Paper 6241-2016

Getting More from the Singular Value Decomposition (SVD): Enhance Your Models with Document, Sentence, and Term Representations

Russ Albright, James Cox, and Ning Jin, SAS Institute Inc., Cary, NC

ABSTRACT

Since its inception, SAS® Text Miner has used the singular value decomposition (SVD) to convert a term-document matrix to a representation that is crucial for building successful supervised and unsupervised models. In this presentation, using SAS® code and SAS Text Miner, we compare these models with those that are based on SVD representations of subcomponents of documents. These more granular SVD representations are also used to provide further insights into the collection. Examples featuring visualizations, discovery of term collocations, and near-duplicate subdocument detection are shown.

INTRODUCTION

The singular value decomposition, or SVD, is a key technique for representing high-dimensional, sparse data in a low-dimensional space. The technique allows for long, sparse document vectors to be represented as compressed, dense vectors that can be used by data mining or machine learning algorithms. SAS Text Miner leverages the SVD for building and applying models (Albright, 2004).

A vector representation works particularly well on short textual observations of a sentence to about a paragraph in length. As documents become longer, the representation can become less effective, particularly if those documents convey multiple themes or concepts. In this case, it might be beneficial to divide the document so that multiple vectors are used for each one. Each individual vector can encapsulate a small portion of text such as a sentence or a group of sentences. The SVD is then applied to these subcomponent vectors and used for text mining.

This subdivision of documents not only makes the SVD more effective, it also permits an investigation and discovery of other characteristics of the collection that are based on local information between terms. By treating sentences or groups of sequential sentences as “documents”, new and interesting analysis might be available to you. In this paper, the following examples are discussed and demonstrated:

- more focused and informative clusters
- document and subdocument similarity
- term collocations
- sentence visualizations
- supervised learning

In the next section, an approach for creating the subcomponents of documents using sentences is explained. Following that, a section is devoted to each of the bullets listed above. In each case, they draw on various nodes within SAS Text Miner. Any additional SAS code that is required to accomplish the task is provided in each section. Finally, some conclusions are made at the end of this paper.

DOCUMENTS, SENTENCES AND OTHER SUBCOMPONENTS

In this section, code is provided to create a new data set in which subcomponents of the original document become separate textual observations. Because SAS Text Miner is designed to work at the document level, these subcomponents are then interpreted as documents when SAS Text Miner runs. Later in the paper, you can see how SAS Text Miner results can then be mapped back to reflect on the original document based on the document identifier variable, or docid.

A data set of the text of 10,000 Wikipedia entries is used to demonstrate the process of transforming your data to a collection of sentences. These sentences are used in subsequent sections of this paper to show various examples.

The Output Term Position Data Set

For each strategy outlined below, PROC HPTMINE is used to create the **outpos** (term position) data set. This data set provides needed information about sentence breaks and the location of parsed terms relative to one another. A display of a few observations of this data set are shown in Figure 1.

Obs	TERM	ROLE	PARENT	_START_	_END_	SENTENCE	PARAGRAPH	DOCUMENT
1	simeon		simeon	0	5	1	0	1
2	ekpe		ekpe	7	10	1	0	1
3	athletics		athletics	0	8	1	0	2
4	at		at	10	11	1	0	2
5	the		the	13	15	1	0	2
6	1920		1920	17	20	1	0	2

Figure 1. Observations and Variables from the outpos Data Set

As you can see, the data set in Figure 1 shows a row for each occurrence of every term in every document in the collection. Note that the natural language processing features such as noun groups and tagging are not being used for this initial run since the goal is only to determine sentence boundaries. The code to accomplish this task is shown below:

```
proc hptmine data=documents;
    doc_id doc_id;
    var text;
    parse
          entities =none
          nostemming
          notagging
          outpos   =position
          nonoungroups
          shownumpunct
          buildindex
          ;
    performance details ;
run;
```

Now that the **outpos** data set has been created, the following sections show how to reconstruct data sets in which the sentences or groups of sentences become the documents in a new data set.

Sentences Become Documents

The following code processes the **outpos** data set that was formed in the previous section into a new documents data set. One DATA step calculates the start and length of each sentence in bytes, and the second DATA step uses the `substr` function to copy the identified sentences from the original text variable into a new observation. The resulting data set, **sentenceObs**, contains a row for each sentence in the collection and three variables: the sentence, a unique identifier for the sentence, and the ID of the document that contains that sentence. This data set can be used as input to SAS Text Miner where the sentences can be reinterpreted as documents for the analysis.

```
data sentenceSize;
retain document start size;
set position;
by document sentence;
if First.sentence then start=_start_+1;
if Last.sentence then do;
   size=_end_ -start+2;
   output;
end;
keep document start size;
run;

data sentenceObs;
length sentences $1000;
merge sentenceSize(in=A ) documents (rename=(docid=document) );
by document;
if A then do;
sentences=substrn(text,start,size);
output;
end;
keep sentences document;
run;
```

In order to improve the performance and effectiveness of some of the analysis done later in the paper, a final DATA step, shown below, removes sentences if they consist of only one or two words and adds a sentence id, sid, to each observation.

```
data tm.sentenceObs;
retain sid 0;
set sentenceObs;
if lengthn(kstrip(sentences)) ge lengthn(kstrip(kcompress(sentences)))+ 2
then do;
    sid=sid+1;
    output;
end;
run;
```

A few sample sentence observations from the Wikipedia data set are shown in Figure 2.

Obs	sid	sentences	document
1	1	Simeon Osuji Ekpe (born 8 April 1934 - 15 March 2010) Nwangele) was a retired Justice of the Court of Appeal of Nigeria and former Chief Judge of Imo State.	1
2	2	Ekpe was educated at Bishop Shanahan College, Orlu and the University of London.	1
3	3	The men's marathon event was part of the track and field athletics programme at the 1920 Summer Olympics.	2

Figure 2. The First Few Sentences of a Wikipedia Data Set

CLUSTERS AND TOPICS

As you might expect, using sentences in an exploratory analysis produces different clustering and/or topic results than does analyzing the entire document at once. Because of its limitations, clustering, in particular, can benefit from this approach.

Clusters partition the observations such that each observation belongs in exactly one group, as opposed to topic analysis, where an observation might relate to any number of topics. When the inputs are documents, the cluster results tend to produce high-level topics or themes. In a sentence-based analysis, the clusters now partition the sentences rather than the original documents. Since each document can contain many sentences, each document might still be associated with multiple clusters. The behavior becomes very similar to that of the Text Topic node.

Producing the sentence clusters in SAS Text Miner is straightforward. Once the data set of sentences is obtained, the Text Parse, Text Filter, and Text Cluster nodes can be run as normal. Then, on output from the Text Cluster Node, the following small piece of code can be run in a SAS Code node to assign documents to clusters based on the sentences that the documents contain.

```
data docCluster;
set emws1.TextCluster_train;
array clus[*] TextCluster_doc1-TextCluster_doc64;
retain clus;
by document;
if First.document then do;
    do i=1 to dim(clus);
        clus[i]=0;
    end;
end;
clus[TextCluster_cluster_]=1;
if Last.document then do;
   numCluster=0;
   do i=1 to dim(clus);
        numCluster= numCluster+clus[i];
   end;
   output;
end;
keep document numCluster TextCluster_doc: ;
run;

proc means data=docCluster;
variable numCluster;
run;
```

The new binary variables TextCluster_doc*1*- TextCluster_doc*n*, where *n* is the number of clusters, contains a 0 or a 1 indicating whether at least one sentence in that document is in that cluster (1) or not (0), allowing documents to now be members of more than one cluster.

When the above technique was applied to the Wikipedia data set, the discovered clusters revealed aspects of the collection that were not apparent when clustering the original articles. For example, the 52nd cluster (out of 64), shown in Figure 3, appears to be about winning or receiving an award of some kind. This theme was not evident when the original articles were clustered with either 64 or even 500 clusters. When the entire document was the context, the cluster was not evident because it cut across several more prominent document clusters about sports figures, military personal, and musicians. While aspects of these types of articles fit this recognition theme partially, because documents can belong to only a single cluster, the theme is not revealed as its own cluster unless the sentence analysis is used.

Obs	_CLUSTER_	clus_desc	freq	percent
52	52	+win +award +championship +medal +best +receive +title +world +winner +prize +honor +nominate +event +final +defeat	1950	0.018814

Figure 3. An Example Cluster from Clustering Sentences

In comparison, when the Text Topic node was run on the original articles, it could pick up on this award theme, confirming that sentence-based clustering behaves more like document-based topics.

There are differences between clustering on sentences and a topic analysis on the documents that contain them, however. In the table below, the two types of runs are compared using 64 and 500 clusters/topics. For clustering, the sentences were clustered and the documents were assigned to the corresponding cluster. The topic run was performed on the original articles. The table shows the mean number of clusters/topics that each document belonged to, as well as the standard deviation of this variable. In addition, the minimum and maximum number of clusters any one document belonged to is shown.

	Number of Clusters	Mean Number per Document	Std. Dev	Minimum	Maximum
Clusters	64	5.78	5.07	1	33
	500	8.28	9.03	1	55
Topics	64	3.62	2.56	0	17
	500	28.7	22.3	0	125

Table 1. Cluster and Topic Statistics

As the requested number of topics/clusters increases, the average number of topics/clusters that a document belongs to also increases. This increase is much more dramatic for topics. For clustering, the maximum number of clusters a document could belong to is obviously bound by the number of sentences that the document contains. The topic functionality has no such upper bound. In fact, the Text Topic node allows for customization of its settings, so how many documents get assigned to individual topics can be controlled if desired.

Perhaps the best reason to use a sentence-based approach when a large number of overall topics or clusters is desired is because the Text Topics node requires as many SVD dimensions to be calculated as there are topics. The Text Cluster node can use the same number of dimensions, typically 50 or 100, regardless of how many clusters you request.

DOCUMENT AND SUBDOCUMENT SIMILARITY

BACKGROUND

Document similarity can be important in a number of contexts. One common application involves identifying and removing redundant information from your collection. This redundancy should be considered because it can inappropriately alter your analysis. For example, news releases frequently get picked up by multiple media outlets and rebroadcasted so that multiple occurrences of any given news story can exist in your collection. Each new release might also be surrounded by text that varies from different outlets. So the entire document is different, but a subcomponent of that document is practically identical. How can you identify these duplicates and remove the multiple versions of them?

Other tasks are also relevant, such as removing the history thread of a set of email or forum posts because the thread is contained in other emails. Plagiarism is another interesting application area where the SVD has proven useful (Ceska, 2008). Plagiarism detection involves determining when either exactly copied, slightly altered, or paraphrased portions of text from some document matches, in some sense, those in a reference data set.

DOCUMENT SIMILARITY AND THE SVD

Subdocument similarity algorithms are similar to search and retrieval tasks. A reference data set is searched against a new observation. In this case, the new observation is a document rather than a query, and the process is to detect whether the new document is similar to any of those in a reference set.

First is a training phase in which the reference data set is prepared for the scoring action. The training phase creates the sentence representation for each document in the reference data set. This can be done with a flow containing either the Text Cluster node or the Text Topic node, since both of them produce the SVD dimensions.

Note that if your reference set itself contains duplications, you might want to first remove them by gradually rebuilding this reference set. Start with a few documents in your reference set and then, before you add additional entries to the set, score them for duplication against the current reference.

Second, the scoring phase creates the sentence representations for the documents that are under consideration and compares the sentences to the sentences of each document in the reference training data. The score code from the cluster or topic node creates the scored sentence representation, but in order to calculate distances from each sentence in your training data to each sentence in the data to be scored, you need the code that appears in the next section.

Euclidean distance, rather than cosine similarity, can be used to determine distance between sentences that have been projected into the SVD space because the vectors have been normalized to unit length in SAS Text Miner. In addition, if the entropy or idf (inverse document frequency) weight has been chosen, frequently occurring terms will have already been down-weighted so that the rarer but concentrated terms have the greatest influence on similarity.

DEMONSTRATING SIMILARITY DETECTION

The following heuristics are used to detect possible duplicated sentences in the Wikipedia data set.

1. A sentence is considered a potential copy if it is within a distance of .0001 of a sentence in the reference set. Allowing some tolerance threshold means that sentences do not have to be exactly the same. They might have been altered by a word or two. This near duplication is important if we were to attempt to detect plagiarism, for example.
2. If a sentence in the document under consideration is close to more than one sentence from the reference set, it is not considered a candidate match as it is unlikely to be a copied version of a sentence. This prevents the detection of short common sentences as duplicates, which can give false positives.
3. A document under consideration triggers further review if it contains at least two potentially copied sentences. This helps avoid spurious matches.

If hold-out data were available, these threshold values could be tuned for the particular data you are analyzing, your number of SVD dimensions, and the type of copying that you are detecting.

The code for the Euclidean distance computation between the observations in the training set and the validation set is shown below. It does not assume that the sentences are sequential, although typically that is the case.

```
/* for performance reasons, eliminate variables then place them back on */
data traincluster;
set  emws1.textcluster4_train(rename= (sid=trainsid document=traindocument
textcluster4_SVD1-textcluster4_SVD&numdim = trainsvd1-trainsvd&numdim));
```

```
keep trainsid traindocument trainsvd1-trainsvd&numdim;
run;

data validatecluster;
set  emws1.textcluster4_validate;
keep sid document textcluster4_SVD1-textcluster4_SVD&numdim;
run;


/* distance calculation */

data cartesianDist;
  set traincluster;
  array trainloc[*] trainSVD1-trainSVD&numdim;
  do i=1 to n;

  set validatecluster point=i nobs=n;
    array valloc[*] textcluster4_SVD1-textcluster4_SVD&numdim;
    sse=0;
     do j=1 to dim(valloc);
         sse=sse+((trainloc(j)-valloc(j))**2);
     end;
    if sse<.0001 and textcluster4_SVD1>0 then output;
    keep trainsid sid sse;
  end;
run;

/* fold needed variables back onto data */

proc sql;
create table compSent as
select a.*, b.sentences,b.document, c.sentences as trainsentences,
c.document as traindocument
from cartesianDist a, emws1.textcluster4_validate b,
emws1.textcluster4_train c
where a.sid=b.sid and a.trainsid=c.sid;
quit;
```

Now you can select only the documents that match at least two sentences. However, if a possibly copied sentence matches more than one sentence from other documents in the reference set, it is not considered duplicated. This prevents the detection of short common sentences as duplicates, which can give false positives.

```
/* select the unique matches of at least 3*/
proc sort data=subcompSent;
by sid traindocument;
run;


data triggerMatch;
length composite $3000 traincomposite $3000;
retain matchcount composite traincomposite;
set subcompSent;
by sid traindocument ;
```

```
if first.sid and first.traindocument then matchcount=0;
if first.traindocument then do;
    matchcount=1;
    composite = left(trim(sentences));
    traincomposite = left(trim(trainsentences));
end;
else do;
    matchcount = matchcount +1;
    composite = left(trim(composite))||"... " || left(trim(sentences));
    traincomposite = left(trim(traincomposite))||"...
"||left(trim(trainsentences));
end;

if last.sid and last.traindocument and matchCount gt 1 then output;
run;
```

To investigate how well the approach works, 80% of the Wikipedia data set was placed into the reference data set (train) and the remaining 20% into a data set to score (validate). The split was done across articles so that sentences from the same article always remain in the same split.

Using the process described above, 31 documents from the 2000 found in the validation data triggered as potential duplication. Initially this was surprising but after examining the matches, it was easily explained. Two of the 31 are shown in Figure 4 and are representative of the others. The matched sentences have been concatenated into composites (called `composite` for the test document and `traincomposite` for the corresponding train document) with an ellipse between them and are shown in the figure, along with the titles of the matched articles. The matches are not exact but rather indicate that the same structure or pattern was used in creating the entries.

Obs	composite	traincomposite	title	traintitle
1	This weather squadron is responsible for base or post forecasting, developing weather products, briefing transient aircrews, and weather warnings for all of their geographical units.... Using automatic observing systems located at all military installations and communicating with their combat weather flights, the squadron is able to 'watch' the weather in their entire area of responsibility from one central location.... Personnel and resources.... List of duty assignments and parent units from 1997 to present.	This weather squadron is responsible for base or post forecasting, developing weather products, briefing transient aircrews, and weather warnings for all of their geographical units.... Using automatic observing systems located at all military installations and communicating with their combat weather flights, the squadron is able to 'watch' the weather in their entire area of responsibility from one central location.... Personnel and resources.... List of duty assignments and parent units from 1945 to present.	21st Operational Weather Squadron	28th Operational Weather Squadron
2	Starters by position.... "Note: Pos = Position; G = Games played; AB = At bats; H = Hits; Avg.... = Batting average; HR = Home runs; RBI = Runs batted in" Other batters.... "Note: G = Games played; AB = At bats; H = Hits; Avg.... = Batting average; HR = Home runs; RBI = Runs batted in" Pitching.... "Note: G = Games pitched; IP = Innings pitched; W = Wins; L = Losses; ERA = Earned run average; SO = Strikeouts" Other pitchers.... "Note: G = Games pitched; IP = Innings pitched; W = Wins; L = Losses; ERA = Earned run average; SO = Strikeouts" Relief pitchers.	Starters by position.... "Note: Pos = Position; G = Games played; AB = At bats; H = Hits; Avg.... = Batting average; HR = Home runs; RBI = Runs batted in" Other batters.... "Note: G = Games played; AB = At bats; H = Hits; Avg.... = Batting average; HR = Home runs; RBI = Runs batted in" Pitching.... "Note: G = Games pitched; IP = Innings pitched; W = Wins; L = Losses; ERA = Earned run average; SO = Strikeouts" Other pitchers.... "Note: G = Games pitched; IP = Innings pitched; W = Wins; L = Losses; ERA = Earned run average; SO = Strikeouts" Relief pitchers.	1931 Cincinnati Reds season	1970 Boston Red Sox season

Figure 4. Example Document and Sentence Matches Found from the Partitioned Wikipedia Data Set

As a secondary example, we made an individual copy, and then made minor alterations to sections of text from an article in the reference data set and placed the altered text into one of the holdout articles. Then the holdout article was scored against the reference data. Figure 5 shows an example of the detection. Again, the `composite` variable holds the sentence that matched those of the training document and placed in `traincomposite`.

Obs	composite	traincomposite	body	trainbody
1	The Reull Vallis valley on Mars appears to have been carved by water.... Lineated Floor Deposits.... The floors of some channels have features called lineated floor deposits.... They are ridged and grooved materials that seem to deflect around obstacles, and they are believed to be ice-rich.... Some glaciers on Earth show such features.	Reull Vallis is a valley on Mars that appears to have been carved by water.... Lineated Floor Deposits.... On the floors of some channels are features called lineated floor deposits.... They are ridged and grooved materials that seem to deflect around obstacles.... Some glaciers on the Earth show such features.	Some glaciers on Earth show such features. KCOH (1230 AM) is a radio station in Houston, Texas. The station brand is "Radio Ranchito". The radio station began in 1948 when KTHT (now KBME) vacated this frequency for a stronger signal at 790 kHz. On June 4, 2012, at 7:00 a format change is underway, as a Talk/Oldies format. The last Talk/Oldies format in the Houston area was the now-defunct KKHU You 106.9 FM now KHPT which is now a simulcast of KEGL KQUE was once part of a quadrocast of KTJM 98.5 "Houston's Jammin Hits" from May 2001 to July 2001 along with 880 AM, and 103.3 FM. "Jammin Hit's up and down the dial". The Reull Vallis valley on Mars appears to have been carved by water. The valley, which runs westward into Hellas Planitia, is named after the Gaelic word for planet. Lineated Floor Deposits. The floors of some channels have features called lineated floor deposits. They are ridged and grooved materials that seem to deflect around obstacles, and they are believed to be ice-rich. Before that KQUE was a simulcast of KKRW 93.7 "The Arrow 70's Rock". During a short period KQUE was continuing their MOR standards format previously on their 102.9 counterpart during the ownership days of SFX Broadcasting Corporation which took over ABC radio affiliate KNUZ "K-News" News/Opinion	Reull Vallis is a valley on Mars that appears to have been carved by water. It runs westward into Hellas Planitia. It is named after the Gaelic word for planet. Lineated Floor Deposits. On the floors of some channels are features called lineated floor deposits. They are ridged and grooved materials that seem to deflect around obstacles. They are believed to be ice-rich. Some glaciers on the Earth show such features. Lineated floor deposits may be related to lobate debris aprons, which have been proven to contain large amounts of ice. Reull Vallis, as pictured below, displays these deposits.

Figure 5 Subdocument Duplication Detection Example

Not all types of near duplication will be detected with this approach. If the duplicated sentence contains more than a couple of different terms, it is likely to be too far away in the SVD space to be considered a match. A rich synonym mapping could help in these situations.

Also, if the sentence is restructured, the sentences will not match. In one example, a pair of sentences from the reference collection was changed to a compound sentence. In this case, no match was detected because the sentence boundary had changed significantly. Other features such as word n-grams would have to be used instead of sentences in order to detect this type of match.

If you need to score in batch, performance and memory can be an issue. A performance improvement that is not shown but that should be straightforward to implement involves using the cluster prediction for the sentences you are scoring. The reference data sentences are already being clustered at train time. In order for you to obtain the SVD dimensions during scoring, the sentences you are scoring receive a cluster membership prediction. The distance comparison for any sentence should need to be done only against those in the same cluster from the reference data rather than with all the reference data sentences.

TERM COLLOCATIONS

BACKGROUND

The word *collocation* has several definitions. Some use the word to describe a set of terms that form a phrase and have a distinct meaning and usage such as “to pay attention” or “to feel under the weather”. Others define a collocation with a broader scope. For our purposes, a collocation of interest is a set of terms that satisfy the two following conditions:

1. The terms in the set tend to occur near one another. For our purposes, the set should occur within the same sentence.
2. The terms in the set occur together at an unexpectedly high rate relative to the rate of their individual occurrences.

In addition, the collocations that are the most interesting are the ones that also occur frequently in the collection. Examples of collocations we might be interested in finding include frequent noun groups such as “data mining” or common non-sequential terms that are used together such as “launch” and “attack”.

The noun group functionality in SAS Text Miner is a concept related to a collocation but that uses linguistic information within a sentence to identify when sequential terms are functioning together as a phrase. A collocation can form a noun group, but it also can just be a set of terms that co-occur within a sentence regardless of the linguistic structure. The elements in the collocation don’t even need to be sequential.

Recognizing collocations is important and useful because often a set of terms is used to characterize, summarize, or describe a document or a group of documents. This list of descriptive terms can be more informative if you know which of those terms actually appear together and near one another rather than being independent of each other within the context.

COLLOCATIONS AND THE SVD

Sentence-level analysis and the SVD can be useful in identifying collocations without doing direct counts of the intersection of terms. Direct counting can be expensive, particularly when identifying collocations consisting of more than three terms. The SVD approach, shown in the following section, can be a fast and effective alternative if you are already doing a cluster analysis or topic analysis.

The main insight in the approach is that collocations of interest are strong influencers of clusters and topics. They influence because they are both multi-term and relatively frequent as a multi-term. So, it is likely that the best collocations will be evident as important influencers in the clusters of sentences.

If a good collocation does exist in your cluster, then the individual terms that make up the collocation will necessarily be near one another in the SVD space. The next section demonstrates a process to identify the collocations from the current set of descriptive terms included in a label for a cluster in the Text Cluster node and for topic labels in the Text Topic Node.

DETECTING COLLOCATIONS IN A TERM LIST

In this example, the descriptive terms from a cluster discovered by the Text Cluster node are analyzed to identify potential collocations within the elements of this set. Currently the list of terms is ordered by importance and presented as a list but there is no indication if some of the terms on the list actually are dependent and form a collocation. In the approach that follows, if any of the terms are near one another in the SVD space, they are considered a collocation and the precise counts are calculated to confirm the decision.

Figure 6 shows the 50 descriptive terms from one of the 64 clusters found on the Wikipedia data set. The plus sign in front of a term means it is parent term that represents many related terms that have been assigned as a synonym or stem.

Obs	clus_desc
7	species +family +habitat natural +forest +land freshwater +plant +lake +genus +endemic tropical +river +loss +threaten +area +marsh subtropical moist intermittent +dry +contain +lowland +snail +grow +gastropod +swamp +island +mollusk +occur +garden +water +savanna +common +fish terrestrial shrubland +distribution iucn +frog +pond +tree +flood rocky +grassland agricultural heavily china air-breathing aquatic

Figure 6. Fifty Terms from a Sentence-Based Cluster

The terms were extracted, stripped of their leading plus sign if they had one, and placed in a data set to be scored, one term per observation. You can use the DISTANCE procedure to calculate distances and find terms that are near one another from the term list. When the data is sorted by distance, the shortest distances indicate the purest collocations among the members of the term list.

In order to get an indication of how well this performed, the actual counts were also generated for how often each term in the list occurred independently and with the other terms in the list. This allows for a mutual information calculation. The ranking comparison of the top 25 out of the possible 1275 combinations is shown in Figure 7. The SVD approach emphasizes word pairs that occur very rarely outside the cluster.

Obs	term1	term2	mirank	svdDistRank
1	gastropod	mollusk	4	1
2	gastropod	snail	5	2
3	subtropical	tropical	66	3
4	intermittent	marsh	13	4
5	mollusk	snail	7	5
6	moist	subtropical	58	6
7	moist	tropical	68	7
8	marsh	swamp	18	8
9	air-breathing	terrestrial	1	9
10	freshwater	intermittent	21	10
11	freshwater	marsh	28	11
12	flood	pond	40	12
13	intermittent	swamp	17	13
14	air-breathing	gastropod	2	14
15	air-breathing	snail	6	15
16	air-breathing	mollusk	3	16
17	savanna	swamp	25	17
18	freshwater	swamp	43	18
19	snail	terrestrial	10	19
20	gastropod	terrestrial	8	20
21	lowland	subtropical	64	21
22	grassland	shrubland	29	22
23	dry	shrubland	36	23
24	lowland	tropical	73	24
25	mollusk	terrestrial	9	25

Figure 7. Highest-Ranking Collocations of a Cluster's Descriptive Terms

VISUALIZING TERMS, SENTENCES, AND DOCUMENTS

The SVD dimensions from either the Text Topic node or the Text Cluster node can be used for two-dimensional visualizations of terms, sentences, documents, or groups of documents when the mean of each SVD has been calculated. All of these objects can be projected into the same space and plotted.

Once the representation is obtained, there are several ways you can project them into two-dimensional coordinates. One approach is to use multidimensional scaling with the MDS procedure. PROC DISTANCE is used prior to PROC MDS to create the input table for PROC MDS. In Figure 8, the "win award" cluster is plotted along with several clusters about sports, military, and music using ODS and PROC MDS. Note how the award cluster is located in the center of the diagram as it shares aspects with the other clusters; it spans the topic of sports, military, and music.

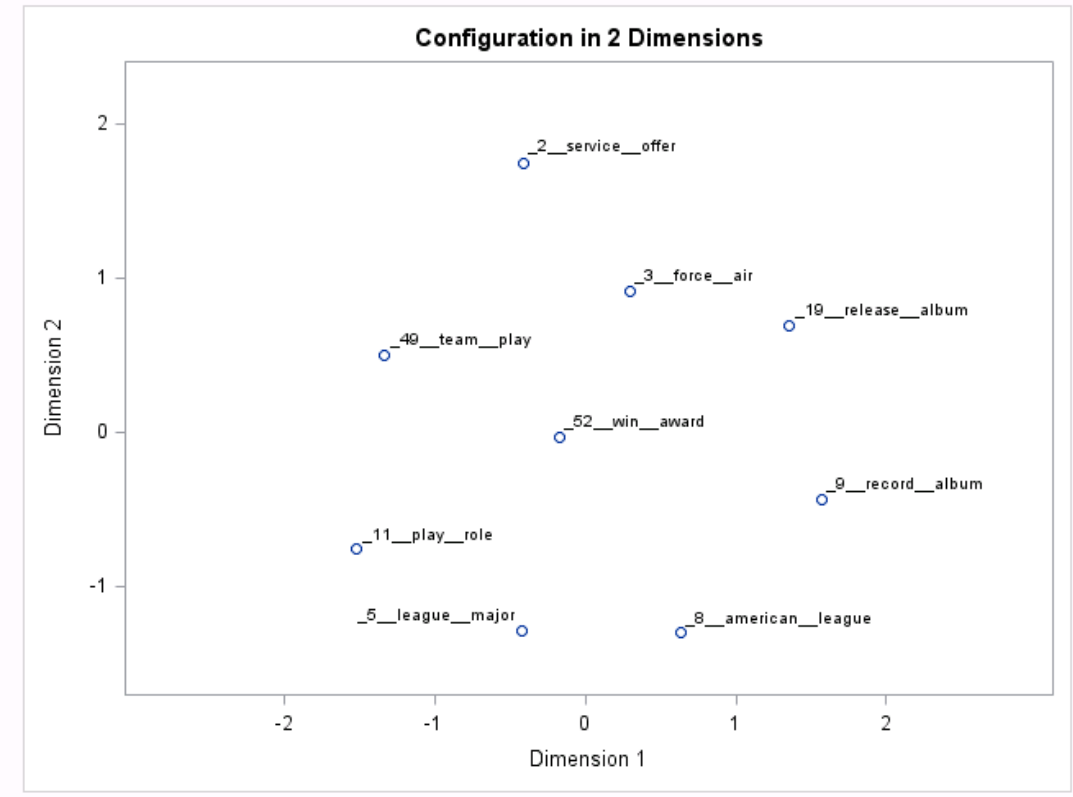

Figure 8. Multidimensional Scaling Plot of Several Clusters

The SAS code to generate Figure 8 from a table of cluster means is shown below:

```
data plotclusters(drop =_cluster_);
   length title $32;
   set emws1.textcluster_clusters;
   cluster=put(_cluster_,best12.);
   title=_cluster_||":"||scan(clus_desc,1," ")||" "||scan(clus_desc,2," ");
   if  _N_=49 or _N_=9 or _N_=2 or _N_=3 or _N_=5 or _N_=11 or _N_=19 or
_N_= 52 or  _N_=8;
run;
```

```
proc distance data=plotclusters out=plotit method=EUCLID nostd;
    var RATIO(_mean1-_mean100);
    id title;
run;
proc mds data=plotit out=plotit
    level=ratio dimension=2 noprint;
  run;
```

In the above example, a higher number of dimensions (100) was used with the SVD so MDS was applied to create two dimensions. If only two SVD dimensions are computed to begin with, then they can be used directly to plot.[1] The two documents from the subdocument duplicate detection example of Figure 5 are shown in Figure 9. The sentences of both documents along with the documents themselves are projected into this two-dimensional representation. The two colors represent sentences from the two different documents, and those of opposite color that are near one another are the sentences that were copied and altered from one document to the other. The points were also jittered so that the overlapped ones can be seen.

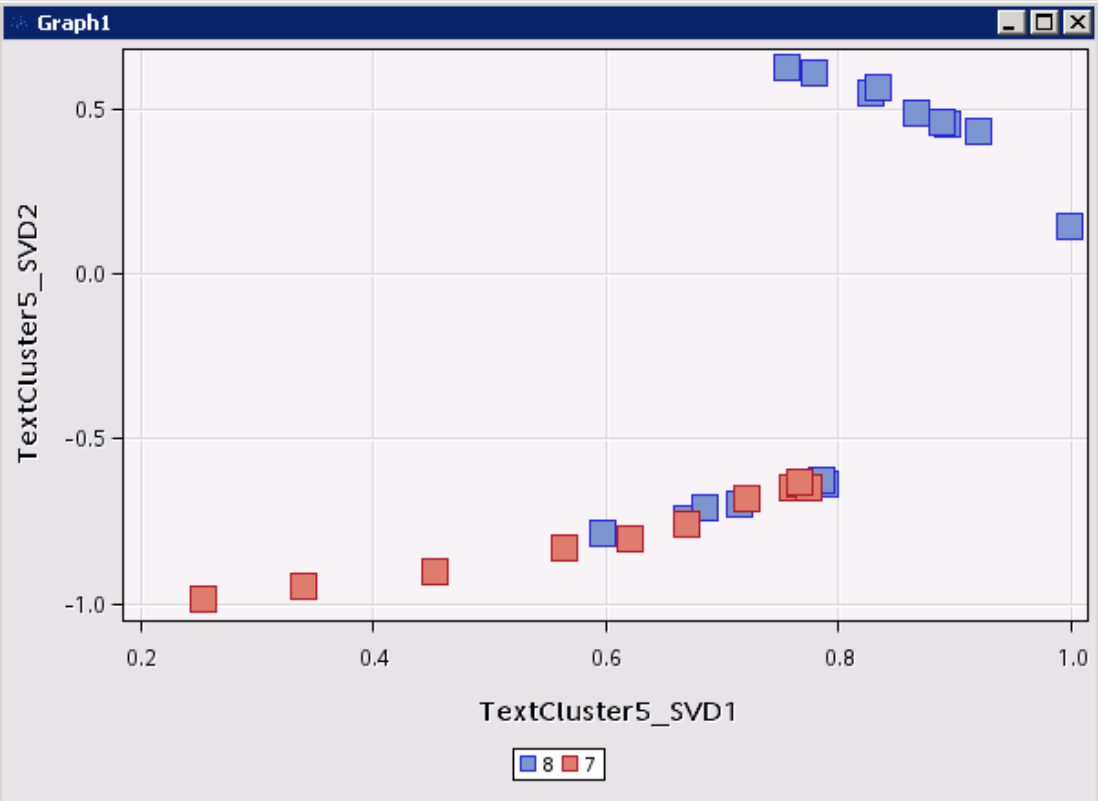

Figure 9. A Plot of a Subdocument Duplication Example

SUPERVISED LEARNING

Supervised learning presents a challenge when sentences or other subcomponents of documents are to be used. Each sentence needs a label for training, but since the label provided corresponds to the document, it is not always clear how to do this assignment. You could transfer the label from the

[1] SAS Text Miner normalizes the projections to unit length, so you shouldn't choose to plot only two if more were calculated.

document to each of its sentences, but this defeats the purpose of isolating aspects of a document that might be more informative versus those that perhaps misrepresent the label.

Manually labeling sentences is too time-consuming and not a realistic solution either, so ensuring that each sentence is labeled properly often requires a different approach. Active learning or other semi-supervised approaches are possible. They involve labeling some representative sentences and then making a prediction about your unlabeled sentences. The previously unlabeled sentences with the highest posterior probability are then labeled according to the model (and perhaps reviewed by the modeler) and the process repeats. Eventually you have a collection of sentences from your document, some of which are labeled positive and the others labeled negative. However, unless they all are reviewed, not all of these sentences will be accurately labeled and predictions will suffer.

Another approach is to just select sentences that are the most likely to be indicative of the label and throw the others out. The subset receives the same label as the original document, so there is no decision to be made. The desired outcome is that the selected sentences create a smaller, but information-rich, document for the prediction. Depending on the content you are learning with, this might be done by choosing the first few sentences of each document, the abstract of a document, or, if you have additional information, you could choose sentences with certain characteristics such as sentences containing a keyword.

CONCLUSION

Vector space models are dependent on the context that you generate term counts from. Typically, in text mining, the document provides the context. In this paper, the perspective shifted from documents to the sentences they contain. The results of the analysis also shifted from a discussion about main themes and topics in a document to a more refined analysis driven by the local information within each sentence.

The sentence analysis led to the following results:

- clusters were delivered that were no longer exclusively describing a document's primary theme but also included secondary themes found within document
- duplication within a document was detected, even if the documents themselves were quite different overall
- term pairs were discovered that were important collocations
- visual relationships between sentences in multiple documents were created

The applications for a sentence-based approach are not limited to what was shown here. Whenever you feel that more local information between terms is important, you should consider this approach.

Finally, the sentences themselves provided a convenient and natural context for the creation of the vectors, but you are not limited to that context. Your context could be based on document structure such as an abstract or elements in a list, or your context could be done with a sliding window across your document that captures n-grams for each vector. Your text mining goals and your data will drive these decisions.

ACKNOWLEDGMENTS

The authors would like to thank Joan Keyser for her editorial contributions.

REFERENCES

Albright, R. 2004. SAS Institute white paper. "Taming Text with the SVD." Available at ftp.sas.com/techsup/download/EMiner/TamingTextwiththeSVD.pdf.

Ceska, Z. 2008. "Plagiarism Detection Based on Singular Value Decomposition." *Proceedings of the International Conference on Advances in Natural Language Processing*,108–119.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Russ Albright
russell.albright@sas.com

James Cox
james.cox@sas.com

Ning Jin
ning.jin@sas.com