SAND2013-5334P

Project Grandmaster Final Report

Nathan Fabian*, Warren Davis*, Jonathon T. (JT) Mcclain*, Derek Trumbo*

List of Figures

List of Tables

1 Introduction

People use social media resources like Twitter and Facebook to share and discuss various activities or topics they are interested in talking about. Resources like LinkedIn reflect a person's accomplishments as he or she progresses through his or her career. By combining these sources of data about individuals and aggregating trends across many individuals using these services, it may be possible to construct a rich portfolio of a person's activities and interests as well as provide a broader context of those activities.

*{ndfabia,wldavis,jtmccl,dtrumbo}@sandia.gov

We access these social media sources in order to examine the data stored at each site. These services provide API's to external applications through an opt-in procedure that allows pulling out and individual's data to be processed by the external application. Although there is a rich set of data in the link graph of associations between users using the system, there are many existing approaches to analyzing the graph. Instead our approach considers that much of this data will be unstructured, free-form text. By analyzing this free-form text directly, we may be able to gain an implicit grouping of individuals with shared interests based on shared conversation, and not as necessarily on explicit linking between them.

In this report, we discuss an application we have developed, called Project Grandmaster. It has been built to pull a person's social media data together, and provide analysis and allowing visual exploration, summarization and understanding of the data in total. Using text analysis algorithms previously developed in Titan[9] and web-crawling technology developed in Avondale[2], we pull the data in and process it. On top of that, we developed custom visualizations to show groupings of individuals, allowing an aggregate understanding of a group by reinforcing and amplifying patterns within, identifying a stereotypical group identity. This can further feed into a learning system by ascribing properties to these stereotypes, and then by determining how new individuals align with these stereotypes infer those properties to the new individuals.

Project Grandmaster works by allowing people to opt-in and give access to their publicly available social data sources. Although it requires authorization in some instances to access the APIs, the data it gathers is only that which is already available to anyone through regular web access. For instance, LinkedIn provides a public profile that is available to anyone who has the URL. We do not use crawling to find these URLs; instead we require that an individual explicitly add their links in order for the system to download the data.

It is important to note that although the data is

from an individual, it is the data in aggregate which is important to the results. We are not making a personality profile for an individual[7, 8], instead we consider only stereotypical behaviors for a group. The association of individuals with groups is probabilistic and can only infer qualities, not to guarantee their existence. In addition, we make the effort to allow an individual control over his or her data. He or she may add data and also delete it at any point. The system will reprocess to remove results pertaining to removed data.

In the remainder of this report we will discuss the implementation of the application, how we collect and process the data, and the algorithms involved in doing so. We show our proof-of-concept results with a bootstrap data set. Finally, we discuss future steps that could be taken to use the results presented here to build learner profiles for automated learning, as well as discuss ways we might continue to improve the results themselves.

2 Implementation

Project Grandmaster is an open source, web-centric application. The data collection and processing happen in the background and update a Mongo database. The visualization code runs as part of a webpage that queries new data each time the page refreshes and builds the visualizations from that.

The documents and profiles are pulled from each of the social sites and treated as a collection of the smallest atomic units from each site. For instance, from LinkedIn we treat a profile as a collection of work experiences, education backgrounds, a summary section, etc. Each of these pieces is a separate document, but each tied to the original profile. Similarly the Twitter data is composed of individual tweets. Each of these is a separate document also associated with the individual. Therefore the final data representation for an individual is a collection of documents from all the data sources involved. This allows us to cluster each of these documents independently for content and find a range of possible document clusters for the individual, reflecting a diversity of categorical content with which the person should be associated.

At a high level, Project Grandmaster is made up of three main parts: A web crawler, an analysis pipeline, and a visualization suite. The web crawling framework collects the data and writes it out to a Mongo database for further processing by the analysis pipeline. It is implemented in Avondale, which has a plugin framework to allow specialized access to the various social media APIs. It is described in more detail in Section 2.1. The analysis pipeline is handled in large part by the Titan Toolkit. We access Titan through a collection of Python scripts and store partial results at each step out to a Mongo database for display by the visualization code. The analysis pipeline is described in more detail in Section 2.2. Finally, the visualizations were developed using D3, a JavaScript based visualization library, and are described in Section 2.3.

2.1 Capturing Social Media Data with Avondale

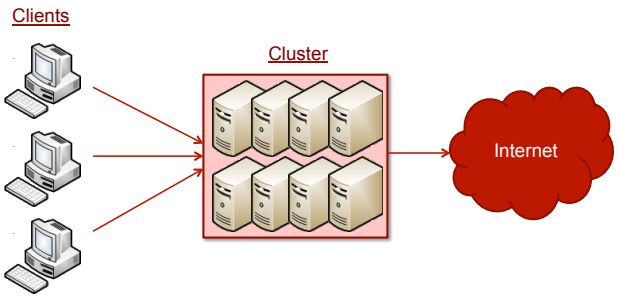

Figure 1: Web crawling framework. The framework controls a single crawl across multiple jobs on multiple nodes.

Avondale was originally written to crawl the web searching for documents. It was architected as cluster of computers each running a crawler, Figure 1, and each storing the results to the database. This allows it to scale to crawl a very large number of pages very quickly.

As part of this effort, we rearchitected Avondale to allow plugins to the framework to crawl the more specialized APIs of the social media sites, Figure 2. This allows the plugins to run on the same cluster infrastructure as the web crawling allowing for potentially large scale social media crawling as well. For this work, we use only one node.

The social media crawl exists as a single long running job on one or more nodes. When a new user joins the system, their data will be passed into the job through a web communication back-end. The job runs each of the plugins at a regular interval specified by the individual plugin type. This allows the system to correctly throttle API queries based on the limitations specified by each source. For instance, Twitter only allows 200 API calls every 15 minutes.

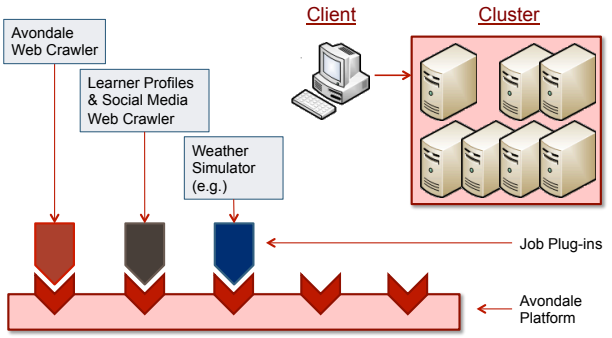

Figure 2: The Avondale plugin structure allow code specialized for particular web APIs, such as the social media used here, to run in the same multi-node framework as the original web crawl.

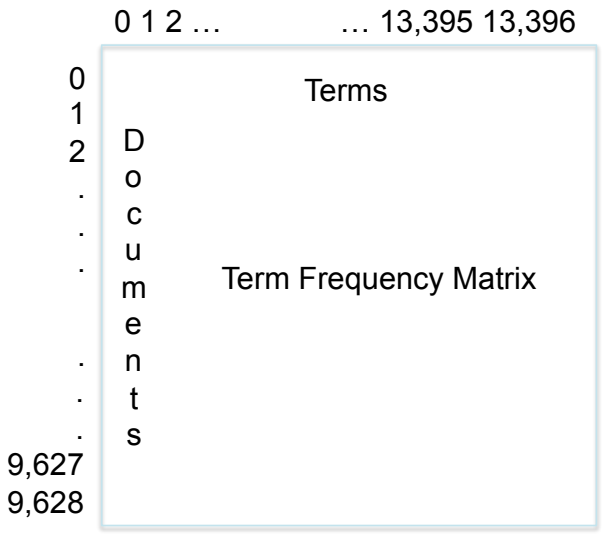

Figure 3: A spare term frequency matrix representing the count of each term (horizontal axis) in a document (vertical axis). This matrix consists mostly of 0s with scattered 1s and one or two places with higher numbers.

When new user data comes in, each social media URL is added to each individual crawler plugin. When that crawler plugin next runs it will check if there is new or updated information for its existing individuals and then start downloading the data from the new individual. It maintains a state about each individual it knows about and can persist this state out to disk in case of shutdown. This allows the system to only grab the data which it has not already downloaded, as well as pause the downloading at any point in order to obey the throttling limitation of the site's API.

As this data is collected, it is stored into the Mongo database, associating each document with the individual who created it. In a separate process, the analysis pipeline checks will reprocess the data, as described in Section 2.2.

2.2 The Data Analysis Pipeline

Once the data is stored into the database by Avondale, the analysis pipeline commences in processing the records. Although it is possible to process the data incrementally, in this proof-of-concept version, we simply reprocess the entire collection each time we want to update the data set. The processing is broken up into four distinct algorithms: Latent Dirichlet Allocation (LDA), clustering of documents, word cloud preprocessing, and user clustering.

We use a version of LDA known as Parallel Latent Dirichlet Allocation (pLDA) [1, 3, 4]. pLDA works from a corpus of documents represented as a term frequency count matrix, Figure 3. In our case the corpus of documents are a collection of the atomic components of each social media site, e.g., tweets, work experiences, summaries, etc, for the entire collection of individuals.

From this complete corpus of all individuals, pLDA constructs a set of topics representative of the whole set of documents, Figure 4. Each topic is a collection of weighted terms and each document is a weighted set of topics. Essentially this clusters terms into collections of synonyms and distinguishes homonyms into separate topics. This reduces dimensionality; instead of treating documents as a collection of individual terms, potentially numbering in the thousands or millions as in Figure 3, we reduce the documents to a collection of topics numbers in the tens or possibly hundreds at most.

By placing all documents into one unified lower dimensional space, we will be able to compare documents which use synonymous words and contrast those which use words in different contexts. Figure 5 shows the documents assigned to two different users, a gamer and a senator. Although the lower dimension of 27 is easier to work with than $13,396$, we still have some challenges remaining in comparing these two individuals. We could compare each individual document by assigning a numeric distance metric, in our case cosine similarity, in topic space between the documents. However, this would result in D^2 com-

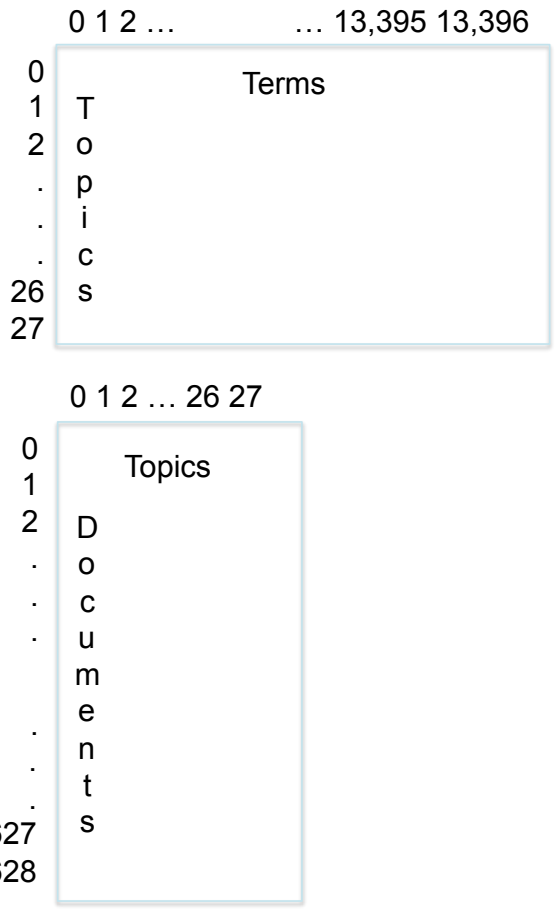

Figure 4: The two matrices produces as a result of processing through LDA. Both are probability matrices, the top gives a probability distribution for each topic across all the terms, see Table 1 in Parameterization, Section 3.1. The bottom gives a probability distribution for each document across all the topics.

parisons where, D is the number of documents. This would not scale well.

To counter this scaling issue, we aggregate the documents down to a single vector for each user. One approach, would be to average the term distributions across all documents for that user and then use cosine similarity to compare the final vectors. However, averaging tends to squash out a lot of the diversity contained in an individuals documents, see Figure 6. We will show later the importance of maintaining these individualities in the examples shown in Section 3.2. Thus, our approach is to cluster the documents first, and then use those clusters to project a user vector that highlights the variety of topics the user engages.

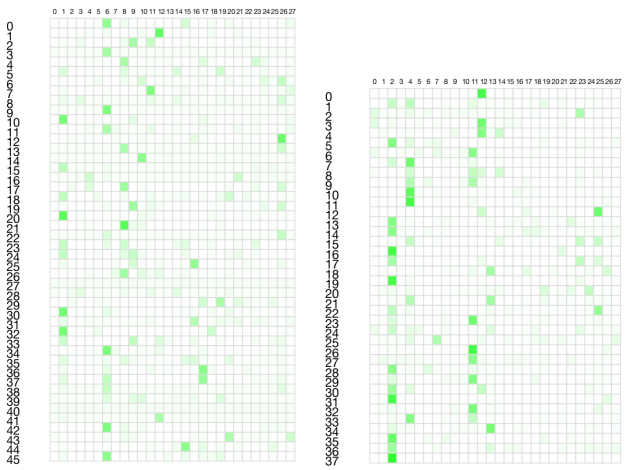

Figure 5: Document distribution for a collection of documents assigned to two users, a gamer and a senator. Brighter green in a column means higher probability of that topic in the document at that row.

Figure 6: Averaged topic vectors from the two individuals in Figure 5. Averaging tends to squash the topic distribution, removing diversity which would be useful later for finding distinctions between individuals.

Using this distance metric we can use K-Means clustering to group the documents into a set of K distinct clusters. K-Means clustering works by taking a parameter K and create a set of K initial cluster centers. Each document is assigned to the closest cluster center using the distance metric. Once each document is assigned, the centers are recomputed as the mean value of all the documents assigned. The algorithm iterates back and forth between these two states until it converges and no document further changes cluster assignments. Figure 7 shows the averaged topic distribution for each document cluster. Note, how the topic vector is reinforced by the similarity instead of squashed in diversity. We can then understand the diversity of an individual by how they project across these different clusters.

Now, we are able to treat these document clusters as vectors of document counts for each person. For each document a person has that is assigned to

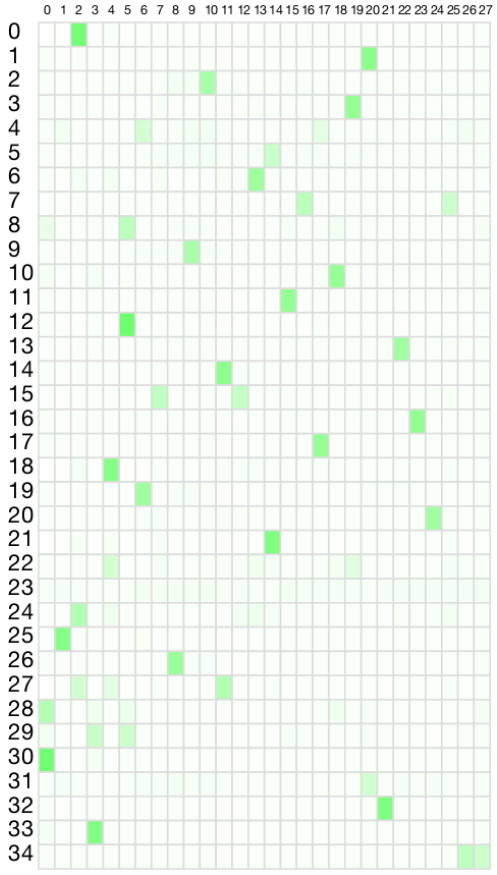

Figure 7: Averaging the topic vectors over a clustered group of documents no longer squashes the vector as it did in Figure 6. Instead certain subsets of topics are highlighted and reinforced in each document cluster

.

a particular cluster, that person's document vectors gets a weighted increment in the associated index, see Figure 8. The weighted increment is based on how close to the center the document is, documents will fall between 0.5 and 1.0 in this scale. Figure 9 shows the resulting document vector for the two individuals from Figure 5. Using this new vector space for individuals, we once again create a distance metric and cluster the individuals using K-means.

Finally we preprocess each document cluster for a word cloud visualization[5]. A word cloud, also known as tag cloud, is a useful visual summarization of a collection of documents. It scales up words which are frequently used in the collection and scales down words which are infrequently used. There are many variations of word clouds and many ways to sort. In our case, we use an arbitrary spatial arrangement and

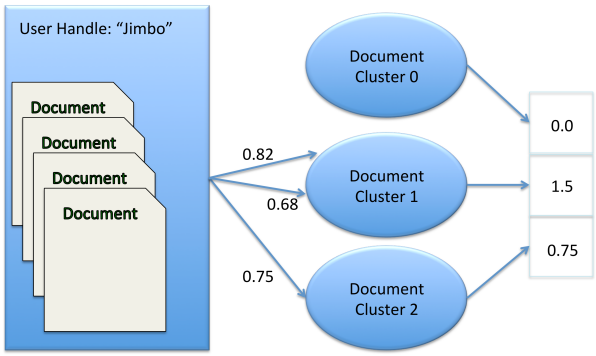

Figure 8: The creation of a document vector for a user using the assignment of the user's documents into document clusters.

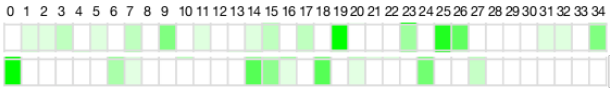

Figure 9: User vectors produced by projecting a users documents through the document clusters as shown in Figure 8.

use only scale to communicate the frequency. The preprocessing step involves counting terms in each document cluster to determine the count for the visualization to use for later scaling. Note, that this frequency count is entirely independent of the topic weighting. We can use this as a separate verification of sense-making in the pLDA/K-means clustering processes.

The result of each of these steps, i.e., LDA, word cloud and both clusterings, is stored back into the Mongo database to enable the visualization.

2.3 Data Representations, Visualizations, and User Interactions

The two main data representations we need to visualize are the user clusters and document clusters described in Section 2.2. This is represented by two sets of bubbles in the main view of the application, Figure 10. The visualizations are developed through a web-based interface using the Javascript library D3 to do the main part of the visual control. The cluster data is read from the Mongo database server-side and passed to for client-side rendering and interaction.

On the left side of the main view are the user clusters. These are organized into a force-directed layout with the cluster centroids represented as rings and

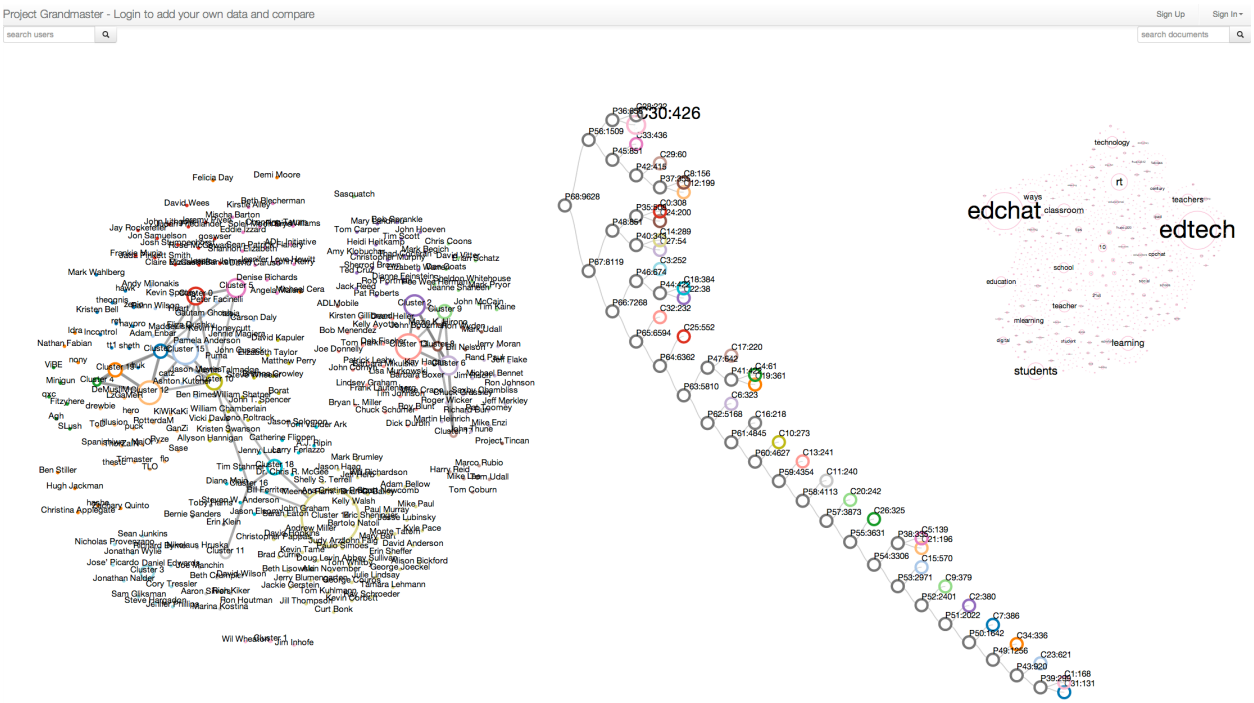

Figure 10: Main view of Project Grandmaster. On the left are the user cluster. In the middle are the hierarchically organized document clusters. On the right is the word cloud associated with the selected document cluster.

individuals represented as points. In a force-directed layout, all items have an implicit repelling force from each other. There is only an attractive force between an individual's point and it's cluster center. There is a secondary force between two cluster centers, represented with a semi-transparent gray line, if the two cluster centers are on average similar to each other, i.e., the individuals contained in the cluster are similar to the individuals in the other cluster, and that similarity is above some threshold. This allows us to understand a two-layer of hierarchical arrangment between the cluster centers and the individuals.

In the middle view, the document clusters are organized into a full hierarchical agglomerative clustering (HAC). The initial clustering is done for all documents using K-means for a given K, here 50. Then the HAC procedure finds the nearest two clusters and merges them into a single parent cluster, replacing the two with this parent in the set. It then iterates this procedure, replacing the nearest two clusters, one or both of which could be a previously merged cluster, until only one cluster remains at the root. Using this hierarchical arrangements we can understand groupings of topics and sub-topics within the document collection. The labels assigned to the document clusters are either "P", representing parent, and a number or "C", representing original cluster, and a number. The number is a unique identifier for each cluster or parent. The number after the colon represents the number of documents contained in that cluster.

On the right hand side of the view is the bubble word cloud associated with the highlighted document cluster. As described previously, in Section 2.2, the word cloud is generated by simply counting term frequencies in the document cluster. The terms that are easiest to see are the ones which are most frequent. There is no meaning to the spatial arrangement only to the size. The purpose of this display is to quickly summarize the content of potentially hundreds of documents contained in the cluster.

The view updates as a user of the application moves the mouse over various sections of the display. As he or she hovers over a user cluster, the other clusters fade out to emphasize the cluster in focus. Blue lines are connected from the user cluster to the various document clusters the users in that cluster are talking about. The document cluster with the highest connection to that user cluster is highlighted and its

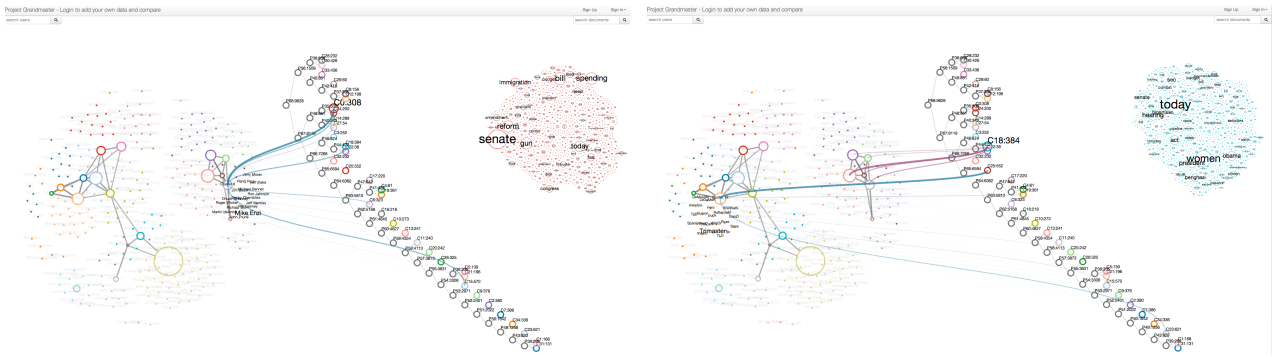

Figure 11: The main view changes to highlight the group being hovered over with the mouse. Blue lines, Figure 11a represent the various document clusters the highlighted user cluster is talking about. As a user hovers over document clusters, Figure 11b, the cluster is highlighted and the associated word cloud is shown. Purple lines are extended out to the user clusters talking about this set of documents.

word cloud is displayed. This is shown in Figure 11a.

In addition to hovering over user clusters, a user may also hover over the document clusters directly. This will highlight the document cluster being hovered and bring up its word cloud. It will also extend purple lines from the document cluster to each of the user clusters talking about this particular document collection. This is shown in Figure 11b. Note that neither the blue lines or the purple lines disappear in order to maintain context as the user is moving around in the view.

Finally there are two search boxes on the upper left and upper right. The search box on the upper left allows one to find users either by partially matching the name or complete matching. Users are highlighted by creating a large blue dot to replace the normally small dot connected with individuals. This can also be seen in Figure 11b. In the Figure 12, the other search box will highlight a document cluster. The cluster is highlights is the one with the highest frequency occurrence of the term in the search box. This search is an exact match. As before, highlighting the document cluster also brings up the associated word cloud and the purple lines back to the user clusters.

All of these interactions allow a user of the application to search through all of the data in the collection and globally understand this data in terms of which groups the users have collected into and what sorts of things they are talking about. If the he or she has a targeted query, such as a particular person or term, they can search for that directly.

A final view is shown in Figure 13. This is an early step at making automated recommendations to a user

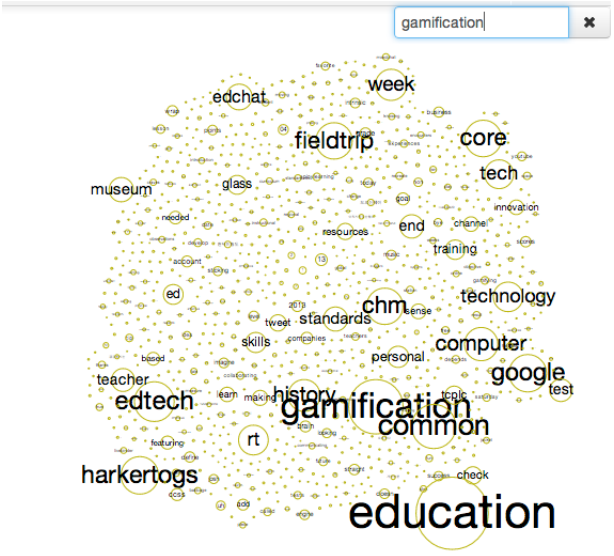

Figure 12: Searching for a term in the upper right box brings up the document cluster where that term has the highest frequency occurence among the clustered documents.

based on the processing we are doing here. This view shows the nearest user and most average user within the cluster of a queried individual. It also shows a number of web links which may be of interest to the queried user. The links are sorted by the proximity between the posting user and the queried user. The proximity is the same calculation used to cluster the individuals, as described in Section 2.2.

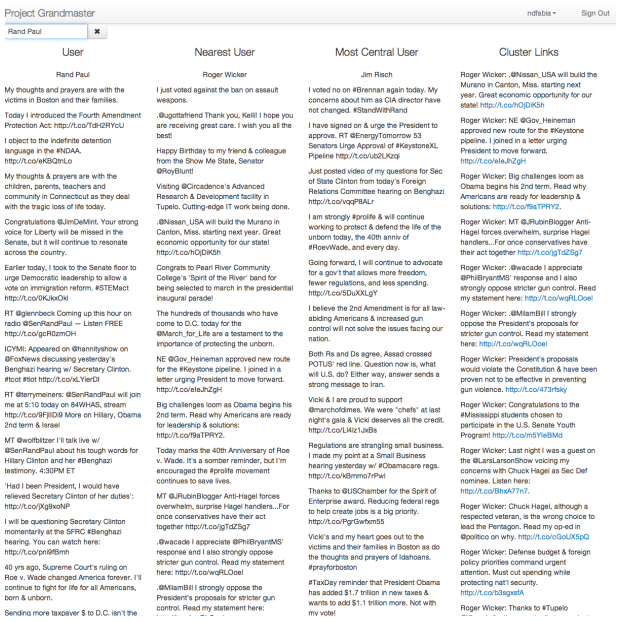

Figure 13: An experimental user focused view. It shows the searched user's data for comparison. The nearest user within the cluster. The most average user in the cluster. And a list of html links sorted by the closeness of the user posting them to the user queried.

0	edchat	edtech	students	rt
1	streaming	stream	hots	going
2	senate	bill	reform	immigration
3	elearning	edtech	learning	online
4	today	women	act	hearing
5	edtech	edchat	google	ipad
6	thanks	lol	love	don
7	today	school	thanks	dc
8	thanks	don	today	fun
9	follow	rt	sure	thank
10	thanks	thank	love	rt
11	live	ll	tune	watch
12	boston	today	families	thoughts
13	jobs	tax	help	health
14	forward	looking	look	working
15	help	thanks	rt	awesome
16	happy	birthday	hope	thanks
17	2	1	4	3
18	education	common	rt	gamification
19	today	congrats	vote	2013
20	think	working	writing	read
21	today	top	stories	¿
22	re	going	trying	ve
23	news	rt	didn	senator
24	american	stop	tonight	center
25	today	support	air	honor
26	lol	online	internet	account
27	week	info	check	call

Table 1: This table shows the top four highest probability terms in each of the 28 topics used for the proof-of-concept data set.

3 Results

The data set we chose to bootstrap the proof-of-concept consists of the public data of professional Starcraft players, United States senators, E-learning experts and Hollywood celebrities. Because they did not opt-in as we would normally operate, we chose them based on the public nature of their roles and selected only the data they explicitly made public. We have a collection of 274 individuals and $9,628$ tweets. It consists of an arbitrary sampling of a few days between the last few months of 2012 and first few months of 2013.

3.1 Parameterization

In order to parameterize the processing, we began by experimentally finding a number of topics to pass to pLDA which produced topic distributions that made sense. We have found through the experimentation that the system works better with as few topics as we can reasonably choose. In addition to the number of topics, we also have to choose values for α and β inputs to pLDA. For β we chose 0.01 as recommended by Steyvers and Griffiths[6]. However, while they recommend setting α to $50/t$ where t is number of topics, we've found that because tweets are small they are less likely to be distributed over as many topics and therefore use $2/t$ here. The resulting set of topics for this data set, using 28 topics is shown in Table 1.

Next we must determine the proper number of clusters to use for the documents and the users. We do this by measuring the performance of the clustering for various values of k number of clusters, Figure 14. In the graph, vertical is the measure of average similarity between the elements of the clusters. The maximum value is 1. We want to pick a number that is as small as possible to allow generalization without going too low in performance. By picking values at the knee in the graph we can optimize for both those criteria. In the graphs, the knee for documents begins

Figure 14: Performance curve for document 14a and user 14b clustering for the collection. Performance is measured on the vertical by how similar the items in a cluster are to each other. Max value is 1. We look at the performance for all values of k clusters between some large value and 1.

at $k = 32$. We choose a value of 35, a couple steps to the left to avoid being too close to the steep drop off in performance. Similarly for the user clusters, the knee begins at about $k = 17$ and we choose a value of 20.

3.2 A few examples

Although understanding the data really requires interacting with it through the tool, we provide a few examples here to give a gist for how the data appears in the final results. We show a few examples of the kinds of groupings we can achieve with this system. It is important to note that the nature of this data is probabilistic and there is some noise and potential error with some users in the system. In part this can depend on the amount of data we have available, error rates would go down as we increase the numbers of documents we include as well as the numbers of individuals.

The first two examples in Figures 15 and 16, we show two different groups of senators. We picked the number of user clusters as described in Section 3.1 based on the quality of the clustering. However, while we maintain these clusters as distinct, for the purposes of visualization we also keep track of how close, in topic space, those user clusters are to each other. Those which are close above some threshold, in this case above 0.5 on a scale of 0 to 1, we draw a light gray line between them and apply some force in the visualization to keep them together. Thus its possible to see two to three larger groups form out of the smaller clusters.

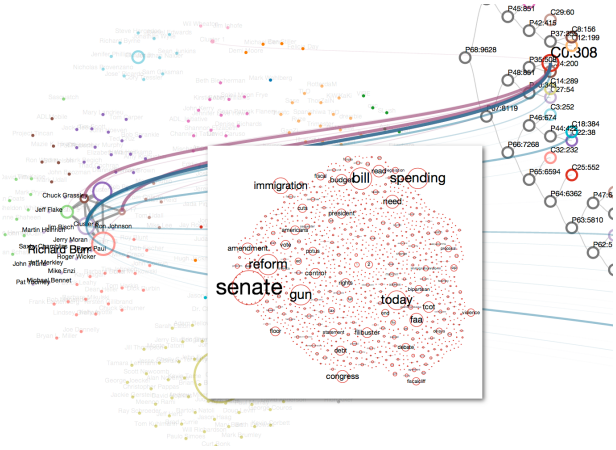

Figure 15: A group of senators talking about a variety of bills going through Congress at the time of data capture.

Having these larger groups can help in navigating each individual group as we see in Figure 16. The word cloud shows prominently the words "forward" and "looking", these words are very abstract in some sense, but because we know through navigation that this group is close to the group talking about "senate", "bill", and "spending", we may conclude that this is a group of senators who are more interested in talking at a higher level rather than about specific bills or votes.

The second two examples in Figures 17 and 18, show two different groups of E-learning experts. Again we find two connected, but distinct groups. The first group in Figure 17 appear to talk more gen-

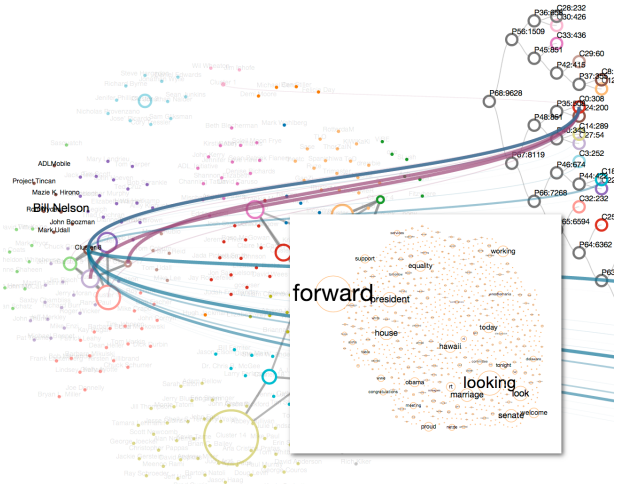

Figure 16: A group of senators talking at perhaps a higher level than those pictured in Figure 15.

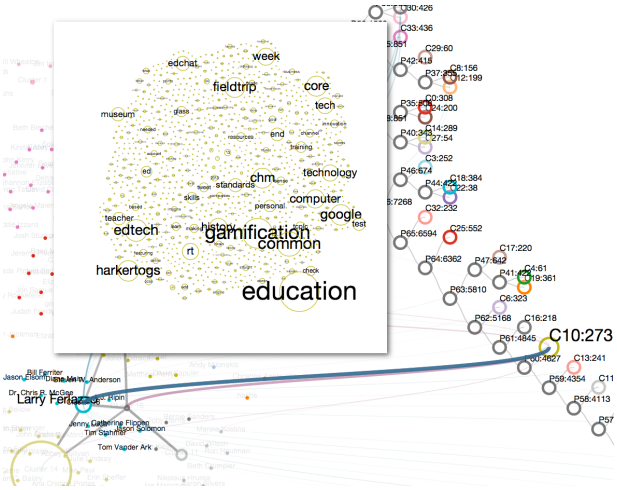

Figure 18: A group of E-learning talking about particular technologies and techniques.

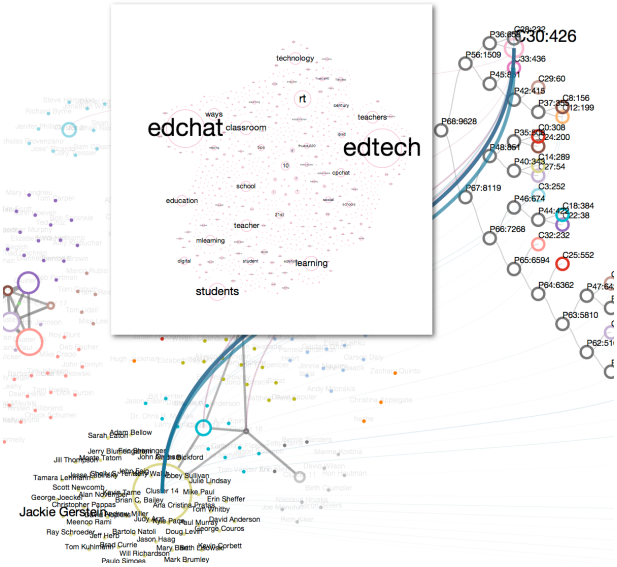

Figure 17: A group of E-learning experts consistent in their use of edchat and edtech as hashtags to describe their tweets.

erally about the topic and use the hashtags (a way of labeling topics in Twitter) "edchat" and "edtech" very consistently to mark the information they're sharing. The second group pictured in Figure 18 are talking more about specific technologies and techniques related to E-learning such as "gamification" and "google". The hashtag "edtech" is present, but not as dominant as the other group, which may imply the tweets of this group are going more in depth on the subject.

To reiterate, the nature of this data is probabilistic so it is important to remember that these descriptions are in aggregate and only relate to an individual at a confidence level related to their closeness to the group. For users much closer to the center as literally shown in the visualization, we might have a stronger confidence about using these general descriptions for them. Contrary, users who are at the fringe of a group, we should have a lower confidence of ascribing traits of the group to them, but not to absolute zero. One way to think of the fuzziness is that those individuals at the fringe may actually have had an affinity for another group. The affinity could be 0.49 where the affinity for the group they are finally assigned is 0.51. We will discuss ways we might improve the visualization to capture this fuzziness in Section 4.

Our final example group, Figure 19, shows the group of professional gamers. Because professional gamers make a living by sponsorship and live streams of their games, it is unsurprising to find that their dominant communication is about the live streams of their activity. While we do not necessarily have a measure for the accuracy of our data, we can at least use the common sense of roles we are aware of to determine whether or not there is enough sense being made out of the data with the analysis presented here.

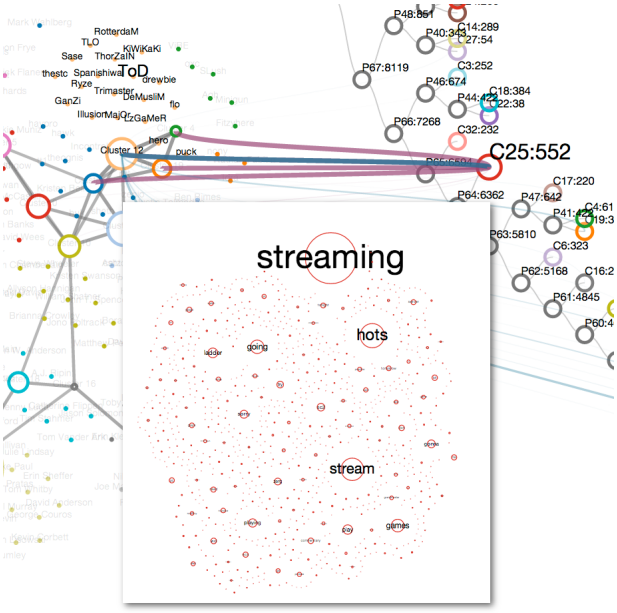

Figure 19: A collection of gamers talking about streaming live footage of themselves playing the game.

4 Future Work

There are two ways we can expand this work. One is to focus on improving the visualization itself, by adding new features and capabilities. The second is by exploiting the information provided by this tool to new capabilities, especially in aiding automated learning systems as stated previously.

First, there are a few ways we consider for expanding the capabilities of this tool. As mentioned previous, there is a fuzzy connection between individuals and their clusters centers than is otherwise implied by the thresholded connectivity we use here. We may be able to expand the visualization to accommodate users who bridge different groups and give a better indication of some individuals' hybrid natures. Although we had considered using, but did not have time to actually use, further text analysis methods such as sentiment analysis, we would consider doing so in the future. Although we are able to group people by talking about a particular topic, we are not distinguishing between those people who are like what they are talking about versus those who are against it. This would apply especially in the case of the senators. Finally, while we use cosine similarity and k-means clustering there may be some other or better methods for performing proximity measurements or handling the clustering itself. It may be worth investigating an incremental improvement in the algorithm methods themselves.

Considering that we have not quite 275 people in this data set, it is a small data set in term of the size of social media data. If we were to collect data from 275 random individuals we might be looking at 275 distinct roles, as opposed to the more defined roles we consider here, e.g., senator, e-learning expert, gamer. In that situation, there would likely be no repeated patterns among the discussions to reinforce the signals and dampen noise, allowing the groups to cluster in a meaningful way. However, if we were to have 275 distinct roles and have 20000 individuals there is a better chance we will start to see repeated patterns. In general, we would expect that as we add individuals the number of roles increases, but it tapers off at an asymptote. There are only so many different kinds of things people can talk about or do. With a large enough dataset we should always be able to find repeated patterns that allow people to cluster. By exploring a much larger data set, we may be able to determine at what point we reach that asymptote.

There are a few ways we could extend the use of this analysis to other applications. One example, using data entirely available within the application, is the user view. This shows web links filtered by the clustering to show only those links posted by people in the same cluster, and sorted so that the top links are by the closest people. With additional meta data, such as a person's interest in a class, we can potentially apply a similar inference to suggest others within the cluster take that class. The strength of the suggestion would be weighted by the proximity of each member.

Another strategy of interest with this data set is the idea of archetype or stereotype definition. Because we have this notion of a cluster center and a set of textual data about the cluster, we can expand the tool to capture new information about that center, treating it as a stereotype for the group. For instance, an expert examining the word cloud recognizes and suggests that the learning system should modify its examples to game related learning. We allow the expert using Project Grandmaster to modify the document clusters changing "C 0 : 125" to a more meaningful learning system flag, "Interest in Games". Now, any user group which links to this document cluster automatically gains the attribute "Interest in Games". Because this would modify the stereotype for the group, any current user or any fu-

ture user assigned to the group would automatically receive the attribute "Interest in Games" which could then be used to bootstrap a learning system to take advantage of that information. Going further there may be more meaningful ways we can add these kinds of attributes to the stereotypes or document clusters and is worth further investigation.

5 Conclusion

This work intended to show a proof-of-concept application for exploring the use of social media data to understand individuals for the purposes of aiding and improving automated learning systems. We have demonstrated a tool which is capable of collecting and processing social media free-form text and providing visualizations which aid a user understanding the text and the individuals in aggregate. We believe the results presented here represent a strong foundation from which to build systems targeted to particular learning applications by making use of the data simplified and summarized by this work.

6 Acknowledgements

Funding for this work was provided by the Office of the Secretary of Defense through the Advanced Distributed Learning (ADL) Initiative.

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

References

[1] David Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[2] Avondale Web Crawler. Todo.

[3] Patricia J. Crossno, Andrew T. Wilson, Daniel M. Dunlavy, and Timothy M. Shead. Topicview: Understanding document relationships using latent dirichlet allocation models. In *Interactive Visual Text Analytics for Decision Making Workshop*, 2011.

[4] Andrew Wilson et al. Text analysis tools and techniques of the pubmed data using the titan scalable informatics toolkit, 2011.

[5] Owen Kaser and Daniel Lemire. Tag-cloud drawing: Algorithms for cloud visualization. *arXiv preprint cs/0703109*, 2007.

[6] Mark Steyvers and Tom Griffiths. Probabilistic topic models. *Handbook of latent semantic analysis*, 427(7):424–440, 2007.

[7] C. Sumner, A. Byers, R. Boochever, and G. J. Park. Predicting dark triad personality traits from twitter usage and a linguistic analysis of tweets. In *Proceedings of the IEEE 11th International Conference on Machine Learning and Applications*, 2012.

[8] C. Sumner, A. Byers, and M. Shearing. Determining personality traits and privacy concerns from facebook activity. In *Proceedings of the Black Hat Briefings '11*, 2011.

[9] Titan Toolkit. titan.sandia.gov.