Human Readable Feature Generation for Natural Language Corpora

Tomasz Dryjanski

tomekd789@gmail.com

Abstract

This paper proposes an alternative to the *Paragraph Vector* algorithm, generating fixed-length vectors of human-readable features for natural language corpora. It extends *word2vec* retaining its other advantages like speed and accuracy, hence its proposed name is *doc2feat*. Extracted features are presented as lists of words with their proximity to the particular feature, allowing interpretation and manual annotation. By parameter tuning focus can be made on grammatical aspects of the corpus language, making it useful for linguistic applications. The algorithm can run on variable-length pieces of texts, and provides insight into what features are relevant for text classification or sentiment analysis. The corpus does not have to, and in specific cases should not be, preprocessed with stemming or stop-words removal.

1. Introduction

Multiple machine learning algorithms exist for automated feature or topic extraction, but they have got certain weaknesses: widely used bag-of-words models that focus on words co-occurrence in documents (e.g. Latent Dirichlet Allocation; Blei et al., 2003) lose information about words proximity across documents, making these methods less precise; *Paragraph Vector* (Quoc V. Le, Tomas Mikolov, 2014) overcomes this weakness, but it generates features that are not easily available for human interpretation.

The *doc2feat* algorithm proposed in this paper runs *word2vec* (Mikolov et al., 2013) on the corpus and interprets its output as a *semantic space* further denoted as ***S***, which is then used to generate a *feature space* further referred to as ***F***. The supporting intuition for ***F*** is following: from the classical "King" - "man" + "woman" = "Queen" equation the author derives an unproven hypothesis that ***S*** is a linear space generated by some set of hidden features. By direct observation canonical vectors of ***S*** do not contain readily available information; then it is assumed that ***S*** is a linear combination of some more fundamental aspects, or features, of the corpus. Hence the *K-means* algorithm (described later) is used to extract these features, and they become the linear basis for ***F***. For this reason *doc2feat* assumes by default that ***S*** and ***F*** have the same dimensionality, but this can be overridden by command line parameters. To avoid confusion, note that a single feature has a vector representation in ***S***, but it also is a canonical vector in ***F***, being an element of its canonical basis. ***F*** is then used to annotate documents: for every document in the corpus a fixed-length vector of features is generated, denoting their similarities in ***S*** to the document; sparse representation with a cut-off threshold is also possible for real-life applications. The generated features are presented as lists of words close to them in ***S***, with the similarity information.

For all similarity calculations *cosine similarity* is used because of good experimental results. This can be supported by the following intuition: with *K-means* we calculate features as centers of word clouds in ***S***, then we are interested in how much a word, or a document represented as a sum of its words, point in a feature center direction. The vector length is then disregarded as not relevant. Cosine similarity is defined in the standard way as the cosine of the angle between two vectors; the range of similarity can vary from 1.0 (the same) down to -1.0 (opposite), i.e. the higher the number the more similar these vectors are.

S can be later disregarded and deleted to release a potentially large amount of storage space. It can also be reused for subsequent annotation of other corpora, assuming they share similar features. Note that after ***S*** is generated, all next steps are deterministic and can be rerun giving the same results. This does not stand for the ***S*** generation.

1.1. Linguistic Analysis

Author's experiments show that setting a narrow *sliding window*, like e.g. 5, promotes generation of features related to grammatical aspects of the corpus language. This is particularly visible in fusional languages, as in the example shown in Table 1. A larger window of e.g. 10 makes the *doc2feat* generate features more related to their meaning. Presentation of the sliding window concept is available in (Quoc V. Le, Tomas Mikolov, 2014); the *word2vec* and *Paragraph Vector* algorithms use it to decide what words should be considered as occurring together, and it is decided by their relative distance, measured in words, in the analyzed document.

*Table 1.*Example grammatical features generated from 100,000 documents crawled from Polish automotive portals. An outlier is marked with the strikethrough font. By a courtesy of the *Applica sp. z o.o.* company (http://www.applica.ai/).

Nouns, Plural, Genitive Case		*Verbs, Plural, 3rd Person*	
przedsięwzięć	0.542953	mają	0.641664
placówek	0.523090	są	0.607566
modeli	0.521040	uzyskują	0.553228
pojazdów	0.507653	stosują	0.542330
określeń	0.506733	uzyskają	0.541744
programów	0.505354	będą	0.517885
metod	0.505253	wprowadzają	0.512920
ekip	0.500873	modyfikują	0.503788
pakietów	0.498335	kierują	0.502854
projektów	0.497558	testują	0.501535
wzorów	0.495480	traktują	0.500137
akcentów	0.493637	wykorzystują	0.487044
aut	0.492617	zajmują	0.483918
hoteli	0.491948	wytwarzają	0.483917
komentatorów	0.491538	patrzą	0.483068
grup	0.490884	mogą	0.482310
funkcji.	0.488429	~~indywidualni,~~	0.481537
kroków	0.483650	oferują	0.481276
udogodnień.	0.483207	kupowali	0.479684
zastosowań.	0.482917	tworzą	0.477042

2. Algorithm

The *doc2feat* algorithm is arranged in the following steps:

Semantic Space Creation: The corpus is processed by *word2vec* to generate vectors for vocabulary words; they are interpreted as the semantic space ***S***.

Feature Space Creation: *K-means* is then run on ***S*** to generate the feature space ***F***, in the same way it is done in *word2vec*, with the only difference that the L_1 *norm* is used. This is advised in (Aggarwal et al., 2001), to avoid the *curse of dimensionality* (i.e. the diminishing ability to distinguish between two vectors based on their distance, with growing dimensionality). The L_1 norm is also known as the *taxicab distance*, and is defined as

$$d_1(\mathbf{p},\mathbf{q}) = \|\mathbf{p}-\mathbf{q}\|_1 = \sum_{i=1}^{n} |p_i - q_i|,$$

providing a computationally feasible tool to measure distances between vectors in high-dimensional spaces. As described in the cited publication a generalization exists, known as the *fractal norm* L_k, defined as

$$x, y \in \mathcal{R}^d, k \in \mathcal{R},\ \mathrm{L}_k(x,y) = (\sum_{i=1}^{d} |x_i - y_i|^{\,k})^{1/k}$$

Multiple fractal norms were tried out by the author, but their computational cost makes them impractical in real-life applications, while they do not add significant value: e.g. features presented in the Table 1 were generated with the L_1 norm, and top 50 words are matching with just one outlier for each feature. Similar precision is observed in other features, and it is considered by the author to be a very stable result.

doc2feat lists words closest to these features with similarities as the output.

Feature Vectors Creation: Next, every document in the corpus is reviewed and linked to a fixed-length feature vector in ***F***, with cosine similarity provided between the document and a particular feature. Document vectors are calculated as sums of word vectors in ***S***.

3. Experiments

Experiments were performed to see if extracted features can be connected to observable aspects of text corpora, and to look for clues for the tool usage.

Experimental protocol

doc2feat was run separately on two datasets with the window set to 10, and the dimensionality set to 200. Each annotated phrase was taken as a separate document to make use of the available labeled data. Documents were then looked up for features.

3.1. Stanford Sentiment Treebank Dataset

The dataset is maintained by (Socher et al., 2013) as a benchmark for sentiment analysis. It has 11855 sentences taken from the movie review site Rotten Tomatoes. Every sentence in the dataset has a label which goes from very negative to very positive in the scale from 0.0 to 1.0. In total there are 239,232 labeled phrases in the dataset. The dataset can be downloaded at: http://nlp.Stanford.edu/sentiment/ [as of Feb 2017].

Results: a feature labeled further as A was found to correlate with the sentiment; see Figure 1. If the feature is found in a phrase with similarity 0.3 or closer, its sentiment falls in the [0.2, 0.8] range with a 94.4% probability. Without checking the similarity the probability would decrease to 88.9%. The example is shown to demonstrate that removing stop-words as part of a preprocessing can lead to a loss of important data.

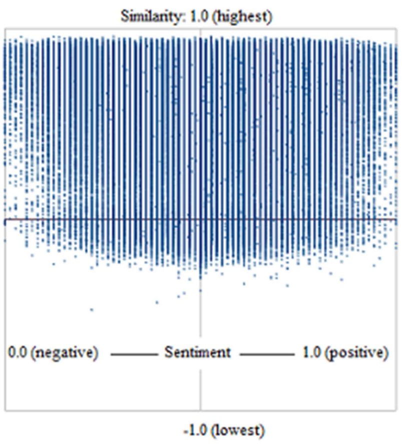
Feature A		Sentiment (horizontal) / Similarity
.	0.922	
to	0.919	
that	0.908	
,	0.906	
the	0.898	
a	0.878	
in	0.847	
of	0.845	
it	0.838	
but	0.835	
and	0.829	
is	0.796	
's	0.777	
for	0.776	
this	0.774	
with	0.767	
as	0.761	
so	0.667	
or	0.666	
have	0.661	
at	0.659	

Figure 1. A plot of 239,232 phrases with their proximity to the feature A, and sentiment. Two overlapping clouds can be visually distinguished: an ellipsoid one around the center that is typical for other features as well, and a parabolic-shaped one correlating strongly with the sentiment.

3.2. UCI Dataset

This dataset was created for (Kotzias et al., 2015). It contains 3,000 sentences labelled with positive or negative sentiment (tagged '1' or '0' respectively), extracted from reviews of products, movies, and restaurants. The correlating features are presented in Table 2.

Table 2. Features extracted from the corpus and their correlation to the sentiment. (a) Number of documents having the feature (similarity > 0.3; the threshold was selected arbitrarily, based on author's experience); (b) Ratio of documents annotated to have the positive sentiment.

Feature A		**Feature B**	
super	0.758627	great!	0.682735
performance	0.633872	headphones	0.679787
cool	0.626024	Battery	0.631001
beautiful.	0.614319	Nice	0.612433
staff.	0.542778	BEST	0.582279
friendly.	0.512033	priced	0.484854
cinematography	0.434619	steak	0.476392
Service	0.424387	new	0.449579
staff	0.408308	Good	0.448354
atmosphere	0.399221	reasonable	0.447734
gives	0.391106	fine.	0.437000
fit.	0.388146	simple	0.428868
ending	0.387151	arrived	0.421529
Our	0.379828	Best	0.411903
makes	0.376765	Its	0.403946
phone,	0.369432	terrible.	0.388678
An	0.359260	device	0.379474
Works	0.357251	Food	0.370406
Feature C		**Feature D**	
someone	0.517472	However,	0.583670
show	0.510152	minutes.	0.571949
lines	0.484348	directing	0.484432
bad.	0.411627	piece	0.373891
involved	0.336632	finally	0.372256
wasted	0.332233	worse	0.352793
said	0.322294	kept	0.325823
action	0.322079	years.	0.323748
sucks,	0.310266	movie.	0.313669

Feature	**(a)**	**(b)**
A	139	82.7%
B	107	85.0%
C	63	14.3%
D	96	20.8%

4. Source Code

The source code can be found here: https://github.com/tomekd789/doc2feat [as of March 2017]. It also provides a tool to look up a corpus to find documents matching a selected feature. The *word2vec* part of *doc2feat* is taken directly from the original source code available at: code.google.com/p/word2vec/ [Feb 2017]. Clear annotations are made in the source code for parts added during the research published in this paper.

5. Discussion

In this paper the concept of a *feature space* is proposed, and L_1 norm is used for its generation. Features spanning the feature space as its canonical basis are presented as sets of words, allowing their manual annotation and tagging. Documents are then mapped to vectors in the generated feature space. Experimental results show state of the art stability and precision of generated features, and their ability to describe important aspects of analyzed corpora.

References

Aggarwal, Charu C., Hinneburg, Alexander, Keim, Daniel A, On the Surprising Behavior of Distance Metrics in High Dimensional Spaces. *ICDT '01 Proceedings of the 8th International Conference on Database Theory; Pages 420 – 434*, 2001.

Blei, D., Ng, A., Jordan, M. *Latent Dirichlet allocation. J. Mach. Learn. Res. 3 (January 2003), 993–1022*, 2003.

Kotzias et al., *From Group to Individual Labels using Deep Features, KDD*, 2015.

Le, Quoc V., Mikolov, Tomas, Distributed Representations of Sentences and Documents. *arXiv preprint arXiv:1405.4053*, 2014.

Mikolov, Tomas, Chen, Kai, Corrado, Greg, and Dean, Jeffrey. Efficient estimation of word representations in vector space. *arXiv:1301.3781*, 2013.

Socher, Richard, Perelygin, Alex,Wu, Jean Y., Chuang, Jason, Manning, Christopher D., Ng, Andrew Y., and Potts, Christopher. *Recursive deep models for semantic compositionality over a sentiment treebank. In Conference on Empirical Methods in Natural Language Processing*, 2013.