Cosine Similarity and PCA

Linas Vepstas

June 19, 2017

Abstract

A report on the results of applying the principal component analysis (PCA) algo to the word-disjunct pairs, and some spot checks of the cosine similarity between words. Results: the cosine similarity looks excellent. Naive PCA and sparse PCA does not work.

Also a very short note that points out that Link Grammar disjuncts are the same thing as the sheaves of sheaf theory. This has the side-effect of explaining why LG seems to be described by modal logic: apparently, the relation betwen sheaves and modal logic was noticed in 1965, and goes under the name of 'Kripke-Moyal semantics'. Now I know the precise mechanism of why the grammar of natural language also appears to be modal. Interesting.

Introduction

The report is organized in three sections: First, a bried summary of the "clean" datasets I have, where most of the various bugs and issues have been squashed. Second, a quick review of PCA, and the results of applying PCA to one of the datasets. Next, a sampling of the cosine similarity of various word-pairs in the dataset. This looks to be very good; the data appears to be high-quality, based on these samples. Next, the short note on sheaf theory. Finally a conclusion summarizing what's been found.

The primary issue is that naive PCA stinks, or rather, is inappropriate. The data is good, but we still need a fast, efficient clustering algo, or rather, one that is not ad-hoc, but is based on first principles. I'm working on this.

Multiple datasets 3 June 2017

Some summary reports from various different datasets. First, datasets that hold word-pairs, parsed using the LG "ANY" link type: i.e. random parse trees.

Size	Pairs	Obs'ns	Obs/pr	Entropy	MI	Dataset
395K x 396K	8.88M	418M	47.0	19.28	-3.02	en_pairs_sim
138K x 140K	4.89M	140M	28.6	17.73	-2.03	en_pairs_tone_mst
183K x 187K	8.05M	268M	33.3	17.83	-1.84	en_pairs_ttwo_mst
425K x 432K	15.2M	557M	36.6	18.32	-1.93	en_pairs_tthree
134K x 135K	5.54M	174M	31.4	17.67	-1.94	en_pairs_rone_mst
185K x 188K	8.95M	321M	35.9	17.77	-1.79	en_pairs_rtwo_mst
						en_pairs_rthree_mst

The legend is as follows:

Size The dimensions of the array. For word-pairs, this would be the number of unique, distinct words observed occurring on the left-side of a word pair, times the number of words occurring on the right. We expect the dimensions to be approximately equal, as most words will typically occur on both the left and right side of a pair. For word-disjunct pairs, the left dimension counts words, the right dimension counts unique, distinct pseudo-disjuncts.

Pairs The total number of distinct pairs observed.

Obsn's The total number of observations of these pairs. Most pairs will be observed more than once. Distributions are typically Zipfian, as previous sections point out.

Obs/pr The average number of times each pair was observed.

Entropy The total entropy of these pairs in this dataset, as defined previously.

MI The total mutual information for these pairs in this dataset, as defined previously.

Left-Right The left and right entropies, as defined previously. Note that $MI = H - H_{left} - H_{right}$ holds, by definition. Not given for the word-pairs table, because these two are nearly equal, and are half the difference between the entropy and the MI.

The datasets are as below.

en_pairs_sim This contains text parsed from Wikipedia, only. As noted previously, Wikipedia is painfully short of verbs, pronouns.

en_pairs_tone_mst Text from Project Gutenberg "tranche one", mostly all "famous authors", popular, well-known 19th century books. Includes six modern sci-fi/fantasy novels from other sources, and some 20th century non-fiction, including a military appraisal of Vietnam.

en_pairs_ttwo_mst Tranche two - Everything from tranche one, plus fan-fiction from http://archiveofourown.org. Most of the selected texts were 10K words or longer. See the 'download.sh' file for the precise texts.

en_pairs_tthree Tranche three - Everything in tranche two, plus several hundred of the most recently created Project Gutenberg texts, whatever they may be. See the 'download.sh' file for the precise texts.

en_pairs_rone_mst Same as en_pairs_tone_mst, but with minor issues fixed. Ditto rtwo, rthree.

Next, datasets that hold disjuncts.

Size	Csets	Obs'ns	Ob/cs	Entropy	H_{left}	H_{right}	MI	Notes
37K x 291K	446K	661K	1.48	18.30	16.00	10.28	-7.98	en_pairs_sim
133K x 1.86M	3.57M	7.66M	2.14	20.55	14.71	10.00	-4.16	en_pairs_tone_mst
176K x 3.40M	6.43M	14.3M	2.23	21.01	14.91	10.01	-3.91	en_pairs_ttwo_mst
378K x 7.18M								en_pairs_tthree_mst
131K x 2.24M	4.28M	9.33M	2.18	20.72	14.93	9.94	-4.15	en_pairs_rone_mst
179K x 3.98M	7.50M	17.0M	2.27	21.15	15.11	9.95	-3.91	en_pairs_rtwo_mst
								en_pairs_rthree_mst

Principal Component Analysis (PCA)

The definition of PCA requires a matrix X that connects columns and rows in some way. In the conventional definition, it is a matrix connecting variables and measurements. The variables (the features being measured) are organized in the columns; the measurements in rows. The PCA algorithm effectively computes the eigenvectors of the matrix X^TX, with X^T denoting the transpose of X.

The matrix X^TX is proportional to the covariance matrix between the different features being observed. The principal component is the the direction of the greatest variation in this matrix.

What plays the role of X in the current situation, and how should the principal component be understood and interpreted?

What we have on hand, foundationaly, is the frequency matrix P with components $p(w,d)$ connecting words with disjuncts. It was defined previously as $p(w,d) = N(w,d)/N(*,*)$, and where $N(w,d)$ is the number of times word w has been observed with disjunct d. As noted earlier, $N(w,d)$ is very large and very sparse: typically $200K \times 4M$ in recent datasets, with only 1 entry out of 2^{15} being non-zero.[1] The extreme sparsity indicates that a power-iteration algorithm will be the most efficient for implementing a PCA algorithm.

What we will examine will be the results on several different kinds of matrices M derived from (constructed from) the base data matrix P. In all cases, the features are words, and so in all cases, it is appropriate to write $X = M^T$; that is, we work mostly with the transpose of the matrix X as usually given in standard texts. This follows from the standard Link Grammar dictionary: the word is followed by the disjuncts it can employ.

PCA of the frequency matrix

Should we identify P and X^T, so that $X^TX = PP^T$? We can, but then we don't get what we want. Computing the principal component of this matrix for a recent dataset (see

[1] I plan to send out the revised, expanded statistical analysis "real soon now".

later section below), we get the following vector shown below. The "weight" gives the magnitude of the vector component. The other two columns are the support for the word, and the number of observations, and are shown for comparison.

word	weight	$\|(w,*)\|$	$N(w,*)$
.	0.9358	2031	341112
the	0.1212	1403	106378
and	0.1098	1225	96276
to	0.1035	1276	96308
”	0.0920	506	45809
,	0.0904	1703	111982
a	0.0842	958	73760
in	0.0808	750	56751
of	0.0783	890	64753
his	0.0666	691	48728
it	0.0567	606	44211
with	0.0531	480	33681
him	0.0482	425	30345
that	0.0464	729	49714
for	0.0450	479	33092

What does this mean? What can we do with this? Why is the weight of the period so high? In essence, this vector is stating that the greatest variety of disjuncts are associated with the period. Since periods are sentence enders, and every sentence has one, a link to the end of the sentence will attach to just about any word. That is, the period almost single-handedly accounts for almost all of the variance of the disjuncts in the dataset. The rest of the list is filled out with words that also attach freely and easily to just about anything: "the" should attach only to nouns, but common nouns wildly outnumber all of the other parts of speech put together. Similar remarks for "and" and "to". The comma can connect in a large variety of situations, and the closing quotation mark ” behaves much like a sentence-ender (This particular dataset contains a lot of dialog). The two columns labeled as $|(w,*)|$ and $N(w,*)$ confirms this interpretation: so, $|(w,*)|$ is the total number of unique, different disjuncts that were observed with w, and $N(w,*)$ is the summation over all of the counts with which these disjuncts were seen.[2] The top words, in terms of the variety and number of disjuncts, are more or less the makeup of the principal component of PP^T. This should not be a surprise. Anyway, this is not what we wanted: we want to classify sets of similar words; discovering which words account for the greatest variation in disjuncts is of secondary interest.

PCA of the cosine similarity

We had previously defined the cosine similarity of two words as

$$\mathrm{sim}(w_1,w_2) = \frac{\sum_d p(w_1,d)p(w_2,d)}{\sqrt{\sum_d p^2(w_1,d)}\sqrt{\sum_d p^2(w_2,d)}}$$

[2] If these numbers seem small, it is because they were taken from a sharply filtered dataset, the en_pairs_ttwo_mst dataset with the (50,30,10) cut applied. This cut is discussed later, below.

and so, perhaps we should use this as the basis for judging the similarity of words. This suggests defining a matrix S with matrix components

$$S(w,d) = \frac{p(w,d)}{\sqrt{\sum_d p^2(w,d)}}$$

and then setting setting $X = S^T$ so that $X^T X = SS^T$. The idea here is that PCA allows a whole-set analysis of similarity, rather than point-wise similarity. That is, for normal clustering algorithms, one computes a large number of values for $\mathrm{sim}(w_1, w_2)$ and then employs a clustering algorithm to categorize these, word by word. Here, instead, perhaps PCA can reveal entire clusters in one gulp, by simultaneously evaluating the similarity between all words in a cluster.

Power iteration converges at about half of the rate as for the frequency matrix, which is not a surprise, as the off-diagonal entries are closer to one-another. The PCA vector, however, is not all that different: 0.978 "." + 0.137 "," + 0.080 "the" + 0.068 "to" + 0.067 "and" + 0.039 "a" + ... and so on, the remaining entries filled out in roughly the same order, by the same words, as in the frequency PCA.[3] Why is this? It's worth taking a look at the matrix:[4]

	.	,	the	to	and	a
.	1	0.549	0.731	0.6435	0.668	0.627
,		1	0.711	0.824	0.888	0.765
the			1	0.790	0.744	0.896
to				1	0.906	0.857
and					1	0.755
a						1

So what is this saying? There are plenty of pairs that have greater similarity; here's an arbitrary sampling:

pair	sim
(he, she)	0.982
(this, that)	0.894
(run, walk)	0.878
(big, small)	0.908
(high, low)	0.910
(soft,hard)	0.809
(easy,hard)	0.846
(easy,soft)	0.749

[3] Created as follows:
(define fsi (add-subtotal-filter psa 50 30 10))
(define pci (make-cosine-matrix fsi))
(define pti (make-power-iter-pca pci 'left-unit))
(define lit (pti 'left-iterate feig 8))
(pti 'left-print lit 20)

[4] Created with
(define poi (add-pair-cosine-compute pti))
(poi 'right-cosine WORD-A WORD-B)

So why aren't some of these in the PCA vector?

Computing similarity for all pairs is not manageable. However, if we limit ourselves, then there are 803 words that were observed 1500 times, or more. There are 322003 = 803 * 802 / 2 pairs of these. The top twelve, ranked by similarity, are:

pair	sim
(He, She)	0.997
(Her, His)	0.996
(nodded, sighed)	0.996
(There, It)	0.996
(He, It)	0.995
(Dallas, Two-Bit)	0.995
(Dallas, Jim)	0.995
(But, When)	0.995
(Two-Bit, Mai)	0.995
(Aster, Mai)	0.994
(He, There)	0.994
(After, With)	0.994

These are capitalized, mostly because they are almost surely sentence openers. For example, grammatically, "but" and "when" aren't all that similar, but, as sentence openers, they probably behave quite similarly, from the grammatical point of view.

The top of the list is dominated by capitalized words primarily because these are observed less frequently, and thus have fewer disjuncts associated with them. As a result, fewer observations of disjuncts means that these are less likely to differ.

We conclude: cosine similarity penalizes words for being observed more frequently; for their grammatical usage being scrutinized more carefully. That's bad. We don't want more similarity just because there is more uncertainty.

There are only four pairs in the top 200 that are NOT capitalized. These are:

Pair	sim
(nodded, sighed)	0.9960
(smiled, nodded)	0.9916
(laughed, asked)	0.9915
(laughed, sighed)	0.9874

This seems like an adequate identification of similarity, for now. Not great, but adequate.

The list of 322003 similar words contains 239778 pairs where both words are lower-case. Aside from the four listed above, the next ten are these:

Pair	sim
(she, he)	0.9819
(on, with)	0.9817
(with, in)	0.9793
(around, over)	0.9770
(into, from)	0.9769
(off, down)	0.9754
(on, from)	0.9747
(on, in)	0.9746
(into, through)	0.9721
(on, for)	0.9720

This listing skips four pairs (smiled, laughed) (nodded, laughed) (asked, sighed) (nodded, asked) that would appear in it. Curiously, surprisingly, with the exception of the first pair, these are all pairs of propositions! The top pairs to break out of this pattern are:

Pair	sim
(would, could)	0.97188
(an, a)	0.97168
(night, day)	0.97072
(evening, morning)	0.97029
(water, fire)	0.96707
(black, white)	0.96661
(head, hands)	0.96657
(an, some)	0.96565
(throat, chest)	0.96472
(floor, table)	0.96439
(did, really)	0.96431
(bed, window)	0.96428
(shoulders, chest)	0.96369
(child, question)	0.96314
(didn't, didn't)	0.96283
(mind, life)	0.96192
(three, two)	0.96185

The list seems excellent to me, and almost too good to be true. Looking a second time, though, it has a distinctly 19th century feel to it: the corpus is heavily weighted by project Gutenberg titles, which will certainly be more plain-spoken than what we are used to. The only oddball pair in the bunch is (child, question). Note that the pair (didn't, didn't) differs only in punctuation: vertical apostrophe, versus right-apostrophe.

The other very curious result here is that, although we are looking for grammatical similarity, we are finding something stronger: semantic similarity. So, for example, (night, day) (evening, morning) (water, fire) (black, white) are not just antonyms, they

are almost primal in their opposition: they're pre-historic, pre-civilizational antonyms. The next few: (head, hands) (throat, chest) (shoulders, chest) are not just body parts, but are primal to human perception. Then there's indoor living: (floor, table) (bed, window) (window, table) which recalls the Project Gutenberg sources. The pair (three, two) is primal, the pair (mind, life) is surprising and interesting. The pair (child, question) is is bizarre.

So we are getting not just grammatical similarity in the standard sense of structural linguistics, but we are also getting similarity in the corpus linguistics sense: That is, when people construct sentences with the word "night" in them, they apparently also construct very similar sentences that have the word "day" in it's stead. I think this is very interesting. This is new, and I've never seen the corpus-linguistics people talk about this (but my experience is limited).

A bit of sheaf theory

I recently realized that much of what is being discussed here can be anchored in the vocabulary of a generic mathematical theory, namely, sheaf theory. Sheaves allow topological structure to be discussed in a local way: sheaves describe how the local neighborhoods of a point glue together, to form a manifold as a whole. Link Grammar disjuncts and connector sets are really just the stalks and germs of sheaf theory, in mild disguise. This can be seen as follows.

A standard way of expressing a graph is to list all of the vertexes in the graph, and to list all of the edges. Knowing these, one knows the graph. However, this is a global description: One does not know the local structure until one looks at specific vertexes, and what they attach to.

A different way of describing a graph is to make a list of pairs: a vertex v and all the edges that attach to it. More generally, one can consider pairs where a vertex v is attached to a vertex w by means of a path of length N or less.

$$(\text{vertex } v, \{w \text{ s.t. vertex } w \text{ is attached to } v\})$$

This describes the graph, as a whole, just as well as the simpler vertex+edge list does. However, the language is different: these pairs are presheaves, obeying all the axioms of a presheaf, e.g. the composition of restriction morphisms. They become a sheaf because they also obey the gluing or collation axiom as well: they can be glued together to form the original graph from which they were taken.

Thus, we can see that the set

$$(\text{vertex } v, \{ \text{ edges attached to } v\})$$

is the same thing as a Link Grammar dictionary entry.

To be more precise, we need to distinguish the graph-sheaf that arises for a single sentence (from the dependency parse of the sentence) from the sheaf that arises from the entire language. If we take the language to consist of the set of all possible sentences, then the sheafification is to parse each of the sentences in the language, to get a dependency graph for each sentence, create the individual (word, connector-set) lists,

and then take the quotient, identifying together all words that have the same spelling. This gives the sheaf of the entire language.

From what I can tell, this realization that language can be sheafified is not new; when the language is not a natural language, but is instead first-order logic, then it's sheafification gives the Kripke–Joyal semantics. According to Wikipedia, this was noted in 1965 for existential quantification. I don't know if this was ever noted for natural language before, but, as I've blathered on the mailing list before, this provides the "answer" to why the logic of Link Grammar appears to be modal logic: Link Grammar dictionary entries are sheaves, and the disjuncts are the different "possible worlds" that a given word can inhabit. For a natural-language sentence, "there exists" (existential quantification) a collection of disjuncts that can parse the sentence. Bingo.

I used to say that LG disjuncts had something to do with linear logic, because linear logic also has the general whiff of "possible worlds" around it. I now see that in fact its actually modal logic, and it is the language of sheaves that provides the direct route from Link Grammar, to modal logic. It would be very interesting to see all the details worked out.

Most interesting is perhaps this: the sentences of a language are observed with some a priori frequency or probability. What's the correct way of converting this to a probability distribution on the sheaf? Next, given a probability distribution on the sheaf, what is the corresponding probability distribution on the corresponding modal logic?

It seems to me that one could make this very generic: every language, and not just first-order logic, but any language, as considered in model theory, has a set of sentences. This sentences are composed of the terms in thier term algebra, and these terms and how they connect, define a graph. That graph can be viewed in terms of sheaves, germs, stalks, etale spaces. This implies that every model, of model theory, has a corresponding cohomology. Writing this out could be interesting. Perhaps this has already been done; perhaps this is what topos theory is. But I suspect that it's not been sufficiently popularized: certainly, the standard computer-science textbooks that tell you what a language is do not tell you that it has a cohomology associated with it. And yet, this seems blantly obvious, in retrospect, and naggingly it might actually be important for some reason or another.

Anyway: this is not just all-talk, no-action. I've written some code that implements some sheaf-based parsing on the atomspace. It is in the github atomspace repo. The README file there explains more.

Conclusions

Conclusions from the above:

- Pair-wise similarity is very promising.
- The cosine similarity measure penalizes better, more accurate measurements, because better measurements are more likely to find dissimilarity. We need a better measure.

- PCA and sparse PCA, in the naive sense, applied to frequencies, or to cosine similarities, are inappropriate for classification. Its still possible that perhaps PCA applied to some sigmoid of the cosine similarity (e.g. cosine to the fourth power) might work better, but the selection of this sigmoid seems ad-hoc, and not anchored in any principles.
- First principles suggest something Bayesian, based on the Gibbs measure, maybe some sort of hidden multi-variate logistic regression. Hidden, because we don't know the grammatical categories in any a priori sense; we must deduce them.
- It would be great if someone worked out the precise details in going from sheaves to modal logic. This was already done, in 1965, for topos theory; no one has done this for natural language, though.

The End