_FILENAME /etc/mpscp_config
   setenv          MPSCP_BLOCKED_FILE /etc/mpscp_blocked_ports
   setenv          PROJECTS_HOME /usr/cta/unsupported
   setenv          KRB5HOME /usr/krb5/bin
   setenv          KRB5_HOME /usr/krb5/bin
   setenv          ARCHIVE_HOME /admin/thompson
   setenv          ARCHIVE_HOST msas23.arl.hpc.mil
   prepend-path    MANPATH /usr/share/man:/usr/local/man
   prepend-path    PATH /usr/local/bin:/usr/krb5/bin
   prepend-path    PATH /app/unsupported/BC
   prepend-path    PATH /usr/people/PBS/SLB

3 excalibur05>
```

Figure 3:
This module sets 16 environment variables and augments two paths.

So three modules are loaded. To see what a particular module does to your shell, execute the "show" sub-command as shown in Figure 3.

The first line (starting with /usr/cta) shows where the "Master" modulefile is located. This is followed by 16 "setenv" lines, each of which sets the value of an environment variable. When you run a Linux command or execute an application, it can extract the value of an environment variable defined in the shell. For example, when the command "archive" is executed to archive a file, the archive command knows where to copy the file based on the shell's value of the variable "ARCHIVE_HOME," which is the personal directory "/admin/thompson." In this case, the Master module sets a unique value for ARCHIVE_HOME for each user, so that one user's files are not archived into another user's directory.

The last four lines of the module modify an existing shell variable. The first "prepend-path" module statement prepends two directories (/usr/share/man and /usr/local/man) to the variable MANPATH, which is used in finding "man" pages (Linux online documentation). The last three lines each insert one or two directories at the beginning of the variable "PATH," which is used to find Linux commands, as well as application scripts and executables.

Here is a list of improvements provided by "bcmodule" that are not available in "module":

1. It translates native modulefile names into BCT-specified modulefile names for greater uniformity of modulefile names across different systems in the HPCMP. Users can use either the BCT modulefile name or the native modulefile name interchangeably. Two new sub-commands (tran and show_all_trans) show the translation of modulefile names from native to BCT and vice-versa, as illustrated in Figure 4.

2. It returns non-zero return codes when errors occur.

3. Output from bcmodule goes to stdout, which allows users to pipe the output into subsequent Linux commands such as "grep" or "more," as illustrated in Figure 5.

This does not work with "module," since it writes its output to standard error. Each line of output from the bcmodule command above has perftools in it, but it also contains modules that have nothing to do with perftools. This is because "grep" is grabbing entire lines. Next we will show a better bcmodule feature for seeing only the perftools related modules.

4. A new sub-command "find" searches through all known modulefile directories (not just those in use) and lists all modulefile names it finds that contain the given string searched for, thus helping users find modulefiles when the exact name is uncertain and/or the location is unknown. An illustration of the sub-command "find" is included in Figure 6.

Here the output format is a bit different, but all the information you need to locate the perftools modules is given. And the timestamps of the files are also given, which is not the case for "module" commands.

5. A new sub-command "find_in_module" searches for a modulefile which contains the given string, thus locating the modulefile not by a string in its name (the sub-command "find" described above does that), but by finding a modulefile that performs any action involving the string. For example, "bcmodule find_in_module XYZ" finds any modulefile that sets the environment variable XYZ, or that sets any environment variable with XYZ as part of the name, or augments any PATH with a directory having the pattern XYZ, etc.

An illustration of the subcommand "find_in_module" is included in Figure 7. Each modulefile listed has the string "ARCH" somewhere in its contents.

6. A new sub-command "inuse" lists all module directories currently in use.

7. A new sub-command "show_all_dirs" lists all known module directories so that users can easily add module directories for greater access to modulefiles.

8. Even easier, the new sub-command "use_all_dirs" adds all known module directories so that users automatically use all configured module directories when loading and listing available modules.

Figure 8 demonstrates how the number of modules available increases (from 210 lines of output to 266 lines) when all (system configured) modulefile directories are added using the "use_all_dirs" option.

These three commands show that by executing "bcmodule use_all_dirs," the list of module directories known to the shell increases such that an additional 56 lines worth of modules are now locatable, and can thus be loaded. The "inuse" and "show_all_dirs" sub-commands mentioned above can be used to see those directories.

The term "system configured" used above refers to the commonly used module directories configured for bcmodule by the system admin, but not directories containing only modules for certain projects. Users who wish to always have their project specific module directory used can do so by including a "module use" command in their personal shell initialization script.


```
8 excalibur05> bcmodule tran compiler/gcc/6.2.0
real module name is gcc/6.2.0
BCT module name is compiler/gcc/6.2.0
9 excalibur05>
```

Figure 4 (left):
The "tran" subcommand shows translation of a modulefile name.

```
9 excalibur05> bcmodule avail | grep perftools
cray-petsc-complex-64/3.6.3.0   perftools-base/6.3.0
cray-petsc-complex-64/3.7.4.0   perftools-base/6.3.2
cray-petsc/3.6.3.0              perftools/6.4.4
10 excalibur05>
```

Figure 5 (above):
Output from bcmodule can be piped into subsequent Linux commands.

```
12 excalibur05> bcmodule find perftools
/opt/cray/modulefiles/perftools-lite
-rwxr-xr-x 1 root root 7794 Jan 15  2015 6.2.2

-rw-r--r-- 1 root root 1506 Jan 25 10:11 6.4.4
/opt/cray/modulefiles/perftools-base
-rw-r--r-- 1 root root 7707 Oct 19  2015 6.3.0

/opt/cray/modulefiles/perftools
-rwxr-xr-x 1 root root 7794 Jan 15  2015 6.2.2

13 excalibur05>
```

Figure 6 (left):
The "find" subcommand shows all modules with pattern in the name.

Figure 7 (right):
This subcommand finds modules based on content of the file, not its name.

```
13 excalibur05> bcmodule find_in_module ARCH
/usr/cta/modules/3.2.10.1/modulefiles/Master
/usr/cta/modules/3.2.10.1/unsupported/moose/dev-gcc
/usr/cta/modules/3.2.10.1/COTS/fluent/171

plus about 50 more
```

```
14 excalibur05> bcmodule avail | wc –l
  210
15 excalibur05> bcmodule use_all_dirs
16 excalibur05> bcmodule avail | wc –l
  266
17 excalibur05>
```

Figure 8 (left):
The "use_all_dirs" subcommand maximizes availability of modules.

```
excalibur05>  module avail
------------------------------------------ /opt/modulefiles ------------------------------------------

cce/8.3.14(default)                        intel/14.0.2.144
cce/8.3.5                                  intel/15.0.1.133
cce/8.3.6                                  intel/16.0.0.109

chapel/1.9.0.1(default)                    modules/3.2.6.7
ddt/4.2.2.6_39982(default)                 pbs
ddt/5.0.1.3_42607                          pgi/14.10.0
ddt-memdebug/4.2.2.6_39982(default)        pgi/14.10.0-acc
ddt-memdebug/5.0.1.3_42607                 pgi/15.3.0(default)
eswrap/1.3.3-1.020200.1278.0(default)      pgi/15.3.0-acc
gcc/4.8.1                                  pgi/15.7.0

gcc/6.2.0

excalibur05> bcmodule avail –compress

------------------------------------------ /opt/modulefiles ------------------------------------------
cce  compiler/gcc compiler/java compiler/pgi ddt ddt-memdebug eswrap modules pbs xc-sysroot
chapel compiler/intel
```

Figure 9 (above):
The "-compress" flag condenses output by dropping versions.

```
24 excalibur05> bcmodule av -sf perftool
------------------------------------------ /opt/cray/modulefiles ------------------------------------------
perftools-base/6.3.0  perftools-lite/6.2.5(default)  perftools-lite/6.4.4  perftools/6.3.0
perftools-base/6.3.2  perftools-lite/6.3.0           perftools/6.2.2       perftools/6.3.2
perftools-base/6.4.4  perftools-lite/6.3.2           perftools/6.2.5(default)  perftools/6.4.4
perftools-lite/6.2.2
25 excalibur05>
```

Figure 10 (above):
This option is similar to "find."
The difference is explained below.

Acknowledgement
The author would like to express his sincere thanks and appreciation to Mr. Bob Price from the Corporate Media Department at SSC Pacific, San Diego.

9. A new sub-command "count" counts modulefiles and modulefile directories currently used.

10. A new switch (-set_gcc) loads the gcc module when a compiler module is loaded (this facilitates compiling under certain circumstances).

11. A new switch (-set_gxx) sets the environment variable GXX_ROOT when a compiler module is loaded (this facilitates compiling under certain circumstances).

Tired of getting too much output from "module avail"? The new bcmodule features "find", "-compress", "-sf", "-xf", "-sd", and "-xd" all make finding modulefiles much easier.

12. A new option for module avail: "-compress" removes the module version and lists each "family" of modules once instead of once per version. On Excalibur, this reduced the number of lines of output from 202 to 51.

Figure 9 includes an example where "module avail," for the directory /opt/modulefiles, gives 21 lines of output whereas the "-compress" option in bcmodule shortens the output to just two lines:

13. New options for module avail and module list that reduce the amount of output:
a. "-sf pattern" lists only modulefiles having "pattern" in the name.
b. "-xf pattern" lists only modulefiles not having "pattern" in the modulefile name.
An illustration of the sub-command "av" with option "-sf" is in Figure 10.

14. New options for module avail:
a. "-sd dir" lists only modulefiles in directories having "dir" as part of the directory name.
b. "-xd dir" lists only modulefiles in directories not having "dir" as part of the directory name.
The two commands "bcmodule find pattern" and "module avail –sf pattern" are very similar, but not identical. Their output formats are different, but more importantly, "find" searches through all configured modulefile directories, whereas the second command only searches through directories you have "in use."
The new bcmodule does a slightly better job than module of "cramming" as many columns of output as it can to your screen to reduce the number of lines of output. In the process of reformatting the output, it sometimes alphabetizes the modulefiles slightly differently.

Happy moduling!

DoD HPC INSIGHTS

is a semiannual publication of the
High Performance Computing Modernization Program.

Director
Will McMahon

Managing Editor
Denise O'Donnell

Design / Layout
Ron Compton

Questions or Comments
hpcmp@hpc.mil

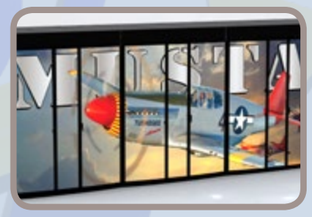
Follow us on Twitter
@DoD_HPCMP

DOD HPC MODERNIZATION PROGRAM SYSTEMS

The High Performance Computing Modernization Program (HPCMP)
provides over 900,000 cores and 45 petaFLOPS
to its users in the pursuit of improved scientific research, weapons design,
force protection, and software development for the Department of Defense

AFRL DSRC

US Air Force Research Laboratory
Wright-Patterson AFB, Ohio
https://www.afrl.hpc.mil



MUSTANG
HPE SGI 8600

Service Date:	12/12/2018
Processor Cores	56,448
Memory (GB)	224,608
Disk (TB)	13,120
GPGPUs	24
teraFLOPS	5004

THUNDER
SGI ICE X

Service Date:	09/08/2015
Processor Cores	125,888
Memory (GB)	460,288
Disk (TB)	17,568
GPGPUs	356
Coprocessors	356
teraFLOPS	5,620

ERDC DSRC

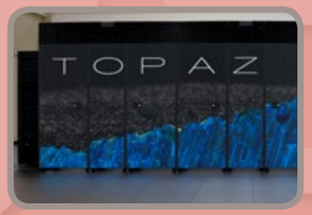
US Army Engineer Research &
Development Center
Vicksburg, Mississippi
https://www.erdc.hpc.mil

COPPER
Cray XE6m

Service Date:	11/19/2012
Processor Cores	14,975
Memory (GB)	29,250
Disk (TB)	443
teraFLOPS	139

ONYX
Cray XC40

Service Date:	05/18/2017
Processor Cores	214,568
Memory (GB)	632,064
Disk (TB)	21,280
GPGPUs	32
KNLs	32
teraFLOPS	7,736



TOPAZ
SGI ICE X

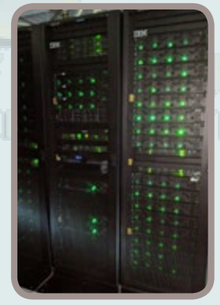
Service Date:	07/01/2015
Processor Cores	125,440
Memory (GB)	443,584
Disk (TB)	17,496
GPGPUs	32
teraFLOPS	4,662

MHPCC DSRC

Maui High Performance
Computing Center
Kihei, Maui, Hawaii
https://www.mhpcc.hpc.mil

HÓKÚLE'A
IBM P8+

Service Date:	03/20/2017
Processor Cores	640
Memory (GB)	10,249
Disk (TB)	359
GPGPUs	128
teraFLOPS	693



ARL DSRC

US Army Research Laboratory
Aberdeen Proving Ground, Maryland
https://www.arl.hpc.mil

CENTENNIAL
SGI ICE XA

Service Date:	06/22/2017
Processor Cores	73,920
Memory (GB)	252,928
Disk (TB)	16,800
GPGPUs	32
teraFLOPS	2,600

EXCALIBUR
Cray XC40

Service Date	04/13/2015
Processor Cores	101,184
Memory (GB)	404,736
Disk (TB)	6,055
GPGPUs	32
teraFLOPS	3,769

NAVY DSRC

Navy DoD Supercomputing
Resource Center
Stennis Space Center, Mississippi
https://www.navo.hpc.mil

GAFFNEY & KOEHR
HPE SGI 8600

Service Date	11/08/2018
Processor Cores	35,328
Memory (GB)	153,856
Disk (TB)	7,914
GPGPUs	16
teraFLOPS	3,137

CONRAD & GORDON
Cray XC40

Service Date	06/19/2015
Processor Cores	61,088
Memory (GB)	204,544
Disk (TB)	1,560
Coprocessors	168
teraFLOPS	2011

POWERFUL RESOURCES

The High Performance Computing Modernization Program (HPCMP) provides the Department of Defense supercomputing capabilities, high-speed network communications, and computational science expertise that enable DoD scientists and engineers to conduct a wide range of focused research, development, and test activities. This partnership puts advanced technology in the hands of US forces more quickly, less expensively, and with greater certainty of success.

Today, the HPCMP provides a complete advanced computing environment for the DoD that includes unique expertise in software development and system design, powerful high performance computing systems, and a premier wide-area research network. The HPCMP is managed on behalf of the Department of Defense by the US Army Engineer Research and Development Center.

To learn more about the HPCMP, please visit our website at: hpc.mil.

To learn more about our High Performance Computing (HPC) centers and HPC Systems, please visit: centers.hpc.mil.

HPCMP User Help Desk
1.877.222.2039
centers.hpc.mil

AFRL DSRC

US Air Force Research Laboratory
Wright-Patterson AFB, Ohio
https://www.afrl.hpc.mil

ARL DSRC

US Army Research Laboratory
Aberdeen Proving Ground, Maryland
https://www.arl.hpc.mil

ERDC DSRC

US Army Engineer Research & Development Center
Vicksburg, Mississippi
https://www.erdc.hpc.mil

NAVY DSRC

Navy DoD Supercomputing Resource Center
Stennis Space Center, Mississippi
https://www.navo.hpc.mil

MHPCC DSRC

Maui High Performance Computing Center
Kihei, Maui, Hawaii
https://www.mhpcc.hpc.mil