Evaluation of LSA performance in Spanish using multiple corpus of text

Facundo Carrillo[1], Guillermo Cecchi[3], Mariano Sigman[2], and Diego Fernández Slezak[1]

[1] Laboratorio de Inteligencia Artificial Aplicada, Dpto de Computación UBA
[2] Universidad Torcuato Di Tella, Ciudad de Buenos Aires, Argentina
[3] Computational Biology Center, T.J. Watson IBM Research Center, Yorktown Heights, NY, USA

Abstract. Latent Semantic Analysis is a natural language processing tools that allows estimating semantic distance between terms. The success of LSA is mainly based on the training corpus choice, which have been studied principally in English. This study focuses on studying LSA with regional Spanish corpus and evaluate the performance by identifying synonyms. We found that performance was slightly better than chance, concordantly with previous results. Standard LSA method cannot dynamically increase the training corpus. By using classifiers we combined multiple LSA models and showed that the use of automatic classifiers increase the performance.

1 Introduction

The way we express ourselves in written texts allows us to understand how the brain organizes ideas and concepts. By analyzing the texts we can both identify and classify higher level cognitive processes through the study of speech characteristics. In this article we set to investigate latent semantic analysis as a technique for automatic extraction of speech features that may detect mental alterations.

The success of most natural language processing tools is based on the training corpus choice. In English there are several text corpus extensively studied, for example: Brown Corpus [1]. However, the validation of the community for the Spanish corpus is not so vast, and almost inexistent for the *Rioplatense* Spanish. Considering these shortcomings in the Spanish corpus, this study focuses on instantiate classic tools with regional Spanish language's corpus and evaluate their performance.

Latent Semantic Analysis (LSA) is a technique for natural language processing based on the relationship of the frequency of terms present in documents, which defines a semantic space where the proximity of terms can be measured [2]. As the frequency of terms is so relevant, the corpus of documents that is used determines the quality of the similarity between concepts. Thus, changes in the training documents have a strong impact on generated semantic spaces.

In this article we propose a way to increase the training corpus without having to retrain all the LSA method.

2 Methods

We used *Latent semantic analysis* to measure similarity between concepts[2]. LSA is a natural language processing technique that proposes that words with close meaning will occur at *similar* frequency in texts.

LSA decomposes a word-by-document occurrence matrix X – with each row corresponding to a unique word in the corpus (n) and each column corresponding to a document (m) – by using Singular Value Decomposition (SVD). Then, the decomposition (U, S, V) is reduced to k dimensions, preserving as much as possible the similarity structure between rows, i.e. preserving the rank of the matrix X. Words are compared by taking the cosine of the angle between the two vectors formed by any two rows. Values close to 1 represent very similar words while values close to 0 represent very dissimilar words. Formally, let D be the frequency matrix $m \times n$, then $SVD(D) = U_d \times S_d \times V_d$ where U_d is a $m \times m$ real unitary matrix. With this factorization, each column of U_d represented a dimension of the new space. Landauer and colleagues studied the importance of the number of components of U_d ($UDMD$) used to compares terms[3]. They showed that changing $UDMD$ the similarity between words changed significantly, concluding that the optimal value of $UDMD$ is around 300 components. This parameter is strongly related with the training corpus and thus one of the most important parameters to optimize.

LSA strongly depends on the training corpus from where the relation between a set of documents and words are learned. Hereinafter, we call a *model* to each LSA decomposition trained by a different training corpus.

To evaluate each model, we defined performance measure based on synonyms and non-synonyms list. The synonyms and non-synonyms lists consisted of 250,000 pairs each of words taken from the dictionary. With the two list, LSA trained with a particular corpus, dimensionality and threshold of related words we defined two rates: Well classified synonyms rate ($TSCB$) and Well classified non-synonyms rate($TNSCB$). The final performance measure was calculated as the mean of these rates.

In this simplified use of LSA, we just want to know if two words are related or not. Then, we define a threshold that splits the cosine distance in two: the discretization threshold (DT). Two words with cosine distance lower than DT are considered not related; otherwise, if the cosine distance is greater than DT then words are related. The optimization consisted in finding the best values of $UDMD$ and DT and that maximized the final performance. We analyzed the performance of each model by fitting $UDMD$ and DT to the following Spanish corpus: Pagina12[4]: 326,466 newspaper's articles. Twitter: 1,000 Tweets (from Bs As) for each dictionary word. Project Gutenberg: 411 Spanish books and Subtitles: 142,181 Tv shows and movies subtitles.

To combine models into a generalized semantic space, we use a classifier. The input of the classifiers consists of the distance of each word on each model. Note that we drop the DT optimization – the threshold to define relation for words –

[4] http://www.pagina12.com.ar/usuarios/anteriores.php

as this task is done by the classifier. Then, after optimizing parameter $UDMD$ of the models – i.e. getting the best number of components that classify synonyms – for each pair of word of the synonyms list and the non-synonyms list we generated a 4-dimension vectors and an associated class: synonym or non-synonym. The classifier used to combine models is REPTree (Weka implementation[4]). To train and test we used a cross-validation 10 folds schema.

3 Results

Based on the training corpus, we generated four LSA models, defining different semantic spaces for comparing words. We calculated the best configuration for each LSA model, by sweeping free parameters: $UDMD$ and DT (see methods for details). Figure 1 shows the performance of every model according to $UDMD$ and DT.

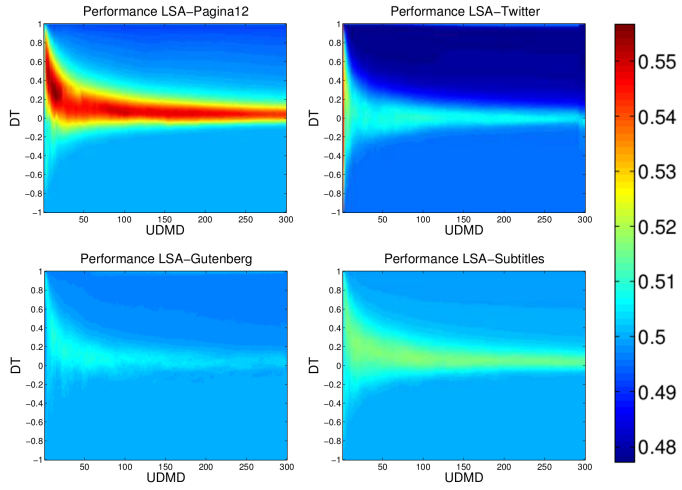

Fig. 1. Performance of the different LSA models ($UDMD$ vs DT)

The performance was different for every corpus, the table 1 summarize the best configurations.

Finally, we tested whether combining many LSA models may increase the method performance. We calculated the distance between each pair of words in each model using the optimized parameter $UDMD$. Then, we trained the classifier using the synonym and non-synonym lists (see methods for details). We obtained a 63.35% of performance in correct classified instances , beating previous performance obtained with individual models. We observe that error of incorrect classification of non-synonym is greater than results obtained in synonyms.

Corpus	Performance	$UDMD$	DT
REPTree	0.6335	-	-
Pagina12	0.5567	14	0.2600
Twitter	0.5456	1	$\forall x \in (-1, 1)$
Project Gutenberg	0.5051	11	0.2900
Subtitles	0.5182	21	0.1800

Table 1. Summary of performance and best configuration by corpus and REPTree performance

4 Discussion

Standard LSA method cannot dynamically increase the training corpus. If a single document is added, all decomposition and parameter estimation must be re-calculated. Motivated by the inability of incremental changes, in this paper we proposed to use classifiers to combine distinct corpus and evaluate the performance. We showed that the use of automatic classifiers increase the performance. While the performance was slightly better than chance, this result beats the previous results found in English[3]. We think that using synonyms list for the definition of the performance measure is not the best way to study method effectiveness. Synonyms relationship is not a well-collected feature by LSA[5]. This is caused by the difficulty of capturing polysemy by LSA. We believe that the methodology proposed (combine distinct corpus with classifiers) allows us to detect this relationship and opens the possibility of dynamically increase training corpus of the method without the necessity of complete recalculation

Acknowledgements

This research was supported by CONICET, ANPCyT, IBM Scalable Data Analytics Innovation Awards and Human Frontiers Program. Mariano Sigman is sponsored by the James McDonnell Foundation.

References

1. Marcus, M.P., Marcinkiewicz, M.A., Santorini, B.: Building a large annotated corpus of english: The penn treebank. Computational linguistics **19** (1993) 313–330
2. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. Journal of the American society for information science **41** (1990) 391–407
3. Landauer, T., Dumais, S.: A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. Psychological Review **104** (1997) 211
4. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. ACM SIGKDD Explorations Newsletter **11** (2009) 10–18
5. Landauer, T.K., Foltz, P.W., Laham, D.: An introduction to latent semantic analysis. Discourse processes **25** (1998) 259–284