Exploiting Gaussian Word Embeddings for Document Clustering

Inzamam Rahaman, Patrick Hosein
University of the West Indies
Email: inzamam@lab.tt, patrick.hosein@sta.uwi.edu

***Abstract*—Every day, millions of documents in the form of articles, tweets, and blog posts are generated by Internet users. Many of these documents are not annotated with labels making the task of retrieving similar documents non-trivial. The process of grouping similar unannotated documents together is called document clustering. When properly done, a cluster should only contain documents that are related to each other in some manner. In this paper we propose a clustering technique based on Gaussian word embeddings which we illustrate using word2gauss. We demonstrate that the method produces coherent clusters that approach the performance of K-means on Purity and Entropy scores while achieving higher recall as measured by Inverse Purity.**

***Keywords*—Document clustering; word embeddings; information retrieval**

I. Purpose

Document clustering is the unsupervised learning task of partitioning a corpus of documents into clusters such that related documents appear in the same cluster. It is an important information retrieval task as it can be used to simplify web searching [1] or aide in the efficient navigation of documents by end-users [2]. This can be used to reduce information overload especially when a large number of documents are involved [3]. Moreover, document clustering can also be used as an initial step in developing recommendation systems [4] and multi-document summarization [5] - both being tasks that reduce the cognitive load on end-users.

Distributed word embeddings, such as word2vec [6], are increasing in popularity as they allow words - non-numerical objects - to be mapped into numerical spaces such that similar words are embedded into "near" to one another under the chosen metric(s), thereby reducing semantic problems to geometric ones amenable to more well-studied machine learning and data mining techniques. One such distributed word embedding is Vilnis and McCallum's word2gauss [7]. In word2gauss, words are embedded in the space of Gaussian Distributions as opposed to points in some vector space like in word2vec. To do this, they utilized energy based learning in which pairs of words that appear together frequently score higher under the supplied energy function than pairs that are infrequent or non-existent in the corpus. For example, if we let E denote the energy function used and our corpus comprises the sentence "The quick brown fox jumped over the lazy dog", with a window size of 2, $E(\text{quick}, \text{fox}) > E(\text{quick}, \text{lazy})$. As discussed in their paper [7], Vilnis and McCallum examined the inner product (IP) and Kullback-Leibler divergence (KL) between the representative distribution of words as energy functions for training their representations using stochastic gradient descent.

If two documents are related, then it is reasonable to anticipate that their constituent words ought to be related as well. In terms of Gaussian word embeddings trained by word2gauss, this would mean that if two documents D_1 and D_2 are related, then we should be able to exploit the energy values for pairs of words between documents to indicate this relationship. In this paper, we explore how Gaussian word embeddings may be exploited to cluster documents.

II. Background

Methods for clustering may be classified as either hierarchal or partitional. In hierarchal methods, clusters are nested in one another and can be visualized as a dendogram where the documents being clustered constitute the leaf nodes. Hierarchal clustering can be either agglomerative or divisive [1]. In the former, each document would start off as a cluster onto itself and clusters are repeatedly merged into larger clusters according to some criteria until one large cluster containing all documents are formed. In the later, the entire corpus is considered one large cluster at the start, and this large cluster is broken down into successively smaller clusters until we are left with clusters containing a single document. Unweighted Pair Group Method with Arithmetic Mean (UPGMA) is an example of a hierarchal clustering method used for document clustering [1]. In contrast, partitional methods of clustering find flat clusters at once without merging or dividing the clusters found [1]. K-means clustering is a quintessential example of a partitional method of clustering.

The task of document clustering is often a challenging one. While many types of data are structured, text often is not. To facilitate clustering documents, we derive structure from text by mapping text to some numerical representation such as a bag of words [8] or TF-IDF [9] vector. In addition, document clustering may involve using external sources of information such as ontologies derived or enhanced from Wikipedia [10].

However, using representations such as TF-IDF vectors or bag of word vectors is not without disadvantages. One such disadvantage is that semantic relationships between related pairs of words are lost. Some clustering methods such as Hu et al. [10] and Hotho et al. [11] address this by using external information to supplement the information provided by the aforementioned numerical representations. However, these methods use data that are external to the corpus. In this paper, we present a method that exploits the relationships between words encoded in Gaussian word embeddings derived

978-1-5386-2823-2/17/$31.00

by word2gauss [7] to cluster documents without resorting to external information.

III. Method

A. Preprocessing

In order to reduce the size of the vocabulary and reduce the noise introduced by suffix usage and formatting, we applied several preprocessing steps to our corpus. First, all non-alphabetical characters were removed from the document and all words were translated to lowercase. After conversion to lowercase, we removed the stop words defined by Porter [12]. Following the removal of stop words, we used a Snowball stemmer [13] to map words to their predicted stems, thereby eliminating the obfuscation of the underlying concepts in the documents by the suffix transformations of its constituent words. After creating the above corpus, we created a vocabulary using all of the words contained in the corpus. We then trained word2gauss using this corpus on the generated vocabulary to obtain the Gaussian word embeddings.

B. Energy between Documents

Suppose that a document D can be represented as an ordered sequence of words w^j where j represents the position of the word in the document. Let $|D|$ denote the number of words in document D. We compute the energy between two documents D_i and D_j as follows:

$$E(D_i, D_j) = \frac{1}{|D_i|} \sum_{m=1}^{|D_i|} \sum_{n=1}^{|D_j|} E(w_i^m, w_j^n) \tag{1}$$

For N documents, we use the above to compute a matrix Π such that $\Pi_{ij} = E(D_i, D_j)$. After its initial generation, this matrix is then normalized using min-max normalization. In addition, we also generate a vector Γ such that $\Gamma_i = \max_j \Pi_{ij}$. Note that the computation of the energies between documents can be computationally demanding, especially when the number of dimensions used to train the Gaussian distributions is high. However, our method for determining the energy between documents can be easily parallelized. This coupled with memoization on the energies between words, reduces runtime and can make the process more scalable over larger document sets.

C. Clustering Process

Next we construct the clusters using a Chinese Restaurant inspired algorithm detailed below. Note, however, our algorithm is sensitive to the order in which documents are processed. Consequently we sorted the corpus in ascending (Asc) and descending (Desc) order based on Γ. We also considered the case of a random ordering of the documents as a baseline to show that the ordering of the corpus affects the quality of the clusters formed.

Algorithm 1 Algorithm for Document Clustering using Energies Between Documents

Input: Π - matrix of energies of document pairs, D - the documents to cluster, N - the number of documents, M - the maximum number of clusters

Output: C - the clusters

Set C to empty list
nc $\leftarrow 0$
pnew $\leftarrow 1$
for $i = 1$ to N **do**
  pnew $\leftarrow \frac{1}{1+\text{nc}}$
  doc $\leftarrow D_i$
  u $\leftarrow$ sample from $U(0, 1)$
  if u $\leq$ pnew **then**
    if nc $< M$ **then**
      $c \leftarrow$ (doc)
      Append c to C
      nc $\leftarrow$ nc $+ 1$
    else
      $c_i \leftarrow \max_{i=1}^{\text{nc}} \max_{j=1}^{|C_i|} \Pi_{doc,C_{ij}}$
      Add doc to C_{c_i}
    end if
  else
    $c_i \leftarrow \max_{i=1}^{\text{nc}} \max_{j=1}^{|C_i|} \Pi_{doc,C_{ij}}$
    Add doc to C_{c_i}
  end if
end for
return C

IV. Experimental Results

A. Dataset

To evaluate our results, we used the Reuters-21578 dataset bundled with *nltk* [14]. Since the categories assigned to articles in the Reuters-21578 dataset would indicate a similarity in the concepts between two articles, we exploited the fact that the documents were annotated to determine the quality of the clusters generated by our method. We extracted 500 documents from the Reuters-21578 dataset that were assigned to a single category. We omitted those documents that were assigned more than one category as evaluating the quality of such clusters accurately would have been more difficult.

After processing these 500 documents using the aforementioned preprocessing steps, we obtained a vocabulary of 3984 words. These words were trained on the document set using word2gauss. When training with word2gauss, we kept the window size, number of samples per word, and the number of dimensions to 15, 10, and 150 respectively. Moreover, we considered, for each case, both covariance types - diagonal (Diag) and spherical (Sph). Our document set contained 40 topics. Since we assumed that in the best case each cluster would represent a single topic, we assumed the "true" number of clusters was 40. However, we also performed experiments with 20 and 60 clusters to yield insight into how the number of clusters affects the quality of the clusters formed.

B. Metrics

Even though document clustering is an unsupervised learning task as discussed by Amigó et al. [15], there are several metrics that can be used to assess the quality of clusters

formed when we use a labeled dataset. These metrics leverage the assumption that the labels provide sufficient insight into relative relatedness of documents in a cluster by framing a cluster as the outcome of a query.

Since clusters are framed as the outcome of a query, some of the metrics we used can be defined in terms of precision and recall [15] as defined below:

$$\mathsf{Precision}\ (C_i, L_j) = \frac{|C_i \cup L_j|}{|C_i|} \tag{2}$$

$$\mathsf{Recall}\ (C_i, L_j) = \mathsf{Precision}\ (L_j, C_i) \tag{3}$$

where C_i is the i^{th} cluster and L_j is the j^{th} label. Using the above, we may compute Purity (Pr) and Inverse Purity (InvPr) as follows:

$$\mathsf{Pr}\ (C) = \sum_{i=1}^{|C|} \frac{|C_i|}{|D|} \max_j \{\mathsf{Precision}\ (C_i, L_j)\} \tag{4}$$

$$\mathsf{InvPr}\ (C) = \sum_{j=1}^{|L|} \frac{|L_j|}{|D|} \max_i \{\mathsf{Recall}\ (C_i, L_j)\} \tag{5}$$

In an ideal clustering, both Pr and InvPr would be equal to 1. It would also be insightful to get some sense of the distribution of the labels in a cluster. This may be determined by the entropy of a cluster defined by:

$$\mathsf{Entropy}\ (C_i) = \frac{-1}{\log |L|} \sum_{j=1}^{|L|} \mathsf{Precision}\ (C_i, L_j) \log \mathsf{Precision}\ (C_i, L_j) \tag{6}$$

For a set of clusters, we may then compute a weighted average over the entropy of each cluster:

$$\mathsf{Entropy}\ (C) = \sum_{i=1}^{|C|} \frac{|C_i|}{|D|} \mathsf{Entropy}\ (C_i) \tag{7}$$

C. Results

To benchmark our method, we considered both K-means [16] and UGPMA [1] applied to the TF-IDF matrix generated from our preprocessed corpus. In addition, we also benchmark against a completely random assignment of documents to clusters where the cluster assignment for any document is drawn from $U(1, k)$. Recall that we varied the energy functions and covariance types used to train the Gaussian word embeddings, as well as the ordering of the documents used in the process described our algorithm detailed above. We record each such scenario as a triple with the format <energy function>-<covariance type>-<ordering of documents by Γ>.

As seen in Tables I, II and III, K-means exhibited the best performance under purity. Consequently, if we frame the clustering as a document retrieval task, K-means exhibited higher precision than the other methods. Moreover, K-means also tended to achieve lower entropy scores, meaning that clusters tended to have less noise. That being said, the inverse purity, analogous to recall, for K-means is noticeably lower than the inverse purity scores for our methods or for UGPMA. Moreover, despite our methods typically performing slightly worse under purity and entropy when compared with K-means, the differences in performance in the highlighted cases are small; however, the gains in inverse purity are quite substantial. Consequently, we argue that, from a document retrieval perspective, our method generates higher quality clusters than K-means or UPGMA.

In addition to the differences in the cluster qualities formed between our methods and K-means and UPGMA, there are also noticeable differences between the quality of the clusters derived for the different training parameters for our method. For example, when we used a sequence of documents ordered by Γ in ascending order, we achieved more coherent clusters

TABLE I. QUALITY OF CLUSTERS GENERATED WHEN $k = 20$

Method	Pr	InvPr	Entropy
Random	0.524	0.116	0.412
K-means	0.628	0.326	0.290
UGPMA	0.532	0.928	0.506
IP-Diag-Asc	**0.600**	**0.888**	**0.390**
IP-Diag-Desc	0.514	0.782	0.492
IP-Diag-Rand	0.556	0.934	0.484
IP-Sph-Asc	0.558	0.958	0.393
IP-Sph-Desc	0.514	0.832	0.480
IP-Sph-Rand	0.560	0.944	0.492
KL-Diag-Asc	0.564	0.744	0.440
KL-Diag-Desc	0.514	0.834	0.479
KL-Diag-Rand	0.546	0.696	0.439
KL-Sph-Asc	0.646	0.668	0.310
KL-Sph-Desc	0.522	0.730	0.476
KL-Sph-Rand	0.544	0.646	0.399

TABLE II. QUALITY OF CLUSTERS GENERATED WHEN $k = 40$

Method	Pr	InvPr	Entropy
Random	0.512	0.134	0.369
K-means	0.648	0.256	0.265
UGPMA	0.600	0.743	0.400
IP-Diag-Asc	**0.640**	**0.900**	**0.355**
IP-Diag-Desc	0.524	0.720	0.460
IP-Diag-Rand	0.592	0.834	0.420
IP-Sph-Asc	0.636	0.912	0.357
IP-Sph-Desc	0.574	0.632	0.474
IP-Sph-Rand	0.580	0.828	0.420
KL-Diag-Asc	0.608	0.802	0.365
KL-Diag-Desc	0.516	0.742	0.459
KL-Diag-Rand	0.572	0.664	0.411
KL-Sph-Asc	0.596	0.714	0.389
KL-Sph-Desc	0.514	0.742	0.497
KL-Sph-Rand	0.562	0.662	0.405

TABLE III. QUALITY OF CLUSTERS GENERATED WHEN $k = 60$

Method	Pr	InvPr	Entropy
Random	0.510	0.128	0.331
K-means	0.684	0.228	0.214
UGPMA	0.622	0.720	0.366
IP-Diag-Asc	0.682	0.812	0.279
IP-Diag-Desc	0.594	0.592	0.460
IP-Diag-Rand	0.638	0.636	0.400
IP-Sph-Asc	**0.684**	**0.826**	**0.284**
IP-Sph-Desc	0.576	0.590	0.479
IP-Sph-Rand	0.642	0.624	0.388
KL-Diag-Asc	0.662	0.686	0.294
KL-Diag-Desc	0.516	0.780	0.475
KL-Diag-Rand	0.576	0.646	0.389
KL-Sph-Asc	0.664	0.688	0.310
KL-Sph-Desc	0.522	0.730	0.476
KL-Sph-Rand	0.576	0.646	0.399

than when using the descending or random ordering. Moreover, using random orderings proved more effective than descending orderings. We hypothesize that those documents with relatively weaker connections to other documents in the corpus act as more effective seeds because they are more likely to represent labels that are less represented in the sample or are more representative of the core concepts underlying a particular label. Furthermore, those embeddings trained using the inner product based energy function also yielded better results than those trained with KL divergence. We suspect that the asymmetry resulting from a KL divergence based energy function leads to a loss of information under our particular algorithm. However, as noted by Vilnis and McCallum [7], these KL divergences tend to capture entailment information. Consequently, we believe that deriving or refining word ontologies using the KL divergence scores [17] might be an interesting area of investigation for future work.

V. Conclusion and Future Work

In this paper we motivated the use of Gaussian word embeddings in conjunction with a Chinese Restaurant Process based algorithm to cluster documents and showed that our algorithm achieves a good trade-off between purity, inverse purity, and entropy when compared to K-means or UPGMA. Work by Hotho et al. [11] has shown that word ontologies can be useful in document clustering. We hope to extend work done by Pembecci et al. [17] in using Gaussian word embeddings to enhance word ontologies to leverage Hotho et al.'s [11] method to produce higher quality clusters.

References

[1] M. Steinbach, G. Karypis, V. Kumar *et al.*, "A comparison of document clustering techniques," in *KDD workshop on text mining*, vol. 400, no. 1. Boston, 2000, pp. 525–526.

[2] K. Eguchi, "Adaptive cluster-based browsing using incrementally expanded queries and its effects (poster abstract)," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999, pp. 265–266.

[3] C.-J. Lee, C.-C. Hsu, and D.-R. Chen, "A hierarchical document clustering approach with frequent itemsets," *International Journal of Engineering and Technology*, vol. 9, no. 2, p. 174, 2017.

[4] P. Jajoo, "Document clustering," Ph.D. dissertation, Indian Institute of Technology Kharagpur, 2008.

[5] C. Shen, T. Li, and C. H. Ding, "Integrating clustering and multi-document summarization by bi-mixture probabilistic latent semantic analysis (plsa) with sentence bases." in *AAAI*, 2011.

[6] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[7] L. Vilnis and A. McCallum, "Word representations via gaussian embedding," *3rd International Conference on Learning Representations*, 2015.

[8] A. Huang, "Similarity measures for text document clustering," in *Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand*, 2008, pp. 49–56.

[9] J. Ramos *et al.*, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the first instructional conference on machine learning*, 2003.

[10] X. Hu, X. Zhang, C. Lu, E. K. Park, and X. Zhou, "Exploiting wikipedia as external knowledge for document clustering," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '09. New York, NY, USA: ACM, 2009, pp. 389–396. [Online]. Available: http://doi.acm.org/10.1145/1557019.1557066

[11] A. Hotho, S. Staab, and G. Stumme, "Ontologies improve text document clustering," in *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*. IEEE, 2003, pp. 541–544.

[12] M. F. Porter, "An algorithm for suffix stripping," *Program*, vol. 14, no. 3, pp. 130–137, 1980.

[13] Martin F Porter, "Snowball: A language for stemming algorithms," 2001.

[14] S. Bird, "Nltk: the natural language toolkit," in *Proceedings of the COLING/ACL on Interactive presentation sessions*. Association for Computational Linguistics, 2006, pp. 69–72.

[15] E. Amigó, J. Gonzalo, J. Artiles, and F. Verdejo, "A comparison of extrinsic clustering evaluation metrics based on formal constraints," *Inf. Retr.*, vol. 12, no. 4, pp. 461–486, Aug. 2009. [Online]. Available: http://dx.doi.org/10.1007/s10791-008-9066-8

[16] B. C. Fung, K. Wang, and M. Ester, "Hierarchical document clustering using frequent itemsets," in *Proceedings of the 2003 SIAM International Conference on Data Mining*. SIAM, 2003, pp. 59–70.

[17] İ. Pembeci, "Using word embeddings for ontology enrichment," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 4, no. 3, pp. 49–56, 2016.