Sentence clustering using continuous vector space representation

Mara Chinea-Rios, Germán Sanchis-Trilles, and Francisco Casacuberta

Pattern Recognition and Human Language Technologies Center
Universitat Politècnica de València
{machirio,gersantr,fcn}@prhlt.upv.es

Abstract. In this paper, we present a clustering approach based on the combined use of a continuous vector space representation of sentences and the k-means algorithm. The principal motivation of this proposal is to split a big heterogeneous corpus into clusters of similar sentences. We use the word2vec toolkit for obtaining the representation of a given word as a continuous vector space. We provide empirical evidence for proving that the use of our technique can lead to better clusters, in terms of intra-cluster perplexity and $F1$ score.

Keywords: clustering, k-means, continuous vector spaces

1 Introduction

With the rapid growth of online information, huge corpora are available, e.g. the wikipedia corpus, the common crawl corpus and others. These corpora may contain text from a variety of domains, especially if they are built from heterogeneous resources such as crawled web pages. The goal of *text categorization* [11] is the classification of documents into a fixed number of predefined categories. Several approaches have been applied to text categorisation, ranging from naive Bayes classifiers [11] to *Support Vector Machines* (SVM) [6, 9].

Text clustering is entailed as a more difficult task than text categorisation. In text clustering we do not know the classes into which the documents should be classified, which means that the only data available is a database of documents without class information. Several attempts have been made in text clustering. For instance, in [7] several kernel-based, text categorisation techniques are adapted to text clustering by using the k-means algorithm.

An especially appealing problem in text categorisation is sentence clustering, in which a document is made up of only one single sentence. This problem has been receiving special attention in the natural language processing (NLP) community since it allows for training specific models for each of the obtained clusters, leading to more task-focused models [8, 1]. Moreover, sentence clustering can also be of interest in other NLP tasks, such as done for text recognition [4] or statistical machine translation [15, 16].

In this paper, we present an approach for sentence clustering based in the k-means algorithm and a continuous vector representation of sentences. This new approach is evaluated in practise under the scope of sentence clustering.

This paper is structured as follows. Section 2, reviews the k-means algorithm. In Section 3, we explain word representation in vector space and extend it to vector representation of sentences. The empirical results are gathered in Section 4, and concluding remarks are discussed in Section 5.

2 *k*-means Methods

The k-means [10] is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters K fixed a priori. The main idea is to define K centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. However, it is common practise to initialise them randomly [5]. The algorithm finds a local optimum for the following minimisation:

$$S = \underset{S}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{\mathbf{x} \in S_k} \parallel \mathbf{x} - m_k \parallel^2 \tag{1}$$

where $S = \{S_1, S_2, \ldots, S_K\}$, $\parallel \mathbf{x} - m_k \parallel^2$ is a distance function between the sample $\mathbf{x}$ and m_k being the centroid of the cluster k, usually the euclidean distance.

$$d(\mathbf{x}_j, m_k) = (\mathbf{x}_j - m_k)^T (\mathbf{x}_j - m_k)$$

Even though the euclidean distance is very well studied in the literature, its application in the context of sentence clustering is not direct. For this reason, other authors [1] propose different alternatives. Here, we resort to a continuous vector space representation of sentences (explained in next section), and compare it with the approach of [1] in Section 4.

3 Continuous vector space representations of sentences

Representation of words as continuous vectors are very used [2]. Deriving from this work, word vectors were used in many natural languages processing applications [3, 19].

In the present paper, we used the *word2vec* [12] toolkit for continuous vector space representation of words. Word2vec is a recently developed technique for building a neural network that maps words to real-number vectors, with the purpose that words with similar meanings will map to similar vectors. Word2vec takes a text corpus as input and produces the word vectors as output. It first constructs a vocabulary from the training corpus and then learns vector representation of words. The representation of word *distance* would be:

$$v(distance) = (0.18094 \ -1.56289 \ -0.73434 \ -0.02778 \ 0.41745)^T$$

Word vectors generated by the neural net have nice semantic and syntactic behaviours. Semantically, "iOS" is close to "Android" and their vector representation is also close. Similarly, but in a more syntactic sense, "boys" minus "boy" is close to "girls" minus "girl".

However, a problem that arises when using continuous vector space representations of words is how to represent a whole sentence with a continuous vector. Following the idiosyncrasy described in the previous paragraph (i.e., semantically close words are also close in their vector representation), we propose to represent a given sentence by adding the vectors of each one of the words within that specific sentence:

$$F(\mathbf{x}) = \sum_{w \in \mathbf{x}} f(w) \tag{2}$$

where w is a word that appears in sentence $\mathbf{x}$ and $f(w)$ is the vector representation of w according to [12].

4 Experiments

In this section, we describe the experimental framework employed to evaluate the proposed model. Then, we show the results for our method in terms of two automatic evaluation metrics, followed by a comparative with a state-of-the-art sentence clustering method based on bilingual word-sequence kernels. This method was presented in [1], alongside with monolingual word-sequence method. The best results were obtained with bilingual word-sequence kernels, so we decided to use this method in this paper so as to compare with the best performing word-sequence kernel clustering method available.

4.1 Experimental set-up

To assess our clustering method, we propose to use labelled data and evaluate to which extent our clustering method is able to recover such labels whenever they are not available. For this purpose, we need a set of data comprised of sentences which can be grouped according to some common characteristic. In the experiments we conducted, such common characteristic was chosen to be the domain the sentence has been produced in. Hence, the training corpus was composed artificially from four different corpora belonging to different domains readily available in the literature. Even though our method can be applied to monolingual text, bilingual word-sequence kernels clustering requires a bilingual corpus. Hence,we considered English-French bilingual corpora. Table 1 shows the main features of the four corpora used. The News Comentary (NC) corpus[1] [18] is composed of translations of news articles. The EMEA[2] corpus [17] contains

[1] available at `http://www.statmt.org/wmt13`
[2] available at `http://www.statmt.org/wmt14/medical-task/`

documents from the European Medicines Agency. The PatTr[3] [20] is a parallel corpus extracted from the MAREC patent collection. The Subtitles (Subs) corpus[4] [17] is composed of subtitles translations in 30 languages.

	EMEA		PatTr		NC		Subs	
	FR	EN	FR	EN	FR	EN	FR	EN
$\|S\|$	1.0M		8.0M		117k		19.1M	
$\|W\|$	12.3M	10.5M	207.0M	189.0M	2.8M	2.4M	177.5M	186.5k
$\|V\|$	45.8k	39.3k	349.3k	356.0k	33.7k	27.6k	228.9k	182.1k

Table 1: Corpora main figures. M denotes millions of elements and k thousands of elements, $|S|$ stands for number of sentences, $|W|$ for number of words (tokens) and $|V|$ for vocabulary size (types).

We selected 2500 random sentence from each corpus. With these sentences, we performed all the experiments, named corpus-train. The corpus-test was created with 51 random sentence from each corpus.

Since the k-means algorithm needs a random initialisation, each experiment was repeated 10 times, reporting average 95% confidence intervals.

We used two different measures for automatically measuring the quality of the produced clusters, intra-cluster perplexity and $F1$ score, defined as follows:

- Intra-cluster perplexity (IC-PPL) [1] is defined as the average perplexity of each one of the clusters, formally

$$ppl_{avg} = 2^{\sum_{k=1}^{K} \frac{1}{K} \frac{1}{W_k} log_2 p(k)} \tag{3}$$

  where $p(k)$ is the probability of the samples of cluster k according to the language model estimated on that same cluster; W_k is the total number of words in the sentences belonging to the cluster k; and K is the total number of clusters. Lower values of IC-PPL are desirable. We decided to compute IC-PPL based on a 5-gram language model.
  We compare the IC-PPL improvement with the IC-PPL oracle, which is computed using as clusters the real labels, i.e., Subs, NC, PatTr or EMEA.
- $F1$ score is a measure of the accuracy achieved by the system. It considers both the precision and the recall of the test to compute the score. The $F1$ score can be interpreted as a weighted average of precision and recall, where an $F1$ score reaches its best value at 1 and worst score at 0.

Word2vec has different parameters that affect the experimental results. We conducted experiments with different vector dimensions, i.e., $v\_size = \{1, 5, 10, 50, 100, 500\}$. In addition, a given word is only considered for building its vector when it appears a given number of times within the training data. If a given word appears nc times in the corpus it is considered an important word and the toolkit computes its vector. We analysed the effect of considering different values $nc = \{1, 3, 5, 10\}$.

[3] available at `http://www.cl.uni-heidelberg.de/statnlpgroup/pattr/`
[4] available at `http://opus.lingfil.uu.se/`

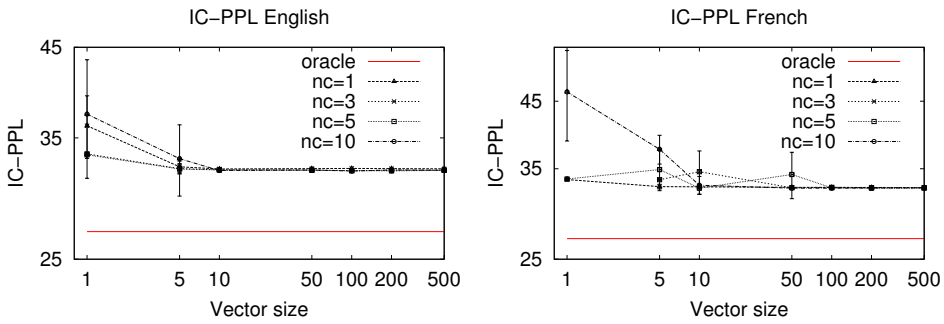

Fig. 1: Effect in average IC-PPL. Horizontal lines represent IC-PPL oracle.

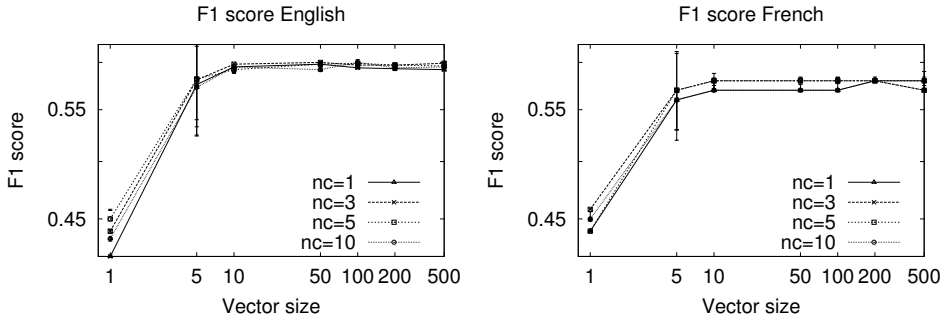

Fig. 2: Effect in average $F1$ score.

4.2 Experimental results

In this section, we present the results using different parameters (i.e., v_size and nc, see above) for obtaining the continuous vector representation of the sentences that is later used within the k-means clustering. Several conclusions can be drawn:

- In terms of IC-PPL, we compared the results with the IC-PPL oracle, which is 26.99. Results are not equal than the oracle, but the difference is less 5 point.
- The results are very similar in both languages, although results were slightly better when considering the English data.
- Above 50 or 100, vector size does not have any effect on the results. However, considering a vector size of 5 or 10 yields do very unstable results (even with high confidence intervals, 0.05 in $F1$ in the worst cases).
- nc does not seem to have a significant impact on the results obtained. We assume that this is due to the fact that the more discriminant words (i.e., those that are important for each domain) appear frequently enough so that they are never left out when computing their vector representation.

4.3 Considering different amount of domains

In this section we analyse the behaviour of our clustering strategy when considering increasing amount of domains. Different conclusions can be drawn:

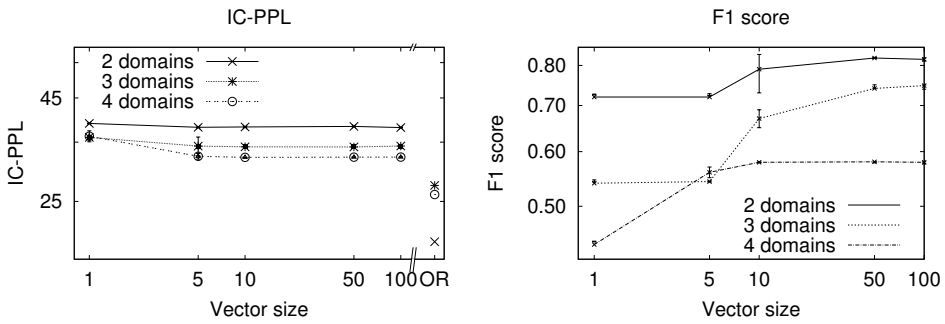

Fig. 3: IC-PPL and $F1$ score when considering increasing amount of domains. OR represent the IC-PPL oracle results and nc=1

- $F1$ value decreases with increasing domains significantly. 2-domain= 0.82 and 4-domain=0.58. This result was expected, since the more the domains, the harder it is to classify the sentences according to the original clusters, but in an unsupervised manner.
- All the experiments with our method do not achieve similar results to the oracle. We obtained the lower difference respect to the oracle with 4 domains.

4.4 Comparative with bilingual word-sequence kernels

We compared our method with the method based on bilingual word-sequence kernels presented in [1]. The authors proposed the direct use of kernels as similarity measure, and applied it to the specific case of sentence clustering via k-means algorithm. This technique has also been used in posterior works [14, 13] and may be hence considered state-of-the-art. The authors assumed that a sentence-aligned bilingual corpus is available, and they clustered the data taking into account such bilingual information. A bilingual word-sequence kernels is defined taking into account two different vocabularies, namely Σ for the source language and Δ for the target language. Let be $\mathbf{s} = \{\mathbf{x}, \mathbf{y}\}$ a bilingual sentence pair, where $\mathbf{x}$ is the sentence belonging to the source language and $\mathbf{y}$ is the sentence belonging to the target language. Then, a bilingual word-sequence kernel can be defined as

$$B_n(\mathbf{s}, \mathbf{s}') = C_n(\mathbf{x}, \mathbf{x}') + C_n(\mathbf{y}, \mathbf{y}') = \sum_{u \in \Sigma^n} |\mathbf{x}|_u |\mathbf{x}'|_u + \sum_{v \in \Delta^n} |\mathbf{y}|_v |\mathbf{y}'|_v \qquad (4)$$

where $|\mathbf{x}|_u$ stands for the number of occurrences of u in sentence $\mathbf{x}$ and $|\mathbf{y}|_v$ for the number of occurrences of v in sentence $\mathbf{y}$. Note that this method depends on strings of fixed-size n, since u and v are elements of Σ^n and Δ^n, respectively. In this work, we will analyse the effect of varying the order of the n-grams considered. Specifically, we will consider $n = \{1, 2, 3, 4, 5\}$.

Figure 4 shows the results obtained with bilingual word-sequence kernels.

- In terms of IC-PPL, results are not equal than the oracle, which is 26.99.

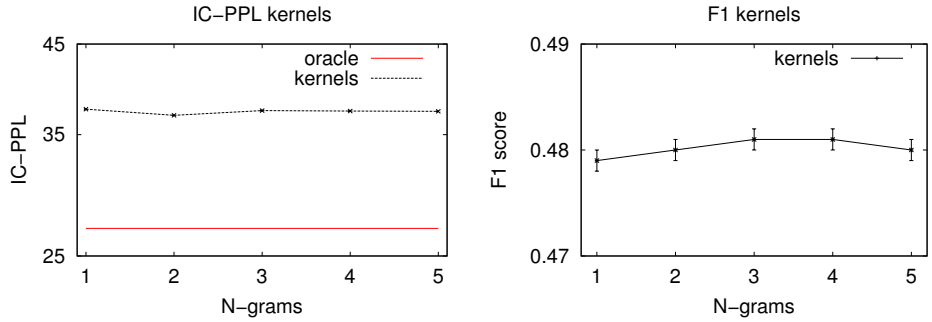

Fig. 4: Word sequence kernel clustering experiments, with $n = \{1, 2, 3, 4, 5\}$.

- n does not seem to have a significant impact on the results obtained in terms IC-PPL score.
- $F1$ score results have an increasing behaviour when n is increased, this trend is broken when $n = 5$.

Table 2 shows the best results in terms of $F1$ score and IC-PPL achieved by the two methods, i.e., continuous vector representation clustering (`Continuous`) and word-sequence kernel clustering (`Kernels`). Our method obtained the best $F1$ results, with a significant difference of 0.12. Comparing the two methods, `Continuous` obtained better results in terms of IC-PPL than `Kernels`. However, the both methods were even able to improve the IC-PPL of the real labels which were present in the data.

Methods	$F1$ score	IC-PPL
oracle	N/A	26.99
Continuous	0.60	31.92
Kernels	0.48	36.93

Table 2: Summary of the best results obtained with each set-up.

5 Conclusions and future works

The main objective in this paper is to group sentences into similar clusters. For this work, we used the k-means algorithm for obtaining the clusters and represented the sentences by means of a novel technique, i.e., continuous vector space representation. The quality of clusters obtained was measured with two metrics, $F1$ score and IC-PPL. The results obtained in terms of $F1$ score were encouraging. In terms of IC-PPL, the clusters obtained by our method were not even able to improve the IC-PPL oracle, but the results are positive. In addition, we compared our clustering strategy with a state-of-the-art strategy, i.e., bilingual word-sequence kernel clustering. In terms of $F1$ score and IC-PPL, our method was able to yield better results.

In future work, we will carry out new experiments with larger amounts of data, and we also plan to perform statistical machine translation experiments

that will validate whether the proposed approach is useful in a statistical machine translation domain adaptation task. In addition, we also intend to propose other more sophisticated ways of representing sentences using the continuous vector space representation of the words each sentence is composed of.

References

1. Andrés-Ferrer, J., Sanchis-Trilles, G., Casacuberta, F.: Similarity word-sequence kernels for sentence clustering. Proc. of S+SSPR. 610–619 (2010)
2. Bengio, Y., Schwenk, H., Senécal, J. and Morin, F.: Neural probabilistic language models. Innovations in Machine Learning. 137–186 (2006)
3. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P.: Natural Language Processing (Almost) from Scratch. JMLR. v.12, 2493–2537 (2011)
4. Cortes, C., Mohri, M., Weston, J.: A general regression technique for learning transductions. Proc. of conference on ML. 153–160 (2005)
5. Hamerly, G., Elkan, C.: Alternatives to the K-means Algorithm That Find Better Clusterings Proc. of Conference on Information and Knowledge Management. 600-607 (2002)
6. Joachims, T.: Text categorisation with support vector machines: learning with many relevant features. Proc. of ECML. 137–142 (1998)
7. Karatzoglou, A., Feinerer, I.: Text clustering with string kernels in R. JSS. v.15, 1–28 (2006)
8. Lagarda, A., Juan, A.: Topic detection and classification techniques. WP4 deliverable, TransType2 (2003)
9. Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C.J.C.H.: Text classification using string kernels. JMLR. v.2, 419–444 (2002)
10. MacQueen, J., and others : Some methods for classification and analysis of multivariate observations. Berkeley symposium on mathematical statistics and probability. 281–297 (1967)
11. McCallum, A., Nigam, K.: A comparison of event models for naive bayes text classification. Proc. of ICML. 41–48 (1998)
12. Mikolov, T., Chen, K., Corrado G., Dean J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781. (2013)
13. Sanchis, G.: Building task-oriented machine translation systems (Doctoral dissertation, Universitat Politcnica de Valncia). (2012)
14. Sennrich, R.: Mixture-modeling with unsupervised clusters for domain adaptation in statistical machine translation. Proc. of EAMT. 185–192 (2012)
15. Serrano, N., Andrés-Ferrer, J., Casacuberta, F.: On a kernel regression approach to machine translation. Proc. of IbPRIA. 394–401 (2009)
16. Szedmak, Z.W.S.T.: Kernel regression based machine translation. Proc. of ACL. 185–188 (2007)
17. Tiedemann, J.: News from OPUS - A Collection of Multilingual Parallel Corpora with Tools and Interfaces. Proc. of RANLP. 237–248 (2009)
18. Tiedemann, J.: Parallel Data, Tools and Interfaces in OPUS. Proc. of LREC. 2214–2218 (2012)
19. Turian, J., Ratinov, L., Bengio, Y.: Word Representations: A Simple and General Method for Semi-Supervised Learning. Proc. of ACL. 384–394 (2010)
20. Wäschle, K., Riezler, S.: Structural and Topical Dimensions in Multi-Task Patent Translation. Proc. of EACL. 818–828 (2012)