Context is Everything: Finding Meaning Statistically in Semantic Spaces

Eric Zelikman
Stanford University
ezelikman@stanford.edu

Abstract

This paper introduces a simple, explicit, and invertible measure of word importance in a global context, even on tiny (10+ sentence) contexts. In this paper, applications of the statistical properties of the contextual word-vector cloud distribution (with a simple bag-of-words model) substantially outperforms the state of the art (SOA) for subjectivity/objectivity analysis, as well as paraphrase detection, and falls within other state-of-the-art models for six other transfer learning tests. After generating a word-vector space containing both 2-gram clauses and single tokens, it became clear that more contextually significant words disproportionately define clause meanings. Applying this idea more generally allowed for a simple and powerful sentence embedding generation technique, which was then extended to a sentence/document summarizer, an improved (and context-aware) cosine distance and a simple document stop word identifier.

1 Introduction

Global Context

All current major approaches to taking global context into account are essentially black boxes: the algorithm takes in a document and returns some function that takes a given word vector is and returns its context measurement, which can be anything from a vector [1] to another deep structure [2]. This unfortunately results in obfuscated approaches that are challenging to meaningfully invert, resistant to further extension. Further, these algorithms often require a fairly large context in the transfer dataset to train, especially when they are unsupervised (auto-encoding)[1]. They act, essentially, as black boxes. However, in this paper, a simple approach relying on normalization of deviation with the Mahalanobis distance is developed, then combined with a variety of machine learning techniques.

It is generally accepted that the words that are important in a document are rarer. This is emphasized by the success of the performance of the inverse-frequency-weighted bag-of-words model for sentence embeddings in Arora et al.'s "A Simple but Tough to Beat Baseline for Sentence Embeddings" [3], which applies that principle directly and remains near the state of the art. A natural extension then, is to measure the "unusualness" of a word vector relative to its document's word-vector distribution. Upon finding that this correlated fairly well with tf-idf, the natural following question is how this correlates to the meaning of a set of words, which yielded a sigmoid [1].

[1] In essence, words that are slightly more globally contextually important than the others in the sentence contribute much more to the meaning of the sentence than words that are slightly less contextually important

Sentence and Document Summarization

Combining the sigmoid pattern with a weighted bag-of-words yields some state-of-the-art results for transfer learning, but as a linear combination of words which should be in the same word-vector space as the initial words, this method carries an interesting implication: One could, knowing a subset of a sentence's words and the sentence's sentence vector, easily calculate the vector of the remaining text. The assumption is made that one of the closest words to a substructure's vector will be somewhat representative of the substructure. Thus, the algorithm solves this as a path finding algorithm for the optimal path from an empty word list to the word list best approximating the sentence embedding.

One of the issues with this was that the metric to evaluate the distance of a word to the remaining substructure was initially the traditional cosine distance metric. However, due to issues inherent to cosine distance[2], there were occasional failures to converge at a meaningful solution. Noting that M-distance can also be calculated between two points, and that the distance between the distribution and the words was already a standard part of my algorithm, an application of the law of cosines resulted in a better and context-aware alternative to cosine distance, capable of realizing that ”cardinal” and ”red” are less related in the context of an essay about Stanford than in an article about the color green.

2 Background and Related Work

2.1 Sentence Embeddings in Context

The context of a dataset is often recognized as useful in tasks from interpreting a user's next search query to document summarization[4], and the significance of global context was highlighted in the recent paper, ”Deep contextualized word representations” [2] which uses a supervised model on a dataset to create a biLSTM encoder. In all of these papers, it is pointed out that the short term nature of the memory of LSTMs makes it difficult for such a model to naturally learn global context, especially with datasets of small snippets. While learning global contextual significance of a word (especially with high-dimensional word vectors) requires degrees of freedom prone to over-fitting, an importance measure can provide this extremely useful information to the learning algorithm. Other previous research generated contextual vectors for words instead, calculated by an autoencoder trained on a translation model [1] or an approach to transform word vectors based on global context using a biLSTM [5]. While the usefulness of transfer learning on a small scale [4], the learning of global context [2], in some cases combined with multiple-word meaning word vectors [6], this approach provides a transparent and easy-to-implement technique which also works on tiny contexts.

2.2 Mahalanobis Distance

The Mahalanobis distance, or M-distance, is a simple way to find a distance between a point and a distribution or between two points in the space of a distribution, normalized for deviation and covariance [7]. Though the implementation simply uses the SciPy built-in function, the formula for two points p_1 and p_2 and distribution covariance S is simply:

$$d(p_1, p_2, S) = \sqrt{(p_1 - p_2)^T S^{-1} (p_1 - p_2)}$$

While the M-distance has been applied in the past in the context of word vectors, the uses have ranged from measuring the distance between ”Gaussian word embeddings” [3] [8], an alternative to cosine distance for one-shot image recognition [4] [9] and movie review sentiment analysis [10] or to help discriminate word sense in context [11].

[2] Further elaborated in background

[3] Word embeddings which have a mean and deviation in every dimension instead of simply a single point, for spanning ambiguities. The covariance used was the joint covariance of the word embeddings

[4] The issue with this approach for a generalist tool is that word vectors that are closely related in meaning to another word vector, but where one has one component that is much more unusual, will result in very unrelated words appearing more similar. For example ”dog” and ”but” are both commonly used words, and in many

2.3 tf-idf

As discussed in Arora et al.'s "A Simple but Tough to Beat Baseline for Sentence Embeddings" [3], a slightly improved weighted average of words by their tf-idf importance yields a remarkably robust approximate sentence vector embedding. tf-idf, however, comes with a variety of issues. First, to produce meaningful results even for fairly common words requires a large context. It is also stratified for rarer tokens, resulting in importance bands for a document as words get increasingly rare. Furthermore, it fails to provide useful results for tokens not in the context or account for covariance in word meaning, and thus is extremely word-choice dependent.

2.4 Cosine distance

Cosine distance is the traditional measurement of similarity between word vectors, varying less with increasing number of dimensions and ignoring differences in importance more than a simple measure of Euclidean distance [12]. However, cosine distance implies that all dimensions are equally valuable, as it calculated as a dot product along the dimensions.

3 Approach and Experiments

3.1 Analyzing M-distance vs tfidf

The initial test that was necessary was a simple implementation of the Mahalanobis distance of words in a corpus compared to their tf-idf. The context used here was the Stanford Sentiment Analysis Treebank dataset [13]. Both the normalized and the unnormalized GloVe word vectors[5] [14] were compared, to see how much of the distance relationship was encoded in the initial distance of the word vectors. The normalized approach resulted in substantially better results, presumably because the length of the original word vectors was correlated with their frequency of use, which caused the document cloud to be less predictive.

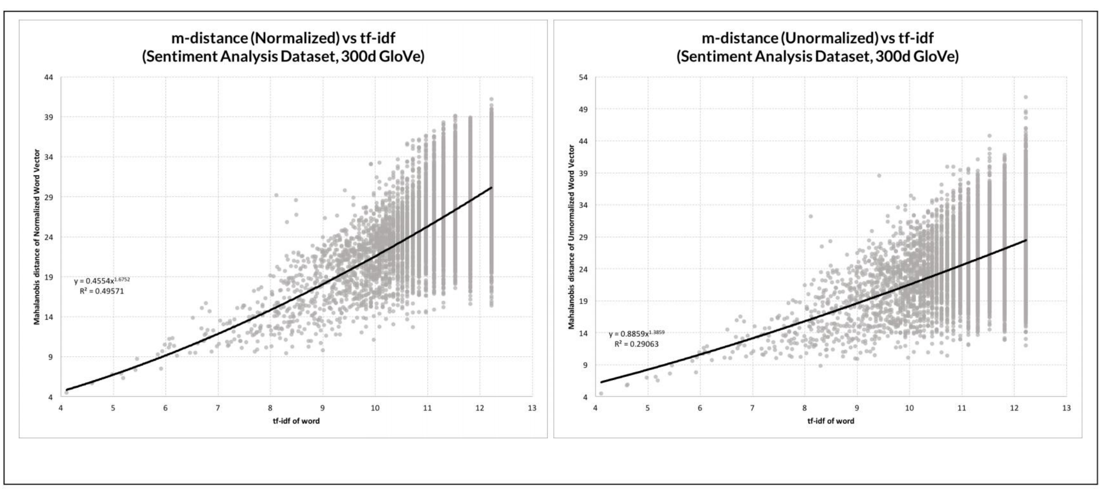

Figure 1: Relationship of Mahalanobis distance with 300d GloVe to tf-idf on Stanford Treebank Dataset normalized and unnormalized

spaces will have a smaller M-distance than "Dachshund" and "dog." One solution to this issue is presented later in this paper

[5] 300 dimensional word vectors from 42B token Common Crawl

3.2 The Unified 2-gram and Token Space

Modifying an implementation of GloVe in TensorFlow, tf-glove [15], by using spaCy [16] to identify two-word clauses, the algorithm randomly treated a two-word clause as a single token 50% of the time, and the remaining times replaced it with one of the underlying words (To prevent the typical clause position in the sentence from biasing its word vector, given the reduced context at the beginning or end of a sentence). In order to analyze the relationship between the clause and its constituent word vectors, the optimal linear combination of the two words to produce the curve [6].

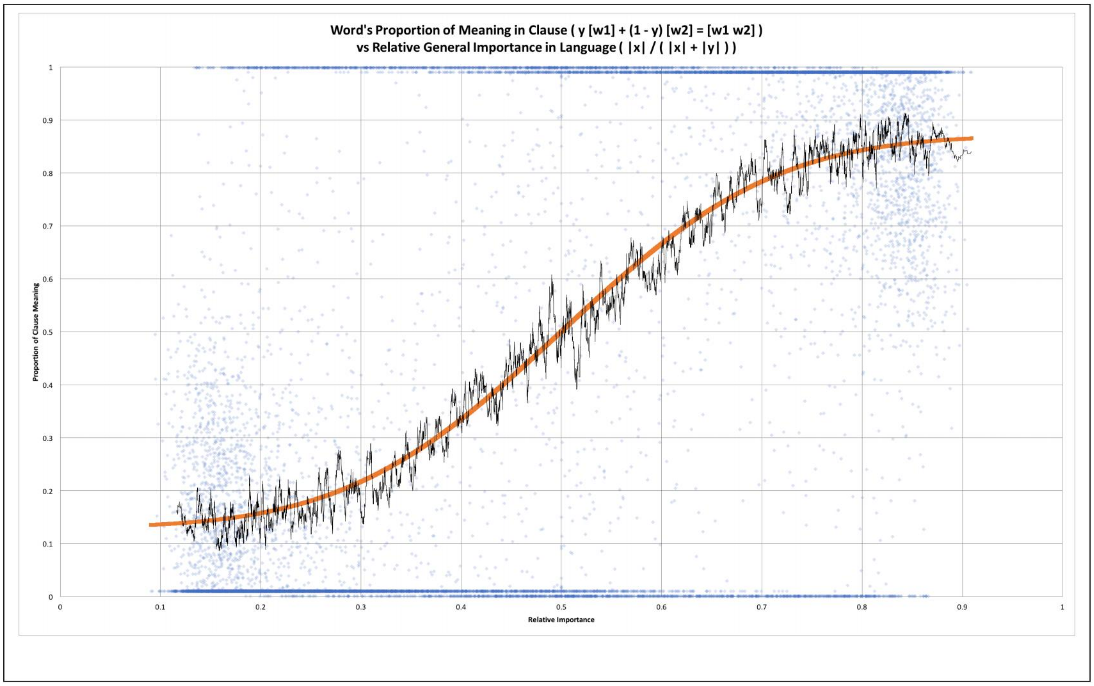

Figure 2: The proportion of a clause's word vector that is determined by a given word, as a function of the word's proportion of the sum of the two words' importances, shown with a moving average (black) and a sigmoid (orange, error function) fit to the data

3.3 Generating and Evaluating Sentence Embeddings

At this point, GloVe was replaced with fastText [17], which, although arguably having less predictive word vectors in general [18], was used due to its ability to predict word vectors for misspelled or extremely esoteric words, by performing a character level prediction to avoid ignoring the importance of any words. This was used with PyMagnitude, which offered extremely fast word vector lookup [19].

Challenges

One of the major challenges of extending the simple two word model to a sentence embedding model is that taking the sigmoid of the proportion of total importance will result in longer sentences necessarily having all the words on the left side of the sigmoid. Thus, in the denominator, some kind of average[7] needs to take the place of the sum of the words. A variety were tested, including a simple non-sigmoidal weight as a baseline, the peak-end approach which was computationally cheapest and surprisingly effective, the mean, the root-mean-square, and the harmonic mean[8]. The

[6]Note that the shape of the curve is almost identical when the clauses containing stop words are removed. The determination of stop words is discussed further in the sentence embedding subsection

[7]Referring to any measure of central tendency

[8]The importance of the final sentence vector was also used, iteratively, but this approach was more costly and had underwhelming performance

Table 1: Performance Compared to State of the Art (SOA).
ST will represent SkipThought in the following table for space reasons. The GloVe word vectors from assignment 1 are used in this model for time reasons. The baseline is Arora et al.'s "...Tough to Beat Baseline for Sentence Embeddings" [3].

Test	Baseline	Unsupervised SOA (SOA*)	BOW
CR: Product Reviews Sentiment	**79.2**	**83.1** ST-LN	79.42
MPQA: Opinion Polarity	82.4	**89.3** ST-LN (90.2[13]	88.28
SUBJ: Subjectivity/Objectivity	90.3	93.7 ST-LN (95.5 AdaSent)	**99.6**
TREC: Question Type (who, what, etc)	**85**	**92.2** ST	79.6
MRPC: (Unclear)	73.6/81.7	73.0/82.0 ST	**74.72/82.16**
SICK-E: SICK Entailment	-	**84.6** SIF (86.3[11])	78.02
SST: Movie Review	-	**82.9** ST (84.6[11])	81.38

other question was which parameters of the sentence were relevant to determining the mean, in other words, what preprocessing was useful.

A recursive approach was also taken, where the sentence was broken down by clause by spaCy [16] and then constituent clauses were recursively merged according to the sigmoid. Unfortunately, it clearly underperformed the bag of words approach from the start, so was dismissed.

Evaluation Technique

Facebook's SentEval [18] suite of tests was used to evaluate the quality of the transfer-learned sentence embeddings, which includes subjectivity detection (SUBJ), product reviews (CR), entailment[9] (SICKEntailment), opinion polarity (MPQA), question-type identification (TREC), movie reviews (MR), paraphrase detection (MRPC), semantic similarity (STS), and other analyses. The tests perform a gradient-descent based classification of the sentence vectors to their corresponding labels.

Stop Word Identification

After a variety of tests, the best results (Using the SentEval-provided dev results) it was found that a stop word cutoff was useful in calculating the mean. Stop words, which happened to closely mirror the bottom 20% of words, were overrepresented[10] and disproportionately skewed certain mean evaluations.

Preprocessing

The harmonic mean turned out to be the most useful measure. Furthermore, in the SentEval tests, tokens which contained no alpha characters ultimately detracted from performance. Furthermore, presumably because FastSent contained fewer capitalized words, and capital words are a small portion of English and therefore have a substantially higher Mahalanobis distance, case-insensitive performance was generally better case-sensitive. Depending on the mean used, augmenting the sentence vectors with an extra dimension containing the sentence importance resulted in better performance.

Curve

The sigmoid curve used, which was an erf function[11] compressed vertically by 3.2x[12] and stretched horizontally by 4.2x, was essentially the same as the fit shown generated in the earlier Figure 2, where the vertical compression of 2.7x and horizontal stretch is also 4.2x.

[9] Whether a given sentence implies another sentence

[10] Did not follow the otherwise normal distribution

[11] $0.5 + erf((x - 0.5) * h)/v$

[12] Note that 2x here refers to a sigmoid function with a range from 0 to 1

[13] BiLSTM-Max A11NLI

Results

As shown in Table 1, this approach vastly outperforms even supervised state of the art models for subjectivity assessment and paraphrase detection, and performs on the higher end of the SOA approaches for all other tests [14]. The tf-idf model is unable to perform in the large movie review dataset (SST) and SICK Entailment.

Accuracy and Amount of Context

One relevant secondary question is how many samples from the training set are actually necessary to establish a context distribution. While the performance is above the ”tough to beat baseline” for most tests after only about 10 sentences, every additional example appears to improve performance logarithmically.

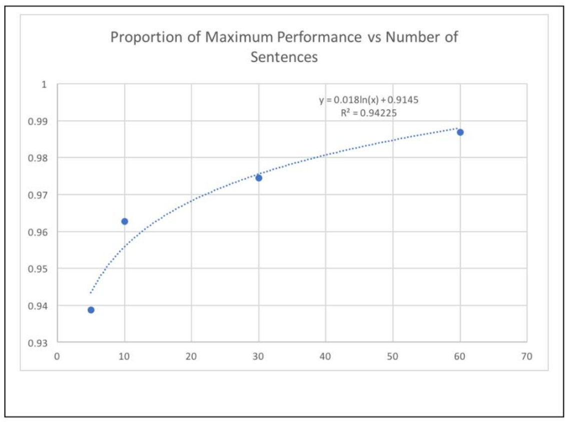

Figure 3: The accuracy in MRPC as a function of number of training tests given, as a proportion of the ultimate accuracy

3.4 Sentence Summarization

In order to summarize a sentence from the sentence vector, the closest 5 words to the sentence are chosen. Then, they are fed into a priority queue by a proportion of the current distance and a proportion of the the distance after they are added. This is a slightly modified A* algorithm.

3.5 M-cosine Distance

Due to the limitations of cosine distance, a normalized approach provides substantially improved results. Treating the M-distance between the words as the opposite leg of the triangle and treating the distances to the the words as the legs, the law of cosines is solved for the cosine of the angle between them [15].

3.6 Document Summarization

Applying the same basic technique used for sentence summarization to documents, the sentence vector of every sentence is calculated, then the document as a composition of sentences. The M-cosine distance is then used to find the closest sentences, then to remove those sentence vectors from the document meaning, until the best distance for a given depth is reached.

[14] Note that the MR test is essentially a toy test, with only 74 examples in the test set, so slight fluctuations result in substantial apparent performance changes, so it is not included in these results. This model scores 70.42 while the baseline scores 73.7. All examples in MR are a subset of SST, so SST is included instead.

[15] $\frac{a^2+b^2-c^2}{2ab}$ where a and b are the legs and c is the opposite side

4 Conclusion

Beyond a set of new useful calculations to augment natural language processing algorithms, and a novel subelement summarization technique, this carries implications about the compositional structure of language beyond that which is typically implied by a deep learning contextual approach. This should serve as a new baseline for simple transfer learning in general, and presents unprecedented results on tiny dataset transfer learning.

Acknowledgments

I would like to express thanks to Zewei Chu, who suggested using SentEval and with whom I hope to further develop this research, with whom I originally implemented (and saw the weaknesses) of the tf-idf sentence embedding approach at a hackathon. I would also like to thank Sarina Wu and Samuel Jacobo for reading over this paper for clarity.

References

[1] B. McCann, J. Bradbury, C. Xiong, and R. Socher, "Learned in Translation: Contextualized Word Vectors," *ArXiv e-prints*, July 2017.

[2] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *ArXiv e-prints*, Feb. 2018.

[3] S. Arora, Y. Liang, and T. Ma, "A simple but tough-to-beat baseline for sentence embeddings," 2017.

[4] Z. Lu, Y. Zhu, S. Pan, E. Xiang, Y. Wang, and Q. Yang, "Source free transfer learning for text classification," 2014.

[5] M. E. Peters, W. Ammar, C. Bhagavatula, and R. Power, "Semi-supervised sequence tagging with bidirectional language models," *ArXiv e-prints*, Apr. 2017.

[6] E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng, "Improving word representations via global context and multiple word prototypes," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2012.

[7] R. Gnanadesikan and J. R. Kettenring, "Robust estimates, residuals, and outlier detection with multiresponse data," *Biometrics*, vol. 28, no. 1, pp. 81–124, 1972.

[8] L. Vilnis and A. McCallum, "Word Representations via Gaussian Embedding," *ArXiv e-prints*, Dec. 2014.

[9] S. Rahman, S. H. Khan, and F. Porikli, "A Unified approach for Conventional Zero-shot, Generalized Zero-shot and Few-shot Learning," *ArXiv e-prints*, June 2017.

[10] V. Balasubramanian, S. Gupta, and P. Veerappagoundar, "Mahalanobis distance-the ultimate measure for sentiment analysis," vol. 13, pp. 252–257, 01 2016.

[11] M.-C. De Marneffe, C. Archambeau, P. Dupont, and M. Verleysen, "Local vector-based models for sense discrimination," 03 2018.

[12] C. Emmery, "Euclidean vs. cosine distance," Mar 2017.

[13] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," 2013.

[14] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation.," in *EMNLP*, vol. 14, pp. 1532–1543, 2014.

[15] G. Simon, "tensorflow-glove," Mar 2017.

[16] M. Honnibal and M. Johnson, "An improved non-monotonic transition system for dependency parsing," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, (Lisbon, Portugal), pp. 1373–1378, Association for Computational Linguistics, September 2015.

[17] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin, "Advances in pre-training distributed word representations," in *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[18] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," *arXiv preprint arXiv:1705.02364*, 2017.

[19] A. Patel and Alex, "plasticityai/magnitude: Release 0.1.14," Mar. 2018.