Fairy tale corpus organization using latent semantic mapping and an item-to-item top-n recommendation algorithm

Paula Vaz Lobo, David Martins de Matos

Spoken Language Systems Laboratory, INESC-ID/IST
R. Alves Redol, 9
Lisbon 1000-029, Portugal
{paula.vaz,david.matos}@l2f.inesc-id.pt

Abstract

In this paper we present a fairy tale corpus that was semantically organized and tagged. The proposed method uses latent semantic mapping to represent the stories and a top-n item-to-item recommendation algorithm to define clusters of similar stories. Each story can be placed in more than one cluster and stories in the same cluster are related to the same concepts. The results were manually evaluated regarding the groupings as perceived by human judges. The evaluation resulted in a precision of 0.81, a recall of 0.69, and an f-measure of 0.75 when using tf*idf for word frequency. Our method is topic- and language-independent, and, contrary to traditional clustering methods, automatically defines the number of clusters based on the set of documents. This method can be used as a setup for traditional clustering or classification. The resulting corpus will be used for recommendation purposes, although it can also be used for emotion extraction, semantic role extraction, meaning extraction, text classification, among others.

1. Introduction

In this paper we present a fairy tale corpus semantically organized and tagged. The aim of this work is to divide a set of fairy tales in semantically related clusters, using an unsupervised method.

Fairy tales are available from different authors all over the Web and are easy to obtain. Because this type of tales are written for children, its plot and language are simpler than tales written for adults. Fairy tales are also easily read and understood.

Fairy tale sentences are shorter and emotions are well defined. A fairy tale corpus can be useful for emotion extraction, semantic role extraction, meaning extraction, recommendation, text classification, among others.

The most popular grouping algorithms that work unsupervised are clustering algorithms (Jain et al., 1999). However, these algorithms have the disadvantage of not being able, in general, of finding the ideal number of clusters. Typically, the user has to feed the algorithm with the number of clusters. Finding this number is not an easy task.

The top-n recommendation (Deshpande & Karypis, 2004) algorithm allows the grouping of stories around an initial story that plays the role of cluster centroid. Stories belonging to the same cluster are semantically related with the centroid story. The resulting clusters will be used for content-based recommendation (Pazzani & Billsus, 2007) purposes.

We applied Latent Semantic Mapping (LSM) (Bellegarda, 2005) to describe stories, getting a suitable representation for semantic processing, and used an item-based top-n recommendation algorithm to define the clusters. Our method is topic- and language-independent, and, contrary to traditional clustering methods, automatically defines the number of clusters based on the set of documents. This method can be used as a setup for traditional clustering and classification algorithms.

This paper is organized as follows: section 2. gives a brief overview of LSM; section 3. describes the top-n recommendation algorithm; section 4 describes our clustering process in which section 4.1. explains the corpus structure, section 4.2. describes the cluster organization method, and section 4.3. analyzes the final clusters; section 5. discusses the results; and section 6. contains the conclusions and future work.

2. Latent semantic mapping & SVD

Latent semantic mapping is a theory and method for extracting and representing information where the discovery of latent structure is needed (Bellegarda, 2005). This technique has the ability to expose global relationships in order to extract useful metadata concerning the underlying data.

LSM uses a mathematical technique called singular value decomposition (SVD) to identify patterns in data. SVD belongs to a class of dimensionality reduction techniques that deal with the uncovering of latent data structures.

The SVD process decomposes the initial matrix in three matrices according to the equation

$$A = U\Sigma V^T \tag{1}$$

To use SVD data must be represented in a matrix. This matrix (A) is then decomposed in three matrices: U, Σ, and V. Matrix U and V relate the lines and columns, respectively, of matrix A with the latent structures in data. Matrix Σ contains the singular values that can be used as rankings of the latent data structures.

When applied to natural language processing, LSM goes under the name of latent semantic analysis or latent semantic indexing (Landauer et al., 1998; Deerwester et al., 1990). Matrix A is the $terms \times documents$ frequency matrix; matrix U will relate terms with bags of words that represent concepts; and matrix V will relate documents with concepts. Cells will have the measure of importance of terms in concepts (matrix U) or documents in concepts (matrix V).

3. Top-n recommendation algorithm

Recommendation algorithms group together related users or items using a similarity function. Item-based recommendation algorithms cluster together similar items with the purpose of recommendation. For this type of algorithms, items are commonly represented by a user vector, where each position of the vector represents user ratings.
Top-n recommendation algorithms are a type of item-based recommendation algorithms. These recommendation algorithms first build a related item model and then apply this model to derive the top-n items for each user (Deshpande & Karypis, 2004).
To build the model, the algorithm inputs the $users \times items$ matrix and the number of similar items that will be stored for each item. The algorithm computes the similarity, typically using the cosine (equation 2) between items using the columns of the $users \times items$ matrix as vectors to represent the items. The cosine measures the distance between the two item vectors.

$$cos(\vec{v}, \vec{u}) = \frac{\vec{v} \cdot \vec{u}}{\parallel \vec{v} \parallel \parallel \vec{u} \parallel} \tag{2}$$

The output is an $items \times items$ matrix such that the j^{th} column stores the n items most similar to item j. Each cell i, j indicates the degree of similarity between j and i.

4. Clustering definition

Our method first builds a $terms \times documents$ matrix (A). SVD is applied to this matrix in order to determine the $documents \times concepts$ matrix (V) according to equation 1. The $documents \times concepts$ matrix is used to compare stories and define the clusters.

4.1. Fairy tale corpus

Fairy tales are short stories with simple plots. Contrary to romances or novels, fairy tales do not have multiple story lines. The number of characters is small and characters often represent stereotypes instead of personal names, e.g., the princess, a king, the shoemaker, the tailor, girl, boy, among others. Further more and mainly in western stories, there is a clear distinction between good and evil. The main character, the hero, concentrates all the good qualities and the villain has all the defects. The limited vocabulary and simple semantics of fairy tales make them suitable for processing and prototyping in several areas of natural language processing. Our main objective is to use this corpus for story recommendation, but it can also be used for semantic analysis and text categorization purposes.
Our corpus comprises a set of 453 fairy tales downloaded from Project Gutenberg (Hart, 1971). The selected authors and life period are shown in table 1.
The corpus also contains a set of Indian, Japanese and Arabian fairy tales translated to English.
This set of stories was tagged and lemmatized using the Stanford part-of-speech (POS) tagger (Klein & Manning, 2003). Figure 1 shoes an example of the resulting tagging. The POS tags are from the Penn TreeBank (Marcus et al., 1993). As shown in the figure 1, the story title and author are also tagged.

Author	Period
La Fontaine	1621-1695
Brothers Grimm	1785-1863
Hans C. Andersen	1805-1875
Beatrix Potter	1866-1943
Arthur Scott Bailey	1877-1949

Table 1: Authors and period of stories downloaded from Project Gutenberg.

```
<title>/NNP −− > NNP/NNP
FATTY/NNP −− > FATTY/NNP
COON/NNP −− > COON/NNP
AT/NNP −− > AT/NNP
HOME/NNP −− > HOME/NNP
</title>/VBD −− > VBD/VBD
<author>/NNP −− > NNP/NNP
Arthur/NNP −− > Arthur/NNP
Scott/NNP −− > Scott/NNP
Bailey/NNP −− > Bailey/NNP
</author>/NNP −− > NNP/NNP
Fatty/NNP −− > Fatty/NNP
Coon/NNP −− > Coon/NNP
was/VBD −− > be/VBD
so/RB −− > so/RB
fat/JJ −− > fat/JJ
and/CC −− > and/CC
round/NN −− > round/NN
that/IN −− > that/IN
he/PRP −− > he/PRP
looked/VBD −− > look/VBD
like/IN −− > like/IN
a/DT −− > a/DT
ball/NN −− > ball/NN
of/IN −− > of/IN
fur/NN −− > fur/NN
,/, −− > ,/,
with/IN −− > with/IN
a/DT −− > a/DT
plumelike/JJ −− > plumelike/JJ
tail/NN −− > tail/NN
for/IN −− > for/IN
a/DT −− > a/DT
handle/VB −− > handle/VB
./. −− > ./.
```

Figure 1: Story tagging example: Fatty Coon at home from Arthur Scott Bailey

Table 2 shows the corpus properties and table 3 shows the number of tales per author.

4.2. Organizing process

The organizing process began with the creation of two sets of stories. The first set comprises the 453 stories with all the words except for stop words. The second set is composed by the 453 stories, but containing only nouns. The choice of using only nouns was made on the belief that users choose

Number of stories	453
Number of words	908 174
Average words/story	1891
Shortest story	75
Longest story	17 694

Table 2: Fairy tale corpus properties

Author	#
La Fontaine	64
Brothers Grimm	193
Hans C. Andersen	60
Beatrix Potter	13
Arthur Scott Bailey	21
Indian	23
Japanese	21
Arabian	15

Table 3: Number of tales per author.

stories based on their characters, e.g., princess stories, ogre stories, and so on.

Our algorithm reads each set of stories and generates the context file. The context file is then used to create the $terms \times stories$ matrix that will be decomposed. We used two matrices for each set of stories: a matrix where each cell represents the frequency of a term in the story; and a matrix where each cell represents the tf*idf (Robertson, 2004) of each term in the set of stories.

Matrices U ($terms \times concepts$) is then used to extract a list of the most popular terms by concept. Matrices V ($stories \times concepts$) and Σ (singular values) are used to organize stories.

If we assume that concepts play the role of users, we can look at matrix V as an $items \times users$ matrix and apply the item-based top-n algorithm. Stories are represented by the lines of the V matrix and each position of the vector dimensionality represents story ratings regarding concepts. We applied the item-based top-n recommendation (Deshpande & Karypis, 2004) algorithm to matrix V and looked for similar stories. The output of the algorithm is the $stories \times stories$ matrix. Each cell of this matrix contains the similarity between stories.

To measure similarity we used the cosine, but story vectors were weighted with the Σ matrix (Bellegarda, 2005) as shown in the following equation,

$$cos(v_i\Sigma, v_j\Sigma) = \frac{v_i\Sigma^2 v_j^T}{\parallel v_i\Sigma \parallel\parallel v_j\Sigma \parallel} \tag{3}$$

To find clusters, we used stories as centroids and calculated the top-10 list of similar stories. Our implementation of the top-n algorithm discards stories with a similarity degree inferior to a threshold of $\frac{\sqrt{2}}{2}$ (cosine of $\frac{\pi}{4}$). Clusters that were subsets of other clusters or with cardinality of 1 were discarded. Also, only 90.95% of the stories could be allocated to a cluster. The remaining 9.05% (42 stories) were discarded. Matrix U (the $terms \times concepts$ matrix) is used to extract the most important terms in each concept, using the obtained values of the matrix as rankings. The terms corresponding to the most important concept of the centroid story for the obtained groups is used to tag the cluster, and some clusters overlap, i.e., some stories belong to more than one cluster.

4.3. Cluster analysis

The corpus is organized in directories. Each directory contains a cluster, i.e., a set of stories, and a text file with the set of the words (tags) semantically related to the stories in the cluster. Our method generated 366 clusters with 411 stories and discards 42 stories from the initial set. Figure 2 shows an example of a cluster and tags.

tags: goat; eat; fill; creep; ear; air; apple; noble; mother

Fatty Coon is mistaken
A terrible fright
Fatty learns something about eggs
The barber shop again
The track in the snow
Fatty meet Jimmy Rabbit
Forty fat turkeys
Fatty and the green corn
Johnnies Green loses his pet
Fatty discovers Mrs. Turtle's secret

Figure 2: Story cluster example: Fatty Coon tales from Arthur Scott Bailey

The organization process has produced two types of clusters. Clusters containing stories from the same author (figure 2) and clusters mixing stories from different authors corresponding to `cluster036` (figure 3).

tags: rich; treasure; live; home; asleep; stove; leg; mouth; head;s dog; horse; fast; pass; time; speed; stop; asleep; night;

The tale of Mr. Jeremy Fisher, by Beatrix Potter
The tale of Tommy Tiptoes, by Beatrix Potter
The butterfly, by H. C. Andersen
The white bride and the black bride, by Brothers Grimm

Figure 3: Story cluster example: Mixed authors cluster (`cluster036`).

In both cases, stories in a cluster are semantically related to the concepts defined by the associated words. Returning to the example of `cluster036`, all stories involve animals and have scenarios inside a character's home. Three of the stories involve running and looking for food. Stories also have other similarities not referred to in the concept words of table 4, e.g., three of the stories involve wives. This suggests that the tags extracted from the concepts need to be

improved.
The text file associated with each cluster contains the set of words that define a concept extracted from the LSM process. These words can also be semantically related. Table 4 shows the concept words for `cluster036`.

rich treasure
live home asleep stove
leg mouth head
dog horse
fast pass time speed stop
asleep night
piece
summer

Table 4: Words in `cluster036`.

As shown in table 4, we can identify groups of words related to common concepts. When the words $rich$ and $treasure$ are used, authors typically refer to wealth and well being. The words $live$, $home$, $asleep$, and $stove$ are typically related to a house.

5. Results

The corpus was manually evaluated by users. Users were invited to analyze clusters, indicating if stories were correctly allocated to each cluster.
Precision was calculated according to equation 4, recall using equation 5, and f-measure using the equation 6.

$$precision = \frac{correctly\ allocated\ stories}{number\ of\ retrieved\ documents} \quad (4)$$

$$recall = \frac{correctly\ allocated\ stories}{total\ number\ of\ stories} \quad (5)$$

$$F = 2 \times \frac{precision\ \times\ recall}{precision\ +\ recall} \quad (6)$$

Table 5 shows the values for precision and recall obtained for the clusters generated using term frequencies with all words and only nouns vs. tf*idf with all words and only nouns.

	Frequency		tf*idf	
	all words	nouns	all words	nouns
precision	**0.78**	0.75	**0.81**	0.71
recall	**0.52**	0.49	**0.69**	0.49
f-measure	**0.62**	0.59	**0.75**	0.58

Table 5: Evaluation measures for story classification

The combination that obtained best results was the use of the $terms \times stories$ matrix using the tf*idf values with all the words. For this case we obtained a precision of 0.81, a recall of 0.69, and an f-measure of 0.75.
Results show that the classification process does not benefit from representing stories only by nouns. In both cases, precision and recall are better if all words are used instead of only nouns.

6. Conclusions & Future work

In this paper we presented a method suitable to semantically organize a corpus, creating similar stories clusters. We have shown that LSM has the advantage of representing documents as vectors and provides a suitable tool to weight document vectors according to concepts. LSM can be a powerful tool for document description. Item-based top-n recommendation algorithms can be used to cluster documents without concerning about defining the initial number of clusters.
The corpus can be improved by exploring LSM to get better sets of tags, e.g., using more than one concept to extract tags. Moreover, the corpus can be enlarged with stories from a different period.
Using this method, we were able to organize a fairy tale corpus, clustering together stories conceptually similar. We intend to use this corpus for recommendation purposes.

Acknowledgments

This work was partially funded by FCT project CMU-PT/0005/2007.

7. References

Bellegarda, J.R. (2005). Latent semantic mapping. *Signal Processing Magazine, IEEE*, 22(5):70–80, Sept.

Deerwester, S., Dumais, S.T., Furnas, G. W., Landauer, T. K., Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407.

Deshpande, M., Karypis, G. (2004). Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst.*, 22(1):143–177.

Hart, M. (1971). Project gutenberg.

Jain, A. K., Murty, M. N., Flynn, P. J. (1999). Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323.

Klein, D., Manning, C.D. (2003). Fast exact inference with a factored model for natural language parsing. In *In Advances in Neural Information Processing Systems 15 (NIPS*, pages 3–10. MIT Press.

Landauer, T. K., Foltz, P. W., Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes*, 25(2):259–284.

Marcus, M.P., Santorini, B., Marcinkiewicz, M.A. (1993). Building a large annotated corpus of english: the penn treebank.

Pazzani, M.J., Billsus, D. (2007). Content-based recommendation systems. In Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl, editors, *The Adaptive Web: Methods and Strategies of Web Personalization*, volume 4321 of *Lecture Notes in Computer Science*, chapter 10, pages 325–341. Springer, Berlin.

Robertson, S. (2004). Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation*, 60.