

UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

Marcelo Rodrigo de Souza Pita

Word embedding-based representations for short text

Belo Horizonte

2019

Marcelo Rodrigo de Souza Pita

Word embedding-based representations for short text

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais, como requisito parcial à obtenção do título de Doutor em Ciência da Computação.

Orientadora: Gisele Lobo Pappa

Belo Horizonte

2019

© 2019, Marcelo Rodrigo de Souza Pita.
. Todos os direitos reservados

Ficha catalográfica elaborada pela bibliotecária Belkiz Inez Rezende Costa
CRB 6ª Região nº 1510

Pita, Marcelo Rodrigo de Souza.

P681w Word embedding-based representations for short text / Marcelo
Rodrigo de Souza Pita — Belo Horizonte, 2019.
xxxii, 123 f. il.; 29 cm.

Tese (doutorado) - Universidade Federal de Minas Gerais –
Departamento de Ciência da Computação;
Orientadora: Gisele Lobo Pappa.

1. Computação – Teses. 2. Modelagem de tópicos – Teses. 3.
Representação de textos - Teses. 4. Processamento de
linguagem natural (Computação) – Teses. 5. Aprendizado de
máquina – Teses. I. Orientadora. II. Título.

CDU 519.6*82.10 (043)



UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

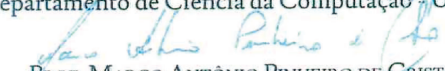
Word embedding-based representations for short text

MARCELO RODRIGO DE SOUZA PITA

Tese defendida e aprovada pela banca examinadora constituída pelos Senhores:


PROFA. GISELE LOBO PAPP - Orientadora
Departamento de Ciência da Computação - UFMG


PROF. MARCOS ANDRÉ GONÇALVES
Departamento de Ciência da Computação - UFMG


PROF. MARCO ANTÔNIO PINHEIRO DE CRISTO
Instituto de Computação - UFAM


PROF. ALEXANDRE PLASTINO DE CARVALHO
Instituto de Computação - UFF


PROF. PEDRO OLMO STANCIOLI VAZ DE MELO
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 2 de Dezembro de 2019.

To my beloved wife Cida and my remarkable son (and best friend) Miguel Ângelo, a great lover of science. My eternal gratitude for your existence in my life.

Acknowledgements

This thesis was supported and funded by the Serviço Federal de Processamento de Dados (SERPRO) through its virtuous Graduate Education Incentive Program. Over this time, SERPRO not only provided me a part time release for research, but also an very pleasant, stimulating and technically challenging working environment. I am excited to return this investment in the form of creative and innovative solutions for the Brazilian state and society.

I also thank the many SERPRO colleagues that directly and indirectly encouraged me in this work, among whom I highlight Gustavo da Gama Torres, one of those who were politically and intellectually responsible for creating the educational programs at SERPRO, and with whom I had many progressive talks about science, technology and politics. I would like to leave proof of my very special thanks to my first leader in SERPRO (and my referral leader), the person who encouraged me the most and authorized my part time release for the doctoral research, my friend Carlos Alberto Moreira Guedes (*in memoriam*). Guedes is missed, but gave me lots of advice I will certainly carry lifelong.

Many thanks to my advisor, Gisele Lobo Pappa, for this long learning period and the patience of advising a part time PhD student. I record here my admiration for her pragmatism, in addition to the great researcher she is. This acknowledgment extends to all alumni and current members of the LaIC (Laboratório de Inteligência Computacional) that get involved in my research, especially Paulo Viana Bicalho, Gabriel Pedrosa and Matheus Nunes. I would like to express my thank to the Computer Science Department of the Universidade Federal de Minas Gerais (UFMG) for the research infrastructure and high profile professors. I extend these thanks to the reviewers of this thesis, namely, Alexandre Plastino (UFF), Marco Cristo (UFAM), Marcos André Gonçalves (UFMG) and Pedro O.S. Vaz de Melo (UFMG), for the kindness of providing many helpful suggestions and criticisms in my doctoral qualification exam. I am also grateful to Fernando Buarque de Lima Neto (UPE), my long time academic mentor, who introduce me to Gisele Pappa at UFMG.

At last but not least, I thank to my family for the love, unconditional support, patience and resilience. My wife, Cida Campos Pita, was my most enthusiastic supporter, energising me the many times I got tired. Miguel Ângelo, my son, was just 3 years old when I became a PhD candidate, so obviously he expects me to have more free time to play with him at the end of this process. Today I suppose he partially understands what those many matrices, vectors, graphs and code mean. I also thank to Thiago Campos, my stepson, for the friendship. And I couldn't forget to thank Nina, our elderly "vira-lata caramelo" pet dog, because her presence has brought joy to this family many times over the years.

*“Que a gente vai, a gente vai, e fica a obra.
Mas eu persigo o que falta, não o que sobra.”*
(Lenine)

Resumo

Textos curtos estão em todo lugar na Web, incluindo mídias sociais, sites de perguntas e respostas (Q&A), textos de propagandas e um número cada vez maior de outras aplicações. Eles são caracterizados pelo escasso contexto de palavras e extenso vocabulário. Estas características tornam a descoberta de conhecimento em texto curto desafiadora, motivando o desenvolvimento de novos métodos.

Técnicas de mineração de texto são dependentes da forma como textos são representados. A necessidade de entradas de tamanho fixo para a maioria dos algoritmos de aprendizado de máquina exige representações vetoriais, tais como as representações clássicas TF e TF-IDF. Contudo, estas representações são esparsas e podem induzir a “maldição da dimensionalidade”. No nível de palavras, modelos de vetores de palavras, tais como Skip-Gram e GloVe, produzem *embeddings* que são sensíveis a semântica e consistentes com álgebra de vetores.

Este trabalho apresenta contribuições em representação de texto curto para classificação de documentos e modelagem de tópicos para texto curto. Na primeira linha, uma investigação sobre combinações apropriadas de vetores de palavras para geração de vetores de documentos é realizada. Estratégias variam de simples combinações até o método PSO-WAWV, baseado na meta-heurística PSO. Resultados em classificação de documentos são competitivos com TF-IDF e revelam ganhos significativos sobre outros métodos. Na segunda linha de pesquisa, um arcabouço que cria pseudo-documentos para modelagem de tópicos é proposto, além de duas implementações: (1) CoFE, baseado na co-ocorrência de palavras; e (2) DREx, que usa vetores de palavras. Também são propostos o modelo Vec2Graph, que induz um grafo de similaridade de vetores de palavras, e o algoritmo VGTM, um modelo de tópicos probabilístico para texto curto que funciona sobre Vec2Graph. Resultados experimentais mostram ganhos significativos em NPMI e F1-score quando comparados com métodos estado-da-arte.

Palavras-chave: Modelo de tópicos para texto curto, representação de texto curto, vetores de palavras.

Abstract

Short texts are everywhere in the Web, including social media, Q&A websites, advertisement text, and an increasing number of other applications. They are characterized by little context words and a large collection vocabulary. This makes the discovery of knowledge in short text challenging, motivating the development of novel effective methods. An important part of this research is focused on topic modeling that, beyond the popular LDA method, have produced specific algorithms for short text.

Text mining techniques are dependent on the way text is represented. The need of fixed-length input for most machine learning algorithms asks for vector representations, such as the classics TF and TF-IDF. These representations are sparse and eventually induce the curse of dimensionality. In the level of words, word vector models, such as Skip-Gram and GloVe, produce embeddings that are sensitive to semantics and consistent with vector algebra. A natural evolution of this research is the derivation of document vectors.

This work has contributions in two lines of research, namely, short text representation for document classification and short text topic modeling (STTM). In first line, we report a work that investigates proper ways of combining word vectors to produce document vectors. Strategies vary from simple approaches, such as sum and average of word vectors, to a sophisticated one based on the PSO meta-heuristic. Results on document classification are competitive with TF-IDF and show significant improvement over other methods. Regarding the second line of research, a framework that creates larger pseudo-documents for STTM is proposed, from which we derive two implementations: (1) CoFE, based on the co-occurrence of words; and (2) DREx, which relies on word vectors. We also propose Vec2Graph, a graph-based representation for corpora induced by word vectors, and VGTM, a probabilistic short text topic model that works on the top of Vec2Graph. Comparative experiments with state of the art baselines show significant improvements both in NPMI and F1-score.

Keywords: Short Text Topic Modeling, Short Text Representation, Word Vectors.

List of Figures

2.1	Histogram for document length in the 20 newsgroups dataset.	32
2.2	Mean F1 and accuracy for the 20 newsgroups classification dataset, considering different ranges of document length.	33
2.3	Box plots of sentence lengths from a sample of 200K documents of English and Portuguese Wikipedia.	34
2.4	CBOV neural network architecture.	42
2.5	SG neural network architecture.	43
2.6	Illustration of the PV-DM model. Both paragraph and word vectors are used as predictors for the next word of the context.	46
2.7	Illustration of the PV-DBOW model. Paragraph vectors are used as predictors for C randomly sample words from the paragraph.	47
3.1	LDA probabilistic graphical model in plate notation.	52
3.2	BTM probabilistic graphical model.	54
3.3	LF-LDA probabilistic graphical model.	54
3.4	DMM probabilistic graphical model.	55
3.5	Illustration of the GPU-DMM model. (Source: Yin and Wang [2014]) . . .	56
3.6	Illustration of the SeaNMF model. (Source: Shi et al. [2018])	59
3.7	Illustration of the NMF topic model applied to TF-IDF CluWords representation.	60
4.1	Illustration of PSO-WAWV optimization problem, with some fictitious particles (blue circles) of size $ V $	66
4.2	Illustrative comparison between normalized and non-normalized vector sum.	68
4.3	Histograms of word vectors and TF-IDF values.	74
5.1	Example of similarity graph for CoFE (“olympic games”).	88
5.2	Example of similarity graph for DREx (“president obama visited cuba”).	89

6.1	Corpus graph generated by the Vec2Graph algorithm. Dataset Sanders, Skip-Gram word vector model with 1000 dimensions, $\sigma = 0.4$	106
6.2	Corpus graph generated by the Vec2Graph algorithm for the Sanders dataset. Colors are mapped to topics (see labels), represented in each node as slices of a pie chart.	109
6.3	Probability distribution of top-10 words per topic (“Spanish”, “Dutch” and “Technology”) for the Sanders dataset inferred with VGTM, 5 topics, for a Vec2Graph corpus graph, $\sigma = 0.4$	110
6.4	Posterior probability distribution of topics for the VGTM, Vec2Graph corpus graph with $\sigma = 0.4$, applied to the Sanders dataset.	110
6.5	Plots of NPMI values (Y-axis) for the VGTM method with different values of σ (X-axis), considering 20, 50 and 100 topics.	114
6.6	Network metrics which achieved significant correlation values with NPMI. .	117

List of Tables

2.1	Mean F1 score for long and short versions of the 20 newsgroups classification dataset. Bold values indicate statistically significant positive differences.	31
2.2	Mean NPMI of topics for the long and short versions of the 20 newsgroups dataset. Bold values indicate statistically significant positive differences.	32
2.3	Distribution of the number of words per sentence extracted from a sample of 200K documents of the English and the Portuguese Wikipedia, and a set of tweets related to NBA topics. The parenthesis after means show standard deviation values.	34
2.4	Characterization of the datasets used in the experiments of this thesis (not shortened), regarding the number of documents, document length and smallness.	37
2.5	Characterization of the datasets with shortened documents, regarding the number of documents, document length and smallness.	37
3.1	General features of short text topic modeling methods.	61
4.1	Dataset statistics.	70
4.2	Results of document classification (mean F1-score) for the 20nshort, Sanders, Snippets and TMN datasets, for each word vector operations and different values of d (word vector size).	72
4.3	Comparison of TFIDF-WAWV-1000 and TFIDF-WAWV-SOFTMAX-1000 methods.	73
4.4	Word clouds of the most representative words in the Snippets dataset by class, for TFIDF-WAWV-1000 and AWV-1000 document vectors. The size of words are proportional to the their frequency in a list of documents' nearest words for each class. Colors do not encode any information.	75
4.5	Number of times each method was statistically the best or presented no statistical difference to other methods.	77

4.6	Comparison of the method PSO-WAWV-1000 with baselines TF-IDF, PV-DBOW-DM-1000, BOHW-1000, AWV-600 and AWV-1000 using F1-score.	79
4.7	Reduction in the representation size and relative classification improvement for AWV-600, PSO-WAWV-1000 and TF-IDF models.	79
4.8	Comparison of the method PSO-WAWV-1000 with PSO-WAWV-dims-1000 using F1-score.	80
4.9	Word clouds of the most representative words in the 20nshort dataset by class, for PSO-WAWV-1000 and AWV-1000 document vectors. The size of words are proportional to the their frequency in a list of documents' nearest words for each class. Colors do not encode any information.	80
4.10	Top-20 and bottom-20 words in the 20nshort dataset in terms of PSO-WAWV-1000 weights.	82
5.1	Datasets statistics.	91
5.2	Average results of NPMI for CoFE and DREx run with LDA (mean of different values of M).	94
5.3	Topics discovered by LDA for Web snippets expanded by CoFE and DREx-GloVe. Column <i>class</i> indicates the respective dataset class(es) the topic refers to. Column <i>Sim</i> indicates the cosine similarity for topics in the same row.	96
5.4	List of words most frequently added to documents labeled as <i>health</i> and <i>business</i> . For both CoFE and DREx, the target document size M is to 60. Each word is followed by the number of times it was added to that class documents.	97
5.5	Examples of TMN documents expanded by DREx-GloVe. The last column shows the selected words during the document expansion step.	98
5.6	Results of NPMI for document expansion methods.	99
5.7	NPMI values for LDA, LF-LDA and BTM methods with 20 topics considering both the original and expanded versions (DREx-GloVe-60) of the dataset. Percentage of improvement of expansion are in parenthesis.	100
5.8	Classification results of mean macro-average F1 score when representing documents by topics extracted from original and expanded documents. Percentage improvements of expansion are in parenthesis.	101

6.1	Statistics for the short text datasets. Columns indicate (left to right): dataset identifier; number of documents; vocabulary size (number of different terms); number of classes or categories; average number of words per document; and average number of unique (distinct) words per document.	112
6.2	Comparison of NPMI results of VGTM ($\sigma = 0.4$ and $\sigma = 0.7$) and baselines for short text datasets. Bold values indicate the best results in the column for a dataset and k	116
A.1	Detailed NPMI results for CoFE and DREx in the NBA dataset.	126
A.2	Detailed NPMI results for CoFE and DREx in the Politics dataset.	127
A.3	Detailed NPMI results for CoFE and DREx in the Sanders dataset.	127
A.4	Detailed NPMI results for CoFE and DREx in the 20Nshort dataset.	128
A.5	Detailed NPMI results for CoFE and DREx in the TMN dataset.	128
A.6	Detailed NPMI results for CoFE and DREx in the Snippets dataset.	129
A.7	Detailed NPMI results for CoFE and DREx in the CLEF dataset.	129
B.1	NPMI results of VGTM for short text datasets, varying σ (0.2 to 0.9) and k . Bold values indicate the best results in the column for a dataset and k value.	130

List of Symbols

DT	Document \times term matrix
V	Vocabulary
D	Corpus, set of documents
A	Document matrix (SVD)
B	Term matrix (SVD)
Σ	Matrix with singular values (SVD)
r	Matrix rank
k	Matrix rank; A topic (LDA-based algorithms); Reduced number of dimensions (HPCA); Context word (GloVe)
n	Matrix number of rows
m	Matrix number of columns
I	Basis vectors (NMF)
J	Coefficient matrix (NMF)
N	Document size (number of words)
K	Number of topics (LDA-based algorithms) (VGTM)
T	Terms in a document
α	Prior Dirichlet distribution of topics per document (LDA-based algorithms)
β	Prior Dirichlet distribution of words per topic (LDA-based algorithms)
φ	Posterior distribution of words per topic (LDA-based algorithms)
θ	Posterior distribution of topics per word (LDA-based algorithms)
θ_{ij}	Angle between vectors \mathbf{v}_i and \mathbf{v}_j (Vec2Graph)
Z	Selected topic for a document (LDA-based algorithms)
s	Sequence of words (MPNML)
s_{ij}	Edge between nodes i and j (Vec2Graph)
\mathbf{W}	Word vectors; Document vectors
W	Set of nodes of \mathcal{G}_C (Vec2Graph)
p	Parent vector (MorphoRNN)
d	Word vectors size; Document

t	Time; Number of neighbors in the general expansion framework
h	Output fo the neural network hidden layer (word2vec)
y	Output of the neural network output layer (word2vec)
C	Context size (word2vec); Decomposition matrix (SVD); Corpus (Vec2Graph)
X	Co-occurrence matrix
Y	Cost function (GloVe)
α^{aux}	Auxiliary dataset prior of topics per document (DLDA)
α^{tar}	Target dataset prior of topics per document (DLDA)
β^{aux}	Auxiliary dataset prior of words per topic (DLDA)
β^{tar}	Target dataset prior of words per topic (DLDA)
γ^{aux}	Prior beta distribution for choosing auxiliary topics (DLDA)
γ^{tar}	Prior beta distribution for choosing target topics (DLDA)
π	Binomial distribution for auxiliary and target topics for each document (DLDA); Bernoulli distribution that controls the choice of topic and background words (Twitter-LDA)
S	Binary switch that chooses between auxiliary and target topic for each document (DLDA); Binary switch that chooses between using or not word vectors (LF-LDA); Binary switch that chooses between topic and background words (Twitter-LDA); Set of edges of \mathcal{G}_C (Vec2Graph)
BT	Set of biterms (BTM)
τ	Matrix of topic vectors (LF-LDA)
ω	Matrix of word vectors (LF-LDA)
w	Word
w_{ij}	Weight of edge s_{ij} (Vec2Graph)
λ	Probability of a word being generated by word and topic vectors, instead of the traditional LDA inference (LF-LDA)
N_u	Number of tweets of user u (Twitter-LDA)
$N_{u,s}$	Number of words in the tweet s of user u (Twitter-LDA)
φ^B	Background words probability (Twitter-LDA)
γ	Prior Dirichlet distribution that controls the choice of topic and background words (Twitter-LDA)
s	Sequence of words (MPNML)
\mathcal{V}	Set of components in a metric space
g	Distance function in a metric space
\mathcal{NN}	Nearest neighbor function in a metric space
G_d	Graph of document d in the general expansion framework
\mathbf{G}	Adjacency matrix of graph \mathcal{G}_C (VGTM)

M	Minimum expected length of documents in the general expansion framework
f	Fitness function (PSO)
$v_i(t)$	Velocity of particle i at time t (PSO)
\mathbf{v}_i	Vector of word i (Vec2Graph)
c_1	Cognitive acceleration (PSO)
c_2	Social acceleration (PSO)
P	Number of particles (PSO)
r_1, r_2	Random numbers (PSO)
$p_i(t)$	Best position of particle i at time t (PSO)
$n_i(t)$	Best position of particle i neighborhood at time t (PSO)
$x_i(t)$	Position of particle i at time t (PSO)
χ	Constriction coefficient (PSO)
\mathcal{G}_C	Word graph for the corpus C (Vec2Graph)
σ	Similarity threshold (Vec2Graph)

Contents

Acknowledgements	5
Resumo	8
Abstract	9
List of Figures	10
List of Tables	12
List of Symbols	15
1 Introduction	22
1.1 Motivation	26
1.2 Objective	27
1.3 Overview of the contributions	28
1.4 Outline of the thesis	29
2 Short Text Representation	30
2.1 Short text definition	30
2.1.1 Text Sparsity and Machine Learning Performance	30
2.1.2 Short Text Definition	32
2.2 Overview of models for text representation	37
2.3 Word embedding representations	39
2.3.1 Related work	39
2.3.2 Continuous Bag of Words	41
2.3.3 Skip-Gram	42
2.3.4 Global Vectors	43
2.3.5 Extensions of word embedding representations	45

3	Short Text Topic Modeling	49
3.1	Probabilistic graphical models	50
3.1.1	Latent Dirichlet Allocation	51
3.1.2	Biterm Topic Model	53
3.1.3	Latent Feature LDA	54
3.1.4	GPU-DMM	55
3.2	Pseudo-document methods	56
3.2.1	Self-term expansion	57
3.2.2	Word network topic model	57
3.2.3	LDA-#	58
3.3	Adding context to topic modeling	59
3.3.1	SeaNMF	59
3.3.2	CluWords	60
3.4	Discussion about the methods	60
4	Short Text Vector Representations for Document Classification	63
4.1	Word vector combinations with PSO	64
4.1.1	Particle Swarm Optimization	64
4.1.2	The PSO-WAWV Model	66
4.2	Arithmetical combinations of word embeddings	67
4.2.1	Sum of word vectors (SWV)	67
4.2.2	TF-IDF-Weighted Sum of Word Vectors (TFIDF-WSWV)	68
4.2.3	Average of Word Vectors (AWV)	69
4.2.4	TF-IDF-Weighted Average of Word Vectors (TFIDF-WAWV)	69
4.3	Experimental setup	69
4.3.1	Datasets	70
4.3.2	Text preprocessing	70
4.3.3	Word vectors	71
4.3.4	Evaluation metric for classification	71
4.4	Results of arithmetical combinations of word embeddings	71
4.4.1	Scale of TF-IDF and word vector values	73
4.4.2	Qualitative analysis of TF-IDF weighting	73
4.4.3	Summary	76
4.5	Results for PSO-WAWV	77
4.5.1	Parameter settings	77
4.5.2	Results	78

5	Methods to Expand Short Text for Topic Modeling	83
5.1	A General Framework to Expand Short Text	83
5.2	Co-occurrence Frequency Expansion	87
5.3	Distributed Representation-based Expansion	87
5.4	Complexity Analysis of CoFE and DREx	89
5.5	Experimental Evaluation	90
5.5.1	Datasets and text preprocessing	90
5.5.2	Evaluation Metrics	91
5.5.3	Parameter setting	93
5.5.4	CoFE and DREx Parameter Analysis	94
5.5.5	Analysis of Topics and Pseudo-documents Generated by CoFE and DREx	95
5.5.6	Comparison of DREx with State-of-the-art methods	97
5.5.7	Combining DREx with Topic Models for Short Text	99
5.5.8	Document Classification Task	101
5.6	Final Remarks	102
6	Vec2Graph Topic Model	103
6.1	Vec2Graph	104
6.2	Vec2Graph Topic Model	105
6.2.1	Topics as communities in the corpus graph	106
6.2.2	Probability of topics per word	108
6.2.3	Probability of words per topic	108
6.2.4	Probability of topics	108
6.2.5	Probability of topics per document	109
6.3	Complexity analysis	110
6.4	Experimental analysis	111
6.4.1	Experimental setup	112
6.4.2	Impact of the similarity threshold	113
6.4.3	Comparison with baselines	115
6.4.4	Analysing structural patterns of the corpus graphs	115
6.5	Final remarks	118
7	Conclusions and Future Work	120
7.1	Strategies for combining word embeddings for short text classification .	121
7.2	Word embedding-based short text expansion	121
7.3	Probabilistic short text topic model based on word embedding networks	123

7.4 Publications	124
Appendix A Detailed results for CoFE and DREx	126
Appendix B Detailed results for VGTM	130
Bibliography	132

Chapter 1

Introduction

Writing is an ancestral technology developed by humans that remains essential in almost all functional aspects of our daily lives. Its product is *text*, pieces of symbols recorded in some kind of persistent media that provides a particularly rich way of representing language, alternatively to, for example, speech.

In the same way as speech, text has the linguistic properties of syntax and semantics (Akmajian et al. [2001]). But unlike speech, text is relatively persistent in the long run, enabling a lot of goods, such as registering our history, codifying legal systems (i.e. laws, contracts), providing a substrate for cultural evolution and the civilization process. From the last century to the present, it contributed to the so called 3rd industrial revolution (Schwab [2017]), which includes automating processes through software development.

The rising of the Web and social media has increased the volume of text produced in the world dramatically. Twitter, for example, was producing about 8,500 tweets per second in early 2019¹, or approximately 730 million tweets a day. Suppose that 1% of these documents are classified as “breaking news”, which results in around 7,3 millions tweets/day. A Twitter user interested in these news will never be able to read all these messages. Hence, we can assume that a smart daily news service, for example, will be able to automatically process, understand and organize this huge volume of data.

Notice that tweets are a small fraction of the Web². Given the severe restriction of available human resources to organize it and the growing demand for general text understanding, building automated processes to extract useful meaning from text is essential.

¹Live Internet statistics available at: <http://www.internetlivestats.com/one-second/>

²The volume of e-mails, for example, is much bigger. In early 2019, about 2,8 millions e-mails per second were being produced.

The main source of complexity in the interpretation of texts lies on the combination of meaning assigned by different participants in the communication, an intersection between the fields of Linguistics and Semiotics (Peirce et al. [1932]). For example, the writer of a text and a reader can have an approximate agreement about the text meaning, but not an exact agreement because of their different contexts and particular historical knowledge. This happens because semantics is not only related to syntax and pragmatics, but also to the learning attributes of communicators, making the automatic interpretation of text by machines a potentially tricky task. A general challenge is to understand how to build systems that are aware of context and “intelligent enough” to achieve an acceptable performance in text interpretation, or to extract concepts from text.

Some properties of texts are of great importance for automatic text understanding (Pinto et al. [2011]). Among them, *smallness* stands out (Rosso et al. [2013]), creating different scenarios of analysis for short and long texts. We are particularly interested in the short text analysis scenario, as it presents more technical challenges and is highly relevant today. It is relevant because short texts are present in a variety of contexts, including microblogging messages (e.g. tweets, Tumblr image captions), short message service (SMS), search engine queries, chat messages (e.g. Telegram and Whatsapp messages), product reviews and descriptions, among others.

Smallness is usually associated with the number of different words in a document and the size of the target vocabulary, i.e. the sparsity of the document-term matrix. The main problem of associating short text with “highly” sparse document-term matrices is that one cannot precisely define how sparse a document-term matrix has to be for characterizing the corpus as containing short texts. Collections of documents that would naturally be defined as long (e.g. books, articles) also have very sparse document-term matrices because of their vocabulary size. The lack of scientific consensus on how sparse a text needs to be to be considered small indicates that an alternative definition has to be pursued.

An important challenge in text processing is to choose a suitable representation for text. For example, if one wants to perform an automatic unsupervised clustering of text documents (Zaki et al. [2014]), it could be more appropriate to adopt a vector representation for texts instead of dealing with the traditional plain representation (i.e. readable text, expressed by means of characters). This is because of the existence of natural distance metrics for scalar and vector values upon which clustering algorithms are developed. A generalization of this argument is that the plain text representation is rarely a good choice for automatic inference on text, with fixed-length vector representations being preferred. Even among the vector-based representations, such as

bag of words and matrix factorization products, some are more appropriate to a given task than others. For example, distance metrics when applied to sparse vectors of high dimensionality are associated with the curse of dimensionality (Houle et al. [2010]). Therefore, dense vector representations are preferred. For short texts, with the little context and low co-occurrence of words in the collection, the representation problem is amplified.

Despite all challenges in text processing, there has been advances in, mostly supported by natural language processing (NLP), statistical inference and machine learning (ML) algorithms. For example, the discovery of latent semantic structures in text has been consolidated in the *topic modeling* field, whose most popular and representative method is *latent Dirichlet allocation* (LDA) (Blei et al. [2003]). Among other well-established methods for topic modeling are probabilistic latent semantic analysis/indexing (pLSA or pLSI) (Hofmann [1999a,b]) and non-negative matrix factorization (NMF) (Sra and Dhillon [2006]; Tandon and Sra [2010]).

In LDA, topics are latent variables in a Bayesian network. Posterior distribution of words per topic and posterior distribution of topics per document are informative about the meaning of topics and documents, respectively. The mixture of topics per document in LDA has been successfully used as feature vectors for text representation in classification (Blei et al. [2003]) and clustering (Lu et al. [2011]), as an alternative to classic vector-based representation models like *one-hot encoding*, *term frequency* (TF) and *term frequency inverse document frequency* (TF-IDF) (Salton et al. [1975]). The latter ones are sparse representations of high dimensionality, i.e. they induce the curse of dimensionality in some situations (Houle et al. [2010]). This has been a major motivation for research on efficient and potentially dense text representations.

While most of traditional NLP techniques consider words as atomic units of processing, the last decade (2008–2018) produced a lot of research contributions in word vector representations (Collobert and Weston [2008]; Turian et al. [2010]; Huang et al. [2012]; Leuret and Collobert [2013]; Socher et al. [2013a,b]; Luong et al. [2013]; Zhila et al. [2013]; Mikolov et al. [2013a,b,d,c]; Levy and Goldberg [2014a]; Pennington et al. [2014]; Faruqui and Dyer [2014]; Yogatama et al. [2015]; Faruqui et al. [2015]). The purpose of these techniques is to produce embeddings in the form of *word vectors*, vector representations that are conceived to capture semantics of words by coding each term and its context in a real vector-space embedding. Word vectors have been explored to bring out interesting representations for texts longer than n-grams (Le and Mikolov [2014]). Word embeddings have also a great potential for application in short text representation and mining, since they provide contextual information of words.

Short text has also received a lot of attention from the topic modeling community.

In general, classic methods, such as pLSA and LDA, present worse results for short text than those obtained with long text. This happens mainly due to the lack of context in documents and scarce word-word co-occurrence. At the same time, the growth of the use of short text documents in the Web and social media motivates the development of novel methods to tackle this problem (Wenyin et al. [2010]; Gao et al. [2015]).

Topic models specially design to work with short text can be broadly classified (but not mutually exclusive) as: (i) expansions of the LDA probabilistic graphical model and (ii) methods that generate pseudo-documents. In the first category, Yan et al. [2013] proposed the biterm topic model for short text (BTM), a probabilistic method that models the co-occurrence of all sequences of two words (biterns) for the entire collection. Yin and Wang [2014], in turn, proposed Dirichlet Mixture Model (DMM), a simplification of LDA to deal with only one topic per document.

In the second category (pseudo-documents generation) we can find the word network topic model (WNTM) (Zuo et al. [2016b]), which generates a pseudo-document for each word using external knowledge. More recently, other short text topic models that fit in both categories have been proposed, such as the self-aggregation based topic model (SATM) (Quan et al. [2015]) and the pseudo-document-based topic model (PTM) (Zuo et al. [2016a]).

Regardless of the category they fall, there is a recent and successful trend of incorporating the idea of word embeddings into the methods of topic modeling for short text. The idea is to enrich the topic modeling process with information about the context in which the words appear, an essential but most of the time missing information in short text documents.

Methods for short text topic modeling that use word vectors include the Latent-Feature LDA (LF-LDA) (Nguyen et al. [2015]) and an extension of DMM called GPU-DMM (generalized Pólya urn dirichlet multinomial mixture) (Li et al. [2016]), which injects semantic information of word vectors into the Gibbs sampling process of the DMM model. A recent method that is not based on probabilistic graphical models or pseudo-documents but that presents state-of-the-art results is the semantics-assisted non-negative matrix factorization (SeaNMF) (Shi et al. [2018]), which is based in the factorization of document and term co-occurrence matrices. Viegas et al. [2019] propose the CluWords, a word embedding-based representation that, combined with non-negative matrix factorization, produce coherent latent topics. There are other short text topic models that are application-specific (e.g. for Twitter) (Hong and Davison [2010]; Mehrotra et al. [2013]).

This thesis presents novel methods that benefit from word embedding models to create suitable short text representations for text analysis and mining. Depending

on the problem to be solved, different word embedding-based representations can be explored. The next sections detail our motivation, what tasks were chosen to be tackled, their associated issues, the research questions and the main contributions of the thesis.

1.1 Motivation

This work is motivated by the problems text mining techniques face when dealing with short text, which is characterized by the scarce context information and low frequency of word co-occurrence. The main hypothesis of this thesis is that *word embeddings can provide the missing contextual information that statistical and machine learning methods need to improve their quality when applied to short text.*

We focus in two groups of machine learning tasks: (i) short text classification; and (ii) short text topic modeling. While short text classification explores supervised learning and an objective view based on categorization of texts, short text topic modeling is unsupervised and exploratory, giving a more general perspective of knowledge discovery. As such, they are representatives of the supervised and unsupervised paradigms in the machine learning research field.

In the light of the hypothesis and the machine learning tasks selected for investigation, we highlight two issues that were the main drivers for the development of this work:

Issue 1 – *Word embedding-based representations for representing document vectors for short text classification are still underexploited.*

Literature reports works in document vector representation that are a natural evolution of word vectors research. Le and Mikolov [2014] propose Paragraph Vector, a method essentially similar to the learning of word vectors described in Mikolov et al. [2013a], but that learns paragraph vectors instead. Rodrigues [2018] proposes the bag of hyperwords representation, which incorporates word vector similarities and an optimization for document classification.

A gap in this research area is the lack of exploration in literature of more obvious and simpler mechanisms that combine word vectors to produce document vectors, such as variations of linear composition, for example, linear combination weighted by TF-IDF, or any other (user-defined) weighting function. More sophisticated algorithms to find sub-optimal combination strategies (e.g. particle swarm optimization, genetic algorithms and genetic programming) can also be applied and investigated. We report contribution in this issue.

Issue 2 – *Traditional topic modeling is not appropriate for short text.*

Due to the lack of context information in short documents in comparison with long ones, traditional topic models when applied to short text tend to produce worse results. A promising line of research is the investigation of short text expansion techniques for topic modeling that makes use of word vectors. As the name suggests, text expansion methods aims to increase the size of text, adding to it words ideally related to the subjects it represents. Another promising line of research is the development of novel short text probabilistic topic models that uses word embedding-based representations.

1.2 Objective

The objective of this thesis is to investigate novel methods for short text representation based on word embeddings, focusing on short text classification and short text topic modeling. To achieve this goal, the following research questions were derived from issues presented in previous section:

Research Question 1 (RQ1) – *Which are the most appropriate strategies for combining word embeddings to produce representations for short text classification?* (relates to issue 1)

In this research line, we investigate strategies for derivation of document vector representation from word embeddings for short text classification. Metaheuristics are used to perform combinatorial optimization of word vectors aiming to maximize document classification accuracy.

Research Question 2 (RQ2) – *Can we enhance short text topic modeling with word embedding-based text expansion?* (relates to issue 2)

In this research line, we explore the usage of word embeddings algebra for expansion of short text and generation of pseudo-documents. The idea is that larger documents can produce more coherent topics than short documents.

Research Question 3 (RQ3) – *Can we develop a competitive probabilistic short text topic model with word embedding-based representations?* (relates to issue 2)

In this research line, we undertake the development of a graph-based probabilistic topic model for short text grounded on word embedding networks.

1.3 Overview of the contributions

This thesis presents contributions in the fields of short text classification and short text topic modeling, according to the research questions stated in Section 1.2. They are listed below.

Definition of short text: The lack of consensus in the literature around a precise definition of short text motivated us to create a particular definition based on a quantitative analysis of real world corpora. The study and the consequent definition are presented in the early pages of Chapter 2.

Strategies for combining word embeddings for short text classification (RQ1): We propose the PSO-WAWV (Particle Swarm Optimization + Weighted Average of Word Vectors) model, which is designed to find sub-optimal weights in a weighted average of word vectors representation aiming to optimize short text classification. In addition to PSO-WAWV, this research also investigates other arithmetical combinations of word vectors. These contributions are presented in Chapter 4.

General framework for short text expansion (RQ2): This framework allows the implementation of several strategies for the expansion of short text. This is achieved by a general expansion algorithm parameterized by a metric space, which defines how new pieces of text are added to the original documents. It is explained in Chapter 5 and provides the base algorithm for the specific implementations CoFE and DREx, two other contributions of this thesis.

Co-Frequency Expansion (CoFE) method (RQ2): The CoFE is a preliminary implementation of the general expansion framework, designed for short text topic modeling and based on the co-occurrence of words. It is presented in Chapter 5.

Distributed Representation Expansion (DREx) method (RQ2): The DREx method is an implementation of the expansion framework the uses the semantic properties of word embeddings to create pseudo-documents for short text topic modeling. It is explained in Chapter 5.

Vec2Graph representation (RQ3): Vec2Graph is a corpus representation model that captures the patterns of semantic similarity of word embeddings in a word graph structure. It is proposed in Chapter 6 and is the basis for the Vec2Graph Topic Model (VGTM) method.

Vec2Graph Topic Model (VGTM) (RQ3): The VGTM method is a graph-based probabilistic short text topic model that explores the Vec2Graph structure to discover topics. VGTM infers topics as overlapping communities in the word graph. The method is presented in the Chapter 6.

1.4 Outline of the thesis

The thesis is organized as follows. Chapter 2 presents our definition of short text and does a literature review of text representations, including word embedding models. Chapter 3 introduces well established techniques for short text topic modeling. Chapter 4 address the RQ1, exploring short text vector representation strategies for document classification on the basis of word embedding algebra and numeric optimization using particle swarm optimization (PSO) (Zhang et al. [2015]). Chapter 5 addresses the RQ2, proposing a framework for expansion of short text, as well as two of its implementations. The first one, CoFE, expands short documents based on co-occurrence patterns of words, while the last one, DREx, uses word embeddings to identify potential good words for expansion. Following, Chapter 6 presents a novel probabilistic topic model that uses a (also proposed) word vector network structure called Vec2Graph, addressing the RQ3. Finally, Chapter 7 makes some conclusions and lists future work.

Chapter 2

Short Text Representation

The majority of definitions for short text are based on the sparsity of the document-term matrix (Pinto et al. [2011]). This cannot be considered a precise definition, since longer pieces of text also have highly sparse document-term matrices, making it difficult to use this single characteristic as the difference of short and long texts. More importantly, how influential the sparsity of a document-term matrix is on the performance of machine learning tasks, and, for short text, how sparse it has to be? Although we do not present definitive answers for these questions, this chapter presents a study that grounds our particular definition.

Following, the chapter does a background on general text representation and how appropriate they are for short text. We will see that most approaches look for fixed-length mathematical objects, namely real-valued vectors, aiming to capture syntactic and semantic patterns in the level of words, sentences or longer pieces of text. It is important to note that this allows not only the representation of unstructured text as input feature vectors to machine learning algorithms, but also creates possibilities of direct algebraic manipulation of the semantic properties represented.

2.1 Short text definition

2.1.1 Text Sparsity and Machine Learning Performance

The most adopted definition for short text in the literature is probably the one based on the *sparsity* of the document-term matrix (Pinto et al. [2011]). Equation 2.1 defines the sparsity of a document-term matrix $\mathbf{D}_{N \times M}$, which is the proportion of zeros in the

whole matrix.

$$\text{sparsity}(D_{N \times M}) = \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} [D_{i,j} = 0]}{N \times M} \quad (2.1)$$

A matrix \mathbf{D} is considered sparse if it has more zeros than other values, i.e. $\text{sparsity}(\mathbf{D}) > 50\%$. However, according to this definition almost all actual text datasets are sparse. For example, the 20 newsgroups dataset¹, although considered long, has a sparsity of 99.56%. In this thesis we work with a version of the 20 newsgroups dataset in which messages are trimmed to a maximum size of 20 words. This adapted dataset has sparsity of 99.79%, just 0.23% of difference in comparison with the long version. Despite this small difference, we experimentally observe significant differences in the results of machine learning tasks, such as classification and topic modeling, in favor of the longer version.

Table 2.1 shows the mean F1 score (5-fold cross validation) for the 20 newsgroups long and short datasets using a TF-IDF representation and a tuned support vector machine (SVM) for text classification. The bold value indicates that the long 20 newsgroups dataset produces the best results with statistical significance (Wilcoxon signed-rank test with 95% of confidence).

Table 2.1: Mean F1 score for long and short versions of the 20 newsgroups classification dataset. Bold values indicate statistically significant positive differences.

	<i>20news_long</i>	<i>20news_short</i>
<i>Mean F1 score:</i>	0.7438022	0.6755164

Following, Table 2.2 compares the mean normalized pointwise mutual information (NPMI) of topics (20, 50 and 100 topics) for the long and short versions of the 20 newsgroup datasets. Again we observe the best results for the long version when compared with the short one.

When looking for the small difference in sparsity of long and short text datasets, but a statistically significant difference on the performance of machine learning tasks, we conclude that the *relationship between sparsity and machine learning performance may not be obvious*.

Figure 2.1 shows the distribution of the number of documents per document length (number of words) for the long version of the 20 Newsgroups dataset. This histogram considers only documents with length up to 150 words, but we observe that the majority of documents have length between 10 and 70 words.

¹Available at: <http://ana.cachopo.org/datasets-for-single-label-text-categorization>

Table 2.2: Mean NPMI of topics for the long and short versions of the 20 newsgroups dataset. Bold values indicate statistically significant positive differences.

	<i>20news_long</i>	<i>20news_short</i>
<i>20 topics</i>	-0.1221579	-0.1966549
<i>50 topics</i>	-0.14280224	-0.19788988
<i>100 topics</i>	-0.16412774	-0.19833106

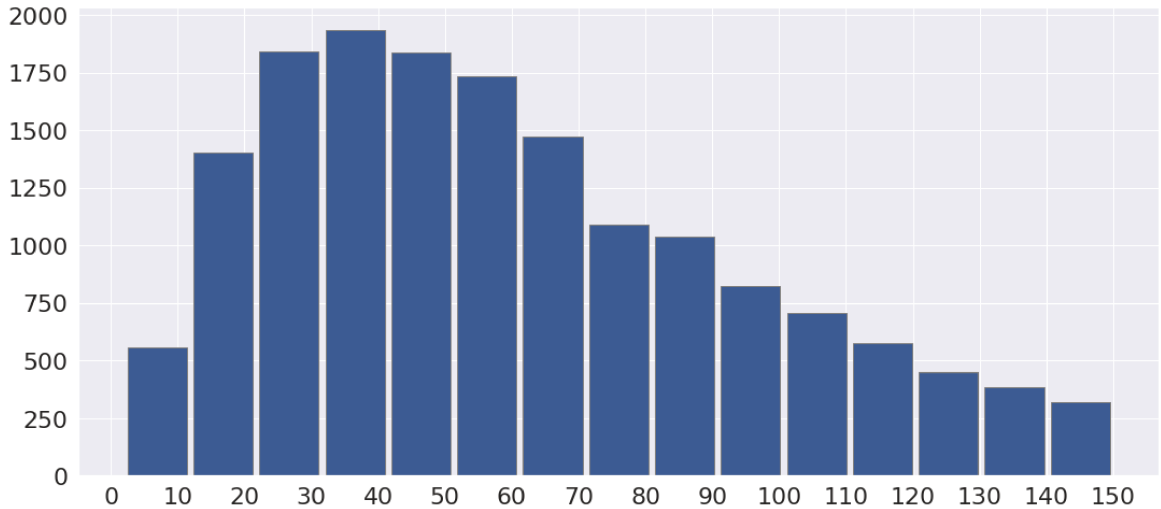


Figure 2.1: Histogram for document length in the 20 newsgroups dataset.

We performed text classification for the 20 newsgroups with stratified samples of different ranges of document length ($[10, 20]$, $[20, 30]$, ..., $[90, 100]$). Figure 2.2 shows the evolution of mean F1 score and accuracy when increasing document length.

It is possible to observe in Figure 2.2 an almost linear growth in accuracy until approximately 0.76, when documents have length above 70 words. We do not observe a similar behavior in the F1 curve, which grows faster, until documents reach the length in the interval $[40, 50]$ and then growth is slower.

These results indicate that direct measures, such as the number of words in a text, could be more relevant to machine learning performance than, for example, the sparsity of the document-term matrix. Therefore, they lead us to a short text definition based on the amount of text, as explained next.

2.1.2 Short Text Definition

We define *short text* as the minimum amount of text needed to express a single thought of the writer. According to The Writing Center of the University of North Carolina

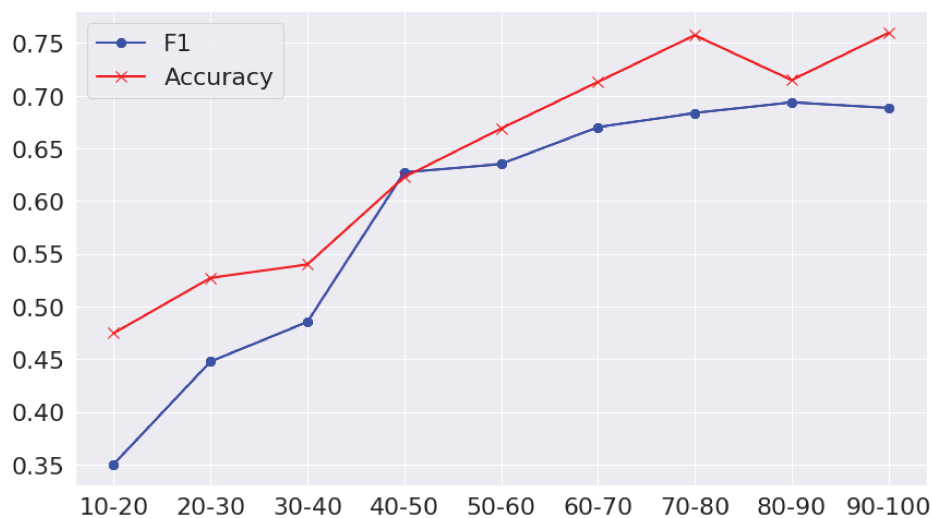


Figure 2.2: Mean F1 and accuracy for the 20 newsgroups classification dataset, considering different ranges of document length.

at Chapel Hill, this is precisely the definition of a paragraph (The Writing Center, University of North Carolina at Chapel Hill [2019]). Lunsford and O’Brien [2008] provide a similar concept for paragraph that relates to a topical unity. In summary, a paragraph is a list of logically related sentences that expresses an idea, topic or thought.

Paragraphs do not have an ideal number of sentences, but they must have a minimum number of sentences to make the underlying idea clear to the reader. In general, this is achieved by some introductory text, followed by the idea development and a conclusion. According to the Grammarly Blog, although there is no rule for paragraph length, regular paragraphs have no more than 5 sentences (Grammarly Blog [2019]). This depends on how formal or informal is the communication channel (e.g. Twitter, scientific journals) and editorial limitation (e.g. limit of 240 characters on Twitter, number of words in a newspaper column). We consider a maximum number of *3 sentences per paragraph* a fair compromise with different writing contexts.

The coincidence between the adopted definition of short text and that of a paragraph, as well as the compromise on the maximum number of sentences per paragraph, paves the way for a numeric definition of short text in terms of a maximum number of words per document. Table 2.3 shows some statistics for real-world sentences extracted from a sample of 200K documents of English Wikipedia, 200K documents of Portuguese Wikipedia² and 83,327 tweets related to topics of the NBA³, regarding their number of words.

²For each language, 200 thousand documents were used. English and Portuguese were used in the study because we have experiments with corpora on both languages.

³Dataset available at: <https://github.com/marcelopita/datasets/blob/master/nba.txt>

Table 2.3: Distribution of the number of words per sentence extracted from a sample of 200K documents of the English and the Portuguese Wikipedia, and a set of tweets related to NBA topics. The parenthesis after means show standard deviation values.

	Eng. Wikipedia	Pt. Wikipedia	Twitter
Average sentence length	18.02 (± 11.6)	17.58 (± 12.55)	4.95 (± 4.33)
Median sentence length	17	16	3
Min. sentence length	1	1	1
Max. sentence length	261	531	29

Regarding the English and the Portuguese Wikipedia, note that both languages have a similar distribution of words per sentence, with close values for average sentence length, standard deviation and median. Figure 2.3 shows the same distribution of sentence length in box plots for the Wikipedia datasets, where we can observe the outliers.

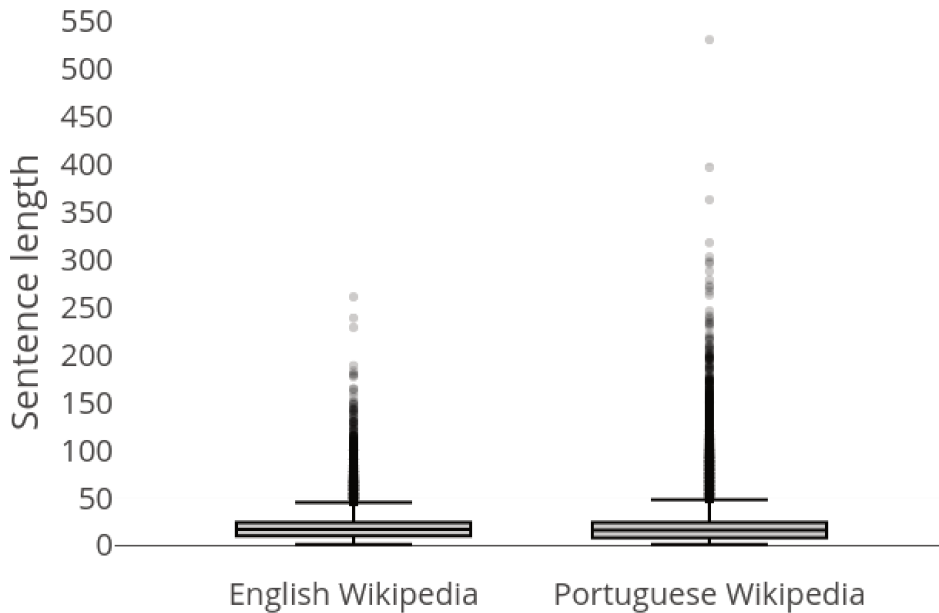


Figure 2.3: Box plots of sentence lengths from a sample of 200K documents of English and Portuguese Wikipedia.

English and Portuguese articles have similar writing patterns with respect to the length of sentences, with Portuguese texts being one word shorter on average. Mostly, box plots indicate similar distributions. Outliers are above the limit of 45 words per sentence in English and 48 words per sentence in Portuguese. Considering a maximum of 3 sentences per document and discarding outliers, we reach a maximum of 135 (3×45) words for a short text. We particularly prefer to be more conservative, assuming the median of 17 words per sentence (English Wikipedia reference).

We can observe how restrictions defined by context affect sentence length in the Twitter statistics shown in Table 2.3. Twitter present shorter sentences than usual long texts. However, the number of sentences in tweets remain similar to regular paragraphs, with a median of 3 and an average of $3.21(\pm 1.33)$ sentences per tweet. This corroborates the compromise of using 3 as the number of sentences per short text.

According to the study above, we define short text as any piece of text *shorter than 51 words* (3 sentences with an average of 17 words). This is also the criterion used to characterize the datasets used in this thesis. Since this threshold can be redefined according to different contexts, we propose a simple metric, shown in Equation 2.2, which defines the *smallness* of a document d , of length $|d|$, relative to a threshold t .

$$smallness(d, t) = \frac{t - |d|}{t} \quad (2.2)$$

According to Equation 2.2, a smallness equals to zero indicates a document with size equals to the threshold; positive values, with a maximum of 1.0, indicate short documents; negative values, without lower limit, indicate long documents⁴.

Characterization of the Datasets of the Experiments

The experiments reported in this thesis used a total of 9 (nine) datasets:

- *20 Newsgroups (20news)* – A collection of messages from 20 public newsgroups. Experiments were carried out only with short messages of up to 19 words, as suggested by Nguyen et al. [2015]⁵.
- *Tweets Sanders (Sanders)* – A Twitter sentiment corpus related to companies Apple, Google, Microsoft and Twitter.
- *Web Snippets (Snippets)* – A sample dataset with summaries of documents (snippets) related to eight subjects, retrieved from queries on a search engine (Phan et al. [2008]).
- *Tag My News (TMN)* – English RSS news from seven different categories. In the short text experiments, only titles were considered, according to Vitale et al. [2012].

⁴Note that there is an upper limit because the shortest document cannot be shorter than zero, but no lower limit, since documents can be arbitrarily long.

⁵At the time of the experimentation we had not yet elaborated our definition of short text.

- *Tweets NBA (NBA)* – A Twitter dataset collected in August 2015 about the NBA teams Golden State Warriors (hashtag #warriors) and Los Angeles Lakers (#lakers)⁶.
- *Tweets Politics (Politics)* – A Twitter dataset collected in August 2015 about the political parties Democrats (hashtag #democrats) and Republicans (#republicans)⁶.
- *CLEF 2012 Tweet Contextualization (CLEF)* – Collection of tweets from informative accounts (e.g. CNN)⁷, created for the INEX 2012 Tweet Contextualization track at CLEF conference.
- *SERPRO Courses' Objectives (Courses)* – Real-world collection of Portuguese texts describing the objectives of 1,706 courses conducted by the Serviço Federal de Processamento de Dados (SERPRO), a Brazilian Federal Government company. The courses are offered to employees and customers and are related to 25 categories of business subjects. In the short text experiments, texts were trimmed to have up to 51 words.
- *SERPRO Customer Service Messages (Customers)* – Real-world messages in Portuguese received by a customer service at SERPRO. The dataset is anonymized⁸ and consists on 17,438 texts obtained by phone call transcription or directly through e-mail and Web form. Messages are related to a business domain and are mapped to 13 categories of technical service attendance group. In the short text experiments, texts were trimmed to have up to 51 words.

Note that documents in the 20news, TMN, Courses and Customers datasets were shorten for the experiments of this thesis. Table 2.4 shows, among other characteristics, the smallness of the original datasets (not shortened). Sanders, Snippets, TMN, NBA, Politics and CLEF datasets are natural short text datasets according to our short text definition, i.e. all documents with less than 51 words. If we consider the average smallness, this list also includes the Courses and Customers datasets. The 20news dataset has very large documents (maximum length of 5,551 words), with negative average smallness and a very low value of minimum smallness (-107.84).

In our experiments we consider only the titles of the messages from the TMN dataset, instead of the whole (short text) document. Even though the Courses and

⁶Available at: <http://www.github.com/gabrielmip/st-topic-modeling>.

⁷Available at: <http://inex.mmci.uni-saarland.de/data/documentcollection.html#qa>

⁸No information about people or organizations.

Table 2.4: Characterization of the datasets used in the experiments of this thesis (not shortened), regarding the number of documents, document length and smallness.

Dataset	#Docs	Avg doc len	Min-max doc len	Avg smallness	Min-max smallness
20news	18768	102.69 (\pm 210.00)	[2, 5551]	-1.01 (\pm 4.12)	[-107.84, 0.96]
Sanders	3770	6.14 (\pm 2.67)	[2, 16]	0.88 (\pm 0.05)	[0.69, 0.96]
Snippets	12117	14.34 (\pm 4.42)	[2, 37]	0.72 (\pm 0.09)	[0.27, 0.96]
TMN	31447	17.99 (\pm 5.06)	[2, 46]	0.65 (\pm 0.10)	[0.10, 0.96]
NBA	70707	8.58 (\pm 3.01)	[2, 18]	0.83 (\pm 0.06)	[0.65, 0.96]
Politics	70712	8.08 (\pm 2.56)	[2, 20]	0.84 (\pm 0.05)	[0.61, 0.96]
CLEF	1001	6.31 (\pm 2.81)	[2, 17]	0.88 (\pm 0.06)	[0.67, 0.96]
Courses	1706	17.65 (\pm 12.65)	[2, 90]	0.65 (\pm 0.25)	[-0.76, 0.96]
Customers	17438	45.65 (\pm 52.44)	[4, 396]	0.10 (\pm 1.03)	[-6.76, 0.92]

Customers datasets are short on average (0.65 and 0.10, respectively), for the experiments we trimmed the documents that exceeds the length threshold (in our case, 51 words). For the 20news dataset we applied a stronger pruning, limiting the documents to a maximum of 19 words, resulting in 1,723 documents. The datasets with documents of reduced size are characterized in the Table 2.5.

Table 2.5: Characterization of the datasets with shortened documents, regarding the number of documents, document length and smallness.

Dataset	#Docs	Avg doc len	Min-max doc len	Avg smallness	Min-max smallness
20news (trimmed)	1723	8.17 (\pm 3.52)	[2, 19]	0.84 (\pm 0.07)	[0.63, 0.96]
TMN (title)	30376	4.94 (\pm 1.51)	[2, 16]	0.90 (\pm 0.03)	[0.69, 0.96]
Courses (trimmed)	1706	17.26 (\pm 11.23)	[2, 51]	0.66 (\pm 0.22)	[0.00, 0.96]
Customers (trimmed)	17438	30.13 (\pm 15.35)	[4, 51]	0.41 (\pm 0.30)	[0.00, 0.92]

2.2 Overview of models for text representation

Text mining and natural language processing (NLP) techniques are highly dependent on the way text is represented. For short text – a scenario characterized by low avail-

ability of contextual information – the expressiveness of the chosen representation is particularly important.

Among the classic document representation models, regardless of whether they have been used for representing long or short text, we highlight three. The first one is the simplified *bag of words* (BoW) binary model, initially proposed as an solution to the problem of indexing terms and documents for information retrieval (Salton et al. [1975]). The BoW representation creates a document-term matrix that indicates which words appear in each document. The second classic representation model is the *term frequency* (TF) model, a variation of the BoW model that represents documents as quantizations over the vocabulary. Instead of producing a binary matrix, which is a simple indication of presence or absence of terms in documents, TF gives the frequency of each term in each document. Finally, the third classic text representation model is the *term frequency–inverse document frequency* (TF-IDF), which besides considering the frequency of terms in the documents also considers the rarity of terms in the corpus. TF-IDF is still a widely used text representation model⁹, usually working as input for a variety of machine learning models. For short text, TF-IDF turns out to be a more expressive and appropriate model, because it considers the rarity of terms in the corpus. TF information, in contrast, is maybe insufficient to make words discriminative in text in which almost all words have low frequency.

Document-term matrices are in general large and sparse. This is due to the potentially large number of documents and terms in a corpus. Therefore, the likelihood of any word appearing in any document is, a priori, very low. Matrix-decomposition-based methods (Kishore Kumar and Schneider [2017]) have been applied in text representation with the objective of obtaining compact versions of document-term matrices, while trying to preserve variational information. A very used technique for this kind of representation is latent semantic indexing (LSI)¹⁰, which obtains low-rank representations (i.e. representations derived from low-rank approximations) for documents by applying the singular value decomposition (SVD) transformation to document-term matrices (Manning et al. [2008]). Another technique is non-negative matrix factorization (NMF), a matrix decomposition method that meets the property of, given a non-negative matrix, producing two other non-negative matrices (Xu et al. [2003]; Wang et al. [2012]). This property is especially interesting for applications where natural interpretation of data is non-negative (Schmidt et al. [2007]; Burkholder and van Antwerp [2013]), and generates more easily auditable matrices. Very sparse document-term matrices

⁹Beel et al. [2016] reports that TF-IDF was the most used weighting scheme in recommender systems until 2016 (the publication date of the report).

¹⁰Also known as *latent semantic analysis* (LSA).

(usual in short text), when used with distance-based methods, can produce bad results because of the curse of dimensionality (Houle et al. [2010]). Matrix decomposition methods also help reducing this sparsity with low rank approximations.

Topic models have also been used for text representation. This is the case of latent Dirichlet allocation (LDA) (Blei et al. [2003]), in which documents are described as distributions of probabilities over discovered topics in the corpus. LDA is further explained in Chapter 3. Topic models like LDA depend on the high frequency of occurrence of words in documents to be effective for discovering topics, and therefore are not recommended for short text corpora (Tang et al. [2014]).

In recent years a lot of progress has been made in the area of vector representation of words (Mikolov et al. [2013b]; Pennington et al. [2014]). Word vectors, or embeddings, were conceived to capture semantics of words by coding each term and its context – an important component for assigning meaning to it – in a real vector-space embedding. Therefore, by incorporating context information to the representation, word vectors can help to enhance the quality of statistical and machine learning methods when applied to short text.

This thesis reports works that extensively use word embeddings for document representation and, mainly, short text topic modeling. The next section provides more details about the field and models used in the contributions of this thesis.

2.3 Word embedding representations

2.3.1 Related work

The literature reports many works that, directly or not, propose vector models for words. Although there are earlier works that marginally contributed to the research field (see for example Deerwester et al. [1990], Schütze [1993], Bellegarda [1997]), the work with neural language models of Bengio et al. [2003] is considered the seminal, because it explicitly addresses the problem. In that work, the authors suggest that improvements in representation quality are achieved by jointly learning word vectors with the subjacent language model. The generalized alternative name “distributed representation of words” for word vectors also came from Bengio et al. [2003], though strictly it is only suitable for connectionist models (i.e. neural networks), in which the expression “distributed representation” is frequently used. Resultant word vectors of Bengio et al. [2003] have been successfully applied in other domains to improve NLP tasks (Turney and Pantel [2010]; Turian et al. [2010]), evidencing the importance of vector representation of words and text in general.

The last decade¹¹ was particularly intense in this research field. Next we briefly explain some of the most cited methods in this period.

Collobert and Weston [2008] (CW) – This work presents a convolutional neural network architecture based on the seminal work of Bengio et al. [2003] and inspired by the convolutional characteristics of the LeNet architecture (LeCun et al. [1998]). The objective is to jointly train a bunch of NLP tasks, such as POS tagging, chunking (parsing), NER, semantic role labeling (SRL), prediction of semantically related words and a language model. The architecture is considered a *deep neural network*, since its output, i.e. features over sentences, can be stacked with subsequent models to produce higher level features.

Hierarchical Log-Bilinear Model (HLBL) – Mnih and Hinton [2009] propose a neural probabilistic language model, the HLBL model, which extends the previous work of Morin and Bengio [2005]. This latter work is an evolution of Bengio et al. [2003], and learns word embeddings in order to achieve smoothed representations of words and contexts to relieve the effects of high sparsity (large vocabularies). It proposes a hierarchical binary decomposition of the language model probabilities based on prior knowledge from the WordNet “IS-A” hierarchy. The objective of the HLBL model is to replace the expert knowledge constraints extracted from the WordNet taxonomy to produce a binary tree only from training data.

Multi-Prototype Neural Language Model (MPNLM) – MPNLM (Huang et al. [2012]) approaches the important problem of homonymy and polysemy by learning multiple vectors per word considering both local and global contexts.

Hellinger PCA (HPCA) – Lebet and Collobert [2013] propose a simplified method for generating word vectors that consists on applying *principal component analysis* (PCA) (Wold et al. [1987]) over a word co-occurrence matrix as an alternative to previously described NNLM models, which are more complex and costly to train.

Morphologically-aware word representations (morphoRNN) – Luong et al. [2013] address the problem of poor estimation of word vectors for rare words, proposing a method that combines recursive neural networks (RNN) and NNLM and builds vector representations from word morphemes.

¹¹Period of 2008 to 2018.

Recurrent neural network language model (RNNLM) – Mikolov et al. [2013d] explores words vector representations that are learned by the input layer of a RNN. It demonstrates that not only word vectors capture the concept of similarity among words, but also other important linguistic regularities, such as the relationships of genre (male-female) and quantity (singular-plural). Relationship regularities are discovered through a simple vector offset method based on cosine distance. For example, one can observe that $v(\text{“apples”}) - v(\text{“apple”}) \approx v(\text{“cars”}) - v(\text{“car”}) \approx v(\text{“mice”}) - v(\text{“mouse”})$ and that this (approximately) same resultant vector captures the singular-plural relationship. The RNNLM model is the basis for the continuous bag-of-words (CBOW) and Skip-Gram models explained, with more detail, in the next section, as they will be explored in this thesis.

2.3.2 Continuous Bag of Words

The continuous bag of words (CBOW) model is one of two the artificial neural network (ANN) architectures proposed in Mikolov et al. [2013a]. The second architecture the Skip-gram (SG), which will be explained in next section. Implementations of these models became known by the machine learning community as *word2vec*.

Word2vec models are an evolution of the RNN model of Mikolov et al. [2013d], eliminating the complexity of the non-linear neuron at the hidden layer, transforming it in a projection layer that performs a weighted sum of inputs, without activation function. The CBOW architecture is illustrated in Figure 2.4.

The objective of the network is to model a classification task, specifically to infer a missing word in a given context of size C . The context is made of a set of words, except the target missing word, and because the order of words in the context is not relevant, the model is named “bag of words”. Notice this mapping makes possible to transform any text corpus into a training dataset for classification.

The input of the model are C one-hot encoding vectors of size V (the vocabulary size), one for each word in the context (x_1, x_2, \dots, x_C) . The desired output is an one-hot encoding vector of size V representing the target missing word, with the actual output being a multinomial distribution achieved by a softmax regression function. Though replicated in the illustration of Figure 2.4, a unique $W_{V \times d}$ weights matrix is shared among all 1-of- V input vectors. Therefore, the size of word vectors, d , is regulated by the number of units on the hidden (or projection) layer. They are extracted from the $W_{V \times d}$ weight matrix, after training, one vector by row. The output of the hidden layer is a simple mean of correspondent on/off vectors in $W_{V \times d}$.

Parameters are learned through backpropagation and stochastic gradient descent.

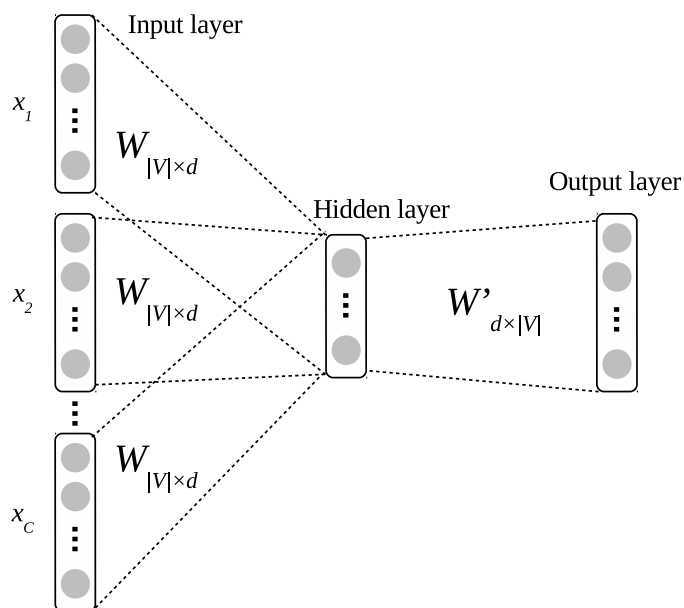


Figure 2.4: CBOW neural network architecture.

Performance is optimized by using hierarchical softmax and negative sampling (Mikolov et al. [2013a]).

Vectors can be qualitatively evaluated considering different kinds of relationships between pairs of words. For example, while it is natural to think of “Brazil” as similar to “Argentina” (both are countries), it is also intuitive that there is some similarity-based relationship between pairs of words like “car” and “cars”, or “large” and “largest”. Mikolov et al. [2013a] propose a general similarity task which answers questions like “Which word is similar to *house* in the same sense that *cars* is similar to *car*?”. As word vectors are expected to be semantically consistent with simple algebraic operations, such as sum and subtraction, the same example question can be formulated as $X = \text{vector}(\text{“cars”}) - \text{vector}(\text{“car”}) + \text{vector}(\text{“house”})$. Because the probability of X being the exact representation of “houses” is very small, the closest vector to X is retrieved.

2.3.3 Skip-Gram

The SG model is an ANN whose architecture is shown in Figure 2.5. Its objective is to learn the opposite task of CBOW, that is, given a word, infer its surrounding context of it. Therefore, the input of the model is a word x (one-hot encoding of size V , the vocabulary size) and the output is a set of words (each word one-hot encoded) within a context window of size C (y_1, y_2, \dots, y_C).

All words in the output layer share the $W'_{d \times V}$ weight matrix and produce a multi-

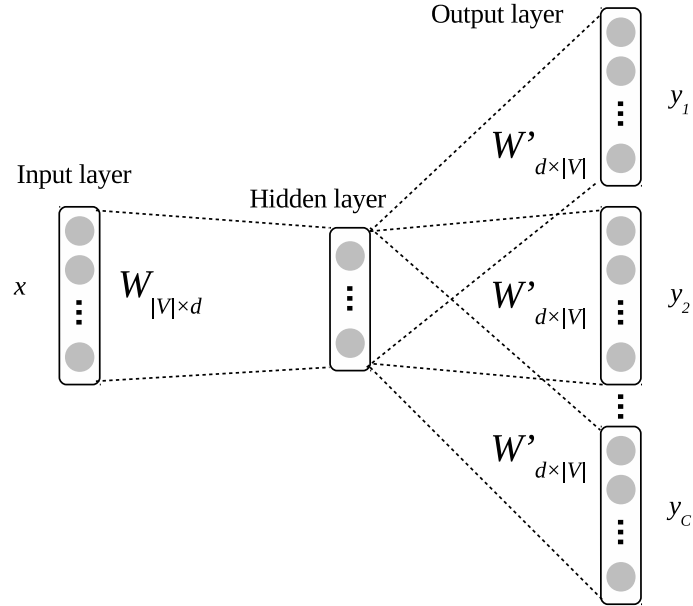


Figure 2.5: SG neural network architecture.

nomial distribution using a softmax function. Word vectors of size d (parameter, and the size of the hidden layer) are extracted from the $W_{V \times d}$ weight matrix, one vector by row. As in CBOW, parameters are learned through backpropagation and stochastic gradient descent, and performance is optimized by using hierarchical softmax and negative sampling (Mikolov et al. [2013a]).

2.3.4 Global Vectors

Pennington et al. [2014] presents a taxonomy that discriminates two categories of methods for generating word vectors: those based on *matrix factorization* and those based on *local context windows*. Methods like LSI and HPCA are in the matrix factorization category. On the other hand, word2vec methods (CBOW and SG) (Mikolov et al. [2013a]), which define context windows of size C , are in the second category. The authors then propose the Global Vectors (GloVe) method, which contains elements from both categories.

Let X be the co-occurrence matrix for the whole corpora. $X_{i,j}$ is the frequency in which the word j co-occurs with the word i in the same observation window. Let $X_i = \sum_k X_{i,k}$ be the total frequency of the word i , or the frequency in which any word co-occurs in the context of i . Therefore, $P_{i,j} = P(j|i) = X_{i,j}/X_i$ is the probability of word j appear in the context of word i .

The GloVe model is fundamentally based on the concept of similarity between words i

and j given the context word k . This similarity is captured by the ratio $P_{i,k}/P_{j,k}$. Let w_i , w_j and w'_k be the vector representation of word i , j k , respectively, where $w_i, w_j, w'_k \in \mathbb{R}^d$ and d is the dimensionality of words and context vectors. Equation 2.3 hypothesizes that there is a function F over vectors which is proportional to $P_{i,k}/P_{j,k}$.

$$F(w_i, w_j, w'_k) = P_{i,k}/P_{j,k} \quad (2.3)$$

The parameters to be learned are the vectors. Restrictions over F impose a way of combining w_i , w_j and w'_k , considering the linearity of the vector space structure. Linear combination of parameters is represented by Equation 2.4.

$$F((w_i - w_j)^T \cdot w'_k) = P_{i,k}/P_{j,k} \quad (2.4)$$

Symmetry between word vectors and context word vectors are partially restored by stating that F is a homomorphism between groups $(\mathbb{R}, +)$ and $(\mathbb{R}_{>0}, \times)$ (equations 2.5 and 2.6).

$$F((w_i - w_j)^T \cdot w'_k) = F(w_i^T \cdot w'_k)/F(w_j^T \cdot w'_k) \quad (2.5)$$

$$F(w_i^T \cdot w'_k) = P_{i,k} = X_{i,k}/X_i. \quad (2.6)$$

The solution to equation 2.5 is $F = exp$. The implication of this over equation 2.6 is shown by equation 2.7.

$$w_i^T \cdot w'_k = \log(P_{i,k}) = \log(X_{i,k}) - \log(X_i) \quad (2.7)$$

In equation 2.8 $\log(X_i)$ is absorbed by a bias b_i and symmetry between words and context vectors is finally restored by adding a bias b'_k .

$$w_i^T \cdot w'_k + b_i + b'_k = \log(X_{i,k}) \quad (2.8)$$

Equation 2.8 produces a cost function Y (equation 2.9) to be minimized through weighted least squares (AdaGrad algorithm).

$$Y = \sum_{i,j=1}^V f(X_{i,j})(w_i^T \cdot w'_k + b_i + b'_k - \log(X_{i,k})) \quad (2.9)$$

The function $f(X_{i,j})$ in equation 2.9 alleviates the effects of extreme values of $X_{i,j}$, and it is defined by equation 2.10. This function obeys the following properties: (i) $f(0) = 0$; (ii) $f(x)$ is not decreasing, avoiding overweighting of rare co-occurrences;

(iii) $f(x)$ is relatively small for large values of x , avoiding overweighting of very frequent co-occurrences.

$$f(x) = \begin{cases} (x/x_{max})^\alpha & \text{if } x < x_{max} \\ 1 & \text{otherwise} \end{cases} \quad (2.10)$$

It is important to clarify that although GloVe obtains the best results when compared to other state-of-the-art methods (SG and CBOW), the computational cost of training GloVe word vectors is much higher, specially because it includes the generation of the full co-occurrence matrix X for the corpus.

2.3.5 Extensions of word embedding representations

This section presents two extensions of word embedding models for document representation, namely, Paragraph Vector (PV) and Bag of Hyperwords (BOHW).

Paragraph Vector

Le and Mikolov [2014] present Paragraph Vector (PV), an extension of Mikolov et al. [2013a] that aims to learn continuous vector representations for longer pieces of text, such as paragraphs or entire documents.

The paragraph vectors are parameters (weights) to be learned by a neural network and used as predictors of words in the paragraph. They eventually capture semantics, in the same way as word vectors, as a side effect of the prediction task. The authors propose two variations of the method, PV-DM and PV-DBOW, as explained below.

The Distributed Memory of PV (PV-DM) is designed similarly to the Word2Vec CBOW model, i.e. $C - 1$ word vectors of a context of size C and the corresponding paragraph vector are learned in the task of predicting the next context word. The context size C , as well the paragraph vectors size d , are user-defined parameters. The model is illustrated in Figure 2.6.

Each paragraph is associated with a unique vector in the matrix $W_{|D|\times d}^1$, where D is the set of paragraphs, or documents. The hidden layer simply average or concatenate the paragraph and word vectors (matrix $W_{|D|\times d}$). Compared to CBOW, a paragraph vector can be seen as another word vector. All parameters of the network are learned using stochastic gradient descent with backpropagation.

The matrix $W_{|D|\times d}^1$, after training, contains a vector for each paragraph in the training set. The inference step for unseen paragraphs is also performed using stochastic gradient descent and backpropagation, but fixing all previously learned weights and

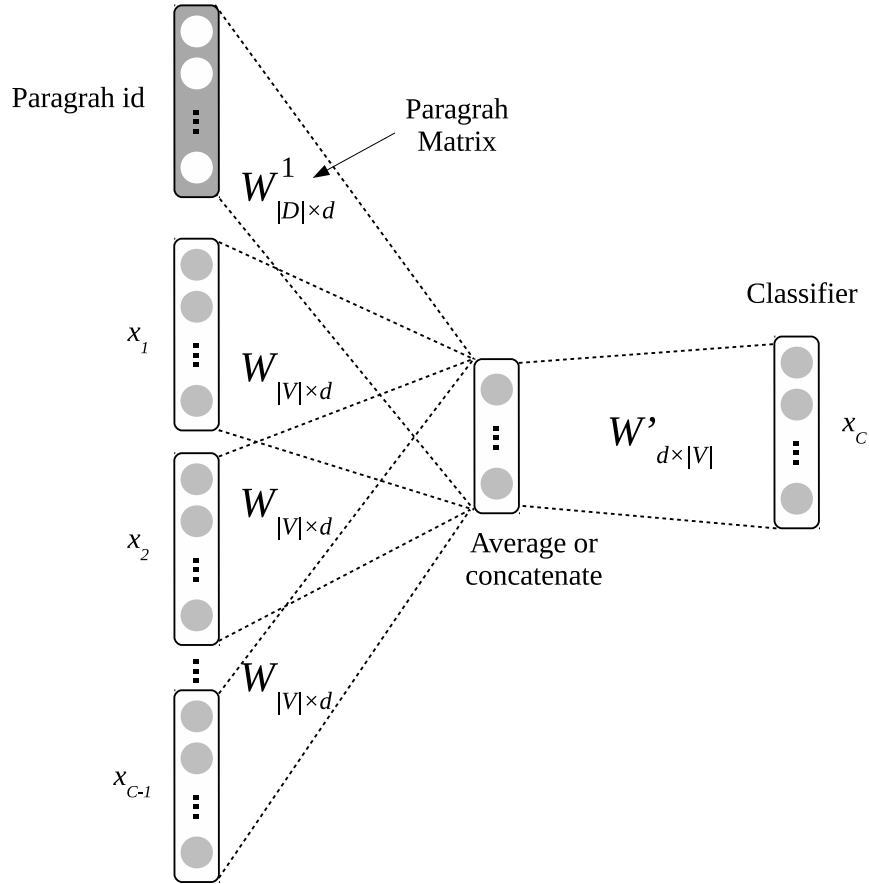


Figure 2.6: Illustration of the PV-DM model. Both paragraph and word vectors are used as predictors for the next word of the context.

adjusting the weights for the new paragraphs. These new paragraphs temporarily becomes new columns added to W^1 .

The alternative PV model is the Distributed Bag of Word of Paragraph Vectors (PV-DBOW), which has a design similar to the Word2Vec SG model. In PV-DBOW, only the paragraph vector is used as a predictor to the C (context size) randomly sampled words picked from the paragraph. Therefore, no words order is considered. The model is illustrated in Figure 2.7

The PV-DBOW is simpler and has less spatial complexity than PV-DM. The authors suggest to use a concatenation of PV-DM and PV-DBOW to represent documents, arguing that this combination produces the best results in general.

Bag of Hyperwords

The bag of hyperwords (BOHW) model (Rodrigues [2018]) brings together elements of the TF-IDF and word embedding representations. A *hyperword* $h_t = (w_1, w_2, \dots, w_{|V|})$

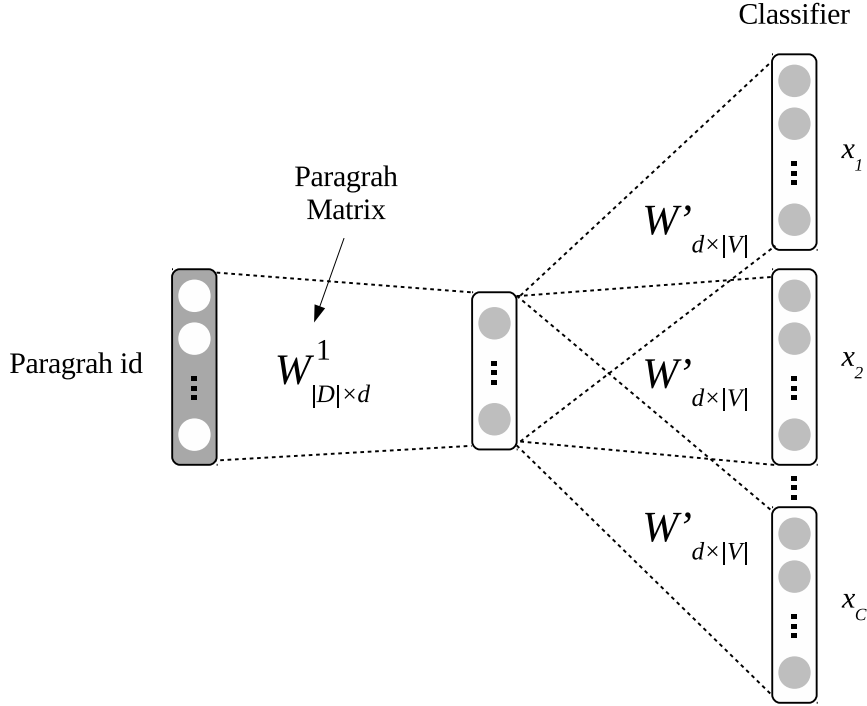


Figure 2.7: Illustration of the PV-DBOW model. Paragraph vectors are used as predictors for C randomly sample words from the paragraph.

for a term t is a similarity vector of size $|V|$. Each i -th element of this vector is the cosine similarity $s(v(t), v(t_i))$, where $v(t)$ and $v(t_i)$ are word vector representations of t and t_i , respectively, and $t_i \in V$. There is a threshold α , below which values of s are 0 (zero). In other words, h_t is a semantically enriched version of the one-hot-encoding representation of t . The word vector model and α are parameters.

BOHW is based on the TF-IDF representation for documents, but instead of words in V it uses the set of all hyperwords, $\mathcal{H} = \{h_t : t \in V\}$. The TF-IDF of a hyperword h_t in a document d is calculated according to Equation 2.11.

$$\text{tf-idf}(h_t, d) = \text{tf}(h_t, d) \text{idf}(h_t) \quad (2.11)$$

The TF of a hyperword h_t in d is defined as follows:

$$\text{tf}(h_t, d) = \sum_{1 \leq i \leq |V|} \text{tf}'(t_i, d) w_i$$

where $\text{tf}'(t, d)$ is the classic TF, i.e. the frequency of term t in d , and w_i is the similarity between $v(t)$ and $v(t_i)$.

The IDF of h_t is calculated according to Equation 2.12.

$$\text{idf}(h_t) = \log \frac{|D|}{\sum_{1 \leq d \leq |D|} \mu_{h_t, d}} \quad (2.12)$$

where $\mu_{h_t, d}$ is the mean of values of weights in the hyperword h_t that occur in hyperwords of d , V_{d, h_t} , as follows:

$$\mu(h_t, d) = \frac{1}{|V_{d, h_t}|} \sum_{t_i \in V_{d, h_t}} w_i$$

For classification datasets (i.e. labeled datasets), the value of α can be optimized to make hyperwords more discriminate regarding categories. This is important because a general value of α for all hyperwords cannot deal with different degrees of generality of related concepts in a same hyperword. Therefore, BOHW implements a dynamic alpha selection mechanism based on mutual information.

Some of the calculated hyperwords can contain very similar hyperwords. The BOHW algorithm defines a β parameter that defines the maximum degree of similarity between two hyperwords to not merge them. When merging occurs, the final hyperword is the mean vector of both hyperwords.

The BOHW method has a strong relationship with the CluWords representation proposed by Viegas et al. [2019]. Since we are using CluWords for short text topic modeling, this method is explained in Chapter 3.

Chapter 3

Short Text Topic Modeling

Topic models are a class of statistical and machine learning models used in the discovery of latent semantic patterns, the topics, in collections of documents (Alghamdi and Alfalqi [2015]). Learned topics are often described as weights for words in the corpus vocabulary, and documents as weights of topics. The statistical foundation of the majority of the topic modeling techniques is the counting of word co-occurrence in documents. It turns out that, in short texts, this count is hampered due the scarce context for words. Therefore, short text-specific topic models have been proposed.

Short text topic models are basically organized in two major groups: (*i*) probabilistic graphical models, usually extensions of LDA; and (*ii*) methods that create larger pseudo-documents from the original short ones. There are also hybrid methods, i.e. models that have elements from both categories *i* and *ii*, such as the self-aggregation based topic model (SATM) (Quan et al. [2015]), which aggregates short text based on topics during the topic inference process in a LDA-based probabilistic graphical model, and pseudo-document-based topic model (PTM) (Zuo et al. [2016a]), where pseudo-documents are latent variables in the probabilistic graphical model. Regardless of their type, some of the methods detailed in this chapter incorporate word embeddings into their formulation to take advantage of the context captured by these word representations.

Literature reports other topic models that have been applied or can be useful to short text. Chen and Liu [2014] propose the Lifelong Topic Model (LTM), a lifelong learning algorithm (Thrun [1998]) for topic modeling which dynamically explores prior knowledge to increase the coherence of inferred topics. LTM iteratively explore topical data from several domains and identifies similarities among the topics discovered, an information that is used to improve their coherence. In the reference paper, LTM was applied to 50 domains of product reviews.

Das et al. [2015] proposes the Gaussian LDA method, which incorporates word vectors into the LDA topic model by modeling each document as a sequence of word embeddings. Although the method was not proposed to overcome the problem of little context in short text, the incorporation of word embeddings adds prior context knowledge that could be useful in this scenario.

Qiang et al. [2017] present the Embedding-based Topic Model (ETM), which also considers word embeddings to overcome the little word-word co-occurrence in short texts. Word embeddings generated from an external large corpus are used to create clusters of short texts, i.e. pseudo-documents. Next, latent topics are inferred using a Markov Random Field Regularized (MRF) model (Xie et al. [2015]), which increases the probability of semantically related words to be assigned to the same topic. A very similar approach is proposed by Gao et al. [2019] recently, which presents the Conditional Random Field regularized Topic Model (CRFTM).

Shi et al. [2017] proposes a method that, instead of using pre-trained word vectors to enhance topic modeling, it learns both word embeddings and the topic model. The method, named skip-gram topical word embedding, can generate topic-specific word embeddings, dealing with the problem of polysemy, at the same time it is capable of generating coherent topics.

The Topic Memory Networks (TMN) method (Zeng et al. [2018]) aims to improve quality of short text classification by jointly learning topic models. The method does not use any external knowledge and is based on a neural topic model (Miao et al. [2017]) and a proposed class-sensitive topic memory mechanism. The reference paper mainly reports results on short text classification, but also topic modeling.

Chen et al. [2019] present the Knowledge-guided Non-negative Matrix Factorization (KGNMF) method, designed to use a word-word semantic graph as regularizer for a low-rank approximation produced by NMF over short texts.

Li et al. [2019] proposes the Relational Biterm Topic Model (R-BTM) as an extension of the Biterm Topic Model (BTM) (see section 3.1.2) that links short texts with similar words. The similarity between pairs of words are inferred using word vector cosine similarity.

3.1 Probabilistic graphical models

The work of Blei et al. [2003] is considered seminal in the topic modeling field. The authors presented latent Dirichlet allocation (LDA), a Bayesian network that implements a probabilistic generative model for document writing. After LDA, probabilistic

graphical models became very influential. Although it was initially proposed to deal with longer texts, LDA is the basis for many other methods with characteristics that suit better for short text.

Among the probabilistic graphical topic models for short text, we highlight the Dirichlet mixture model (DMM) (Yin and Wang [2014]), a simplification of LDA that considers only one topic per document, the biterm topic model (BTM) (Yan et al. [2013]), which creates distributions of topics per biterms (pairs of words in the same short context) instead of individual words, the latent feature LDA (LF-LDA) (Nguyen et al. [2015]), which incorporates word embeddings into a LDA-based probabilistic graphical model and the generalized Pólya urn Dirichlet mixture model (GPU-DMM) (Li et al. [2016]), an extension of DMM that includes word embeddings into the Gibbs sampling process. This section first introduces LDA and then discusses BTM, LF-LDA and GPU-DMM.

3.1.1 Latent Dirichlet Allocation

Blei et al. (Blei et al. [2003]) formalized a generative probabilistic technique especially designed to discover topics in corpora, the *latent Dirichlet allocation* (LDA) method. LDA is a probabilistic graphical model, in this case a Bayesian network that assumes a generative process of document writing, as follows.

Suppose there are latent topics in the mind of a writer (e.g. *politics* and *technology*) that guide the process of writing documents. A particular document is understood as a probabilistic mixture of topics (e.g. 65% of chance for *politics* and 35% for *technology*). Each topic, in turn, is represented as a probabilistic mixture of words in the vocabulary. For example, the word “government” has a higher probability to belong to the topic *politics* than the word “software”, which has a higher probability of belonging to the topic *technology*.

In LDA, a document of size N is assumed to be written following a particular Dirichlet distribution over K topics. Each topic is probabilistic selected from this distribution. Vocabulary words for a selected topic are also probabilistic sampled from the distribution. This process is repeated until N words are selected and the document is complete.

The generative process described above is clearly an oversimplification of the process of document writing, since the real process is much more complex. Given a set of documents, and assuming they were generated using the aforementioned process, LDA learns to infer the original set of topics (latent variables) from the observed words in documents (visible variables).

The LDA Bayesian inference model is shown in plate notation¹ in Figure 3.1. The variable $|D|$ indicates the collection size (number of documents), N is the size of each document (number of terms) and K the number of topics. The only visible variable (with gray background) is T , the terms in documents. The variable α is a positive real parameter, the *a priori* Dirichlet probability distribution of topics per document, which usually is the same for all topics and controls the sparsity of the posterior distribution. Specifically, high values of α tend to produce less discriminative distribution of topics per document. The role of the variable β is similar to the one of α , but it controls the sparsity of the posterior distribution of words per topic. High values of β produce less discriminative distribution of words per topic. Parameters α and β can be estimated using Minka's fixed point iteration technique (Minka [2000]).

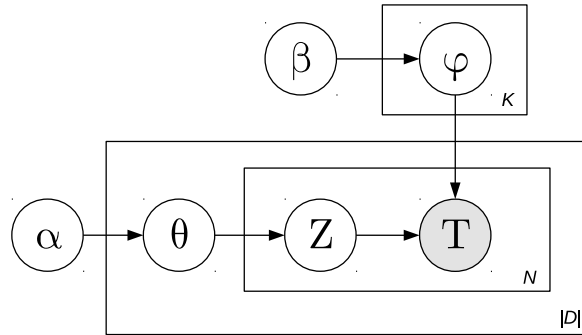


Figure 3.1: LDA probabilistic graphical model in plate notation.

Following with the plate model representation, variable φ_k is the posterior distribution of vocabulary words for topic k , for $k = 1 \dots K$. Variable θ_m is the posterior distribution of topics for document m , for $m = 1 \dots |D|$. Finally, variable Z_{mn} is the selected topic for document m and word n , for $n = 1 \dots N$.

The inference problem of LDA is to calculate the posterior distribution of latent variables θ , Z and φ , according to equation 3.1.

$$p(\theta, Z, \varphi \mid T, \alpha, \beta) = \frac{p(\theta, Z, \varphi, T \mid \alpha, \beta)}{p(T \mid \alpha, \beta)} \quad (3.1)$$

Since calculating this probability distribution is computationally expensive, approximation methods are considered. Blei et al. [2003] use variational inference (Beal [2003]). Griffiths and Steyvers [2004] use a Markov Chain Monte Carlo (MCMC) method known as Gibbs sampling (Geman and Geman [1984]). Another alternative

¹The plate notation for Bayesian models allows the compact representation of wide networks by expressing repetition structures only once.

is to use expectation propagation (Minka and Lafferty [2002]). The inference method most popular in the literature is the Gibbs sampling.

Although LDA is a powerful method to deal with longer documents, it presents some difficulties for dealing with short text. This is mainly due to the limited context available in short text documents, which produces low frequency of word co-occurrence. The next section presents the BTM model, which tries to circumvent this problem by aggregating terms.

3.1.2 Biterm Topic Model

Biterm topic model (BTM) (Yan et al. [2013]) tackles the sparsity problem by explicitly incorporating into the probabilistic graphical model the patterns of co-occurrence of words in documents. When doing this, it ignores the boundaries imposed by document modeling, considering co-occurrences in the whole corpus. While other probabilistic topic models also implicitly capture this notion of word co-occurrence in the learning process through statistical inference, BTM explicitly includes the biterms in the model.

Biterms are pairs of words that co-occur in the same document². They are represented in the graphical model of Figure 3.2 as the pair of variables T . Notice that the BTM graphical model is similar to LDA, except that there is no plate for documents and that probabilities are inferred for biterms instead of individual words. This is convenient for overcoming the sparsity problem, but the posterior probabilities of topics per document have to be derived after the learning process. Specifically, the proportion of topics for a document is inferred from the proportion of topics for the biterms of the document. The generative process of BTM is described in Algorithm 1. Observe that there is only one distribution of probability of topics for the whole corpus (line 4) and that, different from LDA, biterms are drawn instead of single words (line 7).

Algorithm 1 BTM generative process

Require: α, β, D ▷ Dirichlet priors and corpus
1: Initialize Z ▷ Topics
2: **for** $z \in Z$ **do**
3: $\varphi_z \sim Dir(\beta)$ ▷ Multinomial distribution over the vocabulary
4: $\theta \sim Dir(\alpha)$ ▷ Multinomial distribution over the topics for the whole corpus
5: **for** $i, j \in D$ **do** ▷ Biterms in the corpus
6: $Z_{ij} \sim Multinomial(\theta)$ ▷ Draw a topic
7: $T_i, T_j \sim Multinomial(\varphi_z)$ ▷ Draw words

²Do not confuse with *bigrams*, which are pairs of words that co-occur sequentially in the same document. Bigrams are biterms, but not the other way round.

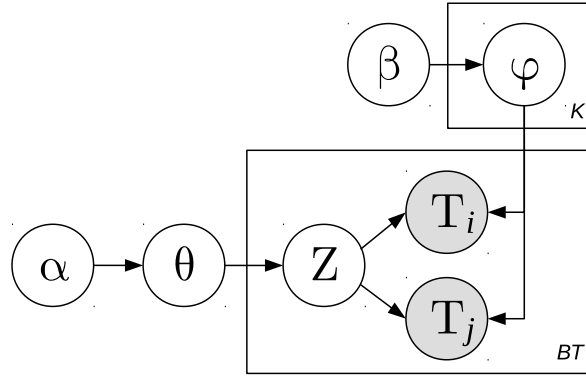


Figure 3.2: BTM probabilistic graphical model.

3.1.3 Latent Feature LDA

Nguyen et al. [2015] proposed an extension of LDA, the *latent feature LDA* (LF-LDA)³, which incorporates knowledge from word vectors to the graphical model to give more density to word semantics.

Figure 3.3 shows the probabilistic graphical model of LF-LDA. Notice that it is largely similar to LDA, except for the relationship between topics and words variables. The new variables τ and ω represent matrices of latent features weights for topics and words, respectively. The probability of LF-LDA generating a word w is a categorical distribution⁴ over $\tau_t \cdot \omega^T$, according to Equation 3.2. The product $\tau_t \cdot \omega^T$ can be interpreted as a vector of scores indexed by words. The variable ω is visible and fixed, because LF-LDA uses pre-trained word vectors.

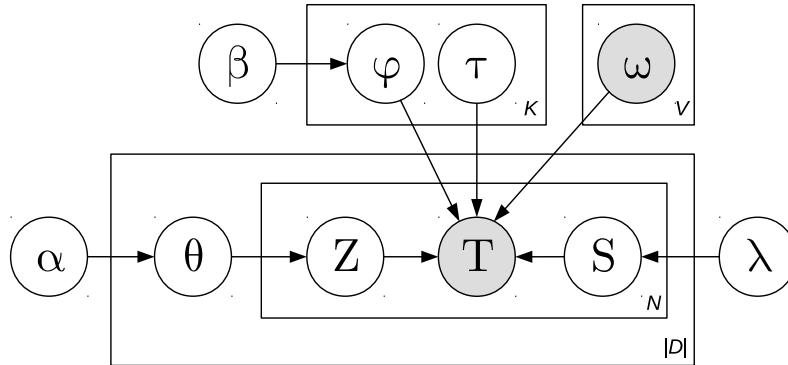


Figure 3.3: LF-LDA probabilistic graphical model.

$$\text{Cat}E(w \mid \tau_t \cdot \omega^T) = \frac{\exp(\tau_t \cdot \omega_w)}{\sum_{w' \in V} \exp(\tau_t \cdot \omega_{w'})} \quad (3.2)$$

³The expression *latent feature vectors* are also used as a synonym of word vectors.

⁴A categorical distribution is the multinomial version of the Bernoulli distribution.

The parameter λ is the probability of a word being generated by word and topic vectors, instead of the traditional inference of LDA. The generative process of LF-LDA is described in Algorithm 2.

Algorithm 2 LF-LDA generative process

Require: $\alpha, \beta, \lambda, D, \omega$ ▷ Priors, corpus and word vectors matrix
 1: Initialize Z ▷ Topics
 2: Initialize τ ▷ Topic vectors
 3: **for** $z \in Z$ **do**
 4: $\varphi_z \sim Dir(\beta)$ ▷ Multinomial distribution over the vocabulary
 5: **for** $d \in D$ **do**
 6: $\theta_d \sim Dir(\alpha)$ ▷ Multinomial distribution over the topics
 7: **for** $w \in d$ **do** ▷ Future words in d
 8: $S_{d,w} \sim Ber(\lambda)$ ▷ Binary switch (0 or 1)
 9: $Z_{d,w} \sim Cat(\theta_d)$ ▷ Draw a topic
 10: $T_{d,w} \sim (1 - S_{d,w})Cat(\varphi_{Z_{d,w}}) + S_{d,w}CatE(\tau_{Z_{d,w}}\omega^T)$ ▷ Draw a word

Gibbs sampling is used as the inference method. After each Gibbs sampling iteration, the topic vectors τ are estimated using regularized maximum a posteriori (MAP) likelihood.

3.1.4 GPU-DMM

Li et al. [2016] propose an extension of the DMM model (Yin and Wang [2014]) that uses an external auxiliary set of word embeddings to find semantically related words in the Gibbs sampling process. The DMM model is similar to LDA, but it assumes only one topic per document, according to the probabilistic graphical model shown in Figure 3.4. The variable d represents the observed document.

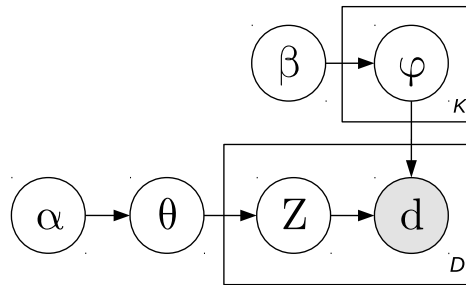


Figure 3.4: DMM probabilistic graphical model.

The GPU-DMM process is illustrated in Figure 3.5. For each document, after the topic inference process in the Gibbs sampling (*Word Filtering* phase), the generalized Pólya urn (GPU) algorithm (Mahmoud [2008]) promotes, for the selected topic, the

words that are semantically related to the words in the document. A word is considered related to another if the cosine similarity between their word vectors is greater than a threshold ϵ . The reinforcement of the probability of related words is probabilistic and proportional to the strength of the original word in the topic.

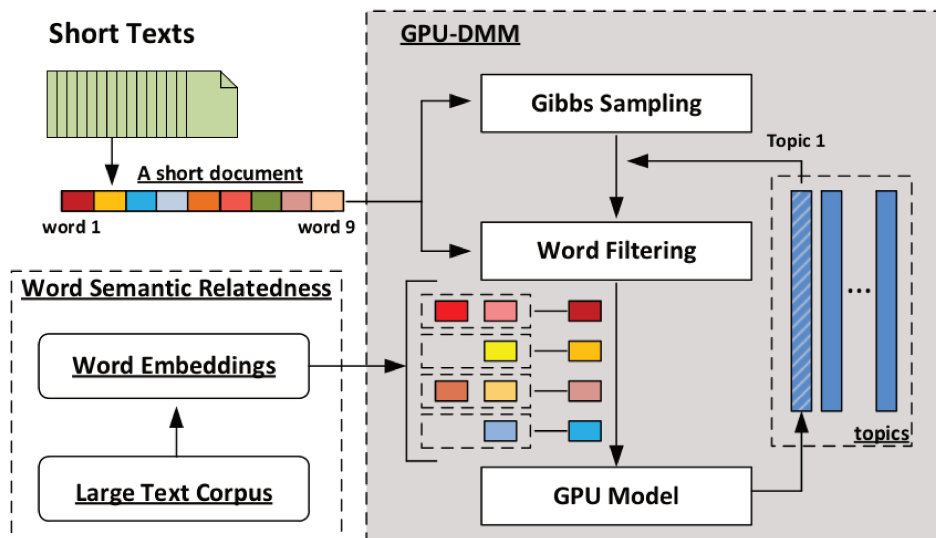


Figure 3.5: Illustration of the GPU-DMM model. (Source: Yin and Wang [2014])

3.2 Pseudo-document methods

This section presents methods that modify the input data to create pseudo-documents that presumably carry more semantic information than short texts. The objective of creating pseudo-documents is the same of the other methods presented so far: deal with the sparsity problem in short text topic modeling.

The task of generating pseudo-documents for topic modeling is closely related to other research areas, such as short text mining (Rosso et al. [2013]; Rafeeqe and Sendhilkumar [2011]) and automatic query expansion (Carpineto and Romano [2012]). In general, these methods are based on the expansion of the original short documents or queries to achieve longer versions of the same text.

As representatives of this category of short text topic model, this section presents the self-term expansion (STE) method and the word network topic model (WNTM). We also describe LDA#, which is an application-dependent method – it was proposed to extract topics from Twitter messages – but that is also relevant in the context of this work.

3.2.1 Self-term expansion

Pinto et al. [2011] propose the *self-term expansion* (STE) method, originally for short document clustering in narrow domains (i.e. domains whose documents have a high vocabulary overlapping). Posteriorly STE was adapted for topic modeling in Bicalho et al. [2017].

STE first calculates a correlation score for each pair of words in the corpus. The authors argue that this score may be any metric based on co-occurrence of terms, so they used pointwise mutual information (PMI), an information theory-based co-occurrence metric defined in Equation 3.3. In the equation, $P(t_i t_j)$ is the number of times terms t_i and t_j appear together in the same context (in this case, the document), and $P(t)$ is the number of times term t appears alone.

$$\text{PMI}(t_i, t_j) = \log_2 \frac{P(t_i t_j)}{P(t_i)P(t_j)} \quad (3.3)$$

This correlation score produces a co-occurrence list, used to expand the documents. The procedure is simple, and consists of concatenating each term of a document with its corresponding set of co-related terms. Correlated terms for each word are defined through a term selection technique. The authors suggest three techniques: (i) *document frequency* – the number of documents the term appears; (ii) *term strength* (Pekar et al. [2004]) – the consistency of a term appearing in similar documents; and (iii) *transition point* (Pinto et al. [2006]). The threshold used for selecting words is a percentage of the vocabulary, which the authors vary from 10% to 90%.

One advantage of STE when compared to other text expansion methods is that it does not use any external dataset to enhance the documents (Pinto et al. [2011]). This makes the method domain independent, and is one less variable the user to worry about.

3.2.2 Word network topic model

Zuo et al. [2016b] present a word co-occurrence graph-based model named *word network topic model* (WNTM) to generate pseudo-documents. The work relies on the following assumptions: (1) while the word-by-document space is sparse, the word-by-word space for the whole corpus remains dense, so it is interesting to learn topics from the word co-occurrence network; (2) as the number of words is larger than the number of documents associated with rare topics, the distribution of words per topic is less skewed, so topics should to be estimated for words instead of documents.

The first step of the method is to generate the word co-occurrence network (or simply *word network*). In this structure, words of the corpus are nodes and undirected edges indicate some degree of co-occurrence between words in the same context. This context can be a document or an arbitrary word neighborhood. The absence of an edge between two words indicates that they never co-occurred in the same context. Nodes also have a degree – which is the sum of all edges weights, and an activity level – which is the average of all adjacent weights. The method creates one pseudo-document per word, with each pseudo-document being formed by the word itself and its neighbors.

The generated pseudo-documents are used as input to LDA. The authors argue that this input transformation changes the original LDA generative model, because it no longer models the generation of documents, but the patterns of co-occurrence in the word network.

Like in BTM, WNTM indirectly estimates the distribution of topics per document, since the characteristics of the original dataset is lost in the process of generating the pseudo-documents. This is done according to Equation 3.4, where $P(z|d)$ is the desired distribution of topics for document d , $P(z|w_i)$ is the known distribution of topics per word and $P(w_i|d)$ can be inferred by the presence of w_i in d .

$$P(z|d) = \sum_{w_i} P(z|w_i)P(w_i|d) \quad (3.4)$$

3.2.3 LDA-#

Mehrotra et al. [2013] propose an input modification method for topic modeling in Twitter, generating pseudo-documents that are the output of pooling schemes and then processed with LDA. The authors highlight the aggregation of tweets by hashtag and an automatic hashtag labeling algorithm for unlabeled tweets. We call it LDA-#.

The following pooling schemes are proposed: (1) *unpooled*, where tweets are not aggregated by any heuristic; (2) *author-wise pooling*, which produces one document per author; (3) *burst-score wise pooling*, which is related to trends in Twitter (trend topics) and it is based on the detection of explosion in the frequency of some words in a given period of time; (4) *temporal pooling*, where tweets posted within the same hour are pooled; (4) *hashtag-based pooling*, which produces one document per Twitter hashtag.

3.3 Adding context to topic modeling

Some methods discussed in the previous section use word embeddings to add context to the topic modeling of short text. This is a successful trend in the literature, as it enriches the topic modeling process with information about the context in which the words appear, an essential but most of the time missing information in short text documents.

From the methods described so far, LF-LDA and GPU-DMM take advantage of word embeddings. The methods proposed in this thesis follow this line, and hence will be compared with these state-of-the-art methods.

Apart from the methods already discussed, there is two recent matrix factorization short text topic model we consider relevant in the context of this work, and for this reason they are discussed in the next subsections.

3.3.1 SeaNMF

Semantics-assisted NMF (SeaNMF) (Shi et al. [2018]) is a short text topic model that learns the semantic correlations between words and their contexts using a skip-gram view of the corpus (Mikolov et al. [2013c]), and then explores this information to discover topics through NMF. The skip-gram model is used to infer the relationship between words and their contexts in a sliding-window. In the case of SeaNMF, the window is the entire short text document. The process is illustrated in Figure 3.6.

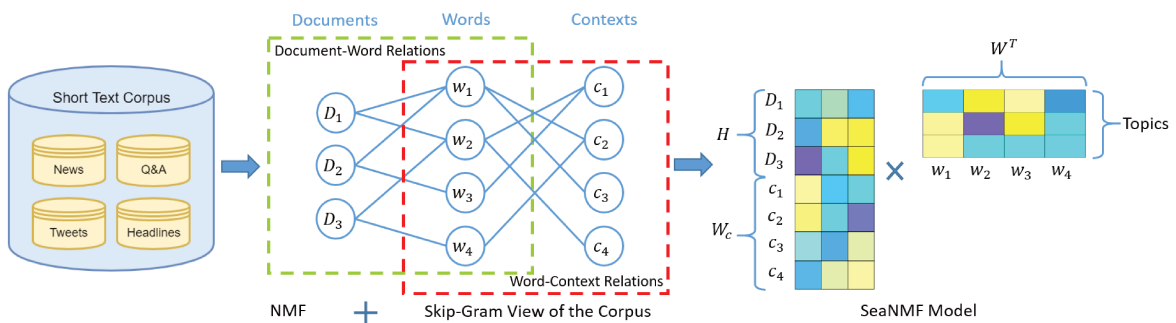


Figure 3.6: Illustration of the SeaNMF model. (Source: Shi et al. [2018])

Note that SeaNMF does not use an external pre-trained word embeddings to define the word context matrix. Instead, it learns the context representation using the corpus. This context consists on the word co-occurrence matrix and it is equivalent to a skip-gram model with variable context window (in this case, the short document) and hidden layer with the size of the vocabulary, constrained only to positive values. Levy

and Goldberg [2014b] documents this process as *shifted positive pointwise mutual information* (PPMI). The document-term matrix (TF), associated with the co-occurrence term-term matrix (PPMI), are factorized using NMF to produce the positive matrices at the end of the process exhibited in Figure 3.6.

3.3.2 CluWords

Viegas et al. [2019] propose a novel word embedding-based document representation named CluWords (Clusters of Words), which uses the similarity between pairs of word vectors previously learned from a large external dataset (e.g. Wikipedia). CluWords are combined with non-negative matrix factorization (NMF) for topic modeling.

The concept of a CluWord is similar to that of a hyperword (see section 2.3.5). Each word in the dataset vocabulary V is mapped to a CluWord, a vector of size $|V|$ whose values are the cosine similarity between the correspondent word vectors above a threshold α (otherwise, it is zero). The representation of documents is equivalent to the BOHW (bag of hyperwords) (section 2.3.5), an adaptation of the TF-IDF representation taking into account the CluWords.

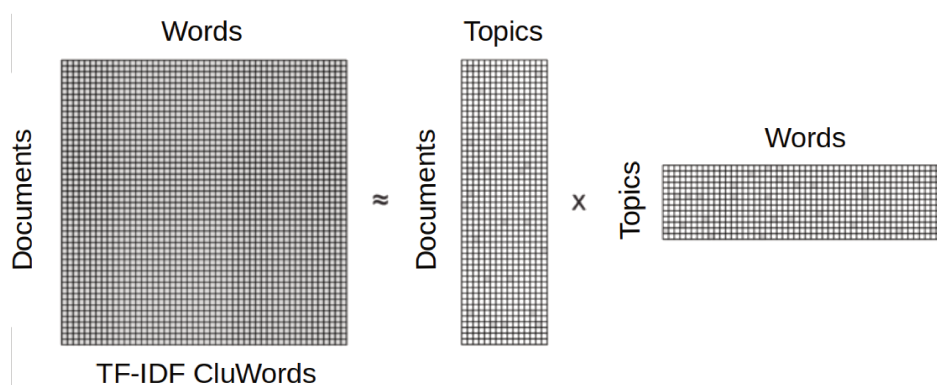


Figure 3.7: Illustration of the NMF topic model applied to TF-IDF CluWords representation.

3.4 Discussion about the methods

This section discusses the similarities and differences of the short text topic modeling techniques presented so far. Table 3.1 lists seven general features of the discussed methods: whether they are LDA adaptations, if they take advantage of word embeddings, if they models explicitly patters of co-occurrence, if they perform document pooling, if they use pseudo-documents and if they are application-dependent.

Table 3.1: General features of short text topic modeling methods.

	BTM (2013)	STE (2011)	WNTM (2016)	LF-LDA (2015)	GPU-DMM (2016)	LDA-# (2013)	SeaNMF (2018)	CluWords (2019)
LDA adaptation	•			•	•			
Word embeddings				•	•		•	•
Explicit patterns of co-occurrence	•	•	•				•	•
Document pooling	•		•			•		
Pseudo-document		•	•			•		
Application-dependent						•		

Previous sections organized the methods according to the following criteria: probabilistic graphical models or adaptations of LDA; and generation of pseudo-documents. These criteria are also features in Table 3.1 along with other general features.

The methods that produce pseudo-documents (feature “Pseudo-document”), compared to probabilistic graphical models (feature “Probabilistic graphical model”), have advantages and disadvantages. The main advantage is the independence of the topic modeling technique, since the usual approach is to transform the input data with the objective of improving the quality of a user-defined short text topic modeling method. On the other hand, a major disadvantage is the extra memory and computational costs to proceed the transformation.

The mentioned word embedding-based short text topic models, namely, CluWords, GPU-DMM and LF-LDA, explore semantic properties of word embeddings. An disadvantage of this approach is the dependence of an external source of knowledge (i.e. the word vectors). The SeaNMF method, although use a word co-occurrence matrix equivalent to a positive skip-gram word embeddings matrix, does not get this context information from an external source, but from the corpus.

In the context of this thesis, we highlight the short text topic models that use word embeddings. Chapters 5 and 6 present contributions in short text topic modeling that explore word embeddings with the same objective of aggregating semantic and contextual information of words into the topic modeling process. The first one proposes a framework for pseudo-document generation. The second, a graph-based probabilistic topic model.

Application-specific methods (feature “Application-dependent”), such as the ones for Twitter, have the advantage of exploiting the domain information. For example, the methods for Twitter topic modeling can use information about authors, hashtags, user geographical location, etc. The same advantage becomes a disadvantage, because the method is rarely enabled to function outside the application context.

The feature of “Document pooling” evaluates if documents are pooled (i.e. aggregated) in some way. This characteristic is important because, in this case, the representation of original documents is lost. The methods BTM, WNTM and LDA-# perform pooling. In BTM, for example, the biterms are defined in the context of documents, but the document object itself is not represented. WNTM redefines the context according to patterns of words co-occurrence in a graph. LDA-# is based on pooling schemes for tweets, that is, tweets are aggregated according to a specified criterion, such as hashtag or author. In any of these techniques, document representations are not directly extracted.

There is also the feature “Explicit patterns of co-occurrence”, which is present in methods where patterns of words co-occurrence are explicitly modeled. For example, the presence of biterms in the BTM model, co-occurrence edges in the WNTM’s word graph, the PMI-based document expansion in the STE method, the word co-occurrence matrix in SeanNMF or the CluWords TF-IDF-based matrix.

Short text topic models presented in this chapter that are cited in literature as reporting state-of-the-art results include CluWords, GPU-DMM, SeaNMF and BTM. All methods shown in Table 3.1 were included in one or more experiments in Chapters 5 and 6.

Chapter 4

Short Text Vector Representations for Document Classification

As explained in Chapter 2, many document representation models have been proposed in the literature and successfully applied to different problems and domains, most of them based on vector representation. This chapter investigates and proposes strategies for creating word embedding-based representations that improve short text classification.

We defined our research assumptions about distributed representation of documents on the top of the representation of text atomic units, *words*. To keep conceptual consistency with distributed representations, *word vector models* are our basic framework from which *document vector models* are derived. Given this premise, the fundamental question on deriving document embeddings from word embeddings is: *How can we consistently combine word vectors?*

The literature shows that useful word vectors models must capture semantics by implementing at least two features: (1) similarity of word concepts, expressed in terms of word vector similarity (usually cosine similarity); and (2) combinations of word concepts, expressed in terms of word vector operations, such as vectors sum and difference. Both features are important to answer the question stated in the previous paragraph, but in practice they have issues that are related to the cumulative error when performing vector operations with relatively many words (Le and Mikolov [2014]).

While exploring word vector arithmetic is certainly a good starting point, the literature states that the direct chained application of such vector operations for minimally long sequences does not produce good results (Le and Mikolov [2014]). We investigate the use of arithmetic operations for short text, but we are particularly interested in other forms of word combination.

We propose a novel method, *PSO-based weighted average of word vectors* (PSO-WAWV), which finds sub-optimal weights in weighted average of word embeddings for document classification, as detailed in the next section.

4.1 Word vector combinations with PSO

The usually recommended way of combining word vectors is through averaging¹ (Socher et al. [2013b]; Le and Mikolov [2014]). In this case, for example, for a short document d “President Trump will visit North Korea”, the vector representation of d , after the removal of the stop word “will”, is $v(d) = (v(\text{president}) + v(\text{trump}) + v(\text{visit}) + v(\text{north}) + v(\text{korea}))/5$, i.e. the average of the corresponding 5 word vectors in the text.

In the case of this average, all words from the document have the same weight. But what if the importance of words is different within a given corpus? Finding an ideal set of weights for words in the vocabulary is a *continuous optimization problem*. In our particular case, it consists of finding real values that, multiplied to their respective word vectors, produce document representations with optimal or sub-optimal classification performance. The PSO-WAWV method performs this optimization using the *particle swarm optimization* (PSO) metaheuristic (Zhang et al. [2015]).

4.1.1 Particle Swarm Optimization

Particle swarm optimization (PSO) is a method for global optimization proposed by Kennedy and Eberhart [1995] and inspired by the social behavior of animals.

In the PSO algorithm, individuals are represented as *particles* that move in a search space and are evaluated by a *fitness function* f . Iteratively, PSO changes the position of particles – the free learning parameters of f – in search of the optimal value, usually the global minimum or maximum. The dynamics of a particle is controlled by a set of combined factors: its current position; a *cognitive factor*, which is the influence of the own particle history of success; the *social factor*, which is the influence of other particles in the neighborhood; and randomness, which helps the algorithm not to get stuck at local optima.

The basis of the PSO algorithm are the equations for updating the *velocity* and *position* of particles. Since particles exist in a multidimensional optimization space,

¹See the recommendation of word vectors averaging or concatenation by Richard Socher in the Lecture 2 of the Stanford class “Deep Learning for Natural Language Processing” (March, 2016). Video available at: <https://youtu.be/aRqn8t1hLxs?t=46m46s> (last access: 2018/04/20). Concatenating, although suggested, is not an actual option, since it does not produce fixed-length document vectors.

their position and velocity are real vectors. At iteration $t + 1$ of the heuristic, the velocity vector of a particle i is updated according to Equation 4.1.

$$\mathbf{v}_i(t+1) = \underbrace{\mathbf{v}_i(t)}_{\text{Inertia factor}} + \underbrace{c_1 r_1 (\mathbf{p}_i(t) - \mathbf{x}_i(t))}_{\text{Cognitive factor}} + \underbrace{c_2 r_2 (\mathbf{n}_i(t) - \mathbf{x}_i(t))}_{\text{Social factor}} \quad (4.1)$$

where $i = 1, \dots, P$, P is the number of particles, c_1 and c_2 are the cognitive and social factors, respectively, r_1 and r_2 are random numbers from a uniform distribution $U(0, 1)$, $\mathbf{p}_i(t)$ is the best position of the particle found so far, $\mathbf{n}_i(t)$ is the best position of the neighborhood found so far, and $\mathbf{x}_i(t)$ is the current position of the particle. The neighborhood of a particle is defined by a communication topology, but in general it consists of the whole set of particles. Once the velocity of all particles are updated, their positions are also updated according to Equation 4.2.

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1) \quad (4.2)$$

The classic PSO heuristic pseudo-code is described in the Algorithm 3.

Algorithm 3 Particle swarm optimization pseudo-code

- 1: Initialize P particles with random \mathbf{x}_i (position) and \mathbf{v}_i (velocity)
 - 2: Set each $\mathbf{p}_i = \mathbf{x}_i$
 - 3: Set each \mathbf{n}_i as the best position found in the neighborhood of i
 - 4: **while** stop criteria not met **do**
 - 5: **for** $i \in P$ **do**
 - 6: Calculate the particle fitness $f(\mathbf{x}_i)$
 - 7: Update \mathbf{v}_i and \mathbf{x}_i (Equations 4.1 and 4.2)
 - 8: Update \mathbf{p}_i
 - 9: Update particles \mathbf{n}_i
-

The PSO algorithm was widely extended and applied in many domains (Zhang et al. [2015]). In the present work we used a variation that incorporates a *constriction coefficient* χ in the velocity equation, with the objective of avoiding the explosion of particles velocities, a frequent problem in the classic PSO. The constriction coefficient is dependent on constants c_1 and c_2 and is calculated according to Equation 4.3.

$$\chi = \frac{2}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|}, \quad \varphi = c_1 + c_2 \quad (4.3)$$

The velocity equation with the incorporation of χ is represented by Equation 4.4.

$$\mathbf{v}_i(t+1) = \chi [\mathbf{v}_i(t) + c_1 r_1 (\mathbf{p}_i(t) - \mathbf{x}_i(t)) + c_2 r_2 (\mathbf{n}_i(t) - \mathbf{x}_i(t))] \quad (4.4)$$

Eberhart and Shi [2000] compared the constriction coefficient approach with the inertia weights, another approach to limit velocities, and concluded that the constriction factor strategy is the best. Clerc and Kennedy [2002] provides a convergence proof when $\varphi > 4$, for which we use the recommended $\varphi = 4.1$ (i.e. $c_1 = c_2 = 2.05$).

4.1.2 The PSO-WAWV Model

We propose a general weighting scheme, in which weights – one per word in the vocabulary – are free parameters to be adjusted in an optimization process. The learning of weights in the PSO-WAWV model is accomplished by the PSO algorithm with constriction factor. The positions of particles are real vectors $t_1, t_2, \dots, t_{|V|}$, where $|V|$ is the number of terms in the vocabulary. The fitness function f is task dependent. In our case it is a metric to assess document classification performance, to be maximized. Figure 4.1 illustrates this optimization problem with fictitious particles and function. Here the fitness is the F1-score.

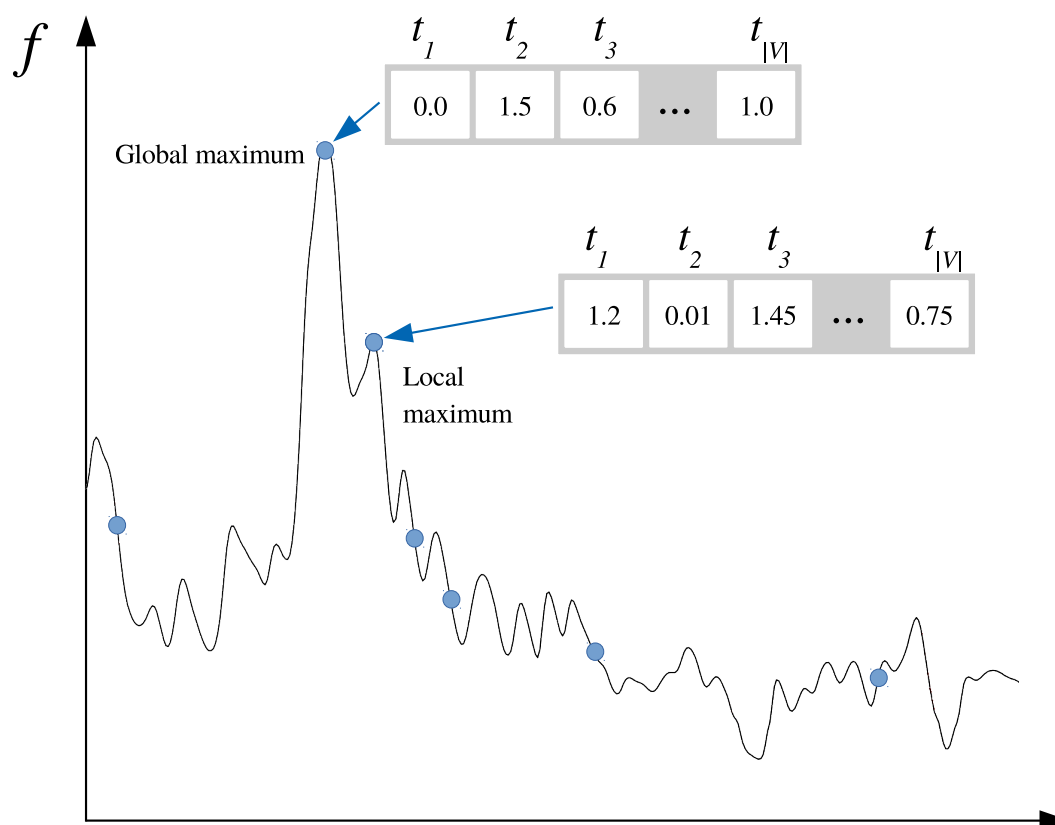


Figure 4.1: Illustration of PSO-WAWV optimization problem, with some fictitious particles (blue circles) of size $|V|$.

4.2 Arithmetical combinations of word embeddings

4.2.1 Sum of word vectors (SWV)

The direct sum of word vectors is reported in literature as a non-effective method for producing general representations of long documents (Le and Mikolov [2014]). The procedure has the advantage of simplicity, and it consists on the sum of all word vectors correspondent to the words of the target document. For example, consider the following document d : “President Trump will visit North Korea”. The vector representation of d , after the removal of the stop word “will”, is the reduction $v(d) = v(\text{president}) + v(\text{trump}) + v(\text{visit}) + v(\text{north}) + v(\text{korea})$.

The word vectors’ literature has mainly reported results for the task of relational similarity² with the objective of verifying semantic capabilities of word vectors. Implementations of the relational similarity task are usually performed with normalized word vectors (length equal to 1), because the interpretation of the vectors’ length is not related to word semantics, but with the word frequency in the dataset. Furthermore, the similarity between words is determined by the cosine metric, which is insensitive to vector length. The interpretation of word vector length is confirmed in the word2vec-toolbox mailing list by Mikolov [2014]:

“The length of the vectors is closely related to the word frequency, ie. more frequent words will be represented by longer vectors than infrequent words. This is related to the training algorithm which starts with very small vectors, and the incremental updates make the vectors longer.”

The SWV procedure is performed using raw word vectors, i.e. unnormalized. This means that words are indirectly weighted by their frequency in the dataset. Figures 4.2(a) and 4.2(b) illustrate, with two bi-dimensional vectors, the normalized and non-normalized sum of word vectors, respectively³. Note that in Figure 4.2(b) the resultant vector is closer to vector A, because A is bigger than B. Therefore, the SWV method overestimates words with higher frequency.

²The classic example of a relational similarity question is: “*man*” is to “*king*” as “*woman*” is to...? – for which the answer is “*queen*”. The test in this case is algebraically expressed by the operation $v(\text{king}) - v(\text{man}) + v(\text{woman})$, which must produce a vector whose nearest word vector is $v(\text{queen})$.

³Figures are merely illustrative and vectors are two-dimensional for didactic purposes. In general, actual word vectors have much higher dimensionality, usually tens or hundreds, sometimes thousands.

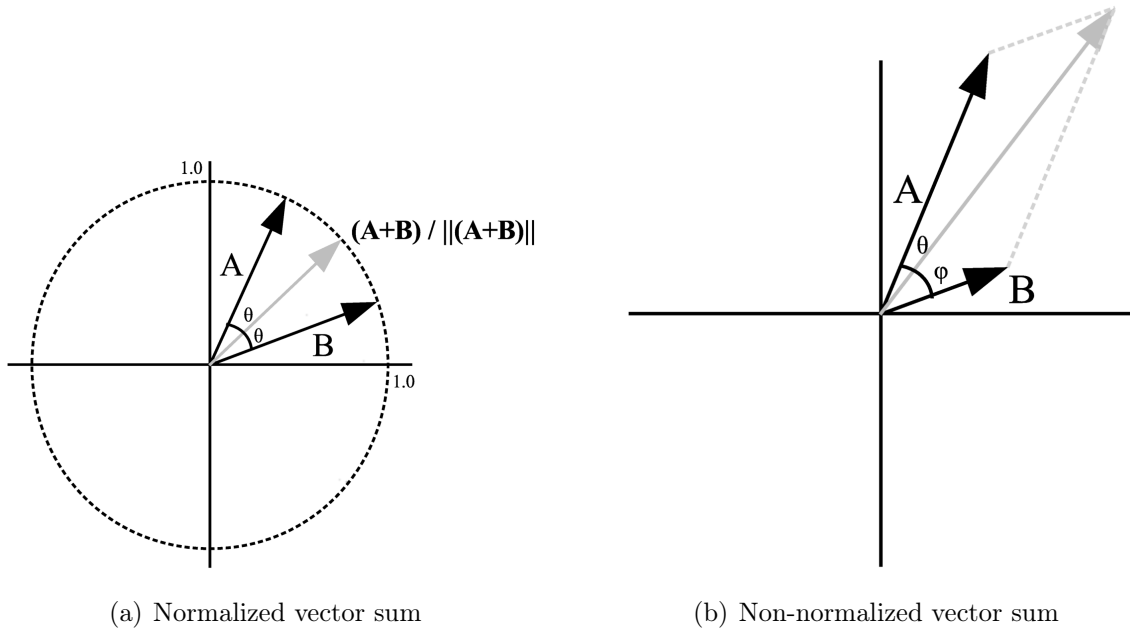


Figure 4.2: Illustrative comparison between normalized and non-normalized vector sum.

4.2.2 TF-IDF-Weighted Sum of Word Vectors (TFIDF-WSWV)

This operation is similar to SWV, except that the correspondent document's word vectors are weighted by their TF-IDF values, instead of the original vectors magnitude. The TF-IDF values are pre-calculated using the target collection and form a matrix $\mathbf{DT}_{|D| \times |V|}$, where $|D|$ is the number of documents in the collection and $|V|$ is the vocabulary size. Ultimately, $\mathbf{DT}_{|D| \times |V|}$ quantifies the importance of terms to documents and the corpus.

We also have the matrix $\mathbf{W}_{|V| \times d}$, which contains the learned word vectors, one vector of size d per row. Naturally, \mathbf{W} can be derived from any distributional semantics method applied to words, like those described in Chapter 2 (e.g. SG, CBOW, GloVe).

The matrix of document vectors $\mathbf{W}'_{|D| \times d}$ is the result of the product of matrices \mathbf{DT} and \mathbf{W} , i.e. $\mathbf{W}' = \mathbf{DT} \times \mathbf{W}$, one document vector per row. Equation 4.5 illustrates

the TFIDF-WSWV method, in matrix notation.

$$\mathbf{W}'_{|D| \times d} = \underbrace{\begin{matrix} & \begin{matrix} term_1 & term_2 & \dots & term_{|V|} \end{matrix} \\ \begin{matrix} doc_1 \\ doc_2 \\ \vdots \\ doc_{|D|} \end{matrix} & \begin{bmatrix} dt_{11} & dt_{12} & \dots & dt_{1|V|} \\ dt_{21} & dt_{22} & \dots & dt_{2|V|} \\ \vdots & \vdots & \ddots & \vdots \\ dt_{|D|1} & dt_{|D|2} & \dots & dt_{|D||V|} \end{bmatrix} \end{matrix}}_{\mathbf{DT}'_{|D| \times |V|}} \times \underbrace{\begin{matrix} & \begin{matrix} feat_1 & feat_2 & \dots & feat_d \end{matrix} \\ \begin{matrix} term_1 \\ term_2 \\ \vdots \\ term_{|V|} \end{matrix} & \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1d} \\ w_{21} & w_{22} & \dots & w_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ w_{|V|1} & w_{|V|2} & \dots & w_{|V|d} \end{bmatrix} \end{matrix}}_{\mathbf{W}_{|V| \times d}} \quad (4.5)$$

4.2.3 Average of Word Vectors (AWV)

This method has already been explained in the Section 4.1, in the context of the proposed method PSO-WAWV. It consists on obtaining the average vector that results from the word embeddings correspondent to the words in the document.

4.2.4 TF-IDF-Weighted Average of Word Vectors (TFIDF-WAWV)

The TFIDF-WAWV is a combination of the AWV and TFIDF-WSWV methods. Instead of performing an arithmetic mean of the word vectors like in AWV, we use the TF-IDF matrix information to perform a weighted average of the word vectors. In matrix notation, we perform the product of the *normalized* TF-IDF matrix $\mathbf{DT}'_{|D| \times |V|}$ by the matrix $\mathbf{W}_{|V| \times d}$ ⁴. So, the TFIDF-WAWV operation is basically similar to the TFIDF-WSWV method, except that the normalized \mathbf{DT}' have rows that sum 1.0.

4.3 Experimental setup

The methods were compared varying parameter d (word vector size). Document vector representations were evaluated using a logistic regressor. Experiments were repeated in a 10-fold cross-validation and F1 scores reported. Comparison among methods were statistically validated with the non-parametric Wilcoxon signed-rank hypothesis test over the means with 0.05 of significance level.

⁴The weighted average of values a , b and c , with weights w_a , w_b and w_c , is usually expressed by $\frac{w_a a + w_b b + w_c c}{w_a + w_b + w_c}$. This is equivalent to $aw_a/(w_a + w_b + w_c) + bw_b/(w_a + w_b + w_c) + cw_c/(w_a + w_b + w_c)$, or the multiplication of vectors $x = [a, b, c]$ and $y = [w_a/(w_a + w_b + w_c), w_b/(w_a + w_b + w_c), w_c/(w_a + w_b + w_c)]$. The vector y is precisely the normalized version (sum equal to 1) of the vector $[w_a, w_b, w_c]$.

Table 4.1: Dataset statistics.

Dataset	#Docs	Vocabulary size	#Classes	Words/doc	Unique words/doc
20nshort	1723	964	20	8.2 (± 3.5)	7.1 (± 2.9)
Sanders	3770	1311	4	6.1 (± 2.7)	5.8 (± 2.5)
Snippets	12117	4677	8	14.3 (± 4.4)	10.3 (± 3.1)
TMN	30376	6314	7	4.9 (± 1.5)	4.9 (± 1.5)

4.3.1 Datasets

Four short text datasets were used in the experiments⁵: 20 Newsgroups (20nshort); Tweets Sanders (Sanders); Web Snippets (Snippets); and Tag My News (TMN). Table 4.1 summarizes the datasets’ main characteristics including average number of terms and unique terms per document. All resultant datasets have few words per document on average, as shown by columns “Words/doc” and “Unique words/doc” in table 4.1. Standard deviation is also small. The largest dataset is TMN (30,376 docs) and the smallest is 20nshort (1,723 docs). The vocabulary size also presents a large variation (from 964 in 20nshort to 6,314 in TMN).

4.3.2 Text preprocessing

All datasets were submitted to the following preprocessing steps:

- Standardization of text to lowercase. This is important to avoid semantic differentiation of similar words based on case.
- Removal of non-alphabetic characters. This procedure is also related to avoiding semantic differentiation of similar words in free natural text (e.g. minimizing differentiation by errors in word accentuation) and exclusion of characters that are not informative to the current application (e.g. punctuation).
- Removal of stop words, i.e. words that are not relevant to analysis, such as articles, prepositions, some verbs, among others⁶.
- Removal of words shorter than 3 characters and longer than 25 characters, minimizing the chance of occurring “noise” words in text.

⁵Datasets available at: <https://github.com/marcelopita/datasets/>

⁶English stop words used in the experiments available at: http://homepages.dcc.ufmg.br/~marcelo.pita/short_text_tm/preproc/english_stopwords.

4.3.3 Word vectors

We used SG word vectors generated with the fastText software (Bojanowski et al. [2016])⁷, for vector sizes 100, 300, 600 and 1000. The training dataset was a dump of the English Wikipedia from 2015/02/06, containing 8,102,107 documents and 2,120,659 words. The size of the context window was defined to be 10, with negative sampling (5 negative examples) and default initial learning rate of 0.025.

4.3.4 Evaluation metric for classification

Macro average F1 score was the evaluation metric used. It consists of averaging the F1 score for all classes, which in turn is the harmonic mean between precision and recall for each class c , as shown in Equation 4.6.

$$F1_c = \frac{2 \times \text{Precision}_c \times \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c} \quad (4.6)$$

Precision_c is the fraction of correct predictions for c , according to Equation 4.7, where tpr_c is the true positive rate for c and fpr_c the false positive rate. Recall_c is the fraction of instances of c that were correctly predicted (Equation 4.8), where fnr_c is the false negative rate.

$$\text{Precision}_c = \frac{\text{tpr}_c}{\text{tpr}_c + \text{fpr}_c} \quad (4.7)$$

$$\text{Recall}_c = \frac{\text{tpr}_c}{\text{tpr}_c + \text{fnr}_c} \quad (4.8)$$

4.4 Results of arithmetical combinations of word embeddings

We first compare the quality of the document vectors generated by the SWV, TFIDF-WSWV, AWV and TFIDF-WAWV models, which will serve as baselines for the proposed method.

Table 4.2 shows the results for all models mentioned above, all datasets and varying the word vector size d . Higher values of d are expected to carry more information about the representation, therefore producing better results. This tendency is observed

⁷There is no essential difference between the SG model generated with fastText and the traditional word2vec implementation, except that fastText also creates vectors for subwords (i.e. parts of words). FastText is available at: <https://fasttext.cc/>.

in Table 4.2. Bold values indicate the best achieved for the dataset, and comparisons are non-blocked, i.e. all combinations of d and models are performed.

Table 4.2: Results of document classification (mean F1-score) for the 20nshort, Sanders, Snippets and TMN datasets, for each word vector operations and different values of d (word vector size).

	<i>Word vector size (d)</i>							
	100	300	600	1000	100	300	600	1000
	20nshort				Sanders			
SWV	0.412	0.441	0.475	0.505	0.973	0.985	0.987	0.984
TF-IDF-SWV	0.396	0.439	0.482	0.493	0.838	0.925	0.952	0.976
AWV	0.430	0.513	0.521	0.528	0.975	0.984	0.986	0.986
TF-IDF-WAWV	0.392	0.471	0.504	0.517	0.831	0.926	0.953	0.963
	Snippets				TMN			
SWV	0.904	0.916	0.927	0.936	0.731	0.753	0.761	0.762
TF-IDF-SWV	0.898	0.911	0.923	0.930	0.721	0.746	0.757	0.760
AWV	0.901	0.926	0.940	0.942	0.727	0.754	0.765	0.770
TF-IDF-WAWV	0.888	0.913	0.928	0.934	0.711	0.735	0.749	0.751

For the 20nshort, the AWV with $L = 1000$ (AWV-1000) method achieves the higher mean, although there is no statistical evidence that it is better than SWV-1000, AWV-300, AWV-600, TFIDF-WAWV-600 and TFIDF-WAWV-1000. AWV has 3 winning variations. The second best method is the TFIDF-WAWV (2 winning variations), followed by the SWV method (1 winning variation).

It is important to notice that the weighting scheme with TF-IDF, both in the TFIDF-WSWV and TFIDF-WAWV methods, seems to worsen the results of pure SWV and AWV, as suggested by the general classification performance drop observed in Table 4.2.

The document classification task for the Sanders dataset provides little opportunity for improvement, because all methods achieve high classification performance, according to Table 4.2. All best values (in bold) are above the F1 score of 0.98. Even so, it is worth to notice the superiority of the SWV and AWV methods.

The classification results for the Snippets and TMN datasets present very similar results. In both cases, AWV was the best method. Results also confirm that the TF-IDF weighting scheme worsen the results of pure SWV and AWV.

4.4.1 Scale of TF-IDF and word vector values

We investigated if the worse results with the TF-IDF weighting schema are related to differences in scale when word vectors are combined with TF-IDF values. Figure 4.3(a) displays a histogram of word vector values (whole Wikipedia collection), while Figures 4.3(b) to 4.3(e) show histograms of TF-IDF values (without zeros) for each short text dataset. Notice that word vector values follow an well shaped zero-mean normal distribution with most of values in the range $[-1, 1]$, while TF-IDF values are left-skewed with variable scale.

To adjust scales, TF-IDF values were transformed by a softmax function (Equation 4.9), which gives a probability interpretation to its values (range $[0, 1]$). Next, we multiplied the TF-IDF values by the average word vectors, originating the TFIDF-WAWV-SOFTMAX document vector model. Table 4.3 compares the results of TFIDF-WAWV-SOFTMAX-1000 with the original TFIDF-WAWV-1000. The conclusion is that results of both models present no statistical difference, suggesting that the difference in scale is not an issue for the logistic regression classifier.

$$\sigma(x_j) = \frac{e^{x_j}}{\sum_i e^{x_i}} \quad (4.9)$$

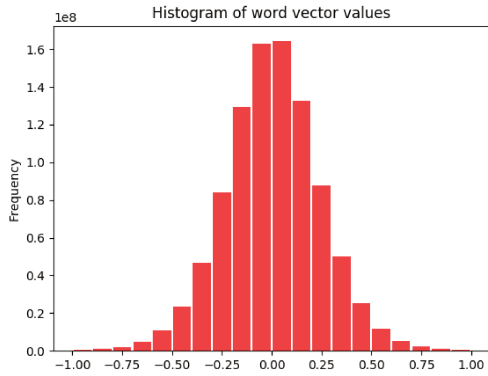
Table 4.3: Comparison of TFIDF-WAWV-1000 and TFIDF-WAWV-SOFTMAX-1000 methods.

Method	20nshort	Sanders	Snippets	TMN
TFIDF-WAWV-1000	0.517	0.963	0.934	0.751
TFIDF-WAWV-SOFTMAX-1000	0.513	0.966	0.933	0.752

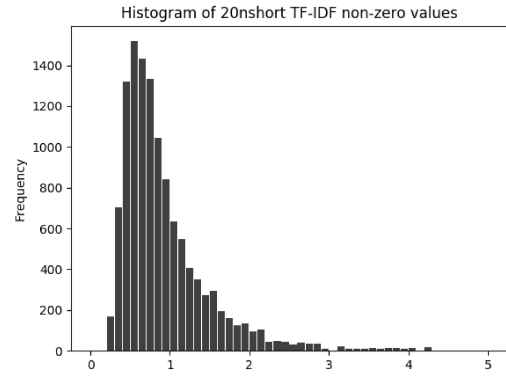
4.4.2 Qualitative analysis of TF-IDF weighting

One advantage of having document vectors represented in the same semantic space as the original word vectors is that they can potentially work together and be compared. We used this characteristic to analyse the document vectors' nearest words vectors for each class in the Snippets dataset. For each document of a class (*business*, *computers*, *culture-arts-entertainment*, *education-science*, *engineering*, *health*, *politics-society* and *sports*), we count the occurrence of the nearest word in the word vector space, and summarize by class. These counts were used to produce word clouds of the most representative words for TFIDF-WAWV-1000 and AWWV-1000, as shown in Table 4.4.

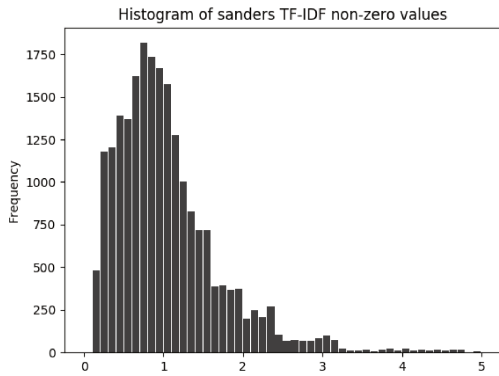
The word clouds of Table 4.4 can explain, at least partially, why the TF-IDF weighting scheme in the TFIDF-WAWV-1000 method produces worse results than



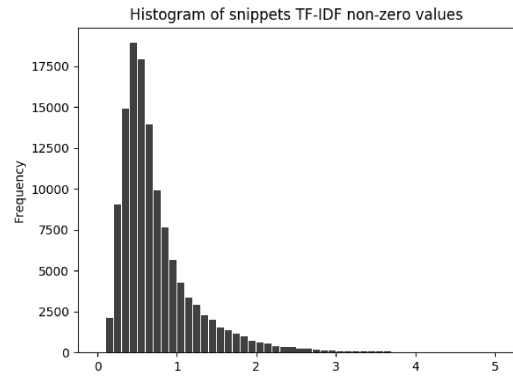
(a) Histogram of word vectors values.



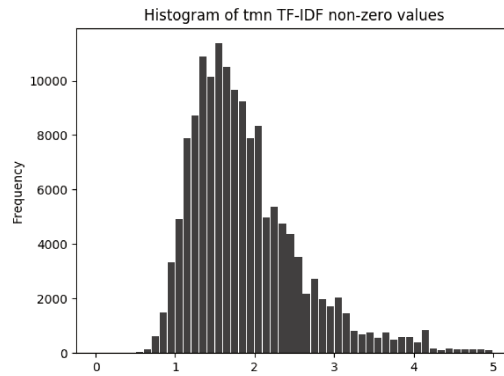
(b) Histogram of 20nshort TF-IDF values.



(c) Histogram of Sanders TF-IDF values.



(d) Histogram of Snippets TF-IDF values.



(e) Histogram of TMN TF-IDF values.

Figure 4.3: Histograms of word vectors and TF-IDF values.

AWV-1000. We can observe that, in general, TFIDF-WAWV-1000 documents are related to more specific words in the domain (class) than AWW-1000 ones, which suggests less generalization power. This can help to explain the worse classification performance

of TFIDF-WAWV-1000 in comparison with AWW.

Table 4.4: Word clouds of the most representative words in the Snippets dataset by class, for TFIDF-WAWV-1000 and AWW-1000 document vectors. The size of words are proportional to the their frequency in a list of documents' nearest words for each class. Colors do not encode any information.

Snippets Class	Method	
	TFIDF-WAWV-1000	AWV-1000
Business		
Computers		
Engineering		
Culture-Arts-Entertainment		

Continued on next page

Table 4.4 – Continued from previous page

Snippets	Method	
	TFIDF-WAVV-1000	AWV-1000
Class		
Sports		
Education-Science		
Health		
Politics-Society		

4.4.3 Summary

Table 4.5 sums up the results, showing the number of times each method was the winner, or was among the winners. Only two variations of AWV, namely AWV-600 and AWV-1000, were the best for all four datasets. These variations will be used as

baselines for the proposed PSO-WAWV method.

Table 4.5: Number of times each method was statistically the best or presented no statistical difference to other methods.

Method	20nshort	Sanders	Snippets	TMN	Total
SWV-300		•			1
SWV-600		•			1
SWV-1000	•	•			2
AWV-300	•	•			2
AWV-600	•	•	•	•	4
AWV-1000	•	•	•	•	4
TFIDF-WAWV-600	•				1
TFIDF-WAWV-1000	•				1

4.5 Results for PSO-WAWV

The classification performance of the representation produced by the proposed PSO-WAWV-1000 model was compared with the best variations of arithmetical combinations of word embeddings. It was compared with TF-IDF, PV and BOHW, as well.

4.5.1 Parameter settings

We used the PSO-WAWV with constriction coefficient and $\varphi = 4.1$, i.e. $c_1 = c_2 = 2.05$. The number of particles in all experiments were 25 and the total number of iterations was set to 150. The algorithm can stop earlier after 50 iterations without any increase in fitness values. We defined the minimum and maximum values for initialization of particles positions as $[0.8, 1.2]$. These values form a uniform distribution around 1.0 with a little standard deviation. When all values are 1.0, the representation becomes equal to AWV, which already presented good results. The objective is to promote a fine tuning of the weights. We also investigated if a second level of optimization in PSO-WAWV, namely, incorporating an optimization of weights for word vectors dimensions as well (PSO-WAWV-dims), would improve classification compared to original PSO-WAWV.

Three other baselines, apart from the best arithmetical combinations of word vectors were considered: TF-IDF, PV and BOHW.

For PV, we have replicated the IMDB document classification experiment of Le and Mikolov [2014] to adjust the parameters of PV. In this experiment, document

vectors have size 800 and are formed from the concatenation of PV-DBOW and PV-DM models, both of size 400. The concatenation is mentioned in Le and Mikolov [2014] as the best PV model. All experimental protocols were followed, including the precise separation of training and testing sets, as well as the usage of a logistic regressor. Test document vectors were generated by inference⁸ with 200 steps (learning epochs), although this parameter has not been defined in the paper. Authors report a classification error rate of 7.42%, result that could not be replicated by the community, including this author, whose best result was an error rate of 19.2%. For the experiments in this section, we used the general parameters of the IMDB replication, but with document vectors of size 1000, for a fair comparison with other methods.

The bag of hyperwords (BOHW), an extension of the bag of word TF-IDF representation that considers word vectors to determine the influence of words in the vocabulary on each document (described in Section 2.3.5), used word vectors of size 1000, dynamic α selection and $\beta = 0.3$.

Document classification was evaluated in a 10-fold cross-validation, i.e. 10 PSO runs. At each PSO execution, a fraction of the training set was used for validation. This validation set, used to evaluate particles, is responsible by the attribution of fitness (F1 score). The 20nshort and Sanders datasets used a validation fraction of 50%, while Snippets and TMN datasets used a validation fraction of 90%. This high fraction of validation data for Snippets and TMN does not affect the learning process, since these datasets have a high number of documents. The validation fraction is a parameter and must be adjusted according to dataset characteristics. Additionally, it controls the fraction of training data for each particle and, ultimately, the general performance of the PSO-WAWV algorithm.

4.5.2 Results

Table 4.6 shows the results for the PSO-WAWV-1000 model and the baselines. Notice that PSO-WAWV-1000 results and AWV results present no statistical difference. An exception to this is in the Snippets dataset, where the classification performance of PSO-WAWV-1000 is really worse than AWV, only with results better than PV-DBOW-DM-1000.

In the 20nshort dataset, BOHW-1000 presented worse results than other methods. In the Sanders dataset, all methods are statistically equivalent. In the TMN dataset, results are competitive with AWV and TF-IDF.

⁸The inference process in PV consists on fixing the neural network parameters for the word vectors and learning the free parameters for the new paragraph.

Table 4.6: Comparison of the method PSO-WAWV-1000 with baselines TF-IDF, PV-DBOW-DM-1000, BOHW-1000, AWV-600 and AWV-1000 using F1-score.

Method	Dataset			
	20nshort	Sanders	Snippets	TMN
TF-IDF	0.518	0.985	0.956	0.769
PV-DBOW-DM-1000	0.496	0.986	0.822	0.713
BOHW-1000	0.483	0.985	0.948	0.751
AWV-600	0.521	0.986	0.940	0.765
AWV-1000	0.528	0.986	0.942	0.770
PSO-WAWV-1000	0.549	0.990	0.914	0.773

The results of PSO-WAWV show that, besides the previous worse results using TF-IDF as weights, the general objective of finding optimal weights for a specific task is feasible. The weights found using the PSO optimization technique are competitive with the baselines for short text classification.

Analysis of the representation size

A clear advantage of the AWV and PSO-WAWV methods over TF-IDF is a more compact representation, i.e. less spatial complexity, and the non-sparsity, which allows a safer application of distance metrics without incurring in the curse of dimensionality. The reduction in the size of representation and correspondent improvements in classification performance are shown in Table 4.7.

Table 4.7: Reduction in the representation size and relative classification improvement for AWV-600, PSO-WAWV-1000 and TF-IDF models.

Dataset	TF-IDF	AWV-600		PSO-WAWV-1000	
	size	% size	% F1 score	% size	% F1 score
20nshort	964	-60.67%	+0.58%	+3.73%	+5.98%
Sanders	1311	-118.5%	+0.1%	-31.1%	+0.51%
Snippets	4677	-679.5%	-1.7%	-367.7%	-4.60%
TMN	6314	-952.33%	-0.52%	-531.4%	+0.52%

Extending PSO-WAWV with Weights for Dimensions of Word Embeddings

Table 4.8 shows the compared results for the variations PSO-WAWV and PSO-WAWV-dims (PSO-WAWV plus optimization of weights for word embeddings dimensions). Note that both methods are statistically equivalent in three datasets, except TMN, in which PSO-WAWV-dims shows an inferior result. We conclude that the addition of a second level of optimization (word vector dimensions) didn't improve classification quality. Therefore, we suggest the usage of the original setting of PSO-WAWV.

Table 4.8: Comparison of the method PSO-WAWV-1000 with PSO-WAWV-dims-1000 using F1-score.

Method	Dataset			
	20nshort	Sanders	Snippets	TMN
PSO-WAWV-1000	0.549	0.990	0.914	0.773
PSO-WAWV-dims-1000	0.524	0.990	0.891	0.764

Qualitative analysis of PSO weighting

We generated word clouds arranged by class similar to those of Table 4.4, this time comparing PSO-WAWV-1000 and AWWV-1000 with some classes of the 20nshort dataset. They are shown in Table 4.9. A total of six classes out of twenty were selected, aiming to show the main differences between the methods.

Table 4.9: Word clouds of the most representative words in the 20nshort dataset by class, for PSO-WAWV-1000 and AWWV-1000 document vectors. The size of words are proportional to the their frequency in a list of documents' nearest words for each class. Colors do not encode any information.

20nshort Class	Method	
	PSO-WAWV-1000	AWV-1000
rec.motorcycles		
comp.sys.mac.hardware		

Continued on next page

Table 4.10 shows the top-20 words in the 20nshort vocabulary with higher and lower weights. The word “jerry” was the third most valued word in the whole vocabulary, what explains its introduction. Maybe this word be a good discriminator for the *rec.motorcycles* class.

Table 4.10: Top-20 and bottom-20 words in the 20nshort dataset in terms of PSO-WAWV-1000 weights.

20nshort		Top-20		Bottom-20	
Position	Word	PSO weight	Word	PSO weight	
1	<i>man</i>	7.392680	<i>quote</i>	-3.021191	
2	<i>band</i>	7.383803	<i>pick</i>	-2.582671	
3	<i>jerry</i>	7.031429	<i>machine</i>	-2.171138	
4	<i>press</i>	5.775304	<i>luck</i>	-1.751085	
5	<i>stop</i>	5.424319	<i>hst</i>	-1.410809	
6	<i>convert</i>	5.405484	<i>modem</i>	-1.284652	
7	<i>sun</i>	5.312865	<i>server</i>	-1.155327	
8	<i>listen</i>	5.292923	<i>rgb</i>	-0.766275	
9	<i>stuff</i>	5.159168	<i>read</i>	-0.678372	
10	<i>performa</i>	5.072030	<i>surface</i>	-0.548096	
11	<i>cute</i>	5.021564	<i>andrew</i>	-0.545905	
12	<i>bitmap</i>	4.993816	<i>literature</i>	-0.500627	
13	<i>compare</i>	4.902909	<i>hate</i>	-0.367920	
14	<i>dec</i>	4.889331	<i>atheists</i>	-0.220502	
15	<i>client</i>	4.864289	<i>exists</i>	-0.056248	
16	<i>widjet</i>	4.780086	<i>keyboard</i>	-0.021405	
17	<i>instructions</i>	4.768953	<i>quadra</i>	0.015236	
18	<i>bruins</i>	4.764550	<i>paul</i>	0.059694	
19	<i>price</i>	4.743247	<i>fact</i>	0.116571	
20	<i>matter</i>	4.647711	<i>advantage</i>	0.231546	

The high and low values for weights of words like the ones listed in Table 4.10 could help to explain why PSO-WAWV-1000 present competitive results in the 20nshort dataset, as well the other datasets in general. But, because finding such weights is an optimization process which involves uncertainty, it is not always possible to generate the best classification results, as reported by the Snippets dataset. We understand that the PSO-WAWV results are encouraging, despite preliminary, and deserve further investigation for enhancement, including the exploration of other optimization techniques.

Chapter 5

Methods to Expand Short Text for Topic Modeling

This chapter addresses RQ2, and proposes a framework to expand short text by creating a generalization for the pseudo-document based approaches reported in Section 3.2. The strategy is to produce larger pseudo-documents representations from the original texts that carry more information for the task of topic modeling.

One advantage of pseudo-document methods for topic modeling is that they are usually simpler than methods that modify LDA or create new probabilistic graphical models, while remaining independent of the underlying topic modeling technique. This is because only input data is transformed by these methods.

We perform two instantiations of the general framework, generating two new methods: (1) *Co-occurrence Frequency Expansion* (CoFE); and (2) *Distributed Representation Expansion* (DREx). CoFE makes use of patterns of words co-occurrence (or correlation) to expand documents. It is perhaps the most intuitive approach to expand documents, and has been previously explored by short text topic models, such as BTM, WNTM and STE.

Our second method, DREx, takes advantage of word embeddings for document expansion. Word embeddings carry the type of semantic information that allows an accurate selection of the nearest neighbors of a word in the process of document expansion.

5.1 A General Framework to Expand Short Text

Section 3.2 reviews a few pseudo-document based methods. The main problems of these methods are the following: (1) they were designed to perform well in other

text mining task than topic modeling (e.g. STE for short document clustering); (2) they degenerate the structure of documents (e.g. WNTM, which creates one pseudo-document per word); and (3) they are context-specific (e.g. LDA-# for Twitter).

We propose a general framework to expand short text that, in this case, is being used for topic modeling. At the same time, it is not a specialization of a particular topic model. In fact, it can be applied to short text and the expanded dataset can feed any topic modeling algorithm. The framework also maintains the structure of documents, meaning that the correspondence between original and expanded documents are kept. Lastly, it is not application-specific, so it can be used in any type of short text.

In our framework, the process of enriching a document with new terms goes through the identification of words in the vocabulary that are similar to components in the original text. These components are usually terms, but this is not a restriction. The proposed mathematical framework is generic because it does not impose any restriction over the definition of components and the similarity metric among them, the metric space. It allows the specification of the desired size of documents as well. A document is enriched with words that already belong to the collection vocabulary, because this does not degenerate the original vocabulary and increases the co-occurrence of similar terms, minimizing the low co-occurrence frequency problem in short text corpora.

The framework is extensible, enabling the implementation of different approaches that define the way words are added to the documents. This is possible due the formalization of a *metric space*, a set for which is defined the distance between any pair of elements. Formally, it is a pair (\mathcal{V}, g) , where \mathcal{V} is a set of components and g is a metric distance between every pair of components $v_i, v_j \in \mathcal{V}$. Different combinations of \mathcal{V} and g generate different metric spaces, hence different approaches for document expansion. The following properties have to be satisfied:

- i. The distance between every pair of components is non-negative, $g(v_i, v_j) \geq 0$.
- ii. The distance between two components is zero if, and only if, they are equal, $g(v_i, v_j) = 0 \iff v_i = v_j$.
- iii. The distance between components is symmetric, $g(v_i, v_j) = g(v_j, v_i)$.
- iv. Distances comply with the triangle inequality, $g(v_i, v_j) \leq g(v_i, v_k) + g(v_k, v_j)$.

The framework defines a generic metric space whose elements of \mathcal{V} are n-grams and g is a distance between them. Specific implementations can determine the cardinality of these n-grams, i.e. if they are unigrams, bigrams and so on. The expansion procedure is regulated by the following definitions.

Definition 5.1. Let \mathcal{V} be the set of n-grams in a corpus, $v \in \mathcal{V}$ one of these n-grams and g a distance function. We define a t -nearest neighbor function, $\mathcal{NN}(v, t)$ which calculates the t closest n-grams to v in \mathcal{V} . Formally, $\mathcal{NN}(v, t) = \mathcal{A} : |\mathcal{A}| = t \wedge \forall p \in \mathcal{A}, \forall x \in (\mathcal{V} - \mathcal{A}), g(v, p) \leq g(v, x)$. In other words, \mathcal{A} contains the t closest n-grams to v .

For each document d , the framework calculates a intermediate graph representation G_d . This representation allows to observe the closest n-grams to the whole document, as stated by the next definition.

Definition 5.2. Let $G_d = (L_d \cup R_d, E_d)$ be a bipartite graph, where $L_d \cup R_d \subseteq \mathcal{V}$, which have two types of nodes: $l \in L_d$, the n-grams extracted from d , and $r \in R_d$, the candidate n-grams to expand d among the t nearest neighbors. An edge $e_d \in E_d$, $e_d = (l, r, w)$, defines a relationship between pairs of nodes l and r , where w is the weight of e_d , equivalent to the similarity s (derived from the distance g) between the referred n-grams. Formally, $e_d = (l, r, w) : l \in d, r \in \mathcal{NN}(l, t), w = s(l, r)$ and $\mathcal{NN}(l, t) \subseteq \mathcal{V}$.

Algorithm 4 shows the general expansion procedure, where D is the corpus to be expanded, (\mathcal{V}, g) is the metric space (\mathcal{V} is the set of n-grams and g is a distance function), t is the number of neighbors words returned by function \mathcal{NN} and M is the minimum expected length of documents. The procedure expands documents shorter than M until it reaches this expected size, if possible.

Algorithm 4 General expansion framework.

Require: $D, (\mathcal{V}, g), t, M$

```

1: for  $d \in D$  do
2:   if  $|d| < M$  then
3:      $G_d \leftarrow$  Graph  $(L_d \cup R_d, E_d)$  generated from  $(\mathcal{V}, g)$  and  $t$             $\triangleright$  Def. 5.2
4:      $C_d \leftarrow \emptyset$                                                                                                 $\triangleright$  Candidate words
5:     for  $e_d = (l, r, w) \in E_d$  do
6:        $C_d \leftarrow C_d \cup \{r, w\}$ 
7:     while  $|d| < M$  do
8:        $h \leftarrow$  SelectionMethod( $C_d$ )                                                                                    $\triangleright$  Selected word
9:        $d \leftarrow d \cup h$ 

```

Algorithm 4 expands each document d in the corpus D according to the condition of line 2, that is, having less than M words. Those documents that already have this minimum expected size are kept as they are originally. For those documents that need to be expanded, the metric space (\mathcal{V}, g) and t are used to generate the similarity

graph G_d , according to definition 5.2 (line 3). The details of the generation of G_d are implementation-specific, since it depends on the precise definition of g . Lines 4-6 extract the set of candidate words C_d , which consists in collecting all nodes in R_d and their corresponding weights of edges to nodes in L_d . Lines 7-9 are responsible for iteratively adding a word h to d until it reaches the expected size M . The selection of h (line 8) is also implementation-dependent, but we recommend a stochastic selection based on the weights, because it avoids a deterministic definition of pseudo-documents.

It is important to clarify that, eventually, documents do not reach the expected size M , because the number of neighbor words t considered for each n-gram in the original document may not be enough. In a first analysis, it may seem obvious to conclude that this can be adjusted, since t is a parameter, but at least two observations related to this adjustment must be taken into account: (i) as t increases, the chance of \mathcal{NN} to return unrelated words to the original ones also increases, because the neighborhood becomes much wider; (ii) depending on the function g , calculating large neighborhoods can be computationally costly, so this is a tradeoff for some implementations.

Another way to work around the issue of documents shorter than M without adjusting t is to recursively apply Algorithm 4 to the target corpus until all documents have at least size M . Unfortunately this does not invalidate the ponderations of the last paragraph, so the probability of arising unrelated words and the computational cost will grow as more words are added to the documents. Because of that, experiments carried out with the methods described in this chapter do not consider this alternative implementation (i.e. recursive expansion of corpus), and some documents can still remain shorter than M even after the expansion procedure.

The word selection procedure (line 8) used in all implementations in this chapter is a probabilistic weight proportionate selection applied over a multinomial distribution of candidate words. This means that words with higher accumulated weights have higher probability of being selected. The probabilistic selection prevents the appearance of the same nearest n-grams in the pseudo-documents. Preliminary experiments with LDA, associated with implementations of Algorithm 4, showed to be sensitive to the selection method, exhibiting better results with the probabilistic approach.

The next sections explain in details two implementations of the proposed framework: CoFE and DREx.

5.2 Co-occurrence Frequency Expansion

The *co-occurrence frequency expansion* (CoFE) method implements the general expansion framework by defining a metric space $(\mathcal{V}_{CoFE}, g_{CoFE})$, where \mathcal{V}_{CoFE} is the vocabulary of the corpus (i.e. unigrams) and g_{CoFE} is a measure of distance between two words in \mathcal{V}_{CoFE} considering the frequency in which these words co-occur in the same context. The intuition is that semantically related words are more likely to occur in the same document.

Formally, the distance function g_{CoFE} is defined according to Equation 5.1. For a word w_k , we define O_k as the set of documents that contains w_k . The equation adapts the Jaccard index, which is a metric of similarity between sets. Words with high co-frequency (i.e. co-occurrence frequency) have high values of Jaccard index, therefore small values of distance.

$$g_{CoFE}(w_i, w_j) = 1 - Jaccard(w_i, w_j) = 1 - \frac{|O_i \cap O_j|}{|O_i \cup O_j|} \quad (5.1)$$

The formal specification of the metric space requires the definition of the distance g_{CoFE} , but the weights in the similarity graph are derived from the similarity $s_{CoFE}(w_i, w_j) = Jaccard(w_i, w_j)$. For a document d , $|d| < M$, the expansion procedure considers the graph $G_d = (L_d \cup R_d, E_d)$, where L_d are nodes that represent each word in the original document, and R_d represents each word that co-occurs at least once with some of the original words (with the limit of t expansion words for each original word). Co-occurrences are represented in the graph as edges E_d and the weights of these edges are similarity values calculated with s_{CoFE} .

Figure 5.1 shows an example of CoFE similarity graph for the text “olympic games” for $t = 4$. For a pseudo-document of size $M = 5$, three words – because the document already have two – from R_d (gold, phelps, pool, rio, silver or brazil) are probabilistically selected. For a word w , its selection probability is proportional to the sum of weights of edges of the node that represents w . For example, the word *rio* have a probability of selection proportional to $0.27 + 0.11 = 0.38$. Precisely, the actual probabilities are the normalized values of the sum of weights for each expansion word.

5.3 Distributed Representation-based Expansion

The *distributed representation-based expansion* (DREx) method uses the semantic properties of word embeddings (see Section 2.3) to build similarity graphs for documents. Word vector algebra is used to quantify the distance or similarity among n-grams. A

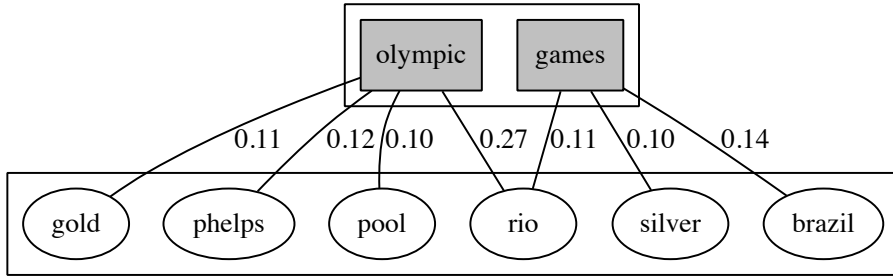


Figure 5.1: Example of similarity graph for CoFE (“olympic games”).

metric space $(\mathcal{V}_{DREx}, g_{DREx})$ is defined, where the set \mathcal{V}_{DREx} contains all vectors v , which represent words and bigrams of the corpus. We use the cosine similarity between two vectors v_i and v_j , $s_{DREx}(v_i, v_j) = \cos \theta = \cos(v_i \cdot v_j / \|v_i\| \|v_j\|)$, where θ is the angle between vectors v_i and v_j . From s_{DREx} we formalize the distance metric g_{DREx} (Zhang and Korfhage [1999]), as shown by equation 5.2.

$$g_{DREx}(v_i, v_j) = 1 - \frac{\cos^{-1} \theta}{\pi} = 1 - \frac{\cos^{-1} \theta(v_i \cdot v_j / \|v_i\| \|v_j\|)}{\pi} \quad (5.2)$$

For a document d in the collection, the expansion bipartite graph $G_d = (L_d \cup R_d, E_d)$ has L_d as the set of nodes for the vector representation of all bigrams in d , and R_d the set of nodes for the the word vectors that are closest to the bigram vectors in L_d (with the limit of t expansion words for each original word).

Word vectors have to be previously trained with a preferably large external corpora (e.g. Wikipedia), with the condition of the vocabulary being large enough to contemplate all words in the target corpus. R_d elements are directly extracted from the full word vectors set. A bigram vector b in L_d is the sum of its two constituent word vectors v_m and v_n , i.e. $b = v_m + v_n$. This calculation is coherent with results already discussed in literature (Mikolov et al. [2013b,d]) and produces a resultant vector – the bigram vector – that “merge” the semantics of both words.

Figure 5.2 shows an example of DREx similarity graph for the text “president obama visited cuba”, with $t = 3$. The bigrams of the text are: “president obama”; “obama visited”; and “visited cuba” – as illustrated in the upper partition of the graph. The expansion words (presidency, barack, reagan, visiting, havana, traveled) – in the lower partition of the graph – are the nearest words relative to the bigrams, with edges weighted by s_{DREx} .

The selection probabilities are calculated in a similar way to CoFE, i.e. normalized sum of weights for each expansion word. In the figure 5.2, the word *reagan* is the more likely to be selected ($0.67 + 0.52 = 1.19$), but the resultant pseudo-document can

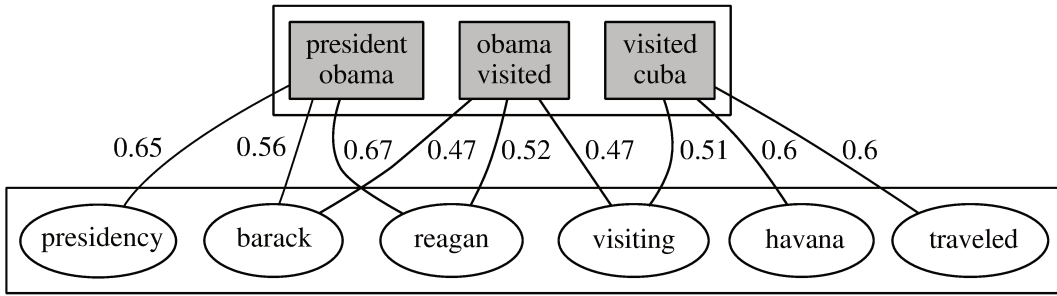


Figure 5.2: Example of similarity graph for DREx (“president obama visited cuba”).

potentially have any of the six expansion words.

5.4 Complexity Analysis of CoFE and DREx

Let N be the size of the target collection (i.e. number of documents), $|V|$ the vocabulary size, M the expected number of words in each pseudo-document after expansion, t the number of nearest neighbors for each word, and L the word vector dimensionality.

Implementations of both CoFE and DREx performs a lazy precalculation of similarities between possible pairs of components in the metric space (i.e. to the extent that components arise), generating a cache. For CoFE, this precalculation passes by all documents, each one of size $|V|$ in the worst case: $O(N|V|)$. Each word in the cache is then compared to all others using g_{CoFE} , therefore the final complexity for cache generation in CoFE is $O(N|V|^2)$.

The cache of DREx is indexed by bigrams. Each document has $|V|$ words in the worst case, so the bigram vectors are calculated with complexity $O(|V|^2)$. For each bigram vector, its distance g_{DREx} is calculated for each other word in the worst case, which is proportional to both the vocabulary size and the vectors dimensionality. The t nearest words are sorted in a partial sorting of time complexity $O(|V| + t \log t)$. The final time complexity in the worst case for the cache generation process in DREx is then $O(|V|^3L + |V|^3 + |V|^2t \log t)$.

The document expansion step in both methods has the same time complexity. For each document, t candidate words are retrieved for each word, from which at most $M-1$ expansion words are selected. This results in a time complexity of $O(NMt + NM) = O(NMt)$.

The afore calculated complexity analysis shows that the expansion procedure cost in both methods is dominated by the cache generation step. This procedure is highly dependent on $|V|$ but, in practice, the worst case is very unlikely to happen.

5.5 Experimental Evaluation

This section reports the experiments carried out with the purpose of evaluating the implementations of CoFE and DREx regarding topic discovery quality on short text. Experiments are organized in five phases:

1. Preliminary evaluation of CoFE and DREx implementations by applying LDA over expanded datasets, varying parameter M (desired length of documents after expansion) for both algorithms and different word vector models for DREx;
2. Analysis of topics and pseudo-documents generated by CoFE and DREx;
3. Evaluation of the best method obtained in the phase 1 (LDA with DREx, $M = 60$ and GloVe word vector model) with pure LDA and other pseudo-document-based topic modeling techniques;
4. Evaluation of different combinations of our selected implementation (DREx with $M = 60$ and GloVe) with LDA and the state-of-the-art techniques for short text topic modeling;
5. Evaluation of different combinations of our selected implementation (DREx with $M = 60$ and GloVe) with LDA and the state-of-the-art techniques for short text topic modeling for document classification.

Next we present the datasets used in the experiments, the preprocessing steps, the evaluation metrics and the experimental setup. Following the results of the mentioned five phases are presented and explained.

5.5.1 Datasets and text preprocessing

Seven short text datasets were used in the experiments. Four of them, namely, 20nshort, Sanders, Snippets and TMN, are described in Section 4.3.1. The other three datasets are:

- *Tweets NBA (NBA)* – A Twitter dataset collected in August 2015 about the NBA teams Golden State Warriors (hashtag #warriors) and Los Angeles Lakers (#lakers)⁶.
- *Tweets Politics (Politics)* – A Twitter dataset collected in August 2015 about the political parties Democrats (hashtag #democrats) and Republicans (#republicans)⁶.

Table 5.1: Datasets statistics.

Dataset	#Docs	Vocabulary size	#Classes	Words/doc	Unique words/doc
TMN	30376	6314	7	4.9 (± 1.5)	4.9 (± 1.5)
NBA	70707	12504	-	8.6 (± 3.0)	8.4 (± 3.0)
Politics	70712	15029	-	8.1 (± 2.6)	8.0 (± 2.5)
20Nshort	1723	964	20	8.2 (± 3.5)	7.1 (± 2.9)
Sanders	3770	1311	4	6.1 (± 2.7)	5.8 (± 2.5)
Snippets	12117	4677	8	14.3 (± 4.4)	10.3 (± 3.1)
CLEF	1001	1639	-	6.3 (± 2.8)	6.0 (± 2.6)

- *CLEF 2012 Tweet Contextualization (CLEF)* – a collection of tweets from informative accounts (e.g. CNN)¹, created for the INEX 2012 Tweet Contextualization track at CLEF conference.

Figure 5.1 shows that datasets have few words per document on average (columns “Words/doc” and “Unique words/doc”) and small standard deviation. The largest datasets are NBA and Politics ($\approx 70,000$ docs) and the smallest is 20Nshort (1,723 docs). The vocabulary size also presents a large variation (from 964 in 20Nshort to 15,029 in Politics). Only four datasets have a set of predefined classes, namely, TMN, 20Nshort, Sanders and Snippets. CLEF has no labels at all, while NBA and Politics have hashtags, which can be indirectly used to label data, but not classes.

Datasets were preprocessed according to the same steps described in Section 4.3.2.

5.5.2 Evaluation Metrics

Experiments were planned to assess both the quality of topic modeling and the power of the representation of documents using topics. In the latter case, documents are represented as vectors of probability distributions over topics and subjected to a classification task. We have selected one evaluation metric for each problem, namely, *normalized pointwise mutual information score* (NPMI-score) (Bouma [2009]) for evaluating topics and *macro-average F1 score* (Yang and Liu [1999]) for document classification.

The NPMI-score is the normalized version of the PMI-score (Newman et al. [2010]), which verifies if semantic agreement between pairs of words in a topic model is coherent with co-occurrence patterns of these words in a larger external reference dataset.

The co-occurrence frequency of words w_i and w_j in a sliding window of arbitrary size (in our case, 10 words) is the basis for the estimation of the probability $p(w_i, w_j)$. According to Equation 5.3, when w_i and w_j always occur together in the external

¹Available at: <http://inex.mmci.uni-saarland.de/data/documentcollection.html#qa>

dataset, then $\text{NPMI}(w_i, w_j) = 1$, and when they never occur together, $\text{NPMI}(w_i, w_j) = -1$. The NPMI-score for a topic is calculated with Equation 5.4, where t is the topic and W_k is the k most probable words for t , and represents the mean of all $\text{NPMI}(w_i, w_j)$ for all $w_i, w_j \in W_k, i \neq j$.

$$\text{NPMI}(w_i, w_j) = \ln \frac{p(w_i, w_j)}{p(w_i)p(w_j)} / -\ln p(w_i, w_j) \quad (5.3)$$

$$\text{NPMI-score}(t; W_k) = \sum_{i=2}^k \sum_{j=1}^{i-1} \text{NPMI}(w_i, w_j) / k(k-1) \quad (5.4)$$

The NPMI-score for a topic model is the mean of the NPMI-score for all topics. In our experiments, $k = 10$ and the external dataset was a sample of 15,000,000 documents from the WMT11 news corpus². We used the NPMI-score implementation described in Röder et al. [2015].

The NPMI metric has a strong correlation with human judgement (Lau et al. [2014]), the main reason why it was selected. Note that the PMI and NPMI, as described in Röder et al. [2015]; Lau et al. [2014]; Nguyen et al. [2015]; Li et al. [2016]; Zuo et al. [2016a], are extrinsic metrics, i.e. they use an external dataset instead of the topic model target dataset. The usage of an external dataset reinforces an independent evaluation of topics.

An important point to be clarified with regard to the NPMI between two words x and y is that they are normalized in the interval $[-1, 1]$, -1 being interpreted as no co-occurrence of x and y in the reference corpus, 0 as independence, i.e. x and y co-occur randomly in the reference corpus, and 1 as complete dependence, i.e. x and y always co-occur in the reference corpus (Bouma [2009]). If the reference corpus is the target dataset, naturally NPMI becomes an intrinsic metric and tends to present higher values, because the topic model was built over the co-occurrence patterns of the same data³. When the reference corpus is an external dataset – usually large and with a broader set of domains and topics – it is natural that, on average, NPMI values are lower, as previously reported in literature (Lau et al. [2014]; Nguyen et al. [2015]). This is the reason why most NPMI values reported in this thesis are negative and near zero. Nevertheless, in this case, NPMI values remain useful for comparative purposes.

The macro-average F1 score metric is the same described in Section 4.3.4.

²Available at: <http://www.statmt.org/wmt11/training-monolingual.tgz>.

³This is the case of Viegas et al. [2019]

5.5.3 Parameter setting

This section details the parameter settings for the methods used in the experiments. All word vectors used in the experiments have 300 dimensions and were generated using the original versions of SG, CBOW and GloVe (see Section 2.3 for details). The training dataset was a dump of the English Wikipedia from 06/02/2015, with 8,102,107 documents and a vocabulary of 2,120,659 words. The size of the context window was set to 10. SG and CBOW used both negative sampling with 5 negative examples and default initial learning rate of 0.025 and 0.05, respectively. GloVe used the default parameter values suggested by Pennington et al. [2014], i.e. $x_{\max} = 100$ and $\alpha = 0.75$.

CoFE and DREx were initially tested with M assuming values 30, 40, 50 and 60. Initially DREx used all word vectors configurations, after which assumed GloVe vectors of size 300 and $M = 60$.

Topic models LDA, LF-LDA and BTM share four parameters: (1) the number of topics k , which assumed values 20, 50 and 100; (2) the prior α and β for the Dirichlet distribution, which regulate the symmetry of topic per document and word per topic probabilities, respectively, and in our experiments were estimated in execution time using the Minka’s fixed point iteration method (Minka [2000]); and (3) the number of sampling iterations, set to 2000.

It is important to notice that defining the number of topics for a collection is still an open problem, and that is why we varied the values in a similar way as done in the literature, including the original papers of baselines (Nguyen et al. [2015]; Yan et al. [2013]; Zuo et al. [2016b]).

LF-LDA has two extra parameters, which are the word vectors and the mixture factor λ . Regarding word vectors, we used GloVe vectors of size 300 from Wikipedia. The factor λ assumed the value 0.6, as suggested in Nguyen et al. [2015].

In the document classification experiment, we used a *support vector machine* (SVM) classifier (Cortes and Vapnik [1995]) with Gaussian kernel, which had parameters γ and C tuned using a grid search procedure. Features describing the documents were derived from the posterior distribution of 20 topics per document.

Experiments for evaluation of topics were repeated 5 times. The experiments of document classification were performed with 5 replications of a 5-fold cross validation (1 fold for tuning SVM parameters through grid search, 3 folds for training the model and 1 fold for testing it), totalizing 25 repetitions. All experiments were statistically validated with the non-parametric Wilcoxon signed-rank hypothesis test over the means with 0.05 of significance level.

Table 5.2: Average results of NPMI for CoFE and DREx run with LDA (mean of different values of M).

	20 Topics	50 Topics	100 Topics	20 Topics	50 Topics	100 Topics
	NBA			Politics		
CoFE	-0.130	-0.145	-0.149	-0.099	-0.116	-0.130
DREx-CBOW	-0.073	-0.083	-0.099	-0.033	-0.064	-0.084
DREx-GloVe	-0.033	-0.023	-0.025	0.020	0.018	0.006
DREx-SG	-0.045	-0.045	-0.051	-0.006	-0.024	-0.041
	Sanders			20Nshort		
CoFE	-0.124	-0.143	-0.146	-0.191	-0.204	-0.205
DREx-CBOW	-0.019	-0.053	-0.078	-0.076	-0.107	-0.129
DREx-GloVe	0.040	0.011	-0.005	0.010	-0.029	-0.060
DREx-SG	-0.021	-0.051	-0.078	-0.091	-0.141	-0.166
	TMN			Snippets		
CoFE	-0.035	-0.067	-0.108	-0.024	-0.062	-0.083
DREx-CBOW	0.003	-0.023	-0.047	0.006	-0.018	-0.046
DREx-GloVe	0.060	0.062	0.053	0.057	0.047	0.023
DREx-SG	-0.041	-0.056	-0.076	-0.018	-0.029	-0.062
	CLEF					
CoFE	-0.138	-0.152	-0.154			
DREx-CBOW	-0.053	-0.063	-0.085			
DREx-GloVe	0.011	0.001	-0.028			
DREx-SG	-0.091	-0.101	-0.116			

5.5.4 CoFE and DREx Parameter Analysis

We compare the performance of CoFE and DREx for topic modeling in all seven datasets when combined with LDA, varying the values of parameters k (number of topics: 20, 50, 100), M (target document size after expansion: 30, 40, 50, 60) and word vector model (SG, CBOW, GloVe). Combining methods with the different values of M , k and word vector models, we reported 48 results (mean over 5 replications each). They are detailed in Appendix A.

Table 5.2 shows summarized results for each method, averaging over the values of NPMI relative to variations of M , where values in bold show the best statistically significant results. Notice that DREx-GloVe shows the best values of NPMI in all datasets, except in the NBA dataset for 20 topics, where it ties with DREx-SG. Detailed results in appendix A suggest the adoption of DREx-GloVe-60 (i.e. DREx combined with GloVe considering $M = 60$), since it was the combination with the best overall results.

The intuition behind the fact that DREx performs better than CoFE in any scenario is related to the difference in the generality of words added to documents for CoFE and DREx. While CoFE uses semantic relationship of words considering only the target dataset, DREx works in a broader domain by exploiting word vectors built

over an external general dataset like Wikipedia. Moreover, in the similarity graph of DREx, the expansion words are sorted by similarity to the sum of n-grams word vectors. In other words, DREx also uses the semantic properties of word vector operations to discover more suitable expansion words.

NPMI values decrease when k increases. This is because the number of predefined classes of the datasets is at most 20. A much higher number of topics can worsen the quality of inferred topics, because this eventually forces the allocation of documents with similar characteristics to different majoritary topic distributions.

5.5.5 Analysis of Topics and Pseudo-documents Generated by CoFE and DREx

We analyzed the distribution of the 10 most probable words for 20 topics discovered with LDA over the Snippets dataset expanded with both CoFE and DREx-GloVe-60 (as it presented the best overall results), regarding the most probable classes for the topics⁴, namely: (1) business; (2) computers; (3) culture-arts-entertainment; (4) education-science; (5) sports; (6) politics-society; (7) engineering; and (8) health.

The results are shown in Table 5.3, whose columns indicate: a sequential topic number; the most probable words of CoFE; the most probable words of DREx; the likely classes (when CoFE and DREx do not agree, two classes are shown); and the cosine similarity between CoFE and DREx topics (probability distributions over words). Topics are paired and ordered in the table according to degree of similarity. It is expected that good topics be aligned with one of the dataset classes.

The majority of topics in Table 5.3 are easily identified as belonging to specific classes. For example, topics 1, 2 and 3 are clearly about engineering, health and sports, respectively. Exceptions are topics 11 and 15, where models seem to disagree about categories.

A general observation of data in this table is that CoFE adds more context-specific words than DREx. For example, in topic 4 (about computers) we see DREx producing words like “hardware” – which is less context-specific than “cpu” or “intel” – with higher probabilities in CoFE. Under the NPMI evaluation perspective, general concepts like the ones exploited by DREx tend to produce better results.

We also analyzed the expansion words to see if the proposed expansion methods change the original meaning of documents. Particularly, this potentially happens with high values of M and very short documents (i.e. with little discriminative words), when the generated expansion graphs have edges with low weights and almost uniform

⁴According to visual inspection of words of topics and likely classes.

probabilities of selecting expansion words. In this scenario, the result is the potential generation of almost random pseudo-documents.

Table 5.3: Topics discovered by LDA for Web snippets expanded by CoFE and DREx-GloVe. Column *class* indicates the respective dataset class(es) the topic refers to. Column *Sim* indicates the cosine similarity for topics in the same row.

#	CoFE	DREx	class	sim
1	car engine electrical motor wheels electric cars gear fuel automatic	car engine cars equipment manufacturing electrical vehicle motor components	7	0.75
2	health cancer medical disease healthy nutrition information diet treatment hiv	health medical care treatment cancer patients disease patient medicine clinical	8	0.74
3	sports football games news game soccer com league team scores	sports football soccer league teams game basketball games sport team	5	0.74
4	intel computer memory chip processor device cpu cache core pentium	computer hardware computers intel software processor memory computing	2	0.72
5	political democracy party democratic social politics parties communist	political government politics party democratic election democracy	6	0.71
6	research edu science university school department graduate program students	university graduate edu faculty education college student students school harvard	4	0.70
7	news information online yahoo web directory com search sites links	information web online internet links external google search websites blog	2, 4	0.65
8	business trade services management marketing gov development international	business industry financial market companies company investment finance	1	0.65
9	movie movies film imdb awards actor video director academy tom	music movie film video movies feature best films shows released	3	0.64
10	software programming computer web data java systems linux code parallel	computer software systems application applications internet information based	2	0.61
11	wikipedia encyclopedia wiki culture history article American ancient category	wikipedia articles article wiki https pages org page encyclopedia doesn	3, 4	0.61
12	theory physics quantum philosophy theorem mathematical newton	theory theoretical analysis methods mathematical instance physics concepts	4	0.55
13	journal theoretical journals biology natural paper papers research theory evolution	research science study scientific technology studies institute development	4	0.40
14	music art rock band pop classical artists lyrics arts album	art work works gallery museum arts photo collection artist painting	3	0.40
15	amazon com books fashion online selection design book shopping manga	published book books publications journal publication literature work	4	0.35
16	system gov house government president presidential republic united congress	development public business government education economic information	6	0.27
17	war military navy force air army nuclear revolution civil weapons	culture history american world part united europe modern america first	3, 4	0.17
18	tickets tennis golf ski buy chicago grand diego maradona woods	news media coverage cnn chicago broadcast bbc interview york washington	5, 3	0.17
19	market stock finance financial exchange bank investment income quotes money	food health healthy diet nutrition calorie eating fitness eat foods	1, 8	0.05
20	network internet security wireless bandwidth test mobile speed access	education students teaching learning school learn work help experience	2, 4	0.04

Table 5.4 shows the most frequent words added to documents of categories *business* and *health* in the Snippets and TMN datasets, using DREx-GloVe and CoFE. Each word is followed by its frequency. Results weaken the hypothesis raised in the last paragraph, since methods do not seem to degenerate the original meaning of the documents, but there are clear differences between CoFE and DREx. First, in most cases DREx adds less words than CoFE, and includes the same relevant words to a greater number of documents. Second, as already confirmed in Table 5.3, CoFE adds

Table 5.4: List of words most frequently added to documents labeled as *health* and *business*. For both CoFE and DREx, the target document size M is to 60. Each word is followed by the number of times it was added to that class documents.

CoFE	DREx-GloVe	CoFE	DREx-GloVe
Snippets			
business		health	
finance (150)	business (1109)	medical (133)	health (803)
financial (136)	financial (856)	health (133)	medical (646)
services (128)	industry (809)	disease (124)	care (586)
business (118)	development (712)	care (119)	research (527)
information (114)	information (683)	nih (110)	education (518)
market (113)	investment (659)	nutrition (107)	patients (499)
research (111)	management (602)	treatment (105)	treatment (495)
online (108)	companies (586)	symptoms (105)	information (484)
money (107)	finance (573)	diseases (104)	study (464)
news (106)	based (550)	research (101)	patient (421)
TMN			
business		health	
stocks (429)	time (1380)	study (195)	health (373)
prices (392)	financial (1296)	risk (188)	patients (344)
billion (390)	business (1120)	cancer (184)	due (338)
oil (362)	investment (1026)	heart (163)	care (321)
rise (357)	increase (977)	drug (158)	treatment (300)
profit (356)	money (924)	linked (156)	disease (299)
shares (354)	market (894)	diabetes (147)	time (294)
report (347)	make (867)	drugs (147)	medical (240)
sales (346)	due (818)	disease (132)	make (238)
higher (343)	brought (679)	kids (131)	life (233)

words that are more context-specific than DREx. For example, in the *health* class of the TMN dataset, DREx produces the word “health” as the most frequent, while CoFE adds more specific terms, like names of diseases (DREx simply adds the word “disease”).

Table 5.5 shows examples of TMN documents (one for each category) and the respective expansion words selected by DREx-GloVe. We can see the overall semantic agreement between the original and expanded words.

5.5.6 Comparison of DREx with State-of-the-art methods

The previous section showed that LDA combined with DREx-GloVe ($M = 60$) presents the best results for NPMI when compared to CoFE. This result reinforces the hypoth-

Table 5.5: Examples of TMN documents expanded by DREx-GloVe. The last column shows the selected words during the document expansion step.

Class	Original text	Words added by DREx
Business	delta air lines q1 loss grows to \$318 million	flight due grown base force billion line lose grow losing jet growing service reaches aircraft lost operations makes
Entertainment	“like a rolling stone” dylan’s best song	miller bob love rolling called singer rock written songs album band wall ones recording times gave live tribute features music
Health	fda to regulate e-cigarettes as tobacco products	produce smoking restricting drug approved company increase drugs manufacturing drink marijuana food smoke reduce industry alcohol regulatory
Science & Technology	apple co-founder wozniak: computers can teach kids	company students early learn ceo computer programs allowed teaching founders dedicated read business teachers based computing studying lessons children named
Sports	nadal cruises past ljubicic into quarters	future open beginning rafael federer tennis djokovic day half cruise sharapova sets years runner roddick finals ferrer time days
US	arizona supreme court stays execution	oregon cases appeals leave states case rest court appeal justice judge ruling kansas texas state months takes oklahoma courts remain nevada united california finally

esis that word vector models, which capture complex semantics of words, are more appropriate to expand documents than a co-occurrence-based approach. In this section we compare DREx with pure LDA and other state-of-the-art document expansion methods, namely, LDA-#, WNTM and STE (see Sections 3.2.3, 3.2.2 and 3.2.1 for a description of these methods). In order to make comparisons fair, STE was adapted to contemplate the parameter M , which in any case was set to 60.

Table 5.6 compares the values of NPMI obtained for each method, varying datasets and the number of topics. Since LDA-# is specific for Twitter data with hashtags, it can only be applied to the NBA, Politics and CLEF datasets. The Sanders

Table 5.6: Results of NPMI for document expansion methods.

Topic Model	20 topics	50 topics	100 topics	20 topics	50 topics	100 topics
	NBA			Politics		
LDA - Original	-0.158	-0.156	-0.154	-0.072	-0.090	-0.095
LDA-DREx-GloVe-60	-0.037	-0.019	-0.021	0.024	0.018	0.006
LDA-#	-0.158	-0.151	-0.153	-0.124	-0.116	-0.117
WNTM	-0.135	-0.141	-0.135	-0.086	-0.089	-0.099
STE	-0.087	-0.070	-0.069	-0.043	-0.045	-0.055
Sanders			20Nshort			
LDA - Original	-0.087	-0.099	-0.116	-0.184	-0.188	-0.193
LDA-DREx-GloVe-60	0.047	0.015	-0.002	0.009	-0.029	-0.056
WNTM	-0.085	-0.113	-0.125	-0.194	-0.194	-0.198
STE	-0.156	-0.164	-0.170	-0.232	-0.241	-0.239
TMN			Snippets			
LDA - Original	-0.062	-0.056	-0.085	-0.061	-0.102	-0.106
LDA-DREx-GloVe-60	0.056	0.062	0.058	0.061	0.050	0.030
WNTM	-0.026	-0.047	-0.067	0.004	-0.034	-0.064
STE	-0.245	-0.217	-0.220	-0.115	-0.141	-0.153
CLEF						
LDA - Original	-0.137	-0.140	-0.135			
LDA-DREx-GloVe-60	0.008	0.004	-0.026			
LDA-#	-0.114	-0.125	-0.126			
WNTM	-0.149	-0.139	-0.141			
STE	-0.176	-0.178	-0.175			

dataset, despite being a collection of tweets, has no hashtag information.

DREx-GloVe-60 combined with LDA was better than other baselines in all datasets for any number of topics. LDA-# was better than pure LDA only in one out of three datasets. STE was better than LDA in three out of seven datasets and WNTM in two out of seven. This shows that state-of-the-art baselines in pseudo-document generation eventually present better results than LDA, and that DREx-GloVe represents a breakthrough over the baselines. Ultimately this is due to the robustness of the combination of word vector representations trained with an external large dataset and a general framework to document expansion.

5.5.7 Combining DREx with Topic Models for Short Text

The previous section compared DREx against other document expansion methods. In this section, we evaluate the combination of DREx with LDA and the state-of-the-art topic modeling techniques for short text, namely, LF-LDA and BTM. The objective is to understand the effect of the expansion method over the topic models, comparing the quality of topics discovered with the original text paired with the quality of topics discovered with the text expanded with DREx-GloVe-60.

Table 5.7: NPMI values for LDA, LF-LDA and BTM methods with 20 topics considering both the original and expanded versions (DREx-GloVe-60) of the dataset. Percentage of improvement of expansion are in parenthesis.

Dataset	Original	DREx-GloVe	Original	DREx-GloVe
		NBA	Politics	
LDA	-0.158	-0.037 (76.58%)	-0.072	0.024 (133.33%)
LF-LDA	-0.149	-0.014 (90.60%)	-0.059	0.027 (145.76%)
BTM	-0.168	-0.036 (78.57%)	-0.085	0.022 (125.88%)
		Sanders	20Nshort	
LDA	-0.087	0.047 (154.02%)	-0.184	0.009 (104.89%)
LF-LDA	-0.079	0.055 (169.62%)	-0.179	0.019 (110.61%)
BTM	-0.085	0.038 (144.71%)	-0.202	0.005 (102.48%)
		TMN	Snippets	
LDA	-0.062	0.056 (190.32%)	-0.061	0.061 (200.00%)
LF-LDA	-0.039	0.055 (241.03%)	-0.061	0.069 (213.11%)
BTM	-0.048	0.070 (245.83%)	-0.042	0.082 (295.24%)
		CLEF		
LDA	-0.137	0.008 (106.01%)		
LF-LDA	-0.129	0.013 (110.07%)		
BTM	-0.149	0.001 (110.07%)		

Results are shown in Table 5.7. We used 20 topics in all experiments, due the time complexity of methods and the fact that previous experiments demonstrated that NPMI is degraded with higher number of topics for the used datasets. In all cases the results show statistically significant improvement of the expanded version with DREx over the non-expanded ones. Improvements range from 76.58% (LDA on NBA dataset) to 295.24% (BTM on Snippets dataset). Despite this impressive improvement in percentage terms, these results should not be overrated due to the very low absolute values of NPMI.

The best results of DREx shows that this implementation of the expansion framework can be used to overcome the low co-occurrence frequency problem in short text without the need of a specific topic modeling technique, since it works on the text input. Notice that the best overall results were obtained with the application of LF-LDA (5 out of 7) or BTM (2 out of 7).

Table 5.8: Classification results of mean macro-average F1 score when representing documents by topics extracted from original and expanded documents. Percentage improvements of expansion are in parenthesis.

	LDA		BTM		LF-LDA	
	Original	DREx-GloVe	Original	DREx-GloVe	Original	DREx-GloVe
20Nshort	0.216	0.24 (+11.1%)	0.252	0.267 (+5.8%)	0.239	0.235 (-1.6%)
TMN	0.599	0.62 (+3.4%)	0.652	0.689 (+5.7%)	0.618	0.624 (+0.9%)
Sanders	0.842	0.901 (+6.9%)	0.88	0.924 (+4.9%)	0.852	0.899 (+5.5%)
Snippets	0.757	0.836 (+10.4%)	0.857	0.872 (+1.7%)	0.729	0.841 (+15.4%)

5.5.8 Document Classification Task

In this section we evaluate the quality of document representations extracted from LDA, LF-LDA and BTM models, both for original and expanded datasets (DREx-GloVe-60). Representations are evaluated through a document classification task, therefore only the datasets with class information are considered: 20Nshort, TMN, Sanders and Snippets. The number of classes for each of these datasets are shown in Table 5.1.

We used the posterior probability distribution of topics to create a vector representation for documents. This feature set f_i for a document d_i and k topics is defined according to Equation 5.5. In this experiment, $k = 20$ (i.e. $|f_i| = 20$). For each topic model two feature datasets were generated, the first one with the posteriors obtained from the application of the topic model on the original dataset, and the second one considering the expanded dataset.

$$f_i = [p(z_1|d_i), p(z_2|d_i), \dots, p(z_k|d_i)] \quad (5.5)$$

Table 5.8 shows the classification results of the mean macro-average F1 score. Pairs of columns “Original” and “DREx-GloVe” are compared and bold values indicate the statistically significant best result. Results suggest that, in general, DREx-GloVe generates better document representations inferred from topics models for classification. An exception to this assertion is the result for the 20Nshort dataset with LF-LDA, but the difference is not statistically significant. All other results show improvement when applying DREx-GloVe.

The results for not expanded datasets presented in Table 5.8 are consistent with the ones reported in Yan et al. [2013]. However, they point out to the relevance of DREx in improving even more the classification results through document expansion. The best overall results are obtained by BTM-DREx.

An important observation is that these results evidence the independence of the expansion method from subjacent topic modeling techniques. In fact, they reveal a

potential for consistent improvement in all topic models considered. Also, the suspicion of document meaning degeneration with the application of DREx to datasets is undermined, since the resultant expanded data contains words that induce topics that are more discriminative regarding class information.

5.6 Final Remarks

This chapter presented a general framework to expand short text for topic modeling, from which two implementations were derived, CoFE and DREx. DREx address the RQ2, showing that the usage of word vectors algebra for text expansion can be explored to produce a new state-of-the-art method for short text topic modeling.

Preliminary results for NPMI shows that DREx, when combined with GloVe and $M = 60$, presents better results than CoFE. Analysis of the most probable words per topic and expanded documents shows that the best results for DREx can be related to the addition of less context-specific words into documents, characteristic which is valued by the NPMI metric.

When compared with baselines for pseudo-document methods, DREx presented the best results for NPMI. Our proposed expansion method based on word vectors also consistently improve the results of state-of-the-art method for topic discovery in short text. This positive result is confirmed with a document classification task. The realization that DREx expansion can produce all sorts of improvement in topic modeling related tasks evidences that word vector models carry a lot of useful semantic information about words that reflects on topics quality.

Chapter 6

Vec2Graph Topic Model

This chapter presents a novel word embedding-based probabilistic topic model for short text. The fundamental structure of the proposed model is a *word network* that represents the semantic relationship between all pairs of words in a target dataset vocabulary, from which topics are inferred. The adopted strategy purposely ignore word-word co-occurrence patterns at the level of documents, taking advantage of similar patterns already represented in word vectors that are learned by algorithms such as Word2Vec, GloVe and fastText.

The word network is generated by our proposed Vec2Graph¹ algorithm, which has the main advantage of not being based on word-word patterns in short texts, admittedly underrepresented due the smallness of documents. Therefore, topics are inferred from a Vec2Graph structure that represents the entire collection, a task that is accomplished by our Vec2Graph Topic Model (VGTM) method. The advantage of not considering word patterns in documents for topic discovery brings no cost to the posterior distribution of topics per document, as they are posteriorly inferred by the strength of topics in the particular document terms.

VGTM is grounded on the hypothesis that communities (David and Jon [2010]) in the Vec2Graph structure are discriminators of topics. This reason is that, while it is natural that in social network analysis we understand the link between two “agents” (usually humans) in a graph as an indicator of social affinity, in a network of words derived from word embeddings, links are naturally interpreted as *semantic affinity*. It turns out that the discovery of high level affinity patterns among nodes in any complex network is precisely the problem of community detection. Therefore, in the VGTM method communities are groups of semantically related words, here an approximation of latent topics in the collection.

¹Vec2Graph because it creates a word graph from word vector similarity.

Among the community detection methods available in complex network analysis research field, we are particularly interested in those suitable for the discovery of *overlapping communities* (Xie et al. [2013]; Fortunato and Hric [2016]), i.e. those in which nodes can belong to more than one community. This is not a restriction, since any community detection algorithm can be plugged to VGTM. Our default implementation expects a probabilistic interpretation of topics, so it is expected that a word can have non-zero probability for more than one topic, what makes overlapping communities more interesting. Precisely, we use a non-negative matrix factorization algorithm (Wang et al. [2011]) to ensure that all words have a degree of membership to all communities (or topics), not only a binary indicator of membership.

Looking at previous works related to VGTM, Wang et al. [2016] presented a method for Twitter topic modeling using graphs that are derived from hashtags. Chen et al. [2016] also proposed a graph-based topic model for question answering. Viegas et al. [2019] proposes an approach in which topics are inferred through non-negative matrix factorization (Lee and Seung [2001]) applied to document vectors derived from word embedding-based representations (CluWords).

6.1 Vec2Graph

Vec2Graph² uses the semantics of words captured by their correspondent word vectors. It uses the cosine similarity of the angle θ_{ij} between two word vectors \vec{v}_i and \vec{v}_j , which is interpreted, to a certain extent, as an estimator of the degree of semantic similarity between words i and j (Mikolov et al. [2013b]). The Vec2Graph algorithm induces a word embedding network \mathcal{G}_C for a corpus C using this estimator.

In \mathcal{G}_C , nodes are mapped to words and edges are weighted by the degree of semantic similarity between words (i.e. cosine similarity between word vectors). Documents are not directly represented in the graph, but they can be obtained by sampling sub-graphs that correspond to words present on them.

Creation of the corpus graph

Let $\mathcal{G}_C(W, S)$ be a corpus graph, where W is the set of nodes (words of the vocabulary) and S the set of edges, where $s_{ij} \in S$ is an edge that represents the relationship between nodes i and j . The weight w_{ij} of an edge s_{ij} is the cosine similarity between

²Code available at: <https://github.com/marcelopita/vec2graph> (2019/09/23).

the correspondent word vectors \vec{v}_i and \vec{v}_j , as shown in Equation 6.1.

$$w_{ij} = \cos \theta_{ij} = \cos(\vec{v}_i \cdot \vec{v}_j / \{\|\vec{v}_i\| \cdot \|\vec{v}_j\|\}) \quad (6.1)$$

The condition for $s_{ij} \in S$ is the following:

$$s_{ij} \in S \iff w_{ij} \geq \sigma, \quad (6.2)$$

where σ is a threshold that indicates the smallest acceptable similarity between any pair of words. That is, a pair of words whose similarity is less than σ are not linked in \mathcal{G}_C . The restriction 6.2 avoids a fully connected \mathcal{G}_C and controls the clustering level of the graph, which is important for the VGTM topic model (described in section 6.2). In practice, the parameter σ is optimized for the target corpus and Vec2Graph application.

On the other hand, the restriction 6.2 can produce a disconnected graph \mathcal{G}_C (i.e. with many connected components). In this case, we relax this restriction by allowing the creation of edges with weight bellow σ , but only the necessary to make \mathcal{G}_C connected. These edges are those that work as bridges between components and with the highest weight values. Furthermore, this property ensures the existence of at least one path between every pair of words in \mathcal{G}_C .

Figure 6.1 shows an example of corpus graph \mathcal{G}_C for the Sanders dataset, which consists on tweets related to IT companies³. We used Skip-Gram word vectors of size 1,000 and $\sigma = 0.4$ ⁴.

6.2 Vec2Graph Topic Model

The Vec2Graph Topic Model (VGTM)⁵ explores the community structure of the corpus graph \mathcal{G}_C with the objective of learning a probabilistic topic model. The structure of \mathcal{G}_C provides an extended context for words in documents, what is positive for the discovery of topics, notably in short text.

As a probabilistic topic model, VGTM must provide at least the following set of posterior probability distributions: probability of topics, probability of words per topic and probability of topics per document. In the remainder of this section we explain how VGTM discovers topics in the corpus graph and how they are converted into the mentioned posterior probability distributions.

³Dataset available at: <https://github.com/marcelopita/datasets/blob/master/sanders.csv>

⁴An interactive version of this graph can be found at: https://homepages.dcc.ufmg.br/~marcelo.pita/vec2graph/corpus_graph.html

⁵Code available at: <https://github.com/marcelopita/vgtm> (2019/09/23).

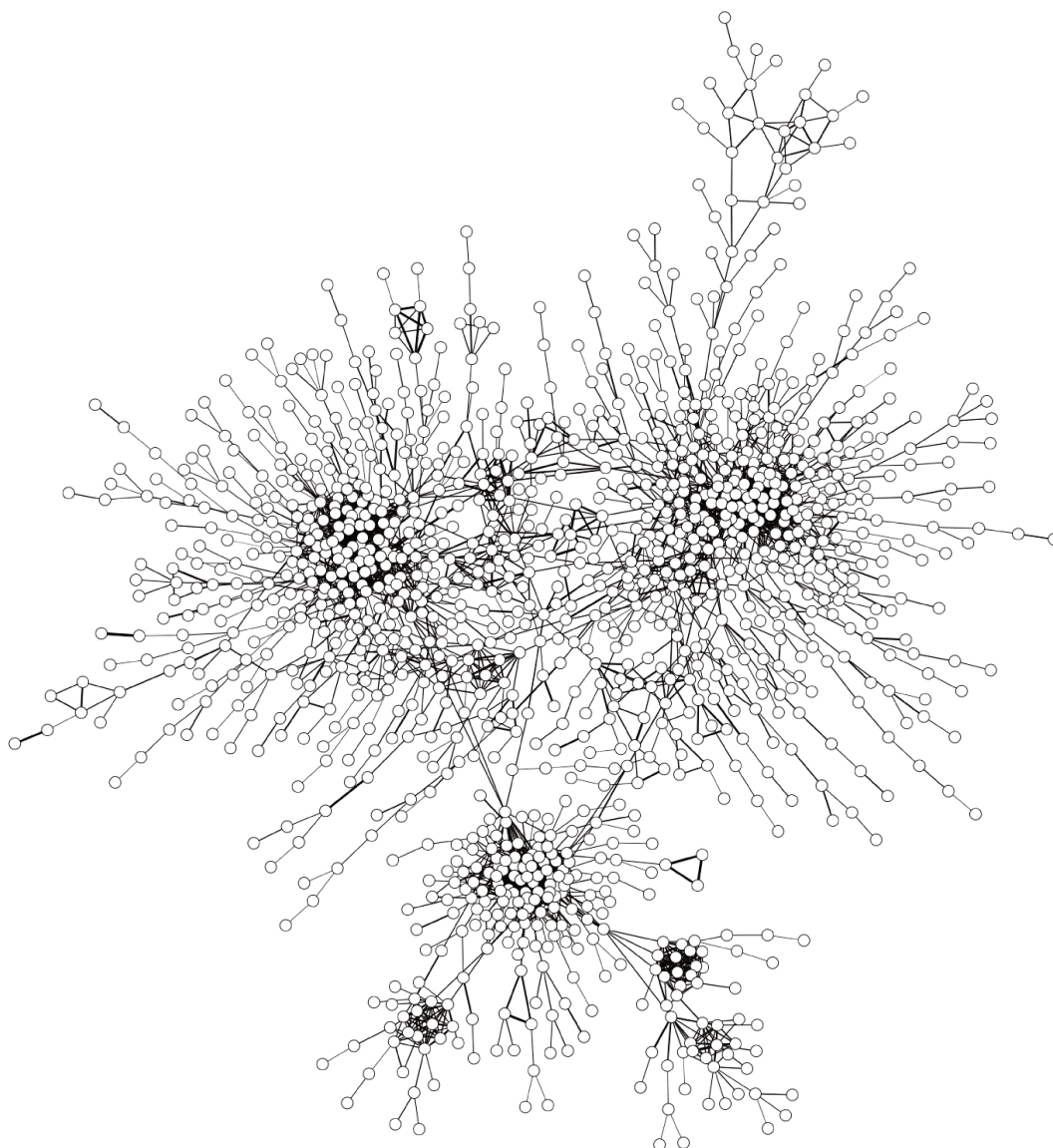


Figure 6.1: Corpus graph generated by the Vec2Graph algorithm. Dataset Sanders, Skip-Gram word vector model with 1000 dimensions, $\sigma = 0.4$.

6.2.1 Topics as communities in the corpus graph

Communities are structural patterns of networks based on node connectivity (David and Jon [2010]). They are identified as subsets of nodes in a graph that are more likely to be connected to each other than to other nodes. Communities can be classified as disjoint or overlapping. Disjoint community detectors performs hard clustering, producing disjoint sets of nodes as communities. Overlapping community techniques, on the other and, performs soft clustering, with overlapping sets of nodes as communities (Fortunato and Hric [2016]).

In the word graph \mathcal{G}_C , discovered communities can be approximated to topics. In

this perspective, and considering a probabilistic point of view, the degree of membership of a word to a community becomes a fundamental variable that defines the probability of words to topics. To achieve this soft clustering of words on the VGTM algorithm, we have to explore overlapping community techniques.

Topics are extracted from overlapping communities in \mathcal{G}_C . Naturally, each topic is represented as a probability distribution over the vocabulary. This means that every word in \mathcal{G}_C belongs to all communities, but with different degrees of membership. Consequently, VGTM demands an overlapping community detection method with full membership of nodes to communities.

The majority of overlapping community detection methods (Amelio and Pizzuti [2014]) have the peculiarity of producing communities with little overlapping. An alternative to them are methods based on Non-negative Matrix Factorization (NMF) (Zarei et al. [2009]; Psorakis et al. [2011]; Wang et al. [2011]; Zhang and Yeung [2012]). For this purpose, VGTM uses the weighted version of the SNMF (symmetric non-negative matrix factorization) proposed by Wang et al. [2011], where edges are symmetric (i.e. undirected), but weighted by similarities among the correspondent word vectors. SNMF can be replaced by any other overlapping community detection method with full membership of nodes. Note that the use of NMF as an overlapping community detector here is different from the classic use of NMF for topic modeling. In our case, \mathcal{G}_C produces a word-word adjacency matrix, while in the traditional case document-term TF-IDF matrices are usually employed.

Let $\mathbf{G} \in \mathbb{R}^{V \times V}$ be the symmetric adjacency matrix of \mathcal{G}_C , where V is the corpus vocabulary. The matrix \mathbf{G} is usually sparse, because the threshold σ reduces the number of neighbors per node. Therefore, since we have potentially many zero values in \mathbf{G} , we reduce the frequency of zero probabilities by replacing all zeros by a very small number (in our current implementation, 10^{-4}). The NMF of \mathbf{G} with rank k is the approximation $\mathbf{G} \approx \mathbf{X}\mathbf{A}$, where \mathbf{A} is the $k \times |V|$ community-word (or topic-word) affinity matrix. From \mathbf{A} , VGTM infers the following probabilities:

- The posterior probability of topics per word, $P(t|w)$;
- The posterior probability of words per topic, $P(w|t)$;
- The posterior probability of topics, $P(t)$;
- The posterior probability of topics per document, $P(t|d)$.

6.2.2 Probability of topics per word

To obtain the probability of topics per word, rows of the matrix \mathbf{A} (i.e. words) must be normalized such that $\sum_t \mathbf{A}_{wt} = 1$, i.e. the affinity of topics to a word w sums 1. In other words, each value of a row for a word w can be interpreted as the probability of w belonging to topic t .

Figure 6.2 illustrates an example with the corpus graph for the Sanders dataset⁶. In this example we use Skip-Gram word vectors of size 1000 and $\sigma = 0.4$. In VGTM we define $k = 5$ (i.e. 5 topics)⁷. Nodes in the graph of Figure 6.2 are drawn as small pie charts with colored slices indicating the proportion of topics. Note the regions dominated by specific topics (topics 2, 3 and 4), as well as the mixture of topics (topics 1 and 5). With the exception of topic 3 (Technology), all other topics in this example capture the notion of idiom (Spanish, Dutch and English).

6.2.3 Probability of words per topic

The probability of words per topic is the inferred posterior probability distribution used to characterize topics. To obtain this distribution, columns of \mathbf{A} (i.e. topics) must be normalized such that $\sum_w \mathbf{A}_{wt} = 1$, i.e. the affinity of all words to a topic t sums 1. In other words, each value of a column for a topic t can be interpreted as the probability of t being represented by the word w . Figure 6.3 shows the top-10 words (in terms of probability) for the topics 1, 2 and 3 illustrated in Figure 6.2 (topic colors remain the same).

6.2.4 Probability of topics

Topics appear in different proportions in the dataset. We infer the posterior probability of a topic t , $P(t)$, from \mathbf{A} by adding up the proportion of t in each word and normalizing this value by the proportions of all other topics in the words, according to Equation 6.3. Figure 6.4 shows the proportion of topics for the Sanders corpus graph of Figure 6.2 (topic colors remain the same).

$$P(t) = \frac{\sum_w P(t|w)}{\sum_j \sum_w P(j|w)} \quad (6.3)$$

⁶Dataset available at: <https://github.com/marcelopita/datasets/blob/master/sanders.csv>

⁷An interactive version of this graph can be found at: <https://homepages.dcc.ufmg.br/~marcelo.pita/vgtm/sanders.html>

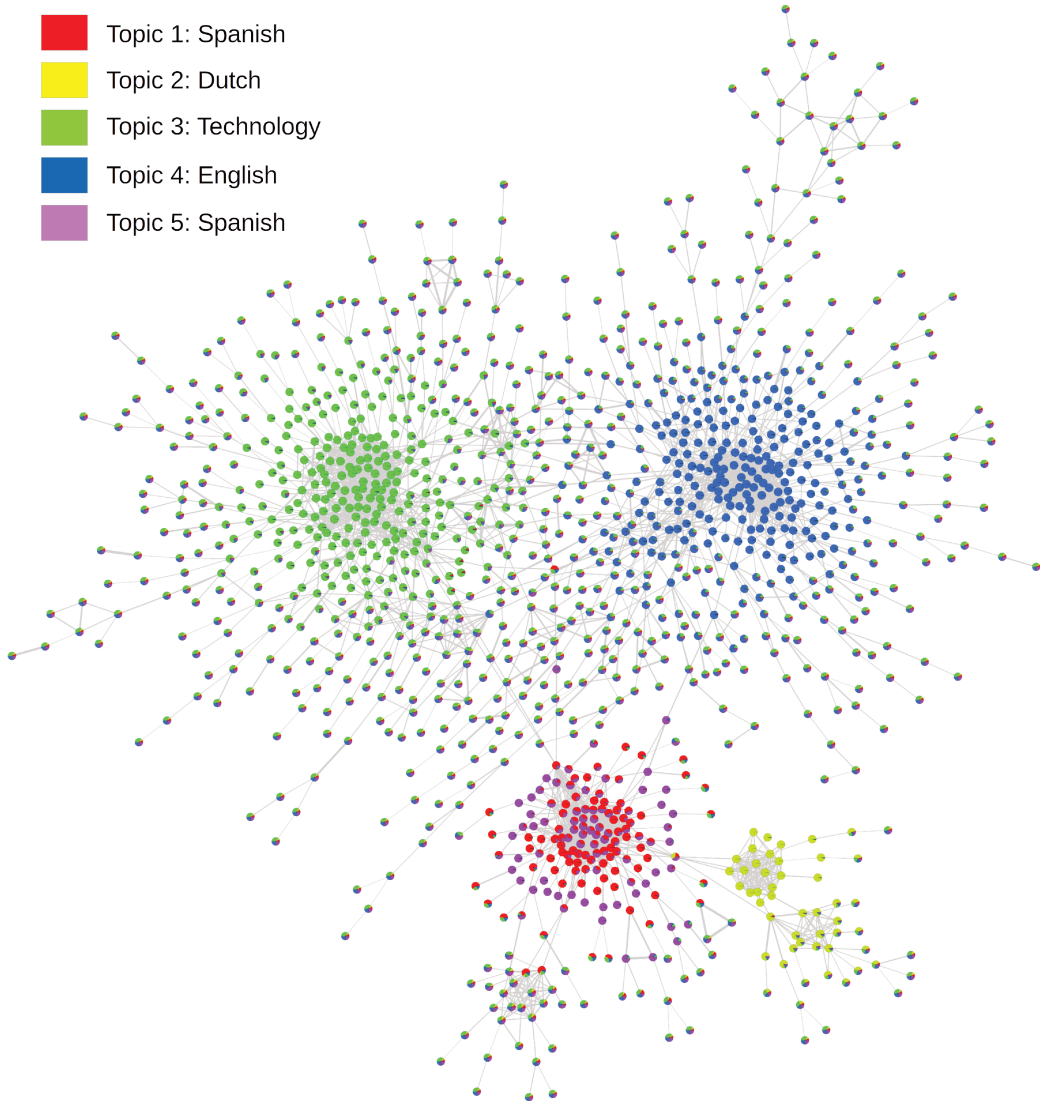


Figure 6.2: Corpus graph generated by the Vec2Graph algorithm for the Sanders dataset. Colors are mapped to topics (see labels), represented in each node as slices of a pie chart.

6.2.5 Probability of topics per document

Documents are represented as probability distribution over topics. In VGTM, the probability of a topic t for a document d , $P(t|d)$, is inferred by the rows in the topic-word affinity matrix \mathbf{A} that match the words in d , and correspondent topic column. It is the partial conditional probability shown in Equation 6.4.

$$P(t|d) = \frac{\sum_{w \in d} P(t|w)}{\sum_j \sum_{w \in d} P(j|w)} \quad (6.4)$$

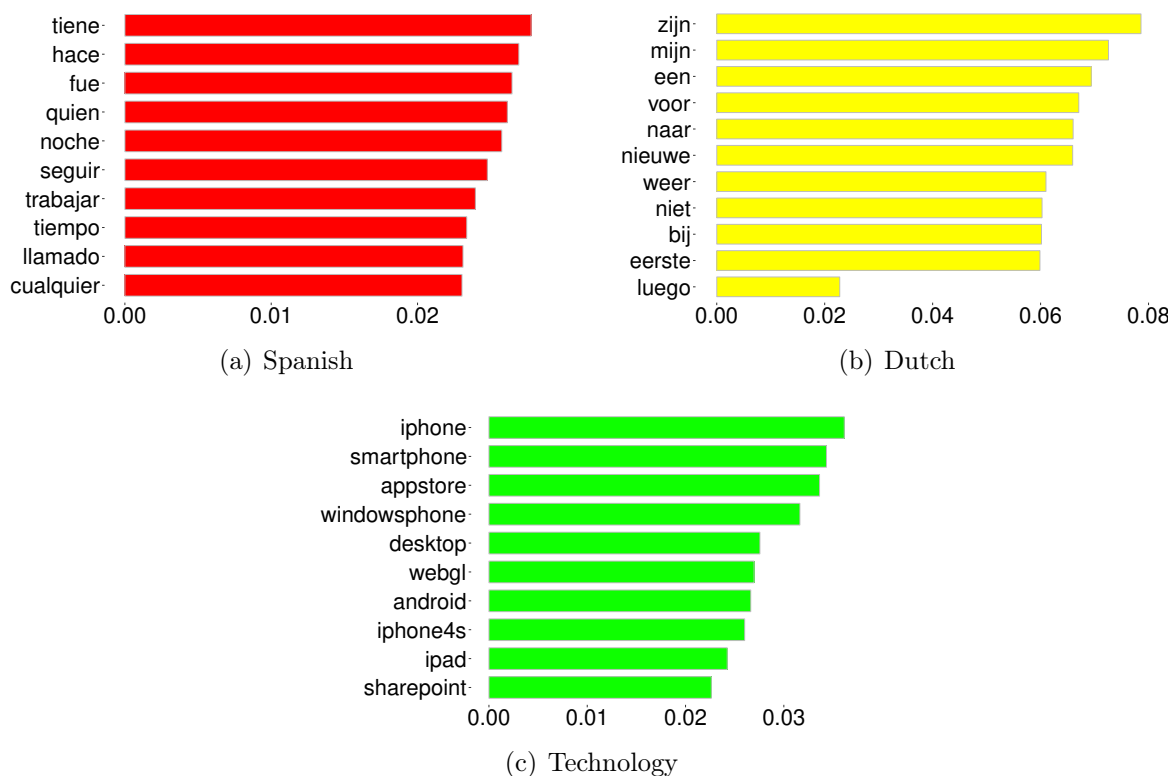


Figure 6.3: Probability distribution of top-10 words per topic (“Spanish”, “Dutch” and “Technology”) for the Sanders dataset inferred with VGTM, 5 topics, for a Vec2Graph corpus graph, $\sigma = 0.4$.

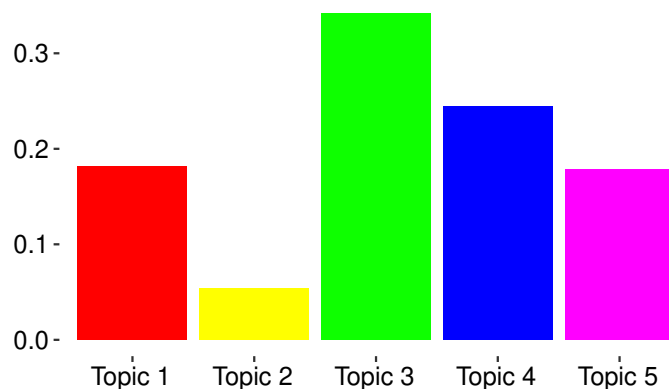


Figure 6.4: Posterior probability distribution of topics for the VGTM, Vec2Graph corpus graph with $\sigma = 0.4$, applied to the Sanders dataset.

6.3 Complexity analysis

The Vec2Graph algorithm creates a corpus graph with $|V|$ vertices, where V is the vocabulary, demanding $|V|(|V| - 1)/2$ word vector cosine distance operations, that is,

$\mathcal{O}(|V|^2)$. This implementation of VGTM algorithm internally runs NMF with rank k (number of topics) over the $|V| \times |V|$ matrix \mathcal{G}_C , for overlapping community detection. We use an implementation based on Févotte and Idier [2011], which is $\mathcal{O}(k \times |V|^2)$. The posterior probabilities are calculated over the $k \times |V|$ matrix \mathbf{A} . $P(w|t)$ and $P(t)$ are both obtained in $\mathcal{O}(k)$, while $P(t|w)$ in $\mathcal{O}(|V|)$. The probability of topics per document, $P(t|d)$, depends on the number of words per document, $|V|$ in the worst case, and k , resulting in a complexity $\mathcal{O}(|V| + k)$. The overall complexity of VGTM is $\mathcal{O}(|V|^2 + k|V|^2 + 3|k| + 2|V|) = \mathcal{O}(k|V|^2)$.

6.4 Experimental analysis

We evaluated the coherence of topics generated by the VGTM method when applied to four short text benchmark datasets, as well as two real-world applications short text datasets. We start the experimental analysis by assessing the impact of the σ parameter (Vec2Graph similarity threshold) on the results, then compare the best results found with consolidated topic models and state-of-the-art topic models for short text. Next, structural patterns of the corpus graph are analyzed to help understanding which factors actually interfere with the coherence of topics in VGTM.

The experiments reported in this chapter used the same four short text dataset introduced in Section 4.3.1. Additionally, two real-world industry short text datasets were used in the experiments: SERPRO Courses' Objectives (Courses); and SERPRO Customer Service Messages (Customers). Table 6.1 shows some statistics for the datasets, all of them with few words per document on average (column w/doc) and unique words per document on average (column unique w/doc). This information was partially introduced in Sections 4.3.1 and 5.5.1, but is repeated here for convenience. All datasets have class labels. The Courses and Customers datasets were manually labeled by experts and attendants, respectively.

Datasets were preprocessed following the same steps described in Section 4.3.2. Topic models were evaluated using the NPMI score, described in Section 5.5.2.

Table 6.1: Statistics for the short text datasets. Columns indicate (left to right): dataset identifier; number of documents; vocabulary size (number of different terms); number of classes or categories; average number of words per document; and average number of unique (distinct) words per document.

Dataset	#docs	Vocab. size	#classes	w/doc	unique w/doc
20Nshort	1723	964	20	8.2 (± 3.5)	7.1 (± 2.9)
Sanders	3770	1311	4	6.1 (± 2.7)	5.8 (± 2.5)
Snippets	12117	4677	8	14.3 (± 4.4)	10.3 (± 3.1)
TMN	30376	6314	7	4.9 (± 1.5)	4.9 (± 1.5)
Courses	1706	4791	25	17.3 (± 11.2)	16.1 (± 9.5)
Customers	17438	19679	13	30.1 (± 15.3)	26.6 (± 12.5)

6.4.1 Experimental setup

We chose as baselines the widely used topic models LDA⁸ and BTM⁹. We also considered as baselines the methods for short text DREx¹⁰, GPU-DMM¹¹, SeaNMF¹² and CluWords¹³.

For all these methods, the number of topics k assumed values 20, 50 and 100, similar to values used in original papers of the baselines (Yan et al. [2013]; Bicalho et al. [2017]). Regarding LDA parameters, the values of α and β (Dirichlet prior distributions) were optimized using the fixed point iteration method proposed by Minka [2000]. The algorithm ran for 2000 iterations. BTM also performed with 2000 iterations, with $\beta = 0.005$ and $\alpha = 50/k$, as indicated in Yan et al. [2013].

DREx, GPU-DMM, CluWords and VGTM use pre-trained word vectors. For English datasets (20nshort, Sanders, Snippets and TMN), word vectors were obtained from a English Wikipedia dump dated from 2015/06/02¹⁴, using the Skip-Gram model with 1000 dimensions, context window of size 10, negative sampling and initial learning rate of 0.025. Pre-trained Portuguese word vectors, used for the datasets Courses and Customers, were obtained from the STIL Corpora 2017 (Hartmann et al. [2017])¹⁵, using the fastText Skip-Gram model (Bojanowski et al. [2016]) with 1000 dimensions.

DREx enriches short texts using correlated words to produce longer pseudo-documents. However, it is not a topic modeling algorithm, so it must be used in

⁸Code available at: <https://github.com/gabrielmip/LDA0pt> (2019/09/23).

⁹Code available at: <https://github.com/xiaohuiyan/BTM> (2019/09/23).

¹⁰Code available at: https://github.com/marcelopita/drex_published (2019/11/06).

¹¹GPU-DMM implementation of the STTM tool (Qiang et al. [2018]). Code available at: <https://github.com/qiang2100/STTM> (2019/09/23).

¹²Code available at: <https://github.com/tshi04/SeaNMF> (2019/09/23).

¹³Code available at: <https://github.com/feliperviegas/cluwords> (2020/12/20).

¹⁴Data extracted from a XML file containing 8,102,107 articles and 2,120,659 words.

¹⁵Data extracted from 17 Brazilian and European Portuguese corpora totalizing 1,395,926,282 words. Available at: <http://www.nilc.icmc.usp.br/embeddings>.

combination with a general purpose topic model for long text. We used it with LDA, producing the LDA-DREx topic model. DREx ran with $M = 60$ (target expanded document size), according to results of the Chapter 5.

The GPU-DMM model used the default parameters proposed in Li et al. [2016], with $\alpha = 0.1$ and $\beta = 0.01$. The algorithm ran for 2000 iterations, just as in the other probabilistic graphic topic models analyzed. SeaNMF also used the default parameters proposed in the original paper (Shi et al. [2018]), with $\alpha = 1.0$ and $\beta = 0.0$, which means disabling the sparsity constraint.

The CluWords method was parameterized with $\alpha = 0.4$, the same used in Viegas et al. [2019] (fastText WikiNews word vectors), indicating that only word cosine similarities above this threshold was considered.

Experiments were repeated 5 times with a 5-fold cross validation, totalizing 25 results per experiment. Results were statistically validated with the non-parametric Wilcoxon signed-rank test with 0.05 of significance level over means.

6.4.2 Impact of the similarity threshold

We assessed the impact of the parameter σ (similarity threshold) on the quality of the topics produced by VGTM. It is important to notice that σ has a direct influence on the connectivity patterns of the Vec2Graph corpus graph \mathcal{G}_C . Since VGTM uses \mathcal{G}_C as the fundamental data structure for topics inference, different patterns can emerge from variations of σ , which in our experiments ranges from 0.2 to 0.9 in 0.1 intervals.

Figure 6.5 shows the NPMI values (Y-axis) for the VGTM method applied to the datasets with different values of σ (X-axis) and k (number of topics). We observe that the best values of σ for the datasets 20nshort, Sanders, Snippets and TMN fall in the range $[0.2, 0.4]$, with the best general results for $\sigma = 0.4$. For the Courses and Customers datasets, our choice is $\sigma = 0.7$, which presents overall higher NPMI values. The values of NPMI shown in the graphs are listed in the Appendix B.

In general, the values of NPMI for the 20nshort, Sanders, Snippets and TMN datasets increase monotonically with the values of σ until a global maximum is achieved, when the values start to decrease. However, the unstable behavior of Snippets, TMN, Courses and Customers datasets deserve a more detailed study, since it indicates that σ and k are not enough to explain the observed variations in NPMI.

For all datasets, in general, models with 20 topics present higher NPMI absolute values, which could be attributed to the proximity of this value to the number of class labels in the data: less than 25 for all datasets (see column #classes of Table 6.1).

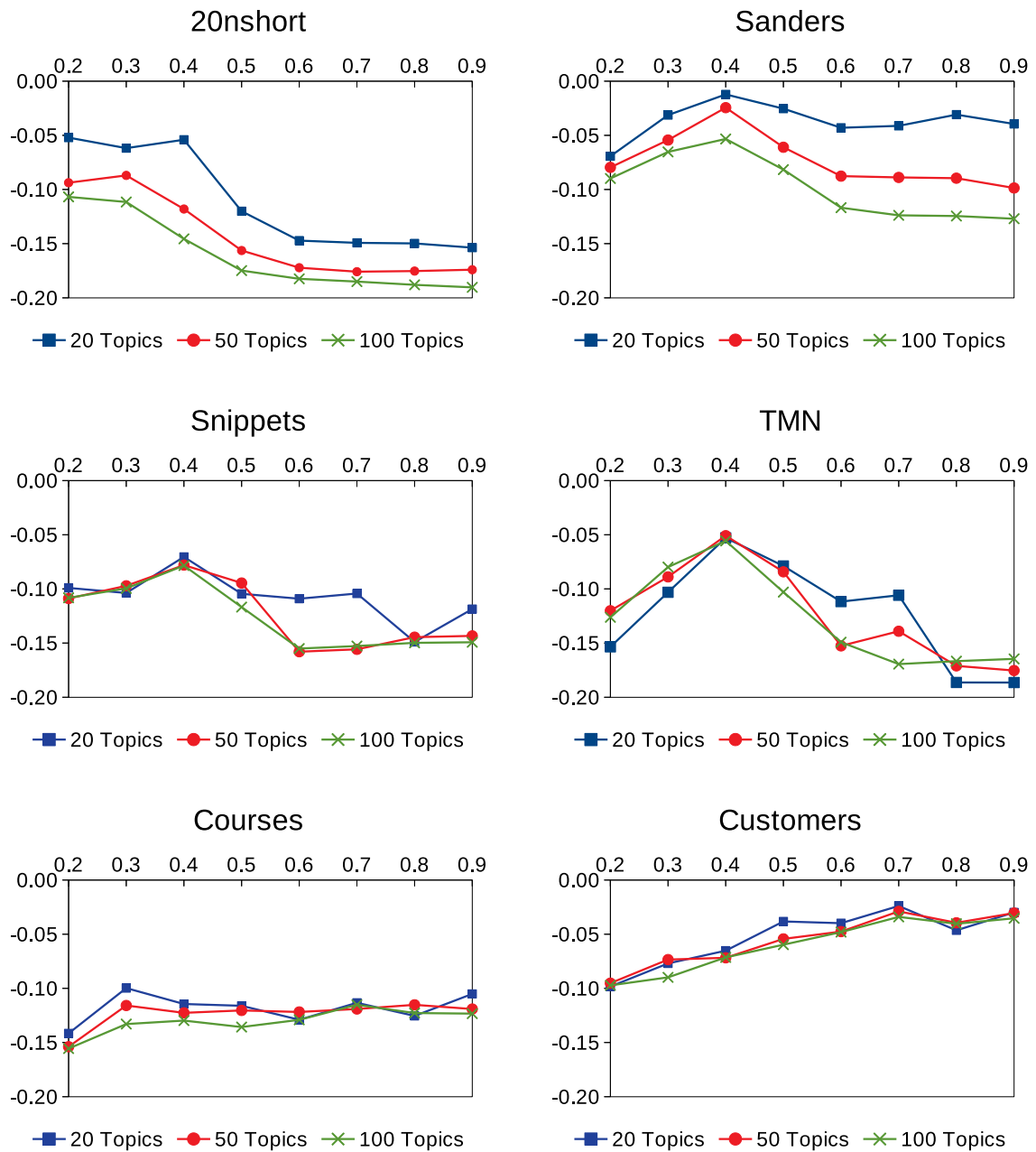


Figure 6.5: Plots of NPMI values (Y-axis) for the VGTM method with different values of σ (X-axis), considering 20, 50 and 100 topics.

6.4.3 Comparison with baselines

This section analyses the coherence of topics generated by VGTM, with $\sigma = 0.4$ and $\sigma = 0.7$, when compared with the baseline methods, namely LDA, BTM, LDA-DREx, GPU-DMM, SeaNMF and CluWords. The results are shown in Table 6.2.

In terms of the number of times a method was statistically the best method or presented no statistical difference to other methods, VGTM was the best with 11 out of 18 experiments, being statistically the best (with no ties) in 8 out of 18 experiments.

CluWords and LDA-DREx also achieved good performance, both being the best method or presenting no statistical difference to others in 7 experiments, while SeaNMF in 2 experiments. CluWords was statistically the best in 3 out of 18, while LDA-DREx in 1 experiments. LDA, BTM and GPU-DMM obtained the worse results in all cases.

For the real-world datasets Courses and Customers, VGTM with $\sigma = 0.7$ had consistently statistically significant better results than the other methods. On the other hand, CluWords was consistently the best method in the TMN dataset, with positive NPMI values for 20 and 50 topics.

The similarities between the operation of VGTM and CluWords, as well as the good results achieved by both approaches for different datasets, suggest further investigation on the situations in which each method is more appropriate, which could lead to a combined method. This is left as a future work.

6.4.4 Analysing structural patterns of the corpus graphs

The properties of the graph generated using Vec2Graph can directly influence the topic discovery task, as showed in the results in Section 6.4.2. In particular, VGTM presented an unstable behaviour as the values of σ increased for the Snippets, TMN, Courses and Customers datasets.

These results may suggest that the corpus graph induced by the similarity threshold σ presents emergent structural properties which are instrumental to achieve better NPMI results. Some of these properties and their correlation with topic coherence quality are investigated in this section.

An extensive analysis of structural network metrics was conducted on the corpus graphs produced by Vec2Graph. The study consisted in selecting the networks generated by Vec2Graph using all values of σ (similarity threshold) tested, for each dataset and number of topics. In total, 144 networks were selected (8 σ values, for 6 datasets and 3 variations of number of topics). We extracted a number of metrics for these networks, and then calculated the Pearson correlation between network characteristics and the performance of the topic modeling algorithm (measured by NPMI). It is im-

Table 6.2: Comparison of NPMI results of VGTM ($\sigma = 0.4$ and $\sigma = 0.7$) and baselines for short text datasets. Bold values indicate the best results in the column for a dataset and k .

	20 topics	50 topics	100 topics	20 topics	50 topics	100 topics
	20nshort			Sanders		
LDA	-0.188	-0.189	-0.203	-0.085	-0.113	-0.120
BTM	-0.211	-0.208	-0.205	-0.088	-0.098	-0.114
LDA-DREx	-0.045	-0.073	-0.099	-0.024	-0.047	-0.063
GPU-DMM	-0.216	-0.203	-0.204	-0.077	-0.091	-0.101
SeaNMF	-0.21	-0.227	-0.228	-0.122	-0.145	-0.143
CluWords	-0.137	-0.162	-0.184	-0.014	-0.052	-0.086
VGTM-0.4	-0.056	-0.118	-0.146	-0.012	-0.025	-0.053
VGTM-0.7	-0.150	-0.175	-0.185	-0.040	-0.089	-0.124
	Snippets			TMN		
LDA	-0.076	-0.092	-0.107	-0.061	-0.058	-0.079
BTM	-0.055	-0.083	-0.087	-0.048	-0.054	-0.064
LDA-DREx	-0.023	-0.037	-0.051	-0.061	-0.046	-0.053
GPU-DMM	-0.077	-0.09	-0.11	-0.051	-0.063	-0.073
SeaNMF	-0.006	-0.057	-0.087	-0.034	-0.075	-0.111
CluWords	-0.045	-0.042	-0.058	0.006	0.002	-0.037
VGTM-0.4	-0.072	-0.078	-0.079	-0.052	-0.051	-0.055
VGTM-0.7	-0.103	-0.155	-0.153	-0.108	-0.139	-0.169
	Courses			Customers		
LDA	-0.258	-0.243	-0.236	-0.164	-0.161	-0.159
BTM	-0.248	-0.242	-0.241	-0.151	-0.151	-0.145
LDA-DREx	-0.200	-0.185	-0.181	-0.150	-0.146	-0.144
GPU-DMM	-0.233	-0.238	-0.238	-0.151	-0.148	-0.136
SeaNMF	-0.183	-0.175	-0.189	-0.099	-0.080	-0.077
CluWords	-0.159	-0.167	-0.177	-0.252	-0.264	-0.257
VGTM-0.4	-0.114	-0.123	-0.130	-0.065	-0.072	-0.071
VGTM-0.7	-0.113	-0.119	-0.115	-0.024	-0.029	-0.034

portant to note that this correlation study was done after the networks were already formed, and not with the goal of optimizing the networks to improve topic modeling performance.

This analysis found interesting relationships between the network metrics of degree assortativity and transitivity and NPMI values. Figure 6.6 presents the values for the Pearson correlations, where the color indicates if the correlation is negative (closer to red) or positive (closer to blue). The ellipses represent the format of the plot (scatterplot) between every pair of variables. All correlations shown are statistically significant, with a significance level (p -value) under 0.01.

Degree Assortativity: It measures the preference for nodes that have similar degrees to attach to each other (Newman [2002, 2003]). The values assumed by this metric,



Figure 6.6: Network metrics which achieved significant correlation values with NPMI.

referred by r , lie in the range $-1 \leq r \leq 1$. A high value for degree assortativity ($r \approx 1$) indicates that nodes that have a high degree are very likely to be connected with other high degree nodes, meaning that the network is composed of a core of interconnected high degree nodes, and a periphery of lower degree nodes. A low value for this metric ($r \approx -1$, also viewed as a high value for disassortativity) indicates that high degree nodes are more likely to be connected with low degree nodes, which means that star-like structures are more likely to appear.

In the context of this work, a high value for degree assortativity means that words in the corpus graph form tightly woven communities, inside which words are highly similar between themselves (therefore connected). Our analysis showed that *low σ (similarity threshold) values produce networks with high degree assortativity*, which makes sense because when σ is low, the networks tend to be denser (closer to a complete graph). We also observed that *high assortativity values correlate to high NPMI values: $r = 0.41$, $p\text{-value} = 5.83 \times 10^{-5}$* . We hypothesize that this correlation happens due to the fact that a closely knit community of words in the graph semantically corresponds to words associated with the same topic.

Transitivity: The most commonly used definition for graph transitivity is the fraction of closed triads over all possible triads in a graph (Newman [2018]). A *triad* in a

graph $G = (V, E)$ is composed by three vertices $\{u, v, w\} \in V$ connected pairwise, *i.e.* $\{(u, v), (v, w)\} \in E$. A triad is considered closed (triangle) if there exists an edge between the vertices on its ends $(u, w) \in E$. The intuition for interpreting transitivity comes from social networks studies, when it is interesting to know the fraction of friendships that began due to a close acquaintance (the idea is that “friends of friends are also friends”).

In our corpus graph, transitivity means the fraction of similar words w_1 and w_2 that have another similar word w_3 as a neighbor. It gives a general idea of how dense the network is, and how tightly woven are the communities of similar words.

Our analysis showed that *the similarity threshold is strongly negatively correlated to transitivity* ($r = -0.86$, with $p\text{-value} < 2 \times 10^{-16}$), which is plausible because the higher σ is, the more strict the connectivity constraints are (two words, or nodes, will only share an edge if their cosine similarity is very high). As σ increases, the graph tends to have less edges, and is more prone to being disconnected. However, *transitivity correlates positively with NPMI* ($r = 0.41$, with $p\text{-value} = 0.0046$), which makes sense due to the fact that a high transitivity value for the graph indicates that words connect to other similar words very easily, therefore forming groups more easily. In these highly connected networks, it is easier for the VGTM algorithm to find communities of words that represent a coherent topic, resulting in a high NPMI value. Both correlations are statistically significant, with $p\text{-values}$ under 0.01.

6.5 Final remarks

VGTM uses the highly informative representation of Vec2Graph to infer topics as overlapping communities in the corpus graph, addressing the lack of context information available in short text. The results obtained by VGTM in terms of NPMI were compared to five other state-of-the-art methods for short text topic modeling.

VGTM obtained the best overall results among the compared methods, and they were particularly expressive for the real-world application datasets.

Note that Vec2Graph is general enough to be not only used in topic modeling for short text, but also in other tasks that involve language, *e.g.* text summarization and query expansion.

The good results of VGTM and CluWords on different datasets deserve further investigation on how this factor affect their performance, especially because both methods have a very similar architecture (*e.g.* both are based on word embedding similarities above a threshold). This is left as a future work that could lead to a combined approach

that merge the best characteristics of both models.

The analyses of structural patterns in the corpus graphs showed that there is no direct relevant correlation between the σ threshold and NPMI. Otherwise, we observe a significant positive correlation of NPMI with both degree assortativity and transitivity. In particular, transitivity has a significant negative correlation with σ , indicating that variations on the threshold do affect NPMI only indirectly. These observations are consistent with the unstable behavior observed in some results in Section 6.4.2, suggesting a complex relationship between NPMI and σ .

Chapter 7

Conclusions and Future Work

This thesis introduced novel methods that explore alternative representations for short text based on word embeddings in scenarios of text analysis and mining. The information provided by word embeddings is explored by the proposed algorithms with the objective of mitigate the low word co-occurrence frequency problem in collections of short texts. This problem makes learning methods suffer with the absence of a minimal context to capture semantics. The proposed methods focus on two groups of applications, namely, short text classification and short text topic modeling.

The first contribution of our work was to propose a definition of short text. The most common definition found in the literature is based on the sparsity of the corpus' document-term matrix. However, this definition does not make clear how sparse this matrix has to be for characterizing a document collection as short text, as both short and long texts present highly sparse document-term matrices despite significant differences in machine learning performance. In Chapter 2 we presented a study that grounds our particular definition. In qualitative terms, we defined short text as *the minimum amount of text needed to express a single thought*, connecting it to the concept of a paragraph. Quantitatively, we analyzed statistics regarding the number of sentences and the number of words in sentences extracted from real-world text datasets (Wikipedia and Twitter). We propose a mathematical definition of *smallness* of a text, parameterized by a length threshold. We concluded that *short texts are documents shorter than 51 words*, a threshold that was used in most datasets of this thesis. Datasets were characterized according to this definition.

The research questions (RQ) stated in Chapter 1 guided the development of the proposed methods. Next we detail the contributions, conclusions and future work in each of these RQ.

7.1 Strategies for combining word embeddings for short text classification

RQ1: Which are the most appropriate strategies for combining word embeddings to produce representations for short text classification?

Chapter 4 reported an investigation on strategies for creating word embedding-based representations that improve short text classification. We propose PSO-WAWV, a method that learns sub-optimal weights for words in documents – represented as weighted average of words vectors – for document classification.

We analyzed simple word embedding arrangements, including the sum of word vectors (SWV), average of word vectors (AWV) and TF-IDF-weighted versions of SWV and AWV. Among these arrangements, AWV presented the best classification results (F1 score).

PSO-WAWV was compared to TF-IDF, PV (paragraph vector), BOHW (bag of hyperwords) and AWV. TF-IDF is a strong baseline, but the method demonstrated to be competitive with AWV and TF-IDF. PSO-WAWV has the advantage of producing more compact instead of lesser representations than TF-IDF. We verified if a second level of optimization, namely weighting the dimensions of word embeddings, produced better results than the original PSO-WAWV, but results presented no statistical difference. Qualitative results with word clouds for the 20nshort dataset also give some examples that help understanding which words are overestimated and underestimated by the heuristic search.

Among directions of future work in RQ1 are:

- Test the proposed methods with other word vector models, such as CBOW and GloVe;
- Evaluate methods with tasks other than document classification, such as clustering and query expansion;
- Verify whether the produced document vectors have dimensions that are discriminative regarding topics.

7.2 Word embedding-based short text expansion

RQ2: Can we enhance short text topic modeling with word embedding-based text expansion?

A major contribution in RQ2 was the general framework for expansion of short text, described in Chapter 5. It allows the implementation of different expansion mechanisms through specific definitions of metric spaces.

As an implementation of the expansion framework, we proposed CoFE, a method based on a metric space whose elements are the words of the corpus vocabulary and the distance function is an adaptation of the Jaccard index, which in turn captures the notion of co-occurrence frequency between pairs of words.

Although CoFE is not our best proposed method in the standard NPMI metric, preliminary published experiments (Pedrosa et al. [2016]) show that CoFE achieved state-of-the-art results using the coherence metric (Mimno et al. [2011]). This metric is similar to NPMI, except that it is based on the co-occurrence of the most probable words of topics in the analyzed dataset (i.e. not an external dataset, as NPMI). Since CoFE is precisely defined by the co-occurrence patterns of words in the dataset, it is expected to produce good results of coherence. Still in Pedrosa et al. [2016], the distribution of topics per document, derived from the application of LDA over datasets expanded with CoFE, are used as feature vectors for short document clustering, indicating positive results for CoFE.

These preliminary results were not extensively documented in this thesis, because the proposed DREx method presented better results in the standard NPMI metric. This leads us to the contributions related to the RQ2.

The DREx method, another implementation of the general expansion framework, uses the semantic properties of word vectors to define a metric space whose elements are bigrams in documents and the distance function is the cosine distance between word vectors. It presupposes a large external set of word vectors that contains the collection terms.

We performed evaluations on topic modeling and document classification for a variety of short text datasets, and compared it with both methods that generate pseudo-documents and methods for short-text topic modeling.

When compared with the pseudo-document generation methods LDA-#, WNTM and STE, results showed the superiority of DREx. When compared with short text topic modeling methods, namely pure LDA, BTM and LF-LDA, applied over the original datasets and the expanded ones, improvement percentages of NPMI varied from 76.58% to 295.24%. Large but less impressive improvements were also obtained when using the distribution of topics per document as predictors for the short text classification

task.

Future works regarding RQ2 include:

- Exploration of other metric spaces for the framework;
- In DREx, we use the sum of the word embeddings in bigrams from the original text to find the most similar expansion words. Other ways of combining word embeddings for n-grams can be explored.

7.3 Probabilistic short text topic model based on word embedding networks

RQ3: Can we develop a competitive probabilistic short text topic model with word embedding-based representations?

Chapter 6 presented a new representation for document corpora named Vec2Graph, which captures patterns of semantic similarity between words in a corpus using the cosine similarity of word vectors. This new representation was exploited to create a graph-based probabilistic topic model for short text, named Vec2Graph Topic Model (VGTM). VGTM infers topics as overlapping communities in Vec2Graph structures. Each word in the corpus graph has different degrees of membership to topics, from which all the posterior probability distributions are derived.

The Vec2Graph algorithm has a σ threshold parameter that regulates the minimum similarity between pairs of words in the graph. We analyzed the impact of a range of σ values over the VGTM quality (NPMI) on 4 benchmark datasets (20nshort, Sanders, Snippets and TMN) and 2 real-world application datasets (Courses and Customers). VGTM was compared with 6 baselines: LDA, BTM, DREx, GPU-DMM, SeaNMF and CluWords.

VGTM was statistically the best method or presented no statistical difference in 11 out of 18 experiments, being the best with no ties in 8 out of 18 experiments. In this very experiment, CluWords and DREx shared the second best results. CluWords was statistically better than the other methods in 3 out of 18 cases, and competitive in 7 out of 18. VGTM was the only winner in the real world datasets, while CluWords the only winner in the TMN dataset.

The irregular behavior of NPMI value for different values of σ suggested an exploratory study of the structural properties of the corpus graphs generated. We analyzed the Pearson correlation between the threshold σ , the degree assortativity and

the transitivity of graphs with the NPMI values obtained. Results showed that there is no direct relevant correlation between σ and NPMI, but a positive correlation between NPMI and assortativity and transitivity. Transitivity has a negative correlation with σ , meaning that variations on σ affect NPMI indirectly.

Directions of future work in RQ3 are the following:

- Use other word vector models, such as GloVe [Pennington et al., 2014], to generate the graph weights;
- Associate VGTM with an optimization mechanism over the Vec2Graph algorithm to find corpus graphs that optimize structural metrics that are highly correlated with topic coherence, such as the ones studied in this paper, i.e., transitivity and degree assortativity;
- Investigate other overlapping community detection methods than non-negative matrix factorization;
- Investigate VGTM regarding document representation quality ($P(t|d)$) using other tasks, such as document classification and clustering.
- Investigate the factors that affect positively the quality of VGTM and CluWords methods on different datasets, leading to a combined approach that merge the best characteristics of both models.

7.4 Publications

The contributions of this thesis generated the following publications:

- Regarding contributions on the RQ1:

Pita, M., & Pappa, G. L. (2018, October). Strategies for Short Text Representation in the Word Vector Space. In 2018 7th Brazilian Conference on Intelligent Systems (BRACIS) (pp. 266-271). IEEE.

- Regarding contributions on the RQ2:

Pedrosa, G., Pita, M., Bicalho, P., Lacerda, A., & Pappa, G. L. (2016, October). Topic modeling for short texts with co-occurrence frequency-based expansion. In 2016 5th Brazilian Conference on Intelligent Systems (BRACIS) (pp. 277-282). IEEE.

Bicalho, P., Pita, M., Pedrosa, G., Lacerda, A., & Pappa, G. L. (2017). A general framework to expand short text for topic modeling. *Information Sciences*, 393, 66-81.

- Regarding contributions on the RQ3:

Pita, M., Nunes, M., & Pappa, G. L. (2019). Probabilistic topic modeling for short text based on word embedding networks. *Information Sciences*. (SUBMITTED)

Appendix A

Detailed results for CoFE and DREx

Here we present detailed results of NPMI for CoFE and DREx combined with LDA. Experiments probe the number of topics (20,50 and 100), the desired document length after expansion M (30, 40, 50 and 60) and, for DREx, three different word vector models (CBOW, GloVe and SG). Tables A.1 to A.7 show complete results of NPMI for all datasets, where the name of the expansion method is followed by its parameters. Results in bold are the best in the column (Friedman statistical test). Notice that DREx-GloVe-60 was the best in 12 of 21 tests, among 16 expansion methods.

Table A.1: Detailed NPMI results for CoFE and DREx in the NBA dataset.

NBA			
	20 Topics	50 Topics	100 Topics
Expansion method	NPMI	NPMI	NPMI
CoFE-30	-0.130(0.005)	-0.146(0.004)	-0.151(0.003)
CoFE-40	-0.130(0.005)	-0.148(0.003)	-0.150(0.005)
CoFE-50	-0.134(0.010)	-0.149(0.004)	-0.152(0.005)
CoFE-60	-0.127(0.007)	-0.138(0.006)	-0.143(0.005)
DREx-CBOW-30	-0.081(0.011)	-0.082(0.003)	-0.099(0.006)
DREx-CBOW-40	-0.076(0.009)	-0.082(0.007)	-0.096(0.005)
DREx-CBOW-50	-0.065(0.011)	-0.083(0.006)	-0.099(0.004)
DREx-CBOW-60	-0.068(0.010)	-0.083(0.005)	-0.101(0.003)
DREx-GloVe-30	-0.044(0.004)	-0.027(0.005)	-0.033(0.005)
DREx-GloVe-40	-0.028(0.011)	-0.024(0.006)	-0.023(0.002)
DREx-GloVe-50	-0.024(0.012)	-0.023(0.007)	-0.023(0.006)
DREx-GloVe-60	-0.037(0.009)	-0.019(0.007)	-0.021(0.002)
DREx-SG-30	-0.071(0.007)	-0.055(0.003)	-0.059(0.004)
DREx-SG-40	-0.038(0.009)	-0.050(0.004)	-0.053(0.005)
DREx-SG-50	-0.040(0.007)	-0.039(0.006)	-0.046(0.004)
DREx-SG-60	-0.031(0.010)	-0.038(0.005)	-0.047(0.006)

Table A.2: Detailed NPMI results for CoFE and DREx in the Politics dataset.

Politics			
	20 Topics	50 Topics	100 Topics
Expansion method	NPMI	NPMI	NPMI
CoFE-30	-0.103(0.004)	-0.115(0.004)	-0.125(0.007)
CoFE-40	-0.104(0.002)	-0.119(0.010)	-0.133(0.005)
CoFE-50	-0.092(0.006)	-0.117(0.004)	-0.131(0.001)
CoFE-60	-0.097(0.008)	-0.112(0.007)	-0.130(0.002)
DREx-CBOW-30	-0.025(0.011)	-0.060(0.007)	-0.081(0.006)
DREx-CBOW-40	-0.030(0.006)	-0.061(0.004)	-0.083(0.005)
DREx-CBOW-50	-0.033(0.011)	-0.068(0.008)	-0.086(0.005)
DREx-CBOW-60	-0.043(0.009)	-0.068(0.008)	-0.086(0.002)
DREx-GloVe-30	0.008(0.007)	0.016(0.006)	0.007(0.004)
DREx-GloVe-40	0.024(0.006)	0.020(0.006)	0.008(0.004)
DREx-GloVe-50	0.025(0.007)	0.017(0.004)	0.004(0.002)
DREx-GloVe-60	0.024(0.007)	0.018(0.002)	0.006(0.005)
DREx-SG-30	-0.013(0.010)	-0.025(0.005)	-0.048(0.005)
DREx-SG-40	-0.001(0.010)	-0.024(0.008)	-0.040(0.004)
DREx-SG-50	-0.003(0.006)	-0.022(0.010)	-0.039(0.003)
DREx-SG-60	-0.006(0.008)	-0.025(0.010)	-0.037(0.005)

Table A.3: Detailed NPMI results for CoFE and DREx in the Sanders dataset.

Sanders			
	20 Topics	50 Topics	100 Topics
Expansion method	NPMI	NPMI	NPMI
CoFE-30	-0.118(0.012)	-0.135(0.004)	-0.133(0.006)
CoFE-40	-0.124(0.010)	-0.145(0.007)	-0.143(0.004)
CoFE-50	-0.127(0.009)	-0.139(0.010)	-0.149(0.003)
CoFE-60	-0.128(0.008)	-0.153(0.006)	-0.158(0.002)
DREx-CBOW-30	-0.025(0.011)	-0.052(0.006)	-0.075(0.006)
DREx-CBOW-40	-0.016(0.016)	-0.055(0.006)	-0.076(0.004)
DREx-CBOW-50	-0.016(0.013)	-0.055(0.007)	-0.081(0.005)
DREx-CBOW-60	-0.020(0.006)	-0.049(0.008)	-0.081(0.001)
DREx-GloVe-30	0.029(0.011)	0.010(0.002)	-0.010(0.003)
DREx-GloVe-40	0.043(0.008)	0.008(0.004)	-0.008(0.004)
DREx-GloVe-50	0.043(0.005)	0.010(0.008)	-0.001(0.002)
DREx-GloVe-60	0.047(0.005)	0.015(0.005)	-0.002(0.004)
DREx-SG-30	-0.020(0.005)	-0.051(0.006)	-0.078(0.004)
DREx-SG-40	-0.023(0.016)	-0.051(0.010)	-0.080(0.005)
DREx-SG-50	-0.017(0.007)	-0.050(0.005)	-0.077(0.004)
DREx-SG-60	-0.025(0.003)	-0.053(0.005)	-0.078(0.005)

Table A.4: Detailed NPMI results for CoFE and DREx in the 20Nshort dataset.

20Nshort			
	20 Topics	50 Topics	100 Topics
Expansion method	NPMI	NPMI	NPMI
CoFE-30	-0.185(0.008)	-0.204(0.006)	-0.208(0.002)
CoFE-40	-0.193(0.009)	-0.209(0.004)	-0.211(0.004)
CoFE-50	-0.194(0.008)	-0.203(0.006)	-0.202(0.005)
CoFE-60	-0.194(0.013)	-0.199(0.004)	-0.200(0.002)
DREx-CBOW-30	-0.075(0.008)	-0.109(0.005)	-0.131(0.003)
DREx-CBOW-40	-0.082(0.004)	-0.109(0.006)	-0.128(0.005)
DREx-CBOW-50	-0.068(0.004)	-0.102(0.004)	-0.131(0.005)
DREx-CBOW-60	-0.078(0.011)	-0.109(0.003)	-0.127(0.003)
DREx-GloVe-30	0.008(0.006)	-0.031(0.007)	-0.063(0.005)
DREx-GloVe-40	0.015(0.007)	-0.024(0.004)	-0.061(0.006)
DREx-GloVe-50	0.009(0.003)	-0.031(0.006)	-0.058(0.004)
DREx-GloVe-60	0.009(0.011)	-0.029(0.007)	-0.056(0.004)
DREx-SG-30	-0.085(0.011)	-0.141(0.004)	-0.167(0.002)
DREx-SG-40	-0.093(0.008)	-0.143(0.007)	-0.169(0.004)
DREx-SG-50	-0.091(0.010)	-0.139(0.006)	-0.161(0.002)
DREx-SG-60	-0.094(0.003)	-0.140(0.005)	-0.165(0.003)

Table A.5: Detailed NPMI results for CoFE and DREx in the TMN dataset.

TMN			
	20 Topics	50 Topics	100 Topics
Expansion method	NPMI	NPMI	NPMI
CoFE-30	-0.025(0.010)	-0.069(0.009)	-0.108(0.008)
CoFE-40	-0.042(0.010)	-0.070(0.008)	-0.108(0.010)
CoFE-50	-0.031(0.012)	-0.061(0.012)	-0.110(0.004)
CoFE-60	-0.040(0.015)	-0.067(0.009)	-0.107(0.006)
DREx-CBOW-30	0.003(0.015)	-0.019(0.012)	-0.038(0.007)
DREx-CBOW-40	0.002(0.015)	-0.024(0.002)	-0.044(0.004)
DREx-CBOW-50	0.000(0.014)	-0.033(0.011)	-0.057(0.005)
DREx-CBOW-60	0.006(0.014)	-0.018(0.008)	-0.051(0.006)
DREx-GloVe-30	0.062(0.007)	0.061(0.005)	0.052(0.003)
DREx-GloVe-40	0.062(0.003)	0.059(0.002)	0.052(0.005)
DREx-GloVe-50	0.061(0.008)	0.063(0.009)	0.049(0.001)
DREx-GloVe-60	0.056(0.011)	0.062(0.005)	0.058(0.004)
DREx-SG-30	-0.046(0.010)	-0.057(0.004)	-0.081(0.001)
DREx-SG-40	-0.036(0.009)	-0.064(0.008)	-0.071(0.006)
DREx-SG-50	-0.051(0.016)	-0.052(0.009)	-0.074(0.011)
DREx-SG-60	-0.034(0.010)	-0.052(0.010)	-0.078(0.002)

Table A.6: Detailed NPMI results for CoFE and DREx in the Snippets dataset.

Snippets			
	20 Topics	50 Topics	100 Topics
Expansion method	NPMI	NPMI	NPMI
CoFE-30	-0.025(0.011)	-0.072(0.008)	-0.092(0.005)
CoFE-40	-0.032(0.009)	-0.066(0.010)	-0.085(0.006)
CoFE-50	-0.018(0.008)	-0.058(0.004)	-0.076(0.004)
CoFE-60	-0.024(0.012)	-0.054(0.004)	-0.079(0.006)
DREx-CBOW-30	0.010(0.007)	-0.013(0.005)	-0.043(0.007)
DREx-CBOW-40	0.001(0.016)	-0.019(0.006)	-0.045(0.004)
DREx-CBOW-50	0.006(0.016)	-0.012(0.003)	-0.049(0.007)
DREx-CBOW-60	0.007(0.015)	-0.027(0.005)	-0.049(0.005)
DREx-GloVe-30	0.053(0.005)	0.034(0.004)	0.007(0.007)
DREx-GloVe-40	0.056(0.012)	0.052(0.005)	0.023(0.004)
DREx-GloVe-50	0.057(0.006)	0.050(0.004)	0.033(0.005)
DREx-GloVe-60	0.061(0.006)	0.050(0.007)	0.030(0.003)
DREx-SG-30	-0.018(0.014)	-0.027(0.009)	-0.059(0.007)
DREx-SG-40	-0.015(0.016)	-0.034(0.008)	-0.062(0.007)
DREx-SG-50	-0.020(0.005)	-0.026(0.004)	-0.064(0.005)
DREx-SG-60	-0.018(0.009)	-0.030(0.011)	-0.063(0.003)

Table A.7: Detailed NPMI results for CoFE and DREx in the CLEF dataset.

CLEF			
	20 Topics	50 Topics	100 Topics
Expansion method	NPMI	NPMI	NPMI
CoFE-30	-0.140(0.000)	-0.141(0.005)	-0.146(0.000)
CoFE-40	-0.148(0.004)	-0.150(0.000)	-0.156(0.001)
CoFE-50	-0.134(0.009)	-0.162(0.001)	-0.160(0.002)
CoFE-60	-0.131(0.005)	-0.155(0.000)	-0.150(0.001)
DREx-CBOW-30	-0.043(0.008)	-0.061(0.002)	-0.085(0.002)
DREx-CBOW-40	-0.052(0.004)	-0.062(0.002)	-0.081(0.005)
DREx-CBOW-50	-0.065(0.014)	-0.060(0.002)	-0.085(0.003)
DREx-CBOW-60	-0.052(0.009)	-0.068(0.011)	-0.086(0.005)
DREx-GloVe-30	0.003(0.005)	-0.003(0.003)	-0.030(0.003)
DREx-GloVe-40	0.016(0.006)	0.001(0.003)	-0.028(0.005)
DREx-GloVe-50	0.016(0.002)	0.002(0.004)	-0.028(0.003)
DREx-GloVe-60	0.008(0.005)	0.004(0.007)	-0.026(0.003)
DREx-SG-30	-0.096(0.006)	-0.104(0.004)	-0.114(0.003)
DREx-SG-40	-0.084(0.006)	-0.104(0.008)	-0.118(0.002)
DREx-SG-50	-0.090(0.009)	-0.096(0.005)	-0.115(0.004)
DREx-SG-60	-0.092(0.012)	-0.099(0.005)	-0.115(0.006)

Appendix B

Detailed results for VGTM

Table B.1: NPMI results of VGTM for short text datasets, varying σ (0.2 to 0.9) and k . Bold values indicate the best results in the column for a dataset and k value.

	20 topics	50 topics	100 topics	20 topics	50 topics	100 topics
	20nshort			Sanders		
VGTM-0.2	-0.051	-0.093	-0.107	-0.068	-0.079	-0.089
VGTM-0.3	-0.061	-0.087	-0.111	-0.032	-0.053	-0.065
VGTM-0.4	-0.056	-0.118	-0.146	-0.012	-0.025	-0.053
VGTM-0.5	-0.118	-0.157	-0.175	-0.025	-0.061	-0.082
VGTM-0.6	-0.146	-0.172	-0.183	-0.045	-0.088	-0.117
VGTM-0.7	-0.150	-0.175	-0.185	-0.040	-0.089	-0.124
VGTM-0.8	-0.148	-0.175	-0.188	-0.030	-0.089	-0.125
VGTM-0.9	-0.152	-0.174	-0.190	-0.040	-0.099	-0.127
	Snippets			TMN		
VGTM-0.2	-0.099	-0.109	-0.108	-0.151	-0.120	-0.126
VGTM-0.3	-0.103	-0.097	-0.099	-0.103	-0.088	-0.079
VGTM-0.4	-0.072	-0.078	-0.079	-0.052	-0.051	-0.055
VGTM-0.5	-0.103	-0.095	-0.117	-0.079	-0.084	-0.103
VGTM-0.6	-0.108	-0.158	-0.155	-0.114	-0.153	-0.150
VGTM-0.7	-0.103	-0.155	-0.153	-0.108	-0.139	-0.169
VGTM-0.8	-0.150	-0.145	-0.150	-0.186	-0.171	-0.166
VGTM-0.9	-0.120	-0.144	-0.150	-0.188	-0.176	-0.164
	Courses			Customers		
VGTM-0.2	-0.142	-0.154	-0.156	-0.098	-0.095	-0.097
VGTM-0.3	-0.100	-0.116	-0.133	-0.077	-0.073	-0.090
VGTM-0.4	-0.114	-0.123	-0.130	-0.065	-0.072	-0.071
VGTM-0.5	-0.116	-0.120	-0.136	-0.038	-0.054	-0.060
VGTM-0.6	-0.129	-0.122	-0.129	-0.040	-0.048	-0.048
VGTM-0.7	-0.113	-0.119	-0.115	-0.024	-0.029	-0.034
VGTM-0.8	-0.125	-0.115	-0.123	-0.046	-0.039	-0.040
VGTM-0.9	-0.105	-0.119	-0.123	-0.030	-0.030	-0.035

Table B.1 shows the NPMI results for the VGTM method applied to the datasets 20nshort, Sanders, Snippets, TMN, Courses and Customers. It compares different values of σ (arranged along the columns) and k (number of topics). Bold font indicates the best values. We observe that the best values of σ for the datasets 20nshort, Sanders, Snippets and TMN fall in the range $[0.2, 0.4]$, with the best general results for $\sigma = 0.4$. For the Courses and Customers datasets, our choice is $\sigma = 0.7$, which presents overall higher NPMI values.

Bibliography

- Akmajian, A., Demer, R., Farmer, A., and Harnish, R. (2001). *Linguistics: An Introduction to Language and Communication*. CogNet. ISBN 9780262511230.
- Alghamdi, R. and Alfalqi, K. (2015). A survey of topic modeling in text mining. *International Journal of Advanced Computer Science and Applications*, 6(1):147--153.
- Amelio, A. and Pizzuti, C. (2014). Overlapping community discovery methods: a survey. In *Social Networks: Analysis and Case Studies*, pages 105--125. Springer.
- Beal, M. J. (2003). *Variational algorithms for approximate Bayesian inference*. University of London London.
- Beel, J., Gipp, B., Langer, S., and Breitingner, C. (2016). paper recommender systems: a literature survey. *International Journal on Digital Libraries*, 17(4):305--338.
- Bellegarda, J. R. (1997). A latent semantic analysis framework for large-span language modeling. In *EUROSPEECH*.
- Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137--1155.
- Bicalho, P., Pita, M., Pedrosa, G., Lacerda, A., and Pappa, G. L. (2017). A general framework to expand short text for topic modeling. *Information Sciences*, 393:66--81.
- Blei, D., Ng, A., and Jordan, M. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993--1022.
- Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2016). Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.
- Bouma, G. (2009). Normalized (pointwise) mutual information in collocation extraction. *GSCL*, pages 31--40.

- Burkholder, T. J. and van Antwerp, K. W. (2013). Practical limits on muscle synergy identification by non-negative matrix factorization in systems with mechanical constraints. *Medical & biological engineering & computing*, 51(1-2):187--196.
- Carpineto, C. and Romano, G. (2012). A survey of automatic query expansion in information retrieval. *ACM Computing Surveys (CSUR)*, 44(1):1.
- Chen, L., Jose, J. M., Yu, H., Yuan, F., and Zhang, D. (2016). A semantic graph based topic model for question retrieval in community question answering. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 287--296. ACM.
- Chen, Y., Zhang, H., Liu, R., Ye, Z., and Lin, J. (2019). Experimental explorations on short text topic mining between lda and nmf based schemes. *Knowledge-Based Systems*, 163:1--13.
- Chen, Z. and Liu, B. (2014). Topic modeling using topics from many domains, lifelong learning and big data. In *International conference on machine learning*, pages 703--711.
- Clerc, M. and Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE transactions on Evolutionary Computation*, 6(1):58--73.
- Collobert, R. and Weston, J. (2008). A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *ICML*.
- Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273--297.
- Das, R., Zaheer, M., and Dyer, C. (2015). Gaussian lda for topic models with word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 795--804.
- David, E. and Jon, K. (2010). *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, New York, NY, USA. ISBN 0521195330, 9780521195331.
- Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391.

- Eberhart, R. C. and Shi, Y. (2000). Comparing inertia weights and constriction factors in particle swarm optimization. In *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, volume 1, pages 84--88. IEEE.
- Faruqui, M. and Dyer, C. (2014). Improving Vector Space Word Representations Using Multilingual Correlation. In *EACL*.
- Faruqui, M., Tsvetkov, Y., Yogatama, D., Dyer, C., and Smith, N. A. (2015). Sparse Overcomplete Word Vector Representations. In *ACL*.
- Févotte, C. and Idier, J. (2011). Algorithms for nonnegative matrix factorization with the β -divergence. *Neural computation*, 23(9):2421--2456.
- Fortunato, S. and Hric, D. (2016). Community detection in networks: A user guide. *Physics reports*, 659:1--44.
- Gao, L., Zhou, S., and Guan, J. (2015). Effectively classifying short texts by structured sparse representation with dictionary filtering. *Information Sciences*, 323:130 – 142. ISSN 0020-0255.
- Gao, W., Peng, M., Wang, H., Zhang, Y., Xie, Q., and Tian, G. (2019). Incorporating word embeddings into topic modeling of short text. *Knowledge and Information Systems*, 61(2):1123--1145.
- Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, (6):721--741.
- Grammarly Blog (2019). How long –and how many sentences– is a paragraph? <https://www.grammarly.com/blog/how-long-is-a-paragraph/>.
- Griffiths, T. L. and Steyvers, M. (2004). Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228--5235.
- Hartmann, N. S., Fonseca, E. R., Shulby, C. D., Treviso, M. V., Rodrigues, J. S., and Aluísio, S. M. (2017). Portuguese word embeddings: Evaluating on word analogies and natural language tasks. In *Anais do XI Simpósio Brasileiro de Tecnologia da Informação e da Linguagem Humana*, pages 122--131, Porto Alegre, RS, Brasil. SBC.
- Hofmann, T. (1999a). Probabilistic latent semantic analysis. *Uncertainty in Artificial Intelligence*, pages 289--296.

- Hofmann, T. (1999b). Probabilistic latent semantic indexing. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 50--57. ACM.
- Hong, L. and Davison, B. D. (2010). Empirical study of topic modeling in twitter. In *Proc. of the first Workshop on Social Media Analytics*, pages 80--88. ACM.
- Houle, M. E., Kriegel, H.-P., Kröger, P., Schubert, E., and Zimek, A. (2010). Can shared-neighbor distances defeat the curse of dimensionality? In *International Conference on Scientific and Statistical Database Management*, pages 482--500. Springer.
- Huang, E. H., Socher, R., Manning, C. D., and Ng, A. Y. (2012). Improving Word Representations via Global Context and Multiple Word Prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 873--882, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Kennedy, J. and Eberhart, R. C. (1995). Particle swarm optimization. In *Proceedings of the 1995 IEEE International Conference on Neural Networks*, volume 4, pages 1942--1948, Perth, Australia, IEEE Service Center, Piscataway, NJ.
- Kishore Kumar, N. and Schneider, J. (2017). Literature survey on low rank approximation of matrices. *Linear and Multilinear Algebra*, 65(11):2212--2244.
- Lau, J. H., Newman, D., and Baldwin, T. (2014). Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 530--539.
- Le, Q. and Mikolov, T. (2014). Distributed Representations of Sentences and Documents. In *ICML*, pages 1188--1196.
- Lebret, R. and Collobert, R. (2013). Word Embeddings through Hellinger PCA. Technical report, Idiap Research Institute.
- LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278--2324.
- Lee, D. D. and Seung, H. S. (2001). Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556--562.

- Levy, O. and Goldberg, Y. (2014a). Linguistic Regularities in Sparse and Explicit Word Representations. In *CONLL*.
- Levy, O. and Goldberg, Y. (2014b). Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177--2185.
- Li, C., Wang, H., Zhang, Z., Sun, A., and Ma, Z. (2016). Topic modeling for short texts with auxiliary word embeddings. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 165--174. ACM.
- Li, X., Zhang, A., Li, C., Guo, L., Wang, W., and Ouyang, J. (2019). Relational biterm topic model: Short-text topic modeling using word embeddings. *The Computer Journal*, 62(3):359--372.
- Lu, Y., Mei, Q., and Zhai, C. (2011). Investigating task performance of probabilistic topic models: an empirical study of plsa and lda. *Information Retrieval*, 14(2):178--203. ISSN 1573-7659.
- Lunsford, A. and O'Brien, A. (2008). *The st. martin's handbook: Instructor's notes*. new york: Bedford/st.
- Luong, M.-T., Socher, R., and Manning, C. D. (2013). Better Word Representations with Recursive Neural Networks for Morphology. In *CoNLL*, Sofia, Bulgaria.
- Mahmoud, H. (2008). *Pólya urn models*. Chapman and Hall/CRC.
- Manning, C. D., Raghavan, P., Schütze, H., et al. (2008). *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge.
- Mehrotra, R., Sanner, S., Buntine, W., and Xie, L. (2013). Improving lda topic models for microblogs via tweet pooling and automatic labeling. In *SIGIR*, pages 889--892.
- Miao, Y., Grefenstette, E., and Blunsom, P. (2017). Discovering discrete latent topics with neural variational inference. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2410--2419. JMLR. org.
- Mikolov, T. (2014). Confirming my understanding of word2vec's algorithm / machine learning pipeline. https://groups.google.com/d/msg/word2vec-toolkit/fwc8-bT0g3Y/3jWkMwQg0_QJ. (Word2vec-toolbox Mailing List).
- Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*, pages 1--9.

- Mikolov, T., Corrado, G., Chen, K., and Dean, J. (2013b). Efficient Estimation of Word Representations in Vector Space. In *ICLR*, pages 1--12.
- Mikolov, T., Le, Q. V., and Sutskever, I. (2013c). Exploiting Similarities among Languages for Machine Translation. *CoRR*, abs/1309.4168.
- Mikolov, T., tau Yih, W., and Zweig, G. (2013d). Linguistic Regularities in Continuous Space Word Representations. In *PNAACL-HLT*.
- Mimno, D., Wallach, H. M., Talley, E., Leenders, M., and McCallum, A. (2011). Optimizing semantic coherence in topic models. In *EMNLP*, pages 262--272.
- Minka, T. (2000). Estimating a dirichlet distribution.
- Minka, T. and Lafferty, J. (2002). Expectation-propagation for the generative aspect model. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pages 352--359. Morgan Kaufmann Publishers Inc.
- Mnih, A. and Hinton, G. E. (2009). A scalable hierarchical distributed language model. In *Advances in neural information processing systems*, pages 1081--1088.
- Morin, F. and Bengio, Y. (2005). Hierarchical probabilistic neural network language model. In *Aistats*, volume 5, pages 246--252.
- Newman, D., Noh, Y., Talley, E., Karimi, S., and Baldwin, T. (2010). Evaluating topic models for digital libraries. In *JCDL*, pages 215--224.
- Newman, M. (2018). *Networks*. Oxford university press.
- Newman, M. E. (2002). Assortative mixing in networks. *Physical review letters*, 89(20):208701.
- Newman, M. E. (2003). Mixing patterns in networks. *Physical Review E*, 67(2):026126.
- Nguyen, D. Q., Billingsley, R., Du, L., and Johnson, M. (2015). Improving topic models with latent feature word representations. *Transactions of the Association for Computational Linguistics*, 3:299--313.
- Pedrosa, G., Pita, M., Bicalho, P., Lacerda, A., and Pappa, G. L. (2016). Topic modeling for short texts with co-occurrence frequency-based expansion. In *Intelligent Systems (BRACIS), 2016 5th Brazilian Conference on*, pages 277--282. IEEE.

- Peirce, C., Hartshorne, C., and Weiss, P. (1932). *Collected Papers of Charles Sanders Peirce*. Collected Papers of Charles Sanders Peirce. Belknap Press of Harvard University Press. ISBN 9780674138001.
- Pekar, V., Krkoska, M., and Staab, S. (2004). Feature weighting for co-occurrence-based classification of words. In *Proceedings of the 20th international conference on Computational Linguistics*, page 799. Association for Computational Linguistics.
- Pennington, J., Socher, R., and Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. In *EMNLP*, pages 1532--1543.
- Phan, X.-H., Nguyen, L.-M., and Horiguchi, S. (2008). Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th international conference on World Wide Web*, pages 91--100. ACM.
- Pinto, D., Jiménez-Salazar, H., and Rosso, P. (2006). Clustering abstracts of scientific texts using the transition point technique. In *CICLing*, volume 3878, pages 536--546. Springer.
- Pinto, D., Rosso, P., and Jiménez-Salazar, H. (2011). A self-enriching methodology for clustering narrow domain short texts. *The Computer Journal*, 54(7):1148--1165.
- Psorakis, I., Roberts, S., Ebden, M., and Sheldon, B. (2011). Overlapping community detection using bayesian non-negative matrix factorization. *Physical Review E*, 83(6):066114.
- Qiang, J., Chen, P., Wang, T., and Wu, X. (2017). Topic modeling over short texts by incorporating word embeddings. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 363--374. Springer.
- Qiang, J., Li, Y., Yuan, Y., Liu, W., and Wu, X. (2018). Sttm: A tool for short text topic modeling. *arXiv preprint arXiv:1808.02215*.
- Quan, X., Kit, C., Ge, Y., and Pan, S. J. (2015). Short and sparse text topic modeling via self-aggregation. In *AAAI*, pages 2270--2276.
- Rafeeqe, P. and Sendhilkumar, S. (2011). A survey on short text analysis in web. In *Advanced Computing (ICoAC), 2011 Third International Conference on*, pages 365--371. IEEE.
- Röder, M., Both, A., and Hinneburg, A. (2015). Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*, pages 399--408. ACM.

- Rodrigues, V. S. (2018). Exploiting semantic similarity for improved text representation. Master's thesis, Universidade Federal de Minas Gerais, Belo Horizonte, Brasil.
- Rosso, P., Errecalde, M., and Pinto, D. (2013). Analysis of short texts on the web: introduction to special issue. *Language Resources and Evaluation*, 47(1):123--126.
- Salton, G., Wong, A., and Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613--620.
- Schmidt, M. N., Larsen, J., and Hsiao, F.-T. (2007). Wind noise reduction using non-negative sparse coding. In *Machine Learning for Signal Processing, 2007 IEEE Workshop on*, pages 431--436. IEEE.
- Schwab, K. (2017). *The Fourth Industrial Revolution*. Penguin Books, Limited. ISBN 9780241300756.
- Schütze, H. (1993). Word space. In *Advances in Neural Information Processing Systems 5*, pages 895--902. Morgan Kaufmann.
- Shi, B., Lam, W., Jameel, S., Schockaert, S., and Lai, K. P. (2017). Jointly learning word embeddings and latent topics. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 375--384.
- Shi, T., Kang, K., Choo, J., and Reddy, C. K. (2018). Short-text topic modeling via non-negative matrix factorization enriched with local word-context correlations. In *Proceedings of the 2018 World Wide Web Conference*, pages 1105--1114. International World Wide Web Conferences Steering Committee.
- Socher, R., Bauer, J., Manning, C. D., and Ng, A. Y. (2013a). Parsing With Compositional Vector Grammars. In *ACL*.
- Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C., Ng, A., and Potts, C. (2013b). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *EMNLP*.
- Sra, S. and Dhillon, I. S. (2006). Generalized nonnegative matrix approximations with bregman divergences. In Weiss, Y., Schölkopf, P. B., and Platt, J. C., editors, *Advances in Neural Information Processing Systems 18*, pages 283--290. MIT Press.
- Tandon, R. and Sra, S. (2010). Sparse nonnegative matrix approximation: new formulations and algorithms. *Rapport technique*, 193:38--42.

- Tang, J., Meng, Z., Nguyen, X., Mei, Q., and Zhang, M. (2014). Understanding the limiting factors of topic modeling via posterior contraction analysis. In *ICML*, pages 190--198.
- The Writing Center, University of North Carolina at Chapel Hill (2019). Paragraphs. <https://writingcenter.unc.edu/tips-and-tools/paragraphs/>.
- Thrun, S. (1998). Lifelong learning algorithms. In *Learning to learn*, pages 181--209. Springer.
- Turian, J., Ratinov, L., and Bengio, Y. (2010). Word Representations: A Simple and General Method for Semi-supervised Learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL '10*, pages 384--394, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Turney, P. D. and Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141--188.
- Viegas, F., Canuto, S., Gomes, C., Luiz, W., Rosa, T., Ribas, S., Rocha, L., and Gonçalves, M. A. (2019). Cluwords: exploiting semantic word clustering representation for enhanced topic modeling. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 753--761.
- Vitale, D., Ferragina, P., and Scaiella, U. (2012). Classification of short texts by deploying topical annotations. In *Advances in Information Retrieval*. Springer.
- Wang, F., Li, T., Wang, X., Zhu, S., and Ding, C. (2011). Community discovery using nonnegative matrix factorization. *Data Mining and Knowledge Discovery*, 22(3):493--521.
- Wang, Q., Cao, Z., Xu, J., and Li, H. (2012). Group matrix factorization for scalable topic modeling. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 375--384. ACM.
- Wang, Y., Liu, J., Huang, Y., and Feng, X. (2016). Using hashtag graph-based topic model to connect semantically-related words without co-occurrence in microblogs. *IEEE Transactions on Knowledge and Data Engineering*, 28(7):1919--1933.
- Wenyin, L., Quan, X., Feng, M., and Qiu, B. (2010). A short text modeling method combining semantic and statistical information. *Information Sciences*, 180(20):4031 -- 4041. ISSN 0020-0255.

- Wold, S., Esbensen, K., and Geladi, P. (1987). Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37--52.
- Xie, J., Kelley, S., and Szymanski, B. K. (2013). Overlapping community detection in networks: The state-of-the-art and comparative study. *Acm computing surveys (csur)*, 45(4):43.
- Xie, P., Yang, D., and Xing, E. (2015). Incorporating word correlation knowledge into topic modeling. In *Proceedings of the 2015 conference of the north American chapter of the association for computational linguistics: human language technologies*, pages 725--734.
- Xu, W., Liu, X., and Gong, Y. (2003). Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 267--273. ACM.
- Yan, X., Guo, J., Lan, Y., and Cheng, X. (2013). A biterm topic model for short texts. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1445--1456. ACM.
- Yang, Y. and Liu, X. (1999). A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42--49. ACM.
- Yin, J. and Wang, J. (2014). A dirichlet multinomial mixture model-based approach for short text clustering. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 233--242. ACM.
- Yogatama, D., Faruqui, M., , Dyer, C., and Smith, N. A. (2015). Learning Word Representations with Hierarchical Sparse Coding. In *ICML*.
- Zaki, M. J., Meira Jr, W., and Meira, W. (2014). Data mining and analysis: fundamental concepts and algorithms.
- Zarei, M., Izadi, D., and Samani, K. A. (2009). Detecting overlapping community structure of networks based on vertex–vertex correlations. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(11):P11013.
- Zeng, J., Li, J., Song, Y., Gao, C., Lyu, M. R., and King, I. (2018). Topic memory networks for short text classification. *arXiv preprint arXiv:1809.03664*.

- Zhang, J. and Korfhage, R. R. (1999). A distance and angle similarity measure method. *Journal of the Association for Information Science and Technology*, 50(9):772.
- Zhang, Y., Wang, S., and Ji, G. (2015). A comprehensive survey on particle swarm optimization algorithm and its applications. *Mathematical Problems in Engineering*, 2015.
- Zhang, Y. and Yeung, D.-Y. (2012). Overlapping community detection via bounded nonnegative matrix tri-factorization. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 606–614. ACM.
- Zhila, A., tau Yih, W., Meek, C., Zweig, G., and Mikolov, T. (2013). Combining Heterogeneous Models for Measuring Relational Similarity. In *NAACL-HLT*.
- Zuo, Y., Wu, J., Zhang, H., Lin, H., Wang, F., Xu, K., and Xiong, H. (2016a). Topic modeling of short texts: A pseudo-document view. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 2105–2114. ACM.
- Zuo, Y., Zhao, J., and Xu, K. (2016b). Word network topic model: a simple but general solution for short and imbalanced texts. *Knowledge and Information Systems*, 48(2):379–398.