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

Felipe Augusto Resende Viegas

On the Role of Semantic Word Clusters — CluWords — in Natural Language Processing (NLP) Tasks

Belo Horizonte
2023

Felipe Augusto Resende Viegas

On the Role of Semantic Word Clusters — CluWords — in Natural Language Processing (NLP) Tasks

Final Version

Dissertation presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Doctor in Computer Science.

Advisor: Marcos André Gonçalves
Co-Advisor: Leonardo Chaves Dutra da Rocha

Belo Horizonte
2023

Viegas, Felipe Augusto Resende.

V656o On the role of semantic word clusters — CluWords — in natural language processing (NLP) tasks [recurso eletrônico] / Felipe Augusto Resende Viegas – 2023.
1 recurso online (124 f. il, color.) : pdf.

Orientador: Marcos André Gonçalves.
Coorientador: Leonardo Chaves Dutra da Rocha.

Tese (Doutorado) - Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Ciências da Computação.
Referências: f. 115-124

1. Computação – Teses. 2. Processamento de linguagem natural (Computação) – Teses. 3. Processamento de textos (Computação) – Teses. I. Gonçalves, Marcos André. II. Rocha, Leonardo Chaves Dutra da. III. Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Computação. IV. Título.

CDU 519.6*73(043)

Ficha catalográfica elaborada pela bibliotecária Irenquer Vismeg Lucas Cruz
CRB 6/819 - Universidade Federal de Minas Gerais - ICEx

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

FOLHA DE APROVAÇÃO

UM ESTUDO APROFUNDADO SOBRE GRUPOS SEMÂNTICOS DE PALAVRAS - CLUWORDS - EM TAREFAS DE PROCESSAMENTO DE LINGUAGEM NATURAL (PLN)

FELIPE AUGUSTO RESENDE VIEGAS

Tese defendida e aprovada pela banca examinadora constituída pelos Senhores(a):

Documento assinado digitalmente
gov.br MARCOS ANDRE GONCALVES
Data: 19/04/2024 08:46:31-0300
Verifique em https://validar.iti.gov.br

Prof. Marcos André Gonçalves - Orientador
Departamento de Ciência da Computação - UFMG

Documento assinado digitalmente
gov.br LEONARDO CHAVES DUTRA DA ROCHA
Data: 19/04/2024 09:30:21-0300
Verifique em https://validar.iti.gov.br

Prof. Leonardo Chaves Dutra da Rocha - Coorientador
Departamento de Ciência da Computação - UFMG

Prof. Pedro Olmo Stancioli Vaz de Melo
Departamento de Ciência da Computação - UFMG

Prof. Rodrygo Luis Teodoro Santos Departamento de Ciência da Computação - UFMG

Documento assinado digitalmente
gov.br VIVIANE PEREIRA MOREIRA
Data: 19/04/2024 11:01:44-0300
Verifique em https://validar.iti.gov.br

Profa. Viviane Pereira Moreira
Departamento de Informática Aplicada - UFRGS

Profa. Renata Vieira
CIDEHUS - Universidade de Évora

Belo Horizonte, 10 de julho de 2023.

This thesis is dedicated to my wonderful wife, Juliana, and my beloved parents, Fátima and Sérgio, for their unwavering support, encouragement, and understanding.

Acknowledgments

I would like to express my deepest gratitude to all those who have supported and guided me throughout the journey of completing this PhD thesis.

First and foremost, I would like to thank my parents for their unwavering support and encouragement. Your love and sacrifices have been the foundation upon which I have built my academic career. I am eternally grateful for the values and perseverance you have instilled in me.

To my wife wonderful Juliana, for her endless love, patience, understanding, and unending support have been my anchor throughout this journey. Your steadfast support and encouragement have been my constant source of strength throughout this journey. Thank you for standing by my side and believing in me, even during the most challenging times. Your presence in my life has been a source of immense strength and motivation.

I am profoundly grateful to my advisor, Marcos Gonçalves, for their invaluable guidance, insightful advice, and continuous encouragement. I would also like to extend my heartfelt thanks to my co-advisor and friend, Leonardo Rocha, for his support in the thesis and since my first year of Computer Science, for his constructive feedback, and for always being there to assist me in my research endeavors. Your expertise and dedication have been crucial to the completion of this work.

I want to acknowledge my colleagues, whose collaboration and assistance have been instrumental in developing this thesis. Your help, ideas, and camaraderie have made this journey enjoyable and intellectually stimulating. I am thankful for the knowledge and experiences we have shared.

Lastly, I would like to thank everyone else who has contributed, in any way, to the successful completion of this thesis. Your support and encouragement have been greatly appreciated.

"*Our virtues and failings are inseparable, like force and matter. When they separate, man is no more.*"

(Nikola Tesla)

Resumo

A capacidade de representar dados de maneira significativa e eficiente é crucial para as aplicações de Processamento de Linguagem Natural (PLN), pois isso afeta drasticamente o resultado dos métodos de aprendizado de máquina. Nesse contexto, esta tese de doutorado se concentra em projetar uma nova representação de documentos que agrupa palavras semanticamente relacionadas, combinadas com filtragem específica para a tarefa e esquemas de ponderação, chamada **CluWords**. Conceitualmente, as CluWords correspondem a grupos que incorporam palavras semanticamente relacionadas, construídas por meio de funções de distância e mecanismos de filtragem. Mais do que simples grupos de palavras relacionadas (filtradas), as CluWords são combinadas com esquemas de ponderação específicos usados para capturar sua importância em uma tarefa específica. Nossa principal hipótese é que a representação das CluWords pode melhorar a eficácia das aplicações de PLN, aprimorando a representação do documento e permitindo lidar com problemas como ruído e falta de informação. O framework das CluWords é decomposto em três etapas bem definidas e flexíveis, e pode ser aplicado em aplicações específicas. Foi explorado quatro aplicações de PLN: modelagem de tópicos, modelagem de tópicos hierárquica, léxicos de sentimento e análise de sentimento. As contribuições incluem: (i) a introdução de uma nova representação de dados, composta por três etapas gerais (agrupamento, filtragem e ponderação). Essas etapas são especialmente projetadas para superar desafios específicos relacionados a ruído e falta de informação em cada tarefa; (ii) o design dos componentes das CluWords capazes de melhorar a eficácia na detecção de tópicos relevantes para modelagem de tópicos, modelagem de tópicos hierárquica e análise de sentimento; (iii) a proposição de um conjunto de evidências experimentais empíricas para demonstrar que as relações semânticas podem ser eficazes para léxicos de sentimento; (iv) proposta de duas novas métricas de qualidade de tópico para avaliar a qualidade tópica das estruturas hierárquicas.Nossos experimentos mostram que as CluWords são o estado da arte em modelagem de tópicos e modelagem de tópicos hierárquica. No contexto de léxicos de sentimento, nossos resultados experimentais mostram que as relações semânticas fornecidas pela incorporação de palavras podem ser eficazes para o respectivo contexto. No contexto da análise de sentimento, nossos experimentos mostram que a filtragem e a ponderação dos CluWords são capazes de mitigar o ruído semântico.

Palavras-chave: representação de dados; word embedding; modelagem de tópicos; modelagem de tópicos hierárquica; análise de sentimento.

Abstract

The ability to represent data in meaningful and tractable (i.e., efficient) ways is crucial for Natural Language Processing (NLP) applications since it drastically impacts the outcome of machine learning methods. In this context, this Ph.D. thesis focuses on designing a new document representation that groups semantically related words coupled with task-specific filtering and weighting schemes called **CluWords**. Conceptually, CluWords correspond to clusters of semantically related word embedding built through distance functions and filtering mechanisms. More than simple groups of (filtered) related words, the CluWords are coupled with specific weighting schemes used to capture their importance to a specific task. Our main hypothesis is that the CluWords representation may improve the effectiveness of NLP applications by enhancing the document representation and enabling it to deal with issues such as noise and lack of information. The CluWords framework is decomposed into three well-defined and flexible steps, and it can be applied to overcome specific-task applications. This Ph.D. thesis explores four NLP applications: topic modeling, hierarchical topic modeling, sentiment lexicons, and sentiment analysis. The expected novel contributions of this thesis include (i) the introduction of a new data representation composed of three general steps (clustering, filtering, and weighting). These steps are specially designed to overcome task-specific challenges related to noise and lack of information; (ii) the design of CluWords' components capable of improving the effectiveness in detecting relevant topics for Topic Modeling, Hierarchical Topic Modeling applications, and Sentiment Analysis; (iii) the proposal of a set of empirical, experimental evidence to show that semantic relationships can be effective for Sentiment Lexicons; (iv) proposal of two new topic quality metrics to assess the topical quality of the hierarchical structures. In this Ph.D. thesis, our experiments show that CluWords is state-of-the-art in topic modeling and hierarchical topic modeling. In the context of sentiment lexicons, our experiment results show that semantic relationships provided by word embedding can be effective for the respective context. In the context of sentiment analysis, our experiments show that CluWords filtering and weighting can mitigate semantic noise.

Keywords: data representation; word embedding; topic modeling; hierarchical topic modeling; sentiment analysis.

List of Figures

List of Tables

Summary

Chapter 1

Introduction

1.1 Motivation

Much has been developed recently in terms of text and particular word representations. In this Ph.D. thesis, we pay special attention to a particular type of representation: *static word embedding models*, exemplified by models such as Word2Vec [58], GloVe [71] and FastText [59]. Word embedding representations can be divided into two groups: Static word embedding and Contextual word embedding. Static word embedding is the class of word embeddings that generate a single embedding representation for a word. These models are based on co-occurrence statistics and contextual information of textual datasets. They represent words as high-dimensional vectors so that their similarities correlate with semantic and/or positional relatedness. Contextual word embedding, which is transformer architectures [21] capable of generating contextual (or more than one representation) for words. This means contextual embeddings can capture the meaning of the words given their context, despite being state-of-the-art in several domains, such as Document Classification [20], some works [24] that show that static word embeddings, such as FastText, outperform contextual embedding representations. As shown in [4], embedding models consistently outperform count models (e.g., TF-IDF) in several tasks, such as concept categorization, synonyms detection, and semantic relatedness, providing strong evidence in favor of the supposed superiority of word embedding models. Our proposed solution exploits such static embedding representations to enhance data representation without being limited to any of them.

Historically, in Natural Language Processing (NLP), texts have been represented with fixed-length vector representations (Vector Space Models), referred to in the literature as the bag-of-words (BoW). In this representation, each vector is of length equal to the size of the collection's vocabulary. Each vector element has a value corresponding to the weight of the term that the vector represents in the document. Several robust strategies exist to weigh the importance of terms, e.g., Term Frequency-Inverse Document Frequency (TF-IDF) [80].

Although conceptually simple and cheap (from a computational cost perspective), the BoW representation suffers from problems in sparse collections, in which most documents contain only a small portion of the collection‘s vocabulary. Moreover, the BoW representation usually does not consider the information about the positions occupied by the words in the original document, which can often be correlated with semantic relationships among terms. In this context, document enrichment strategies, such as n-grams (a.k.a. Bag-of-n-grams), have been adopted [36]. N-grams are rudimentary models that exploit word ordering, capturing the local relation between them. However, these strategies may not be able to capture complex semantic relationships among terms, which have a large potential to determine class assignments. [27] proposed a strategy to capture complex semantic relationships based on the co-occurrence of words in the documents. The strategy called *compound features* (or *c-features*) generates pairs of discriminative words to reduce the ambiguity and noise of the BoW representation.

Over the past years, since the first proposal of static word embeddings, works in the literature [43] adopted techniques to enrich the data representation, exploiting the use of word embeddings [58]. Static embedding models are Vector Space Model types that have recently been extensively used in Natural Language Processing (NLP). Word2Vec [58], for instance, uses an n-gram model with gaps (Skip-gram) to predict words composing the context of a given input word, capturing co-occurrence relationships between words and their meanings. FastText [59], on the other hand, learns vectors for the sub-words (i.e., character n-grams) found within each word and the complete word. Both models serve the same purpose: *Words that occur in similar contexts tend to have similar vector representations, i.e.,* embeddings*, allowing for more valuable semantic relationships and producing good descriptions of a word's general meaning.*

Though richer than previous models, embedding models, in general, may still suffer from semantic noise, in which words with different definitions have high similarity in the vector space. The semantic noise happens mainly in two scenarios:

- The application domain is distinct from the domain in which the embeddings were created. In this case, it is important to note that the corpus used to train the embedding is relevant. If the domain is different, but the number of documents is large enough, the quantity of information may help the generalization of the embedding space [59, 8];
- When a small set of training documents is used to train the embedding vector space. The absence of (enough) training information makes the vector space inaccurate in capturing semantic information among words.

In both scenarios, the learned embedding model may not capture the correct information about the words, especially the infrequent words [70], leading to semantic noise due to the lack of information.

In this work, we propose to advance the research on word/text representations by proposing a new powerful concept called **CluWords** [98, 95, 96, 93, 94] - clusters of semantically related words coupled with task-specific filtering and weighting schemes – to solve the described challenges in NLP Tasks, including Sentiment Analysis, Topic Modeling, and Automatic Text Classification.

1.2 Our Proposal

Conceptually, our CluWords are composed by applying three generic steps (clustering, filtering, and weighting) to build a more informative representation for a collection of textual data given an application scenario. These three steps can be observed in Figure 1.1. In more detail, the CluWords correspond to clusters of semantically related word embeddings [59] built by means of the application of distance functions[1] and filtering mechanisms. More than simple groups of (filtered) related words, CluWords are coupled with specific weighting schemes to capture their importance to a specific task.

Our main hypothesis with the CluWords is that by exploiting the word embeddings similarities, **and mainly,** by filtering out potential noise (i.e., irrelevant words from the cluster, which depends on the task) and properly weighting them according to the task at hand, we should be able to construct rich word representations. In other words, due to the filtering and weighting steps, CluWords can deal with semantic noise from static word embedding (including pre-trained representations), which favors its use, especially for small collections. There is a deficiency in training and embedding representation in small datasets due to the lack of information. The CluWords representation can capture word relationships that are very useful for applications that have a semantic component, such as Topic Modeling [98, 95, 96], Sentiment Analysis [93] and Text Classification [17, 20]. In this Ph.D. thesis, we exploit the CluWords representation in three application scenarios: topic modeling, hierarchical topic modeling, and sentiment analysis.

In the context of *topic modeling*, we propose to exploit the CluWords representation to generate "meta-words" capable of enhancing the document representation. Explicitly exploiting similarity between word embeddings to find the nearest words provides fine-grained information about relationships between words. The CluWords in this context combine traditional syntactic evidence (occurrences of words in a document) and semantic similarity among a word and its neighbors using a parameter α that weights the contribution from these sources of evidence. Our goal is to mitigate the potential drawbacks

[1] CluWords are not limited by any particular type of word embedding or distance function, being flexible enough to accommodate many options.

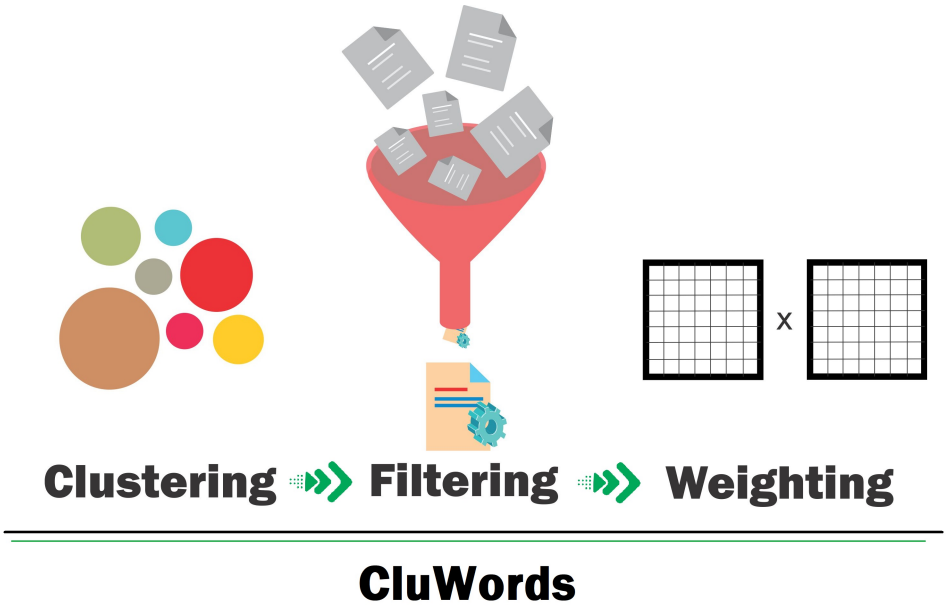


Figure 1.1: CluWords schema – The CluWords concept is composed of three flexible steps: (i) Clustering; (ii) Filtering; (iii) Weighting.

(e.g., noise) of using the projected space of word embeddings by exploiting only clear similarity evidence. Another novel contribution is a specific TF-IDF weighting scheme for the CluWords, as described in [95].

In the context of *hierarchical topic modeling*, we propose the CluHTM method, which exploits the CluWords representation combined with a non-probabilistic hierarchical topic method framework to build hierarchical structures. The idea is to use the CluWords to capture the semantic relationships among words at all hierarchy levels. CluWords can solve the lack of information in generating cohesive topics with non-probabilistic matrix factorization, especially in the deeper levels of the hierarchy. More specifically, information is absent because factorization methods exploit only local information to build the topics over the hierarchy. Thus, information about documents becomes increasingly scarce at deeper levels of the hierarchy. Thus, our hypothesis is to explore the (global) information provided by CluWords to assign more information to matrix factorization methods, as described in [96]. In CluHTM, we exploit two variants: the f-CluHTM, which exploits the static word embeddings, and c-CluHTM, which exploits static transformations of BERT contextual embeddings. In addition to the CluHTM proposal, we observed a lack of metrics to evaluate the quality of topics across the hierarchy structure. So, we design two new topic quality metrics to assess topical quality, considering two aspects: (i) *Topic topological consistency (or redundancy)* and (ii) *Semantic hierarchical structure*. Thus, we propose the Uniqueness and Semantic Hierarchical Structure (SHS), which can capture redundancy and semantic relationships over the topics of the hierarchical structure.

In *sentiment analysis* contexts, we propose a set of hypotheses to investigate if word vector embeddings may carry relevant information about word sentiment value. In our hypotheses, as well as the experimental evaluation that corroborates them, we use sentiment analysis tools that can discriminate the sentiments' *polarity* – i.e., their *positive* or *negative* inclination – and their *intensity* (or *strength value*) – i.e., how strong they are. Usually, these strategies make use of a *lexicon* (or *dictionary*) to detect the polarity and intensity of words. A lexicon is formed by a set of words tagged with their respective *sentiment value*, consisting of a number (within a defined range) expressing both the words' polarity (given by the number's sign) and intensity (given by number's absolute value) [37]. Our main insight is to show that the proximity of word vectors is not necessarily related to polarity but to intensity. For instance, in the sentence "the movie was good" sentiment intensity increases if we replace the word "good" with "awesome", while sentiment polarity remains positive. On the other hand, replacing "good" with "bad" changes the sentence's sentiment polarity to negative but leaves its intensity essentially unaltered. In addition, we propose a simple method for *lexicon expansion*, exploiting word embedding distances, to show that the newly expanded lexicons may improve the quality of detecting the sentiment of a sentence (or document) [93].

Inspired by the *lexicon expansion*, we propose CluSent. This CluWord variant exploits the nearest words of a given pre-trained static word embedding to generate "meta-words" to expand and enhance the document representation in terms of syntactic and semantic information. CluSent's main hypothesis is that by exploiting word embeddings similarities, **and mainly,** by filtering out potential noise (i.e., irrelevant words from the cluster for sentiment inference) and by properly weighting them (in the case of sentiment analysis, with the appropriate polarity and intensities), we should be able to construct rich word representations for the sake of sentiment analysis. In other words, by exploiting customized dataset-oriented filtering and weighting mechanisms, CluSent can deal with semantic noise from pre-trained static word embeddings, especially for short texts. Thus, we propose a new TFIDF-like representation that exploits polarity and intensity, what we call $TF\text{-}AL$. We use this $TF\text{-}AL$ concept as **a filtering/weighting mechanism** in the CluSent representation. The idea here is to build *cluster of words* (a.k.a CluWord) of similar polarity and intensity, keeping only words of the same Part-of-Speech (PoS) into a CluWord, e.g., only adjectives or nouns with the same polarity and similar intensity would belong to the same CluWord for this task. In sum, we exploit information in the sentiment lexicon, i.e., polarity and the lexicons' intensity, to filter out words from a CluWord. The intensity is also used as a weighting measure for each CluWord, collaboratively with the semantic information.

1.3 Hypotheses

In this section, we present the fundamental hypotheses of this thesis:

Main hypothesis: The use of CluWords to enhance document representations in NLP applications such as *1- topic modeling*, and *2- sentiment analysis*, may improve the effectiveness of such applications by helping to deal specifically with issues of noise and information shortage present in traditional word embedding models. Since the CluWords generation comprises three general steps, it can be designed to overcome specific-task challenges related to noise and lack of information by adapting these steps to its task.

- More specifically, we believe that the filtering and weighting mechanisms may be able to adapt CluWords to different NLP scenarios, overcoming problems in representing a document with embeddings, including semantic noise and lack of information in each specific application.

1. *Topic Modeling:*

- Contextual information (semantic similarity) of words available in the word embeddings vectors allows the non-probabilistic topic modeling strategies to build more cohesive topics.
  - More specifically, topic modeling strategies generally adopt document representations commonly used in classification tasks. Word embedding models are more effective in representing documents than traditional representations since embeddings capture semantic information of words [43]. Accordingly, we believe that CluWords may exploit the embedding information in the clustering and filtering steps to build robust topic representation.

2. *Hierarchical Topic Modeling:*

- Contextual information (semantic similarity) of words available in the word embeddings vectors allows the non-probabilistic topic modeling strategies to build more cohesive topics in deeper levels of the hierarchy.
  - More specifically, hierarchical topic modeling strategies generally adopt fewer documents in deeper levels of the hierarchy. We believe that CluWords may exploit the embedding information in the clustering and filtering steps to build richer representations.

- Exploiting different types of embeddings – static and pooled contextual embeddings – to show that the CluWords are versatile to different types of embeddings.

3. *Sentiment Analysis:*

- Semantics can be derived from the meaning of the word itself and the context in which it is used. The sentiment of a word is somehow related to this meaning. So, it is possible that word vectors can carry relevant information about words' sentiment values (i.e., their polarity and intensity).
  - More specifically, we hypothesize that the "closer together" two-word vectors of a lexicon are, the more likely they share "similar" polarities and values of intensity.
- Combining polarity and intensity information of words in the representation of CluWords may improve the effectiveness of detecting the sentiment of sentences.
  - We believe that the polarity information and Part-of-Speech may be used to remove semantic noise in the CluWords generation (filtering mechanism).
  - In addition, the intensity information may be explored to build a new weighting scheme specially designed for the CluWords in sentiment analysis.
- The CluSent, which is the CluWords version designed for sentiment analysis, may improve the effectiveness of a classification model in detecting the correct class for documents.
  - In addition, we believe that exploring different filtering techniques and unlabeled sentiment information may enhance the document representation to smooth semantic noise.

1.4 Research Questions

Our main general research questions are: *(i) Can CluWords be effectively exploited to advance the state-of-the-art in NLP and Information Retrieval tasks? (ii) Are task-specific filtering and weighting effective mechanisms capable of adapting the CluWords to different NLP application scenarios?*. The specific research questions that may support our general research questions, considering three application scenarios, include:

1. **Topic Modeling:** *(i) Can we exploit the CluWords to enhance document representation for topic modeling? (ii) Can the CluWords add more information to hierarchical topic modeling models at deeper levels of the hierarchy?*

2. **Sentiment Analysis:** *(i) Can the CluWords be used to overcome issues of lack of information in sentiment analysis tasks? (ii) Can polarity/intensity and Part-of-Speech (PoS) be used to filter out words from CluWords for sentiment analysis?*

1.5 Contributions

So far, the development of our work has resulted in the following contributions:

1. *Topic Modeling:* We exploit the CluWords representation to enhance non-probabilistic topic modeling [98, 95]. One of the main advantages of CluWords is that they can be used in standard BoW representations, just as the original words, along with standard algorithms, without having to resort to sophisticated solutions such as pooling. The key ideas that make CluWords work so well in this application are a dynamic threshold that controls the size of the neighborhood (to control noise) and a new weighting scheme specially designed to weight this type of evidence in a BoW-like representation. Our results are currently state-of-the-art in topic modeling, with gains of 80%

2. *Hierarchical Topic Modeling:* We exploit the CluWords concept in hierarchical topical modeling. We have proposed the CluHTM, a non-probabilistic hierarchical topic modeling strategy – based on non-negative matrix factorization (NMF) [41], and CluWords [96]. One main advantage of CluHTM is that it originally exploits a cross-level stability analysis metric for defining the number of topics and ultimately 'the shape' of the hierarchical structure; as far as we know *this metric has never been applied with this goal*. In our first experimental evaluation, some of our results improve over the state-of-the-art by more than 500%. Our experimental results show that the CluWords information can improve the document representation, generating more cohesive topics in all evaluated hierarchy levels. We also propose a variant of the CluHTM using pooled transformations of BERT's contextual embeddings, called c-CluHTM. This c-CluHTM variant exploits the BERT's hidden layers by exploiting several pooling strategies (e.g., average, maximum, concatenation combined with pooling) to build a single-word embedding representation for each word in CluHTM's meta-word construction. The idea is to evaluate the quality of the

topics built with these new word embedding representations in the CluHTM solution compared to the standard CluHTM solution (a.k.a. f-CluHTM). Our experimental evaluation demonstrates the superiority of c-CluHTM with gains between 12% and 21% regarding the NPMI and Coherence. We also proposed two supplement topic metric evaluations – *Uniqueness* that considers topic topological consistency (or redundancy), and the *Semantic Hierarchical Structure (SHS)* that captures the semantic relatedness of the hierarchies (a.k.a. topologies). Regarding these two metrics, the CluHTM variants can achieve gains of up to 60% when compared to the same (best) baselines.

3. *Sentiment Analysis:* We challenge the notion popularized in [85] that it is not possible to exploit (general) semantic relationships based on vector proximity by expanding existing manual lexicons with unknown words. The affective (sentiment) values of these new words (previously unknown in the lexicon) can be automatically derived from the neighborhood of word embeddings in the lexicon. We perform empirical experiments to show that semantic relationships can be effective and exploited for Sentiment Lexicons.

Inspired by the semantic analysis previously described, we propose a CluWords extension, called CluSent, to build document representations for sentiment analysis that exploits the three steps of the CluWords' concept adding new filtering and weighting instantiations using sentiment lexicon's polarity and intensity to tackle the problems of information shortage and noise, commonly found in sentiment analysis applications. The CluSent proposal is a *demonstration* of how to build and dynamically instantiate the CluWords' filtering (aiming at de-noising) and weighting mechanisms by exploring polarity and intensity information from unsupervised lexicons. In our experimental evaluation, comparing *CluSent* with five strong state-of-the-art sentiment analysis baselines in a large benchmark with 19 datasets, our solution achieved the best results in 30 out 38 possibilities (19 datasets considering MacroF1 and MicroF1), with gains up to 14.21% (*ss_bbc*), 7.60% (*ss_digg*) and 7.17% (*ss_rw*) compared to the *best baseline in each dataset*, in terms of MacroF1.

Our work on building and applying the concept of CluWords in NLP tasks have been validated and published in the main Information Retrieval and Natural Language Processing conferences and journals in the past years, including:

1. Contextual CluHTM – Exploiting Contextual Embeddings in Hierarchical Topic Modeling and Investigating the Limits of the Current Evaluation Metrics. (*Submitted to a journal*)

2. CluSent – Combining Semantic Expansion and De-Noising for Dataset-Oriented Sentiment Analysis of Short Texts, WebMedia 2023.

3. Exploiting Semantic Relationships for Expanding Sentiment lexicons. Information Systems 2020.

4. CluHTM - Semantic Hierarchical Topic Modeling based on CluWords, ACL 2020.

5. CluWords: Exploiting Semantic Word Clusters for Enhanced Topic Modeling, WSDM 2019.

6. Semantically-Enhanced Topic Modeling, CIKM 2018.

Our work also supported other research scenarios where we contributed and validated the CluWords concept:

1. New Metrics for Assessing the Quality of Hierarchical Topic Modeling Strategies. Electronic Journal of Undergraduate Research on Computing 2023.

2. A framework for automatic construction of search profiles based on semantic topic modeling. Electronic Journal of Undergraduate Research on Computing 2023.

3. Evaluating Topic Modeling Pre-processing Pipelines for Portuguese Texts. XXVIII Simpósio Brasileiro de Sistemas Multimídia e Web 2022.

4. Semantic Academic Profiler (SAP): a framework for researcher assessment based on semantic topic modeling. Scientometrics 2022.

5. CluWords: Exploiting Semantic Word Clusters for Enhanced Topic Modeling. CSBC 2019.

Our work granted two Google Latin American Research Awards (LARA) – eighth and ninth editions. In addition, this Ph.D. thesis also gave me the opportunity and the expertise to contribute to other research works, listed as follows:

1. On the class separability of contextual embeddings representations – or "The classifier does not matter when the (text) representation is so good! Information Processing & Management 2023

2. A Comparative Survey of Instance Selection Methods Applied to NonNeural and Transformer-Based Text Classification. ACM Computing Surveys 2023.

3. Stroke Outcome Measurements From Electronic Medical Records: Cross-sectional Study on the Effectiveness of Neural and Nonneural Classifiers. JMIR Medical Informatics 2021.

4. An Article-Oriented Framework for Automatic Semantic Analysis of COVID-19 Researches. ICCSA 2021.

5. PCV50 Automatic Classification of Electronic Health Records for a Value-Based Program through Machine Learning. Value in Health Journal 2021.

6. On the Cost-Effectiveness of Neural and Non-Neural Approaches and Representations for Text Classification: A Comprehensive Comparative Study. Information Processing & Management 2021.

7. Extended pre-processing pipeline for text classification: On the role of meta-feature representations, sparsification and selective sampling. Information Processing & Management 2020.

8. NetClass: A network-based relational model for document classification, Information Sciences 2018.

9. Exploiting efficient and effective lazy Semi-Bayesian strategies for text classification, Neurocomputing 2018.

10. A Feature-Oriented Sentiment Rating for Mobile App Reviews, WWW 2018.

11. A Genetic Programming Approach for Feature Selection in Highly Dimensional Skewed Data, Neurocomputing 2017.

12. A Two-Stage Machine learning approach for temporally-robust text classification, Information System 2017.

1.6 Roadmap

The remainder of this work is organized as follows.

Chapter 2 In this chapter, we describe related works. We start by introducing word embeddings. Then we describe some problems found in Topic Modeling. We discuss the main strategies proposed in the literature, including word embeddings. Finally, we describe relevant works about hierarchical topic modeling, presenting their strengths and weaknesses.

Chapter 3 In this chapter, we present our data representation called CluWords (cluster of words). We begin the chapter by describing the methods used to generate the CluWords. Then, we present the method based on TF-IDF we developed to ponder them.

Chapter 4 This chapter presents an experimental evaluation of our proposed CluWords in the context of Topic Modeling. We initially describe the datasets used in the experimental evaluation and the evaluation metrics. Next, we present the results compared with the main Topic Modeling methods in the literature.

Chapter 5 In this chapter, we present our CluHTM solution for the context of Hierarchical Topic Modeling. We begin the chapter by presenting the Stability method that we exploit in our proposal. Then, we describe our proposed solution that exploits CluWords to represent documents and the Stability method to automate the generation of topics in the hierarchy. We also present an experimental evaluation of our proposed CluHTM, comparing it with the main state-of-the-art baselines. We initially describe the datasets used in the experimental evaluation and the evaluation metrics. Next, we present the results compared with the main Hierarchical Topic Modeling methods.

Chapter 6 In this chapter, we present our theoretical hypothesis and a set of empirical experiments to demonstrate unequivocally that semantic relationships can be effectively used for sentiment analysis. Then, we present the CluSent, a CluWords instantiation designed for Sentiment Analysis. We present the results compared with the main sentiment analysis methods.

Chapter 7 Finally, in this chapter, we conclude the proposal, summarizing our main findings and proposing some directions for further investigation.

Chapter 2

Related Work

We begin the chapter by introducing the concept of Word Embedding since the proposed solution is built by exploring the information of word vectors. Then, we introduce the similarities and differences between topic modeling and clustering domain. Then, we present the main strategies proposed in the literature and the challenges in the Topic Modeling scenario. We also introduce the challenges and the two main groups of strategies developed for Hierarchical Topic Modeling. We also present the sentiment lexicons strategies that inspired our study with word embedding with sentiment analysis. And finally, we present the related work, regarding the sentiment analysis domain.

2.1 Word Embedding

Normally, natural language processing methods traditionally treat words as discrete atomic symbols (a.k.a tokens), and therefore 'cat' may be represented as *token_1* and 'dog' as *token_2*. These encodings are arbitrary and provide no useful information about how these words relate to each other. This means the model can leverage very little of what it has learned about 'cats' when processing data about 'dogs' (such as whether they are both animals, four-legged pets, etc.). Representing words as unique, discrete tokens furthermore leads to data sparsity, which usually means that we may need more information to train statistical models [45] successfully. Using vector representations can overcome some of these obstacles.

Vector space models (a.k.a Word Embeddings) represent (embed) words in a continuous vector space where semantically similar words are mapped to nearby points (are embedded near each other). Word Embeddings are similar to an autoencoder, encoding each word in a vector. Still, rather than training against the input words through reconstruction, as a restricted Boltzmann machine does, Word2Vec trains words against other words that neighbor them in the input corpus. Word embedding models are vector spaces whose similarity of vectors is related to the semantic information of words. All word em-

bedding methods depend in some way on the Distributional Hypothesis [34], which states that words that appear in the same contexts share semantic meanings. The approaches adopting this principle can be divided into two categories: methods based on counting (latent semantic analysis) and predictive methods (Word2Vec).

This distinction is elaborated in much more detail by [4]. Still, in a nutshell, Count-based methods compute the statistics of how often some word co-occurs with its neighbor words in a large text corpus and then map these count statistics down to a small, dense vector for each word. Predictive models directly try to predict a word from its neighbors regarding learned small, dense embedding vectors (considered model parameters).

In this work, we focus on the predictive method Word2Vec. Word2Vec [58] is a computationally efficient predictive model for learning embeddings from words in a text, using a gigantic text corpus to train a neural network that represents each word as a vector. Each word is mapped into a vector of $\mathcal{D}$ dimensions (typically between 100-300) in a continuous vector space. In this vector, words occurring in similar contexts (semantically related) tend to be represented as vectors at close distances.

It comes in two flavors: the Continuous Bag-of-Words model (CBOW) and the Skip-Gram model (Figure 2.1). Algorithmically, these models are similar, except that CBOW predicts target words (e.g. '*mat*') from source context words ('*the dog sleeps on the*'), while the skip-gram does the inverse and predicts source context-words from the target words. This inversion might seem like an arbitrary choice. Still, statistically, it has the effect that CBOW smoothes over a lot of the distributional information (by treating an entire context as one observation). For the most part, this turns out to be a useful thing for smaller datasets. However, skip-gram treats each context-target pair as a new observation, and this tends to do better when we have larger datasets.

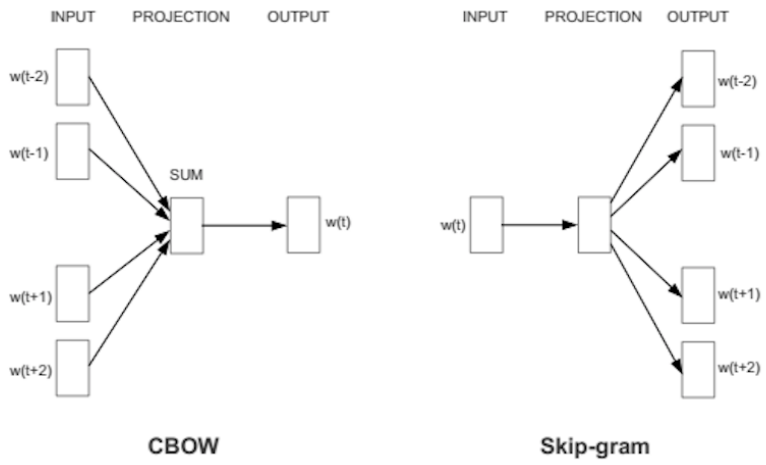


Figure 2.1: The two Word2Vec approaches, the Continuous Bag-of-Words model (CBOW) and the Skip-Gram model proposed in [58].

FastText [59], on the other hand, learns vectors for the sub-words (i.e., character n-

grams) found within each word and the complete word. At each training step in FastText, the mean of the target word and subword vectors are used for training. The adjustment calculated from the error is then used uniformly to update each of the combined vectors to form the target. This adds a lot of additional computation costs to the training step. The trade-off is a set of word vectors that contain embedded sub-word information. In [59], the authors claim that the potential benefits of FastText are: (i) it generates better word embeddings for rare words; (ii) The usage of character embedding for downstream tasks has recently been shown to boost the performance of those tasks compared to using a word embedding like Word2Vec.

BERT (Bidirectional Encoder Representations from Transformers) [21] is a method that exploits pre-trained language representations. Traditional word embedding models, such as Word2Vec and fastText, create *static* representations at the word level. Each token has a unique vector representation, whereas, in BERT models, words are dynamically represented with information from the surrounding words.

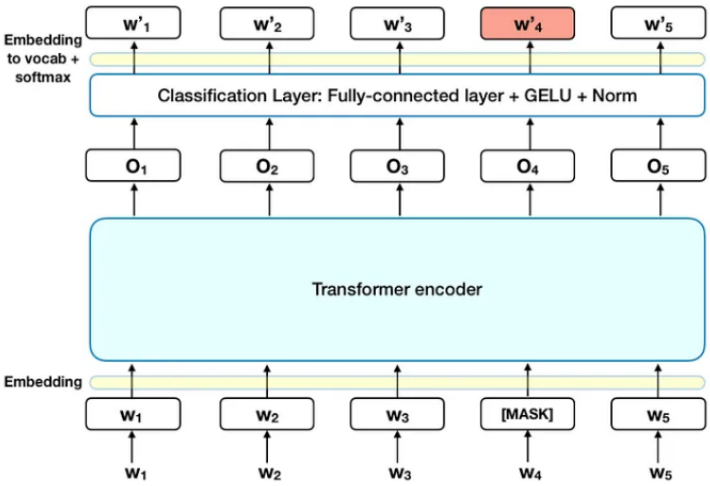


Figure 2.2: BERT structure.

To create the models, BERT uses a Transformer, an attention mechanism that learns the correlations between words in a text. Figure 2.2 illustrates its structure[1] with a classification layer at the end. A transformer is built with two separate mechanisms, an encoder that reads text input and a decoder that produces a prediction for the task. BERT differential lies in its bidirectional approach. Instead of using only the previous or subsequent words to construct the representation of a term, BERT reads the entire sequence of words at once. Its methodology comprises two phases. The first consists of replacing 15% of the words in each sequence with a MASK token, and then the model tries to predict the value of the masked words using the surrounding unmasked ones. In the second phase, the model receives pairs of sentences as input and learns to predict

[1] https://shorturl.at/aFHZ4

whether the second sentence is subsequent in the original document. In the end, we have very semantically rich word representations that consider each word's context.

2.2 Topic Modeling and Clustering – Similarities and Differences

Topic modeling methods [91, 16] aim to build latent topics given a textual collection. Each latent topic can summarize semantic meanings associated with documents in the collection. Usually, a topic is represented by a set of top κ words (i.e., relevant words) that summarize each created topic. The goal is to create topics in which the words have semantic synergy or co-occur in similar contexts. Figure 2.3 illustrates the three main steps that characterize all topic modeling approaches. The first step – the data input – corresponds to collecting the raw information about the documents (text collection). The second step – data representation – is used to transform the raw textual information into a (data) structure that a topic modeling approach will use to create the latent topics. The third step is the execution of a topic modeling approach which receives the data representation to create the latent topics. In Figure 1, the third step illustrates what is expected as an output of a topic modeling approach – latent topics composed of words. Some topic modeling approaches require the number of topics as an input parameter, while others can automatically select the proper number of topics. Usually, the top κ words selected to compose the topics are configurable.

Hierarchical topic modeling (HTM) is a variant of topic modeling. The main difference is that in HTM, the latent topics (step 3 in Figure 2.3) are organized in a hierarchical structure. This means that some topics have a hierarchical dependency on other topics.

Clustering approaches [61, 26], on the other hand, aims at partitioning the text collection into coherent groups (or clusters) of documents based on some criteria, e.g., semantic similarity. Figure 2.4 illustrates the three main steps of a clustering process. The first and second steps are similar to the steps described in Figure 2.3. The third step in the Figure represents the standard outcome of a clustering approach, assigning each document to a different cluster (or a partition). Some clustering approaches require the number of clusters as input, while others can automatically select the appropriate number of clusters based on some target criterion.

Based on the above descriptions, clustering and topic modeling are similar regarding (textual) data gathering and representation, but the expected outcomes are quite

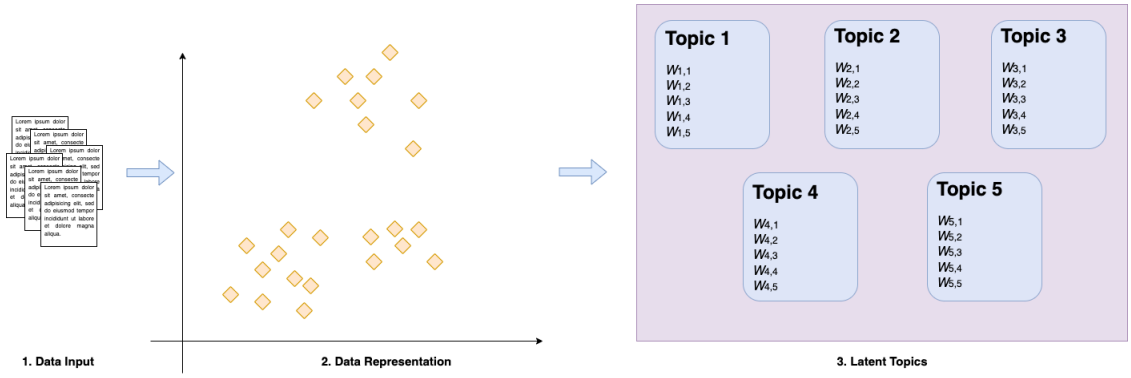


Figure 2.3: Simplistic depiction of the process of Topic Modeling generation – Data input; Data representation, that maps each input document into a data representation (e.g., TFIDF, Word Embedding, etc.), and Latent Topics, illustrating the outcome of a topic modeling approach.

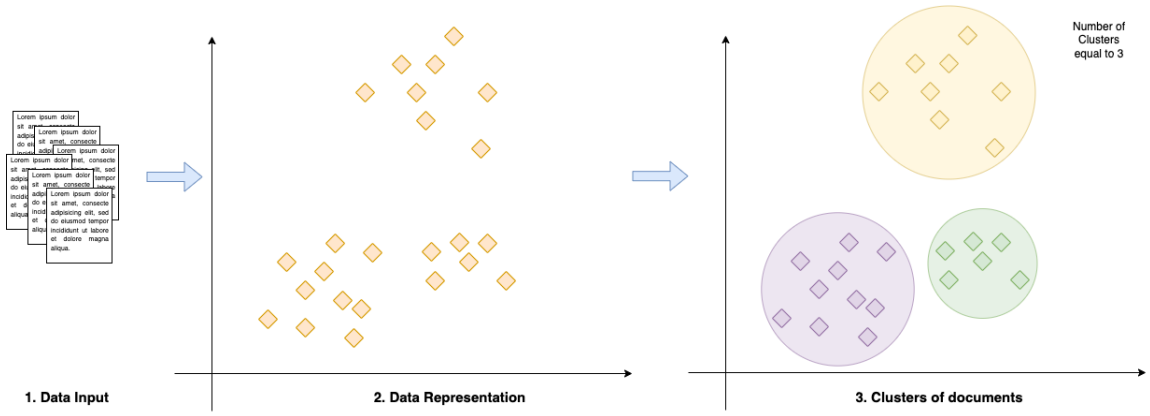


Figure 2.4: Simplistic depiction of the process of Clustering generation – Data input; Data representation; and Clustering of documents into different (semantic) groups.

different. Indeed, clustering approaches may be used internally within a topic modeling approach to help generate topics, but the final goals are still different. For instance, BERTopic [31] is an example of a topic modeling approach that exploits clustering to help generate topics by putting together documents with similar semantics. But as described in Section 2.4, BERTopic has several additional steps that build on top of the clusters to generate the latent topics, the final goal of the method.

In this context, Topic modeling evaluation metrics are designed to assess the quality of topic modeling methods using the top κ words selected for each latent topic (step 3 in Figure 2.3), according to some criteria (e.g., semantic relatedness).

2.3 Topic Modeling

We now turn our attention to algorithms that aim at uncovering abstract topics from data. We start with the probabilistic models. In [6], the authors propose the so-called latent Dirichlet allocation (LDA), which generalizes how $P(w|z)$, the probability distribution over terms w considering documents belonging to the abstract topic z, is estimated. In [15], the authors proposed the Bi-term Topic Model (BTM) method to address the data sparsity challenge. BTM uses the concept of bi-terms generated based on co-occurrence statistics of frequent terms.

In [14], the authors deal with incoherent topics through a Lifelong Topic Model (LTM): an iterative method that exploits data from several application domains that usually show some information overlapping in order to produce more coherent and reliable topics. The basic assumption here is that lexical and semantic relationships are key to uncovering coherent topics.

In the basic pLSA [35], the word-topic distributions (Φ) and document-topic distributions (Θ) matrices are learned by directly optimizing the log-likelihood of the training dataset $L(\Phi, \Theta)$. In the recently developed Additive Regularization of Topic Models, (ARTM) [100] approach, the basic pLSA model is augmented with additive regularizers. More specifically, the Φ and Θ matrices are learned by maximizing a linear combination of $L(\Phi, \Theta)$ and r regularizers $R_i(\Phi, \Theta), \forall i = 1, \cdots, r$, with regularization coefficients τ_i as shown in Equation 2.1.

$$R(\Phi, \Theta) = \sum_{i=1}^{r} \tau_i R(\Phi, \Theta), \;\; L(\Phi, \Theta) + R(\Phi, \Theta) \rightarrow \max(\Phi, \Theta) \tag{2.1}$$

Embedding-based Topic Model (ETM) [73] is another technique that incorporates the external word correlation knowledge into short texts to improve the coherence of topic modeling. ETM not only solves the problem of limited word co-occurrence information by aggregating short texts into long pseudo-texts but also utilizes a Markov Random Field regularized model that gives correlated words a better chance to be put into the same topic.

The FS method [32] is a strategy for building topics with sentiment information. It extracts words that co-occur often (a.k.a., bi-grams). Then, it infers the sentiment strength of the extracted bi-grams based on the sentiment score of the documents in which they occurred. To generate the topics, the strategy applies LDA over these sentimental bi-grams.

We now consider the non-probabilistic topic modeling, comprising strategies such as matrix factorization since it produces top-notch state-of-the-art performance without the limitations of probabilistic approaches, such as lack of observations when applied to

short texts. In this case, a dataset with n documents and m different terms is encoded as a design matrix $A \in \mathbb{R}^{n \times m}$ and the goal is to decompose A into sub-matrices that preserve some desired property or constraint.

The Non-negative Matrix Factorization (NMF) [41] is a well-known matrix factorization applicable to topic modeling. Under this strategy, the design matrix A is decomposed into two sub-matrices $H \in \mathbb{R}^{n \times k}$ and $W \in \mathbb{R}^{k \times m}$, such that $A \approx H \times W$. In this notation, k denotes the number of latent factors (i.e., topics), H encodes the relationship between documents and topics, and W encodes the relationship between terms and topics. The restriction enforced by NMF is that all three matrices do not have any negative element. When dealing with properly represented textual data, the design matrix usually contains non-negative term scores, such as TF-IDF, with well-defined semantics (e.g., term frequency and rarity). It is natural to expect the extracted factors to be non-negative so that such semantics can be somehow preserved. As a final note, as with the probabilistic strategies, the non-probabilistic ones can also generate incoherent topics, which is undesirable.

Recent works have been proposed to improve the construction of topics by means of using a word embedding as auxiliary information for probabilistic topic modeling. Das *et. al.* [19] propose an LDA-based topic model using multivariate Gaussian Distribution with word embedding. In [81], the authors propose the STE framework, which can learn word embedding and latent topics in a unified way. Finally, Li et al. [48] propose a GPU-DMM model, which can promote semantically related words using the information provided by the word embedding within any topic. The GPU-DMM extends the Dirichlet Multinomial Mixture (DMM) model by incorporating the learned word relatedness from word embedding through the generalized Pólya urn (GPU) model [48] in topic inferences.

In [82], the authors propose a semantics-assisted non-negative matrix factorization (SeaNMF) model to discover topics for the short texts. The method incorporates the word-context semantic correlations into the model. The semantic correlations between the words and their contexts are learned from the skip-gram view of the corpus, which was demonstrated to be effective for revealing word semantic relationships.

2.4 Hierarchical Topic Modeling

Hierarchical Topic Modeling (HTM) can be roughly grouped into supervised and unsupervised methods. Considering the supervised HTM strategies, we here highlight some relevant supervised extensions to the traditional Latent Dirichlet Allocation (LDA) [7], a widely used strategy for topic modeling (TM). LDA assumes a Dirichlet probability dis-

tribution over textual data to estimate the probabilities of words for each topic. In [56], the authors propose SLDA, a supervised extension of LDA that provides a statistical model for labeled documents. SLDA allows connecting each document to a regression variable to find latent topics that will best predict the response variables for future unlabeled documents. Based on SLDA, Hierarchical Supervised LDA (HSLDA) [72] incorporates the hierarchy of multi-label and pre-labeled data into a single model, thus providing extended prediction capabilities w.r.t., the latent hierarchical topics. The Supervised Nested LDA (SNLDA) [75], also based on SLDA, implements a generative probabilistic strategy where topics are sampled from a probability distribution. SNLDA extends SLDA by assuming the topics are organized into a tree structure.

We now focus on unsupervised HTM strategies, in which a hierarchical structure is learned during topic extraction. In [62] the authors propose the Hierarchical Pachinko Allocation Model (HPAM), an extension of *Pachinko Allocation* (PAM) [51]. In PAM, documents are a mix of distributions over an individual topic set, using a directed acyclic graph to represent the co-occurrences of topics. Each node in such a graph represents a *Dirichlet* distribution. At the highest level of PAM, there is only a single node, where the lowest levels represent a distribution between nodes of the next higher level. In HPAM, each node is associated with a distribution over the vocabulary of documents.

In [30], the authors propose the hLDA algorithm, which is also an expansion of LDA, considered state-of-the-art in HTM. In hLDA, the nested Chinese Restaurant Process (nCRP) is used to generate a hierarchical tree in addition to using the text Dirichlet distribution. NCRP needs two parameters: the tree level and a γ parameter. At each tree node, a document can belong to a path or create a new tree path with probability controlled by γ. More recently, in [104], the authors propose the unsupervised HTM strategy named a knowledge-based hierarchical topic model (KHTM). This method is based on hLDA and, as such, models a generative process whose parameter estimation strategy is based on Gibbs sampling. KHTM can uncover prior knowledge (such as the semantic correlation among words), organizing them into a hierarchy consisting of knowledge sets (k-sets). More specifically, the method first generates, through hLDA, an initial set of topics. After comparing pairs of topics, those topics with similarity higher than α (a.k.a., k-sets) are filtered so that the first 20 words of each topic are kept, and the remaining are just discarded. Those extracted k-sets are an extra weight when extracting the final topics.

Probably the most similar work to ours is the HSOC strategy, proposed in [52], which proposes to use NMF for solving HTM tasks. To mitigate the main drawbacks of NMF in the HTM setting[2], HSOC relies on three optimization constraints to properly drive the matrix factorization operations when uncovering the hierarchical topic struc-

[2]Namely, the incoherence of topics and unreasonable hierarchical structure caused by the lack of a learned probability distribution that governs the document/topics relationships

ture. Such constraints are global independence, local independence, and information consistency, and they allow HSOC to derive hierarchical topics that somehow preserve topic coherence and reasonable hierarchical structures.

As it can be observed, almost all models, supervised or unsupervised, are based on LDA. Though matrix factorization strategies normally present better results than Dirichlet strategies in TM tasks, for HTM, the situation is quite different. Matrix factorization methods face difficult challenges in HTM, mainly regarding data size as one goes deeper into the hierarchy. More specifically, a matrix factorization must be applied to increasingly smaller data sets at every hierarchical level, ultimately leading to insufficient data at lower levels. These approaches also do not exploit semantics nor any external enrichment, relying only on the statistical information extracted from the dataset.

BERTopic [31] is a hybrid solution that exploits BERT embeddings to represent the documents and exploits a hierarchical clustering approach to build hierarchical topics. In more detail, the method is composed of five steps: (i) Embedded Documents: this step exploits the information provided by a transformer approach (i.e., BERT) to get the embedding vector (i.e., the CLS embedding) of the documents; (ii) Dimensionality reduction: this step exploits PCA or UMAP to reduce the dimension of the document embedding, (iii) Cluster Documents: this step exploits the HDBSCAN method to build the clusters of documents; (iv) Bag-of-Words (Bow): applies a BoW representation in a cluster-level to have a better representation of the words that occur in the clusters; (v) Topic Representation: exploits the c-TFIDF approach over the BoW representation of the previous step to get the topic representation based on words that (co-)occur in the same clusters. This method is capable of building hierarchies of topics. – however, the topological structure is different from other HTM methods. The levels of BerTopic hierarchy have only two topics (like a binary tree), as illustrated in Figure 2.5-B, unlike the other methods adopted as baselines, that produce hierarchies such as those shown in Figure 2.5-A. Since this method exploits BERT embeddings as ours, we use it as a baseline.

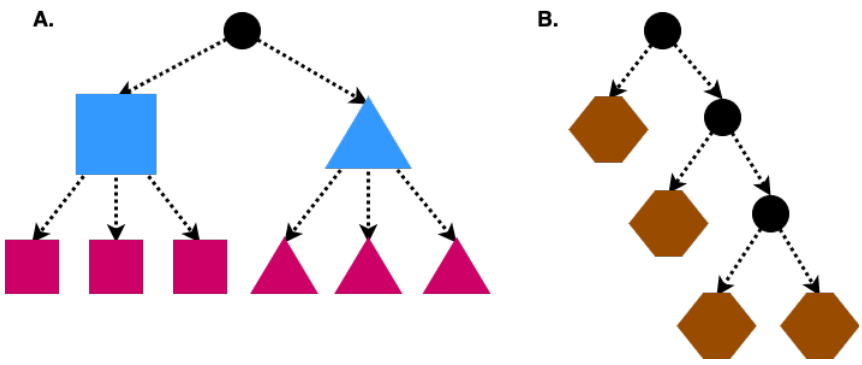


Figure 2.5: A. – Example Hierarchical structure built by CluHTM, HLDA, HPAM, and hARTM; B. – Hierarchical structure built by BERTopic.

hARTM [99] is a Non-Bayesian method hierarchical version of Additive Regularization of Topic Models (ARTM). This method exploits regularization to mitigate problems posed by the LDA-based methods. Bayesian generative models, like LDA, assumes that topic distributions over words and document distributions over topics are generated from prior Dirichlet distributions. This conflicts with two natural practical realities due to sparsity: (i) topics with zero probabilities in a document and (ii) words with zero probabilities in a topic. The authors mention that regularization reduces a potentially infinite set of solutions and helps to select a better one. The hierarchical version has a restriction of building only two levels of hierarchy per topic. We use this method as a baseline, adopting the implementation provided by BigARTM[3].

HTM solutions can also be roughly grouped into non-probabilistic and probabilistic models. In probabilistic strategies, textual data is considered to be "ruled" by an unknown probability distribution that governs the relationships between documents and topics hierarchically. The major drawback of this type of approach is the number of parameters in the model, which rapidly grows with the number of documents. This leads to learning inefficiencies and proneness to over-fitting, mainly for short textual data [86].

To overcome these drawbacks, non-probabilistic models aim at extracting hierarchical topic models through matrix factorization techniques instead of learning probability distributions. Such strategies also pose challenges. They are usually limited to local information (i.e., data limitation) as they go deeper into the hierarchy when extracting the latent topics. As one moves more in-depth in the hierarchical structure representing the latent topics, the available data rapidly reduces in size, directly impacting the quality of extracted topics (in terms of coherence and structure reasonableness). Probabilistic models mitigate this phenomenon by relying on global information when handling the probability distributions[104]. Because of that, the current main HTM methods are built based on probabilistic methods [30, 62].

2.5 Sentiment Lexicons

In this section, we briefly review the main efforts related to lexicons and their applicability to the task of polarity detection. We begin by discussing existing lexicon resources and how they were created. We then consider some works related to using lexicons in polarity detection. Finally, we review existing strategies to expand lexicons that inspired our study with word embedding with sentiment analysis.

[3] https://github.com/bigartm/bigartm

Lexicons have many possible applications and have been recently employed in sentence-level polarity detection. This task is particularly relevant in the context of social media, including the analysis of news [74], of emotional contagion [39], and overall happiness in Twitter messages [9].

The literature presents a wide variety of lexicons. The first construction efforts date back decades ago, usually relying on human-curated strategies. As an example, ANEW [10] was created by psychologists and linguists to provide a set of normative emotional ratings for a large number of words in English. Since then, many new lexicons have emerged. More recently, Nielsen [67] created the lexicon AFINN by combining existing ones. He tentatively expanded the ANEW lexicon by manually incorporating language support for microblogs. Although this expansion was not done using a clear and reproducible procedure, AFINN showed good results in practice in a specific task of SemEval Workshop. [89] introduced a lexicon annotated by humans using their SentiStrength algorithm. LIWC [87] is a widely used paid tool originally proposed to analyze sentiment patterns in English texts. NRC VAD [64] is a new, recently proposed lexicon with human ratings for the valence, arousal, and dominance dimensions for more than 20,000 words. The authors used Best–Worst Scaling to obtain fine-grained scores (and word rankings). They also addressed issues of annotation consistency that plague traditional rating scale methods of annotation. And finally, SenticNet 5 [13] used an ensemble of symbolic and subsymbolic AI to automatically discover lexicons. The method is a three-level Semantic Network: (i) Primitive Level; (ii) Concept Level, and (iii) Entity Level. This generalization process allowed the authors to extend the coverage of SenticNet and to build a better knowledge representation to encode semantics and sentiment.

There are two types of methods for sentence-level polarity analysis: machine-learning-based and lexicon-based methods. Machine-learning methods have the advantage of being able to create trained models for specific purposes and contexts. However, they require labeled data, impairs their applicability to new data. Consequently, many efforts have proposed new sentence-level methods based on lexicons. In particular, VADER [37] and SO-CAL [76] created new lexicons and combined different natural language processing techniques to determine sentence polarity. In VADER, the authors employed the Mechanical Turk annotators to select over 9,000 lexicon candidates. Then, from these candidates, a manual labeling process with ten independent human raters was applied to reach the current 7,500 VADER lexicons. VADER was considered one of the best methods for different datasets in a recent benchmark comparison of many lexicon-based sentence-level sentiment analysis methods [76]. In Chapter 6, we exploit the VADER lexicon in our analysis since it signed the best results over the same gold standard datasets used in [76].

Finally, there are some hybrid approaches. For instance, the authors of [57] combine unsupervised off-the-shelf methods using supervised techniques by exploiting boot-

strapping strategies to produce an initial (automatic) labeled seed. Though not directly compared to our work, our expanded lexicons could serve as a member of the committees used in [57]. Next, we discuss existing strategies to expand lexicons.

There are two common strategies used to create a large lexicon. The first corresponds to the corpus-based approaches that use seed words and patterns in unlabeled corpora to induce domain-specific lexicons. An effective corpus-based approach is constructing lexical graphs using word co-occurrences and then performing some form of label propagation over these graphs [92].

Other works have learned transformations of word embeddings to induce sentiment lexicons [78, 106]. For instance, SENTPRO [33] is a corpus-based based method that extends [92] by incorporating word embeddings to construct lexical graphs by label propagation. The strategy also uses a bootstrap method to obtain confidence sentiment scores. Densifier [78] is another method that creates a domain-dependent lexicon based on a set of embeddings learned on a large corpus of a domain. In [84], the authors proposed a neural architecture for learning the sentiment-specific phrase embedding from massive tweets selected with positive and negative emoticons. Since SENTPRO [33] outperforms Densifier [78], we evaluate our proposed method comparing it with SENTPRO.

RoWE [49] is a method that uses linear regression to build affective meaning in each affective dimension. In this method, each affective dimension represents an emotion. To infer an effective dimension, RoWE uses a set of words that reflects the sentiment of the dimension to be inferred and its respective word embedding as attributes for the regression method. Both SENTPRO and RoWE overcame the state-of-the-art techniques (including Densifier) for inducing sentiment lexicons. We have both, RoWE and SENTPRO as baselines.

Finally, the third way expands a human-curated lexicon (dictionary-based approaches). In [23], the authors used WordNet to expand a lexicon dictionary. WordNet [60] is a lexical repository of the English language commonly used in computational linguistics and natural language processing. WordNet represents words in a structure called synsets, interconnected through semantic concepts and lexical relations. WordNet has 117,000 synsets connected using a small number of "conceptual relationships". The main idea in [23] is to start from a small number of words whose polarity is known and to use the relationships between words in the dictionary to infer the polarities of the remaining words. The words are related if they share at least one synset in WordNet.

A similar strategy was used to create SentiWordNet [3], a WordNet version that includes each term's semantic orientation. This strategy largely improves the coverage of lexicons. However, it is computationally expensive due to the huge search space of the WordNet network. More importantly, it also introduces substantial inaccuracies in sentence-level sentiment classification [76] due to inconsistencies in the semantic classification of words. Determining such inconsistencies across lexicons is known to be an

NP-complete problem [22].

2.6 Sentiment Analysis

BERT [21] [4], is an end-to-end deep learning classifier composed of a bidirectional Transformer encoder. The model is pre-trained with a 3.3 billion word corpus. BERT predicts missing words from a sentence. The authors proposed two pre-trained tasks: *Masked language model (MLM)* – this technique masks words with a probability of 15% and the model is trained to predict the masked words. and (ii) *Next sentence prediction (NSP)* – it trains the model to predict whether two sentences are consecutive or not. BERT uses a multi-layer bidirectional Transformer encoder whose self-attention layer acts forward and backward. BERT redefined the state-of-the-art for several NLP tasks. SentiBERT [105] is a variant of BERT that captures compositional sentiment semantics. During training, SentiBERT exploits BERT to capture contextual information by masked language modeling. Then, the model learns the composition of meaning by predicting the sentiment labels of the phrase nodes. Unfortunately, due to documentation limitations and the unavailability of code description, we could not evaluate the SentBERT as provided by its authors[5]. Based on the experiments in [105], SentiBERT presents gains of 4% on average compared to BERT.

In [90] is described NB-weighted-BON[6], a method that trains document embeddings using cosine similarity. The Cosine similarity helped to reduce overfitting in the embedding generation task. The generated embeddings are combined with Naive Bayes weighted bag-of-n-grams. In the experimental results, NB-weighted-BON showed improved results compared to strong baselines, including BERT. In some comparative analyses[7], NB-weighted-BON is the current state-of-the-art (best-known algorithm) in some sentiment analysis benchmarks.

In [83], the authors proposed the Recursive Neural Tensor Network (RNTN). RNTN uses a tree where each node contains a word, its sentiment, and its associated label (positive, negative, neutral, positive, and negative). The solution represents a sentence using word vectors and an analysis tree. Given a new test document, the tree of this document is generated and compared (by similarity) with existing trees in the training set for predicting the respective label of the test document. RNTN is a classical and popular neural method that explores several paradigms as trees and similarity for sentiment anal-

[4] Available in https://github.com/yaserkl/
[5] https://github.com/DeepakDhana/SentiALBERT1
[6] https://github.com/tanthongtan/dv-cosine
[7] https://paperswithcode.com/sota/sentiment-analysis-on-imdb

ysis. It is still used by many recent methods [1, 38] as a "de facto" baseline to surpass, given its good average results in general.

Finally, in [79], the authors proposed the L-MIXED[8] strategies that exploit a BiLSTM model with pre-trained embeddings. The idea is to propose a training strategy that achieves higher accuracy than more complex models without an extra pre-training step. To do that, the authors explored the applicability of semi-supervised learning (SSL), where there is no prior pretraining step. The authors also proposed a mixed objective function for SSL that utilizes both labeled and unlabeled data to obtain further improvements in classification.

[8] https://github.com/DevSinghSachan/ssl_text_classification

Chapter 3

Cluwords

In this chapter, we introduce the CluWords representation, presenting the three general steps exploited by the method. Then, in Section 3.2, we present how we instantiate the clustering, filtering, and weighting steps of the CluWords representation for the Topic Modeling task.

3.1 CluWords Concept

Our main motivation for CluWords is to build document representation capable of using word embedding models to capture semantic relationships between words. We use them to improve the information about the words in the documents. However, improving this information is not a simple task. Embedding models can present semantic noise, in which words with different meanings may have high similarity in the vector space. To solve this challenge, we propose the CluWords (a.k.a Cluster of Words), clusters of semantically related words coupled with task-specific filtering and weighting schemes.

CluWords builds a richer data representation by exploiting word embedding similarities, filtering out potential noise (i.e., semantic noise, irrelevant words), and properly weighting them according to the application scenario. To build a document representation, the CluWords concept applies three generic steps. Figure 1.1 presents the CluWords concept. Each step is described as follows:

Clustering This step is exploited by strategies that capture semantic relationships between words through embedding models. [1] Methods used in this step may explore distance-based metrics between word vectors. Classic clustering methods, such as k-means, nearest neighbors, or customized clustering methods, can exploit this step. In addition, this step can be combined with the filtering step, as we shall see in Sec-

[1] Note that we are not necessarily limited to embedding models. However, this is the main current approach in the literature to capture relationships between words. It can be easily modified in the future.

tion 3.2.1. The intuition of this step consists of enriching the information about the documents with semantic information about the words.

Filtering This step involves applying strategies capable of filtering any noise in data representation. The filters can be built to smooth out the noise generated by the previous clustering step and noise from the original data representation. For instance, this step can be exploited by strategies that smooth inconsistent words from a semantic neighborhood.

Weighting Weighting is the last step for building the representation of CluWords. It combines the semantic information built in the first step with the document information. This step may explore strategies that weigh the semantic and document representations. For instance, this step can be exploited by strategies capable of weighing the relevance of terms based on the syntactic information in the documents.

3.2 Instantiating CluWords for Topic Modeling

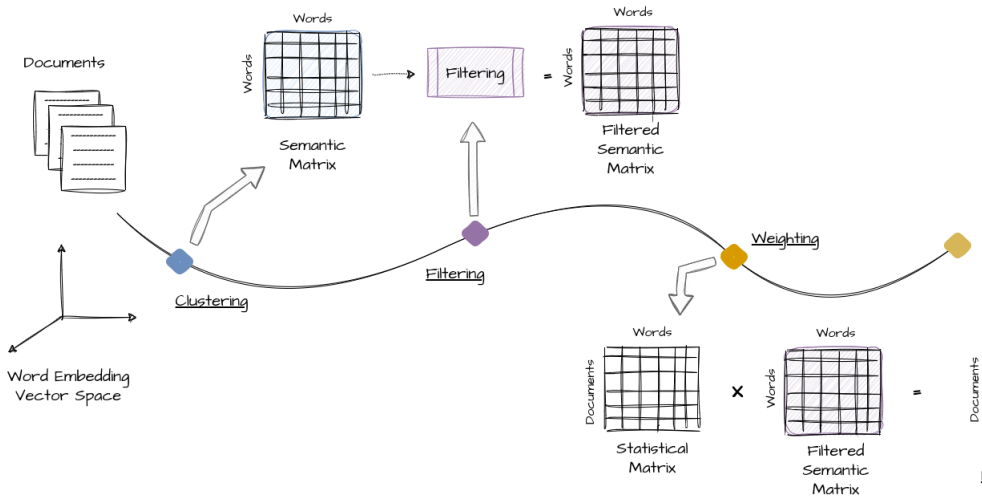


Figure 3.1: Diagram showing the steps for building the CluWords representation.

This Section presents how we instantiate the CluWords steps for the Topic Modeling tasks. The Figure 3.1 presents the process of transforming each original word into a CluWord (cluster of words) representation, described as follows:

- The clustering and filtering steps exploit the nearest neighborhood approach based on the information about the dataset, combined with word embedding representation;

- The weighting step exploits both semantic and statistical information combined to measure the importance of each CluWord in a modified version of the TF-IDF weighting scheme.

3.2.1 Clustering and Filtering

Let $\mathcal{W}$ be the set of vectors representing each word t in the dataset vocabulary (represented as $\mathcal{V}$). Each word $t \in \mathcal{V}$ has a corresponding vector $u \in \mathcal{W}$. The CluWords representation is defined as in Figure 3.1. The semantic matrix in the Figure 3.1 is defined as $C \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$, where each dimension has the size of the vocabulary ($|\mathcal{V}|$), t' represents the rows of C while t represents the columns. Finally, each index $C_{t',t}$ is computed according to Eq. 3.1.

$$C_{t',t} = \begin{cases} \omega(u_{t'}, u_t) & \text{if } \omega(u_{t'}, u_t) \geq \alpha \\ 0 & \text{otherwise,} \end{cases} \tag{3.1}$$

where $\omega(u_{t'}, u_t)$ is the cosine similarity defined in Eq. 3.2 and α is a similarity threshold that acts as a regularizer for the representation. Larger values of α lead to sparser representations. In this notation each column t of the semantic matrix C will be forming a CluWord t. Each value of the matrix $C_{t',t}$ may receive the cosine similarity between the vectors $u_{t'}$ and u_t in the embedding space $\mathcal{W}$ if it is greater than or equal to α. Otherwise, the $C_{t',t}$ receives zero, according to the Eq. 3.1.

$$\omega(u_{t'}, u_t) = \frac{\sum_i^l u_{t'i} \cdot u_{ti}}{\sqrt{\sum_i^l u_{t'i}^2} \cdot \sqrt{\sum_i^l u_{ti}^2}} \tag{3.2}$$

The CluWord (CW_t) for a term t relates t' with its closest words, limiting this relationship with the cutoff value α that filters noisy words (i.e., words that do not have a significant relationship with t) from the CluWord. Since threshold α is a cosine similarity value, it is contained within the interval $[0, 1]$. If $\alpha = 0$ the similarities of every term in $\mathcal{V}_T$ are included in CW_t, otherwise, if $\alpha = 1$ only the similarity of t to itself (i.e. $\omega(u_{t'}, u_t)$) is included in CW_t. Thus, selecting a value for parameter α is an important aspect of generating good CluWords. Moreover, α controls the sparsity of the resulting document representation. With high α values, there are only a few CluWords that relate to a document. This representation is similar to the traditional BOW representation, where the occurrence of a word in a document determines if that word will be used in the document representation. However, with low α values, more CluWords tend to be related to the document, reducing the document representation's sparsity. Note that once we

select an appropriate value for α, each CluWord CW_t keeps the values of similarities of the terms most similar to t according to the semantics established by the word embeddings.

Table 3.1 presents an example of the words belonging to a CluWord whose centroid is the word "chat". The Table shows the words we consider, in a very informal analysis, syntactically, semantically, or unrelated to the respective centroid.

Table 3.1: Example of the words belonging to a CluWord with "chat" as the centroid.

Centroid: chat	
Semantically similar words	audio, communicate, communication, contact, conversation, conversations, discuss, email, emails, forum, hear, interact, interaction, listen, listening, mail, message, messages, messaging, news, phone, post, reply, socialize, socializing, speak, talk, talking, voice, meeting, networking, room, service
Syntactically similar words	chat, chats, chatted, chatting
Unrelated/Undefined words	access, avatar, buddies, buddy, download, dude, evening, evenings, exchange, gallery, game, gaming, girl, girlfriend, guys, homework, interface, mate, mates, pal, server, sip, sit, smilies, strangers, stuff, telephone, thoughts, twitter, video, wander, wanna, web

We intend to use the CluWords to replace the original BOW representation of documents. It is important to note that the goal of CluWords is (mainly, but not only) to enrich the BOW representation by adding semantic information, that is, each term t will be replaced by its corresponding CluWord CW_t in each document d it belongs. To use the CluWords representation, we need to compute the TF-IDF of the CluWords. We describe this weighting scheme in Section 3.2.2.

3.2.2 Weighting

The conventional TF-IDF [80] is a measure of the importance of a term that evaluates two distinct aspects: (i) the relevance of the term in a specific document d (characterized by the TF component of the measure) and (ii) the importance of the term in the collection of documents to be considered (given by the IDF component). $TF_{t,d}$ accounts for the frequency of occurrence of term t in document d. IDF measures the importance of the term t in a collection of documents. The more documents a term occurs in, the less important it is considered. Thus, the IDF of a term should be inversely related to the number of documents in which the term occurs. The TF-IDF score $TFIDF_{t,d}$ of term t in document d is defined in Eq. 3.3.

$$TFIDF_{t,d} = TF_{t,d} \cdot log(\frac{|\mathcal{D}|}{n_t}) \tag{3.3}$$

where n_t is the number of documents in $\mathcal{D}$ where t occurs.

In Figure 3.1, the CluWords representation is defined as the product between the statistical matrix (a.k.a. term-frequency matrix) and semantic matrix C. The statistical

matrix (TF) can be represented as a $TF \in \mathbb{R}^{|\mathcal{D}|\times|\mathcal{V}|}$, where each position $TF_{d,t}$ relates to the frequency of a word t in document d. Thus, given a CluWord (CW) t for a document d, its data representation corresponds to

$$CW_{d,t} = \overrightarrow{TF_d} \times \overrightarrow{C_{,t}} \tag{3.4}$$

where $\overrightarrow{TF_d}$ has the term-frequencies of document d, and $\overrightarrow{C_{,t}}$ is the semantic scores for the CluWord t, according to Eq. 3.1.

The TF-IDF weighting for a CluWord t in a document d is defined as

$$CW_{d,t} = CW_{d,t} \times idf_t. \tag{3.5}$$

The IDF component is defined as

$$idf_t = log\left(\frac{|\mathcal{D}|}{\sum_{1\leq d\leq|\mathcal{D}|}\mu_{t,d}}\right) \tag{3.6}$$

where $\mathcal{D}$ is the number of documents and $\mu_{t,d}$ is the average of semantic weights of the semantic matrix C for the CluWord t ($\overrightarrow{C_{,t}}$) that occurs in the vocabulary $\mathcal{V}_d$.

The average $\mu_{t,d}$ is defined as

$$\mu_{t,d} = \frac{1}{\left|\mathcal{V}_{d,\overrightarrow{C_{,t}}}\right|} \cdot \sum_{t'\in(\mathcal{V}_d\cap\overrightarrow{C_{,t}})} C_{t',t} \tag{3.7}$$

where $\mathcal{V}_{d,\overrightarrow{C_{,t}}}$ is composed by all terms in document d which have the $C_{t',t}$ not equal to zero in CluWord $\overrightarrow{C_{,t}}$. This is formally defined in Eq. 3.8.

$$\mathcal{V}_{d,\overrightarrow{C_{,t}}} = \{t' \in d | C_{t',t} \neq 0 \text{ in } \overrightarrow{C_{,t}}\} \tag{3.8}$$

3.3 Complexity of CluWords

The complexity of building the clustering step is the nearest neighbor search, which can be exploited by using the fast approximate nearest neighbor search (HNSW) [55] with the complexity of $\mathcal{O}(logN)$. The complexity of multiplying the statistical and semantic matrix is $\mathcal{O}(NNZ(\overrightarrow{TF_d})NNZ(\overrightarrow{C_{,t}})/|\mathcal{V}|)$ in average, where $|\mathcal{V}|$ is the size of the vocabulary.

Chapter 4

Experimental Evaluation for Topic Modeling

This chapter presents our experimental evaluation of the CluWords in topic modeling. We initially describe the datasets, baselines, and evaluation metrics. Next, we present the results compared with the main topic modeling methods in the literature.

4.1 Experimental Setup

4.1.1 Datasets

The primary goal of our solution is to perform effective topic modeling so that more coherent topics can be extracted. We consider 12 real-world datasets as a reference to evaluate the topic model coherence. We created two of them by collecting comments from Facebook and Uber Apps in the Google Play Store. The others were obtained from previous works in the literature. For all, we performed stopword removal (using the standard SMART list). We removed words such as adverbs using the VADER lexicon dictionary [37], as most of the significant words for identifying topics are nouns and verbs. These procedures improved both the efficiency and effectiveness of all analyzed strategies. Table 4.1 provides a summary of the reference datasets, reporting the number of features (words), documents, the mean number of words per document (density), and the corresponding references.

[1]https://www.cc.gatech.edu/gvu/ii/jigsaw/datafiles.html
[2]http://qwone.com/~jason/20Newsgroups/

Dataset	#Feat	#Doc	Density
Angrybirds [32]	1,903	1,428	7.135
Dropbox [32]	2,430	1,909	9.501
Evernote [32]	6,307	8,273	11.002
InfoVis-Vast [1]	6,104	909	86.215
Pinterest [32]	2,174	3,168	4.478
TripAdvisor [32]	3,152	2,816	8.532
Tweets [50]	8,029	12,030	4.450
WhatsApp [32]	1,777	2,956	3.103
20NewsGroup [2]	29,842	15,411	76.408
ACM [97]	16,811	22,384	30.428
Uber	5,517	11,541	7.868
Facebook	5,168	12,297	6.427

Table 4.1: Dataset characteristics

4.1.2 Evaluation, Algorithms, and Procedures

We compare the topic modeling strategies using representative topic quality metrics in the literature [68, 69]. In general, there are three classes of topic quality metrics based on three criteria: (a) coherence, (b) mutual information, and (c) semantic representation. Here, we focus on (a) and (b) since they are the most used metrics in the literature [82, 69]. We also consider three topic lengths (5, 10, and 20 words) under each metric in our evaluation—different lengths may bring different challenges.

Regarding the metrics, *coherence* captures the easiness of interpretation by co-occurrence. Words that frequently co-occur in similar contexts in a corpus are easier to correlate since they usually define a more well-defined "concept" or "topic". For coherence, we employ an improved version of regular coherence [69], called TFIDF-Coherence, defined as

$$c_{tf-idf}(t, W_t) = \sum_{w_1, w_2 \in W_t} log \frac{\sum_{d:w_1,w_2 \in d} tf - idf(w_1, d) tf - idf(w_2, d)}{\sum_{d:w_1 \in d} tf - idf(w_1, d)} \tag{4.1}$$

where the *tf-idf* metric is computed with augmented frequency as

$$tf - idf(w, d) = \left(\frac{1}{2} + \frac{f(w, d)}{max_{w' \in d} f(w', d)}\right) log \frac{|D|}{|\{d \in D : w \in d\}|} \tag{4.2}$$

and $f(w, d)$ is the number of occurrences of a term w in document d. It skews the metric towards topics with high *tf-idf* scores since the numerator of the coherence fraction has a quadratic dependence on the TFIDF scores, and the denominator is only linear.

Another class of topic quality metrics is based on the notion of *pairwise point-wise mutual information (PMI)* between the top words in a topic. It captures how much one "gains" in the information given the occurrence of the other word, taking dependencies

between words into consideration. Following the work [68], we compute a *normalized version of PMI* (NPMI), in which, for a given ordered set of top words $W_t = (w_1, \cdots, w_N)$ in a topic, NPMI is computed as:

$$NPMI_t = \sum_{i<j} \frac{log \frac{p(w_i,w_j)}{p(w_i)p(w_j)}}{-log\ p(w_i, w_j)} \tag{4.3}$$

Next, we compare our proposed data representation described in Setion 3.2, as well the best configuration with eight topic model strategies described as follows

LDA: The Latent Dirichlet allocation (LDA) method [6] is a probabilistic model, which generalizes how $P(w|z)$, the probability distribution over terms w considering documents belonging to the abstract topic z, is estimated.

FS: The FS method [32] is a strategy used to build topics with sentiment information. It extracts words that co-occur often (a.k.a., bi-grams). Then, it infers the sentiment strength of the extracted bi-grams based on the sentiment score of the documents in which they occurred. To generate the topics, the strategy applies LDA over these sentimental bi-grams.

BTM: The Bi-term Topic Model (BTM) method [15] is an extension of LDA, proposed to deal with the data sparsity challenge. The BTM method uses the concept of bi-terms generated based on co-occurrence statistics of frequent terms.

LTM: The Lifelong Topic Model (LTM) [14] is an iterative method that exploits data from several application domains that usually overlap information to produce more coherent and reliable topics. The basic assumption here is that lexical and semantic relationships are key to uncovering coherent topics.

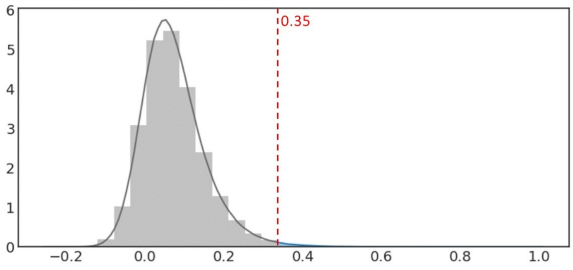
GPU-DMM: The GPU-DMM method captures semantically related words using the information provided by the word embedding within any topic. The GPU-DMM extends the Dirichlet Multinomial Mixture (DMM) model by incorporating the learned word relatedness from word embedding through the generalized Pólya urn (GPU) model [48] in topic inferences.

ETM: Embedding-based Topic Model (ETM) [73] is another technique that incorporates the external word correlation knowledge into short texts to improve the coherence of topic modeling. ETM not only solves the problem of very limited word co-occurrence information by aggregating short texts into long pseudo-texts but also utilizes a Markov Random Field regularized model that gives correlated words a better chance to be put into the same topic.

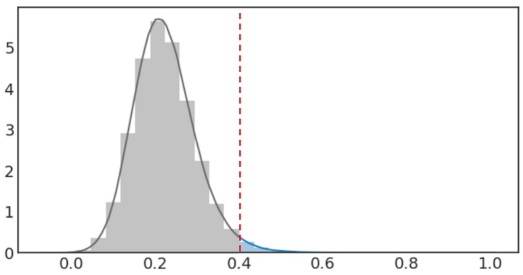
ARTM: The Additive Regularization of Topic Models (ARTM) [100] approach is the basic pLSA [35] model is augmented with additive regularizers.

SeaNMF: The Semantics-Assisted Non-Negative Matrix Factorization (SeaNMF) model [82] discover topics for the short texts. Basically, the method incorporates the word-context semantic correlations into the model. The semantic correlations between the words and their contexts are learned from the skip-gram view of the corpus, which was demonstrated to be effective for revealing word semantic relationships.

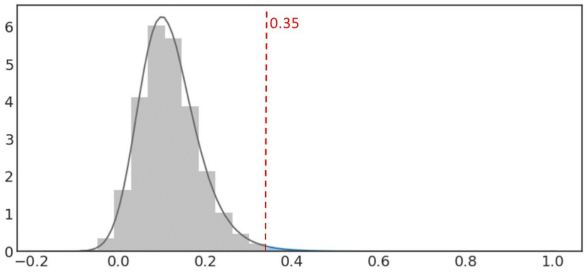
In our experiments, we adopt the Non-negative Matrix Factorization (NMF) [41] to evaluate the CluWords, since it is the main non-probabilistic matrix factorization. We discovered 25 topics for all datasets except 20News, ACM, and Tweets, where 20, 11, and 6 latent topics were discovered for these datasets, respectively. The number of topics for the app datasets was defined based on the choice of latent topics made in [32]. For the 20News, ACM, and Tweets datasets, we chose the number of topics equal to the real number of classes. We assess the statistical significance of our results utilizing a paired t-test with a 95% confidence and Holm-Bonferroni correction to account for multiple tests. In the next section, we present the results of experiments conducted to evaluate the effectiveness of CluWords using three different pre-trained word embedding spaces, considering NPMI scores. Next, we compare the best word embedding configuration with the baseline strategies regarding the scores of Coherence and NPMI.



(a) W2V GoogleNews

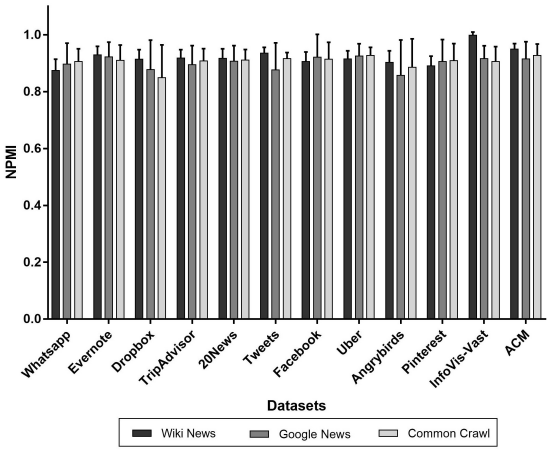


(b) FastText WikiNews

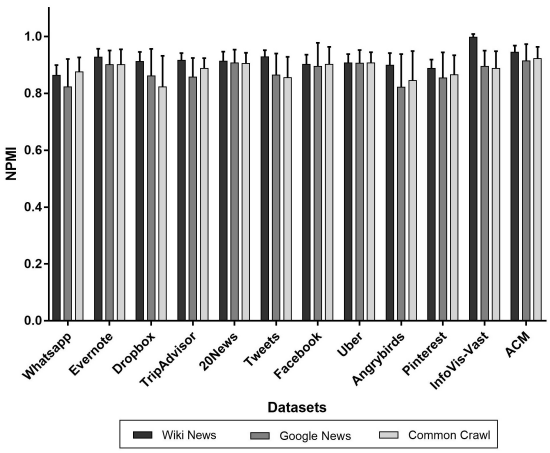


(c) FastText Common Crawl

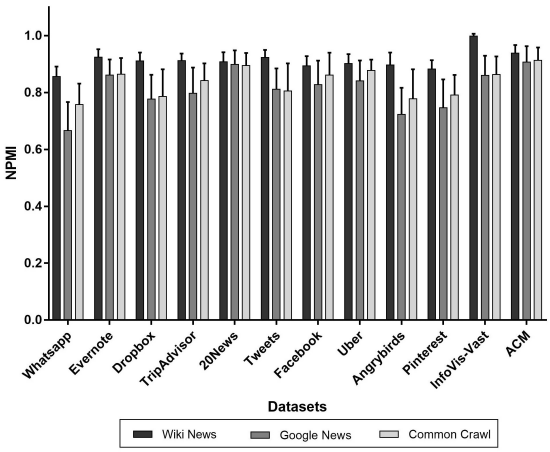
Figure 4.1: Cosine similarity histogram.



(a) 5 Words



(b) 10 Words



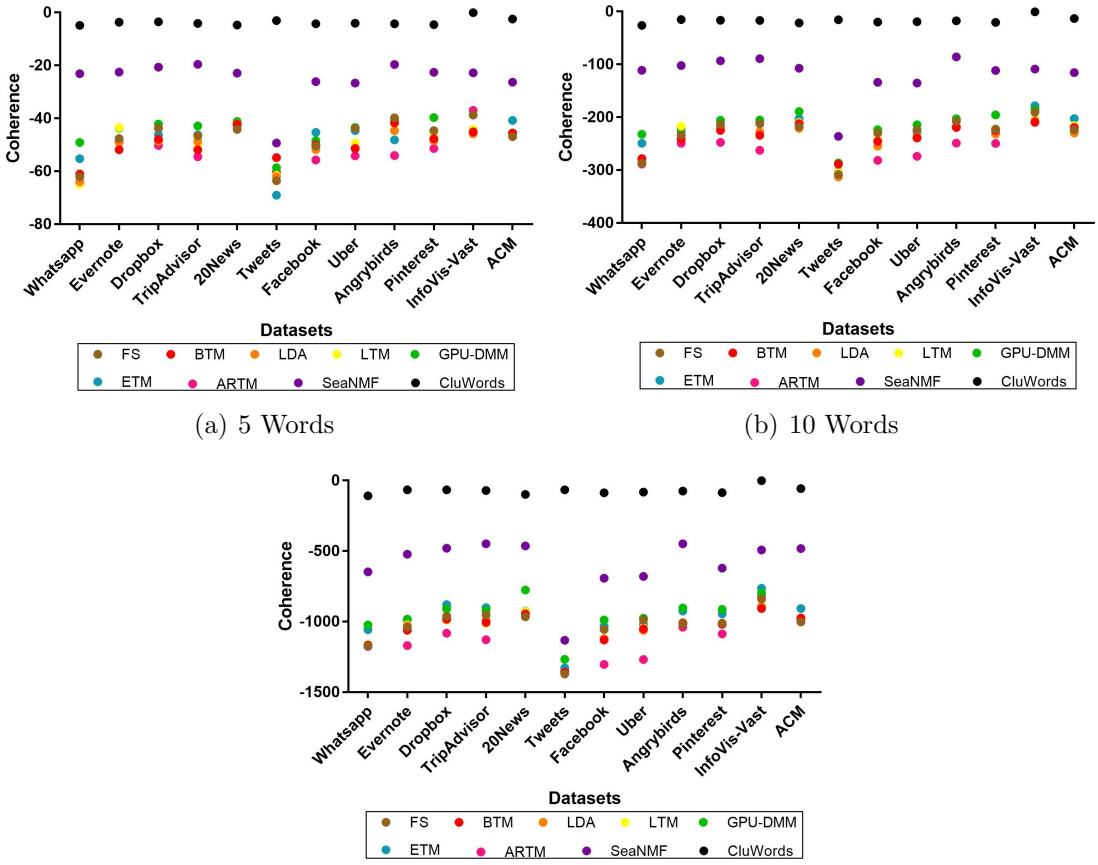
(c) 20 Words

Figure 4.2: Evaluation of CluWords exploring different Word Embeddings, in terms of NPMI score.

4.2 Experimental Results

4.2.1 Choosing the best word embedding space

In this Section, we compare the proposed CluWords with three pre-trained word embeddings spaces: (i) Word2Vec trained with GoogleNews [58]; (ii) FastText trained with WikiNews [59] and (iii) Fasttext trained on Common Crawl [59]. Initially, to build the proposed data representation, as described in Chapter 3, we must select a cosine similarity threshold of α. The idea is to select a threshold α that is restrictive, and capable of filtering pairs of noisy words. For this, we need to find the distribution of similarities between the word pairs of the word embedding to infer a similarity threshold.



(a) 5 Words

(b) 10 Words

(c) 20 Words

Figure 4.3: Comparing the results achieved by each strategy considering top 5, 10 and 20 words for TFIDF-Coherence.

Figure 4.1 shows the distribution of similarities of each pre-trained word embedding. We can observe that the distribution of similarities of the three-word vector spaces is similar and that the FastText WikiNews presents a slightly greater deviation than the other word embeddings. Thus, for our experiments, we chose a threshold α capable of selecting only 2% of the most similar word pairs. We select a high threshold of α to avoid an unexpected pair of terms since the space of the word vectors is not evenly dispersed, according to [63]. Thus, the threshold selected for FastText WikiNews is $\alpha \geq 0.40$, while for W2V GoogleNews and FastText Common Crawl, a threshold of $\alpha \geq 0.35$ has been selected.

Figure 4.2 contrasts the results of the proposed CluWords on the three evaluated word embedding spaces. FastText WikiNews always achieves superior results considering all datasets and topic lengths (5, 10, and 20 words). Most of the results (32 out of 36 results) are statistic ties, which suggests that the proposed data representation is capable of performing with the same quality in the three distinct word vector spaces. The CluWords results presented in the next Section were generated using the word embeddings from FastText WikiNews.

We performed an additional quantitative experiment using the Fasttext WikiNews space to reinforce the evidence that CluWords can also capture syntactic information. In the experiment, our goal is to show that in the process of selecting the neighborhood of a CluWord CW_t (Section 3.2), a part of the terms closest to term t are variations of the same word (e.g., the word *chats* is a variation of the word *chat*). Thus, given a CluWord CW_t, we select each term $t'|C_{t',t} \neq 0$ and derive t' to its stem form. We measured the proportion of terms affected by the stemming process. Table 4.3 illustrates the average affected terms in the CluWords, for the 12 datasets. We can observe that approximately 11% of the terms belonging to a CluWord are variations of the same word.

Strategies	Whatsapp			Evernote			Dropbox			TripAdvisor		
	5 words	10 words	20 words	5 words	10 words	20 words	5 words	10 words	20 words	5 words	10 words	20 words
FS	0.171 ± 0.051	0.201 ± 0.048	0.230 ± 0.043	0.102 ± 0.052	0.090 ± 0.020	0.100 ± 0.018	0.109 ± 0.042	0.097 ± 0.027	0.107 ± 0.018	0.094 ± 0.037	0.092 ± 0.028	0.104 ± 0.021
BTM	0.201 ± 0.057	0.236 ± 0.038	0.284 ± 0.038	0.118 ± 0.057	0.109 ± 0.029	0.120 ± 0.024	0.155 ± 0.050	0.161 ± 0.043	0.166 ± 0.040	0.130 ± 0.052	0.144 ± 0.044	0.158 ± 0.039
LDA	0.172 ± 0.050	0.230 ± 0.030	0.284 ± 0.042	0.114 ± 0.067	0.114 ± 0.036	0.114 ± 0.019	0.165 ± 0.110	0.149 ± 0.056	0.150 ± 0.037	0.114 ± 0.057	0.122 ± 0.029	0.137 ± 0.028
LTM	0.178 ± 0.052	0.225 ± 0.041	0.269 ± 0.040	0.193 ± 0.051	0.168 ± 0.044	0.158 ± 0.033	0.167 ± 0.072	0.160 ± 0.040	0.175 ± 0.046	0.149 ± 0.059	0.144 ± 0.035	0.161 ± 0.037
GPU-DMM	0.312 ± 0.165	0.327 ± 0.141	0.330 ± 0.131	0.258 ± 0.165	0.270 ± 0.149	0.229 ± 0.076	0.284 ± 0.147	0.267 ± 0.125	0.284 ± 0.129	0.286 ± 0.209	0.253 ± 0.144	0.244 ± 0.122
ETM	0.365 ± 0.171	0.378 ± 0.163	0.399 ± 0.154	0.319 ± 0.138	0.320 ± 0.133	0.331 ± 0.131	0.403 ± 0.094	0.399 ± 0.109	0.398 ± 0.119	0.347 ± 0.151	0.349 ± 0.154	0.355 ± 0.163
ARTM	0.174 ± 0.046	0.248 ± 0.036	0.339 ± 0.042	0.125 ± 0.050	0.118 ± 0.019	0.139 ± 0.013	0.158 ± 0.041	0.183 ± 0.036	0.239 ± 0.030	0.128 ± 0.042	0.168 ± 0.030	0.226 ± 0.030
SeaNMF	0.884 ± 0.256 ●	0.803 ± 0.166 ●	0.576 ± 0.112	0.932 ± 0.293 ●	0.901 ± 0.283 ●	0.780 ± 0.241 ●	0.968 ± 0.222 ●	0.927 ± 0.219 ●	0.784 ± 0.185 ●	0.951 ± 0.292 ●	0.938 ± 0.293 ●	0.816 ± 0.262 ●
CluWords	0.875 ± 0.039 ●	0.864 ± 0.036 ●	0.856 ± 0.036 ▲	0.929 ± 0.031 ●	0.928 ± 0.029 ●	0.924 ± 0.029 ●	0.914 ± 0.034 ●	0.912 ± 0.033 ●	0.912 ± 0.029 ●	0.918 ± 0.030 ●	0.916 ± 0.026 ●	0.912 ± 0.025 ●
Strategies	**20News**			**Tweets**			**Facebook**			**Uber**		
	5 words	10 words	20 words	5 words	10 words	20 words	5 words	10 words	20 words	5 words	10 words	20 words
FS	0.119 ± 0.056	0.110 ± 0.026	0.110 ± 0.022	0.071 ± 0.054	0.066 ± 0.033	0.078 ± 0.005	0.061 ± 0.065	0.054 ± 0.033	0.050 ± 0.014	0.056 ± 0.043	0.045 ± 0.023	0.048 ± 0.016
BTM	0.244 ± 0.117	0.217 ± 0.089	0.192 ± 0.059	0.142 ± 0.061	0.100 ± 0.026	0.095 ± 0.019	0.137 ± 0.063	0.110 ± 0.036	0.118 ± 0.029	0.093 ± 0.044	0.094 ± 0.036	0.094 ± 0.026
LDA	0.218 ± 0.121	0.196 ± 0.084	0.174 ± 0.063	0.083 ± 0.055	0.060 ± 0.028	0.079 ± 0.020	0.115 ± 0.067	0.085 ± 0.028	0.095 ± 0.023	0.094 ± 0.053	0.083 ± 0.030	0.089 ± 0.012
LTM	0.224 ± 0.134	0.196 ± 0.074	0.179 ± 0.049	0.109 ± 0.060	0.084 ± 0.022	0.093 ± 0.017	0.146 ± 0.079	0.113 ± 0.048	0.119 ± 0.027	0.097 ± 0.065	0.088 ± 0.032	0.091 ± 0.022
GPU-DMM	0.421 ± 0.044	0.477 ± 0.044	0.471 ± 0.031	0.090 ± 0.062	0.081 ± 0.051	0.092 ± 0.046	0.326 ± 0.170	0.313 ± 0.164	0.282 ± 0.162	0.322 ± 0.241	0.275 ± 0.199	0.240 ± 0.142
ETM	0.249 ± 0.109	0.262 ± 0.092	0.243 ± 0.066	0.057 ± 0.044	0.071 ± 0.038	0.092 ± 0.041	0.198 ± 0.090	0.186 ± 0.087	0.171 ± 0.095	0.180 ± 0.077	0.173 ± 0.074	0.165 ± 0.096
ARTM	0.281 ± 0.105	0.235 ± 0.076	0.216 ± 0.062	0.091 ± 0.055	0.068 ± 0.031	0.080 ± 0.025	0.079 ± 0.044	0.091 ± 0.023	0.136 ± 0.021	0.075 ± 0.043	0.091 ± 0.020	0.135 ± 0.018
SeaNMF	0.897 ± 0.247 ●	0.893 ± 0.249 ●	0.891 ± 0.254 ●	0.237 ± 0.183	0.239 ± 0.145	0.195 ± 0.056	0.718 ± 0.410 ●	0.655 ± 0.396 ●	0.546 ± 0.312	0.684 ± 0.434 ●	0.630 ± 0.417 ●	0.522 ± 0.343
CluWords	0.917 ± 0.034 ●	0.913 ± 0.034 ●	0.908 ± 0.034 ●	0.935 ± 0.021 ▲	0.928 ± 0.024 ▲	0.923 ± 0.027 ▲	0.906 ± 0.034 ●	0.902 ± 0.034 ●	0.894 ± 0.034 ▲	0.915 ± 0.029 ●	0.907 ± 0.031 ●	0.902 ± 0.033 ▲
Strategies	**Angrybirds**			**Pinterest**			**InfoVis-Vast**			**ACM**		
	5 words	10 words	20 words	5 words	10 words	20 words	5 words	10 words	20 words	5 words	10 words	20 words
FS	0.053 ± 0.036	0.077 ± 0.033	0.124 ± 0.028	0.102 ± 0.077	0.096 ± 0.050	0.112 ± 0.031	0.049 ± 0.039	0.057 ± 0.026	0.056 ± 0.019	0.148 ± 0.107	0.136 ± 0.050	0.128 ± 0.044
BTM	0.132 ± 0.075	0.154 ± 0.034	0.193 ± 0.040	0.148 ± 0.074	0.144 ± 0.043	0.147 ± 0.032	0.193 ± 0.079	0.170 ± 0.071	0.149 ± 0.051	0.176 ± 0.084	0.146 ± 0.055	0.136 ± 0.051
LDA	0.137 ± 0.065	0.154 ± 0.038	0.190 ± 0.044	0.144 ± 0.062	0.135 ± 0.043	0.147 ± 0.039	0.154 ± 0.075	0.153 ± 0.064	0.139 ± 0.051	0.138 ± 0.062	0.122 ± 0.051	0.117 ± 0.046
LTM	0.117 ± 0.061	0.154 ± 0.041	0.189 ± 0.041	0.145 ± 0.061	0.137 ± 0.051	0.143 ± 0.035	0.182 ± 0.092	0.158 ± 0.058	0.131 ± 0.042	0.173 ± 0.095	0.163 ± 0.074	0.143 ± 0.054
GPU-DMM	0.260 ± 0.173	0.286 ± 0.141	0.301 ± 0.142	0.330 ± 0.192	0.322 ± 0.194	0.278 ± 0.146	0.264 ± 0.155	0.259 ± 0.104	0.207 ± 0.103	0.233 ± 0.101	0.208 ± 0.113	0.189 ± 0.095
ETM	0.366 ± 0.089	0.373 ± 0.079	0.385 ± 0.085	0.358 ± 0.114	0.355 ± 0.109	0.370 ± 0.115	0.304 ± 0.163	0.304 ± 0.157	0.313 ± 0.158	0.266 ± 0.086	0.230 ± 0.052	0.201 ± 0.035
ARTM	0.209 ± 0.066	0.262 ± 0.054	0.337 ± 0.051	0.167 ± 0.060	0.198 ± 0.030	0.264 ± 0.027	0.102 ± 0.095	0.084 ± 0.077	0.076 ± 0.045	0.178 ± 0.088	0.147 ± 0.058	0.149 ± 0.047
SeaNMF	0.964 ± 0.238 ●	0.955 ± 0.222 ●	0.808 ± 0.194 ●	0.836 ± 0.311 ●	0.754 ± 0.278 ●	0.552 ± 0.167	0.861 ± 0.321 ●	0.840 ± 0.332 ●	0.768 ± 0.288	0.843 ± 0.336 ●	0.857 ± 0.337 ●	0.860 ± 0.345 ●
CluWords	0.903 ± 0.041 ●	0.899 ± 0.043 ●	0.897 ± 0.044 ●	0.891 ± 0.034 ●	0.888 ± 0.031 ●	0.883 ± 0.031 ▲	0.998 ± 0.012 ●	0.997 ± 0.012 ●	0.998 ± 0.009 ▲	0.950 ± 0.019 ●	0.945 ± 0.023 ●	0.939 ± 0.028 ●

Table 4.2: Comparing the results achieved by each strategy considering top 5, 10 and 20 words for NPMI.

4.2.2 Effectiveness Results Against the Baselines

We compare our proposed solution against eight state-of-the-art topic modeling strategies considering the twelve reference datasets. In Figure 4.3, our strategy achieves statistically significant gains in terms of the quality of the discovered topics in all 12 datasets, considering the Coherence score. Most baselines cannot get even close, with the best baseline (SeaNMF) being worse than Cluwords by more than 33% when it obtains its best performance (in TripAdvisor), considering the three evaluated topic lengths. These are very strong results as SeaNMF is considered the state-of-the-art in Topic Modeling (besides being a very recent proposal [82]).

In Table 4.2, we contrast the results of CluWords and the reference strategies, considering the NPMI metric. The best results, marked with ▲, are statistically superior to others. Statistical ties are represented with ●. As we can see, our strategy achieves the single best results in 7 out of 36 results, tying with SeaNMF in the other 29 as the **best** method in terms of the quality of the discovered topics, considering the NPMI score. Again, the other baselines' results are far below, reinforcing that SeaNMF is the baseline to be beaten.

Datasets	**Syntatic information**
Whatsapp	10.37 ± 6.21
Angrybirds	10.76 ± 6.11
20News	14.47 ± 6.54
Dropbox	12.83 ± 6.53
InfoVis-Vast	17.57 ± 7.66
Tweets	13.34 ± 6.37
ACM	15.58 ± 7.54
Evernote	14.02 ± 7.05
Pinterest	12.12 ± 5.99
Uber	12.99 ± 7.05
Facebook	12.62 ± 6.78
Tripadvisor	12.92 ± 6.90

Table 4.3: Syntactic information in the CluWords.

Another perspective of the results can be taken when analyzing the standard deviations of the results. The ones obtained by CluWords are considerably smaller than those of SeaNMF. Figure 4.4 allows us to clearly observe the differences between the NPMI deviations, considering topics with ten words. To better quantify this, we performed two variability tests. Equal variances across samples are also called *homogeneity of variance*. Some statistical tests, for instance, the analysis of variance, assume that variances are equal across groups or samples. Levene's test [44] and Bartlett's test [5] can be used to

verify that assumption. Levene's test is less sensitive than Bartlett's test to depart from normality. On the other hand, if the data come indeed from a normal or nearly normal distribution, then Bartlett's test should perform better. We applied both tests because we cannot assume any of the options. In these tests, if the resulting p-value is less than some significance level (e.g., p-value < 0.05), the obtained differences in sample variances are unlikely to have occurred based on random sampling from a population with equal variances (i.e., different variances).

Table 4.4 presents the test for equality of variances concerning the NPMI scores of both, CluWords and SeaNMF. We marked in ▲ the p-values that present statistically significant differences between the variances and use ● when both strategies have the same variance. Table 4.4 shows that in 21 out of 24 tests, the CluWords and SeaNMF have different variances indeed. Tweets are the only dataset where the test showed equivalent variances between the strategies. However, in this dataset, CluWords outperforms SeaNMF considering all three topic lengths Table 4.2). Thus, we can conclude that our CluWords strategy can generate the best semantically cohesive topics, in terms of TFIDF coherence and NPMI, with less variability in NPMI according to Levene's and Bartlett's tests.

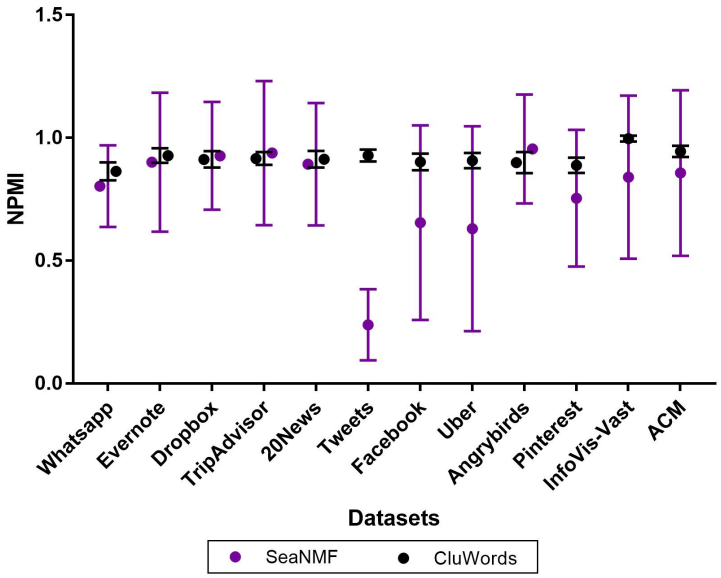


Figure 4.4: Comparison of NPMI scores for Cluwords and SeaNMF strategies considering 10 words.

Datasets	Variance		p-value	
	CluWords	**SeaNMF**	**Levene's test**	**Bartlett's test**
20News	0.0013	0.0644	0.004▲	0.0 ▲
ACM	0.0008	0.0741	0.169●	0.018▲
Angrybirds	0.0020	0.0375	0.002▲	0.014▲
Dropbox	0.0009	0.0343	0.006▲	0.006▲
Evernote	0.0009	0.0582	0.015▲	0.000▲
Facebook	0.0012	0.0971	0.000▲	0.000▲
Infovisvast	0.0001	0.0831	0.000▲	0.000▲
Pinterest	0.0010	0.0278	0.002▲	0.001▲
TripAdvisor	0.0007	0.0687	0.004▲	0.000▲
Tweets	0.0009	0.0032	0.112●	0.138●
Uber	0.0011	0.1179	0.000▲	0.000▲
WhatsApp	0.0013	0.0125	0.001▲	0.000▲

Table 4.4: Test for Equality of Variances considering 20 Words.

4.3 Chapter Summary

Our thorough experimental evaluation (12 datasets, eight baselines, two evaluation metrics, three topic lengths) showed that sometimes by large margins, we outperform the best (state-of-the-art) methods for Topic Modeling known in the literature, with a much smaller variability in terms of the quality of the produced topics. In addition, we perform a test for equality of variances concerning the NPMI scores of the best methods, CluWords and SeaNMF. The results show that both methods have different variances. Thus, we can conclude that the CluWords can generate the best semantically cohesive topics.

Chapter 5

CluWords for Hierarchical Topic Modeling

In this chapter, we introduce a new HTM approach, called **CluHTM**, which combines the CluWords representation with a non-probabilistic factorization method. This new approach allows the execution of the CluWords with a non-probabilistic factorization method in an automated way. In Section 5.1, we describe the [29] stability method, which allows CluHTM automation. The CluHTM method explores the exact configuration of the CluWords described in chapter 3. The intuition of the CluHTM method consists of smoothing the limitation of non-probabilistic methods in exploring local information by using the representation of CluWords to explore *global* semantic information into the hierarchical topic tree creation. In Section 5.2, we present the construction of the CluHTM approach. Finally, we present the experimental setup and evaluation, respectively in Sections 5.3, and 5.4.

5.1 Background: Stability Measure

The Stability measure is motivated by the term-centering approach generally taken in topic modeling strategies, where topics are usually summarized as a truncated set of top words [29].

The intuition behind this strategy is, given some K topics, to measure whether running multiple random samplings for a topic modeling strategy results in Stability in terms of p top words extracted from the topics. Given a range of topics $[\mathcal{K}_{min}, \mathcal{K}_{max}]$, and some topic modeling strategy (in our case, the Non-negative Factorization Matrix method), the strategy proceeds as follows. First, it learns a topic model considering the complete data set representation $\mathcal{D}$, which will be used as a reference point ($\mathcal{W}_{\mathcal{D}}$) for analyzing the Stability afforded by the K topics. Note that the p top words represent each topic. Subsequently, $\mathcal{S}$ samples of the data are randomly drawn from $\mathcal{D}$ without

replacement, forming a subset of $\mathcal{D}'$ documents. Then, $|\mathcal{S}|$ topic models are generated, one for each subsampling ($\mathcal{W}_{\mathcal{S}_i}$).

To measure the quality of K topics, the Stability computes the mean agreement among each pair of ($\mathcal{W}_{\mathcal{D}}$, $\mathcal{W}_{\mathcal{S}_i}$). The goal is to find the best match between the p top words of the compared topics. The agreement is defined as $agree(\mathcal{W}_x, \mathcal{W}_y) = \frac{1}{p}\sum_{i=1}^{p} AJ(w_{xi}, \rho(w_{xi}))$, where $AJ(\cdot)$ is the average Jaccard coefficient used to compare the similarity among the words w and $\rho(\cdot)$ is the optimal permutation of the words in $\mathcal{W}_{\mathcal{S}_i}$ that can be found in $\mathbb{O}(p^3)$ time by solving the minimal weight bipartite matching problem using the Hungarian method [40].

5.2 CluHTM

CluHTM is an iterative method able to automatically define the best number of topics in each hierarchy, given a range of a possible number of topics $[\mathcal{K}_{min}, \mathcal{K}_{max}]$. CluHTM explores Cluwords and Non-negative Matrix Factorization (NMF) [42], one of the main non-probabilistic strategies. Finally, the Stability method (described in Section 5.1) is used to select NMF k parameters (a.k.a number of topics).

CluHTM has five inputs (Algorithm 1), (i) $\mathcal{D}_{max}$ corresponds to the depth down to which we want to extract the hierarchical structure. (ii) $\mathcal{K}_{min}$ and $\mathcal{K}_{max}$ control the range of some topics, such range will be used in all levels of the hierarchy; (iii) $\mathcal{T}$ is the input text data, and (iv) $\mathcal{W}$ is the "pre-trained" word embedding vector space used in the CluWords generation. The output is the hierarchical structure $\mathcal{H}$ of p top words for each topic.

The method starts by getting the root topic (lines 2-3 of Algorithm 1), which is composed of all documents in $\mathcal{T}$. Since the method is iterative, each iteration is controlled by a queue schema to build a hierarchical structure. Thus, at each iteration (line 3), the algorithm produces the CluWords representation for the documents $\in \mathcal{T}'$ (line 5), chooses the number of topics, exploiting the Stability measure (line 6), and runs the NMF method (line 7) to extract the p words for each topic in $\mathcal{O}$ (line 8). Then, in the loop of line 9, each topic is stored in the queue, and the respective documents of each topic.

In summary, the CluHTM (for now we are going to call f-CluHTM) exploits *global* semantic information (captured by CluWords) within *local* factorizations, limited by a stability criterion that defines the "shape" of the hierarchical structure. Though simple (and original), combining these ideas is extremely powerful for solving the HTM task, as we will see next.

Algorithm 1: f-CluHTM

Input: $\mathcal{D}_{max}$ - Hierarchy Depth;
$\mathcal{K}_{min}$ - Number of minimum topics;
$\mathcal{K}_{max}$ - Number of maximum topics;
$\mathcal{T}$ - Term-frequency representation;
$\mathcal{W}$ - Word embedding vectors $\in \mathcal{T}$;
Output: $\mathcal{H}$ - Hierarchical Structure;
parent - Parent dependency

```
1  parent ← −1;
2  queue.push(0, T);
3  while queue ≠ ∅ do
4      depth, T' ← queue.pop();
5      Clu ← GenerateCluwords(T', W);
6      K ← Stability(K_min, K_max, Clu)
7      O ← NMF(Clu, K)
8      topics ← ExtractTopics(O)
9      foreach topic ∈ topics do
10         parent ← parent ∪ topic;
11         H ← H ∪ topic;
12         if depth + 1 ≤ D_max then
13             T' ← ExtractDocs(topic);
14             queue.push(depth + 1, T')
15 return H, parent
```

5.2.1 The c-CluHTM solution

f-CluHTM exploits FastText embeddings in the Clustering step, as described in Section 5.2. This step receives a static representation of embeddings: a vector space containing only a single vector representation per word. c-CLUHTM extends f-CluHTM to receive a contextual embedding representation (i.e., more than one representation per word) by transforming these contextual representations into a static one.

In our solution, we exploit BERT's contextual embeddings. Figure 5.1 summarizes the BERT's embeddings for words in an input text d. Each word w_i has an embedding representation in the output of each encoder (twelve layers), and each embedding has 768 dimensions. Different documents containing the word w_i may have different vectors for w_i because w_i may have different neighbors in each document. Given a layer, BERT generates n vectors for the same word, where n is the occurrence of the word in the collection.

We exploit three pooling operations to transform BERT's contextual embeddings into a static embedding for the Word Clustering step. We will use the same notation for all three operations described below. Let $\mathbb{D} = \{d_1, d_2, \cdots, d_{|\mathbb{D}|}\}$ be defined as the set of documents, where $|\mathbb{D}|$ is the number of documents of the collection. Let $\mathbb{V} = \{w_1, w_2, \cdots, w_{|\mathbb{V}|}\}$ be the vocabulary of words in the collections, where $|\mathbb{V}|$ is the number

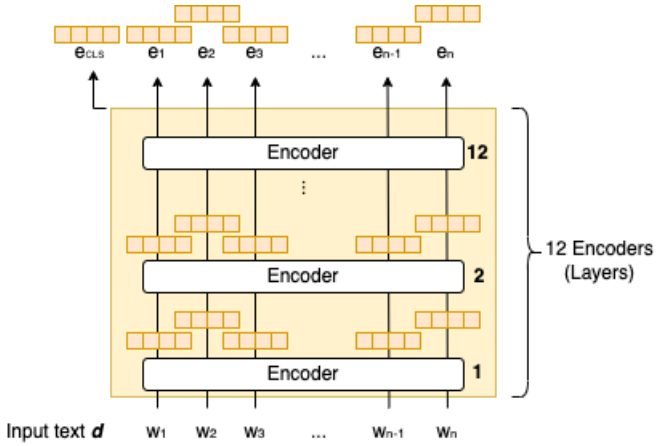


Figure 5.1: BERT's encoders structure given an input text d.

of words in the vocabulary. Each word $w_i \in \mathbb{V}$ has an embedding representation $\overrightarrow{w}_i \in \mathbb{K}_{i,x}$, where $\mathbb{K}_{i,x}$ is the set of each occurrence of w_i in $\mathbb{D}$, and $x = \{1, 2, 3, \cdots, 12\}$ is the embedding layer of BERT.

The max-pooling (Equation 5.1) retrieves the maximum embedding $\overrightarrow{w}_i \in \mathbb{K}_{i,12}$ for the word $w_i \in \mathbb{V}$. In other words, the max-pooling operation takes the maximum value of each embedding dimension. The Average pooling (Equation 5.2) is the average operation of each embedding representation $\overrightarrow{w}_i \in \mathbb{K}_{i,x'}$, where $x' = \{9, 10, 11, 12\}$. And finally, the Concatenation pooling (Equation 5.3) creates an embedding $\overrightarrow{wC}_i$ concatenating the embedding representations of $\overrightarrow{w}_i \in \mathbb{K}_{i,x'}$, then, it computes the mean of $\overrightarrow{wC}_i \in \mathbb{K}_i$, where $\mathbb{K}_i$ is the occurrences of w_i in $\mathbb{D}$.

$$\overrightarrow{wM}_i = Max(\overrightarrow{w}_i), \text{ where } \overrightarrow{w}_i \in \mathbb{K}_{i,12} \tag{5.1}$$

$$\overrightarrow{\mu w}_i = \frac{\sum_j^{\mathbb{K}_{i,12}} \overrightarrow{w}_i}{|\mathbb{K}_{i,12}|} \tag{5.2}$$

$$\overrightarrow{\mu wC}_i = \frac{\sum_j^{\mathbb{K}_i} \overrightarrow{w}_{i,9} || \overrightarrow{w}_{i,10} || \overrightarrow{w}_{i,11} || \overrightarrow{w}_{i,12}}{|\mathbb{K}_i|}, \text{ where } \overrightarrow{w}_{i,9} \in \mathbb{K}_{i,9}, \overrightarrow{w}_{i,10} \in \mathbb{K}_{i,10}, \overrightarrow{w}_{i,11} \in \mathbb{K}_{i,11}, \overrightarrow{w}_{i,12} \in \mathbb{K}_{i,12} \tag{5.3}$$

In summary, the main difference between f-CluHTM and c-CluHTM is using different word embeddings. f-CluHTM exploits the pre-trained fastText, whereas the c-CluHTM exploits the BERT's contextual embeddings pooled in a single representation using three different strategies – Max, Average, and, Concatenation of word vectors.

5.3 Experimental Setup

The primary goal of our solution is to effectively perform hierarchical topic modeling so that more coherent topics can be extracted. To evaluate topic model coherence, we consider 12 real-world datasets as a reference, previously described in Chapter 3. All of them were obtained from previous works in the literature. For all datasets, we performed stopwords removal (using the standard SMART list) and removed words such as adverbs, using the VADER lexicon dictionary [37], as the vast majority of the essential words for identifying topics are nouns and verbs. These procedures improved both the efficiency and effectiveness of all analyzed strategies. Table 5.1 provides a summary of the reference datasets, reporting the number of features (words) and documents, as well as the mean number of words per document (density) and the corresponding references.

Dataset	#Feat	#Doc	Density
Angrybirds [32]	1,903	1,428	7.135
Dropbox [32]	2,430	1,909	9.501
Evernote [32]	6,307	8,273	11.002
InfoVis-Vast	6,104	909	86.215
Pinterest [32]	2,174	3,168	4.478
TripAdvisor [32]	3,152	2,816	8.532
Tweets [50]	8,029	12,030	4.450
WhatsApp [32]	1,777	2,956	3.103
20NewsGroup	29,842	15,411	76.408
ACM [97]	16,811	22,384	30.428
Uber	5,517	11,541	7.868
Facebook	5,168	12,297	6.427

Table 5.1: Dataset characteristics

We compare the HTM strategies using representative topic quality metrics in the literature [68, 69]. We consider two classes of topic quality metrics based on three criteria: (a) coherence and (b) mutual information. In our evaluation, we consider three topic lengths (5, 10, and 20 words) for each parameter since different lengths may bring different challenges.

Regarding the metrics, *coherence* captures the easiness of interpretation by co-occurrence. Words that frequently co-occur in similar contexts in a corpus are easier to correlate since they usually define a more well-defined "concept" or "topic". We employ an improved version of regular coherence [68], called Coherence, defined as

$$c(t, W_t) = \sum_{w_1, w_2 \in W_t} log \frac{d(w_1, w_2) + \varepsilon}{d(w_1)}, \tag{5.4}$$

where $d(w1)$ denotes the number of occurrences of $w1$, $d(w1, w2)$ is the number of documents that contain both $w1$ and $w2$ together, and ε is a smoothing factor used for preventing $log(0)$.

Another class of topic quality metrics is based on the notion of *pairwise pointwise mutual information (PMI)* between the top words in a topic. It captures how much one "gains" in the information given the occurrence of the other word, considering dependencies between words. Following a recent work [68], we here compute a *normalized version of PMI* (NPMI) where, for a given ordered set of top words $W_t = (w_1, \cdots, w_N)$ in a topic:

$$NPMI_t = \sum_{i<j} \frac{log \frac{p(w_i,w_j)}{p(w_i)p(w_j)}}{-log\ p(w_i, w_j)}. \tag{5.5}$$

We compare our approach described in Section 5.2 with four hierarchical topic model strategies described as follows.

hLDA: is an expansion [30] of LDA, and it is considered state-of-the-art in Hierarchical Topic Modeling. The method exploits the text Dirichlet distribution with the nested Chinese Restaurant Process (NCRP) to generate a hierarchical tree structure. The NCRP needs the tree level and a γ parameter. At each tree node, a document can belong to a path or create a new tree path with probability controlled by γ.

HPAM: is an extension of *Pachinko Allocation* (PAM) [51]. In PAM, documents are a mix of distributions over an individual topic set, using a directed acyclic graph to represent the co-occurrences of topics. Each node in such a graph represents a *Dirichlet* distribution. At the highest level of PAM, there is only a single node, where the lowest levels represent a distribution between nodes of the next higher level.

BERTopic: is a hybrid solution that exploits BERT embeddings to represent the documents and exploits a hierarchical clustering approach to build hierarchical topics. In more detail, the method is composed of five steps: (i) Embedded Documents: this step exploits the information provided by a transformer approach (i.e., BERT) to get the embedding vector (i.e., the CLS embedding) of the documents; (ii) Dimensionality reduction: this step exploits PCA or UMAP to reduce the dimension of the document embedding, (iii) Cluster Documents: this step exploits the HDBSCAN method to build the clusters of documents; (iv) Bag-of-Words (Bow): applies a BoW representation in a cluster-level to have a better representation of the words that occur in the clusters; (v) Topic Representation: exploits the c-TFIDF approach over the BoW representation of the previous step to get the topic representation based on words that (co-)occur in the same clusters. This method is capable of building hierarchies of topics.

hARTM: is a Non-Bayesian method hierarchical version of Additive Regularization of Topic Models (ARTM). This method exploits regularization to mitigate problems posed by the LDA-based methods. Bayesian generative models, like LDA, assumes that topic distributions over words and document distributions over topics are generated from prior Dirichlet distributions. This conflicts with two natural practical realities due to sparsity: (i) topics with zero probabilities in a document and (ii) words with zero probabilities in a topic. The authors mention that regularization reduces a potentially infinite set of solutions and helps to select a better one.

For the input parameters of CluHTM (Algorithm 1), we set $\mathcal{K}_{min} = 5$, $\mathcal{K}_{max}$=25, $\mathcal{R} = 10$ and $\mathcal{D}_{max} = 3$. We define $\mathcal{K}_{min}$ through empirical experiments, and the $\mathcal{K}_{max}$ was defined according to the number of topics exploited in [95]. We adopt the parameters suggested by their own works for the baseline methods. We assess the statistical significance of our results employing a paired t-test with 95% confidence and Holm-Bonferroni correction to account for multiple tests.

5.4 Experimental Results

We compare CluHTM variants against state-of-the-art HTM methods considering the twelve reference datasets. Three hierarchical levels for each strategy are used in this comparison.

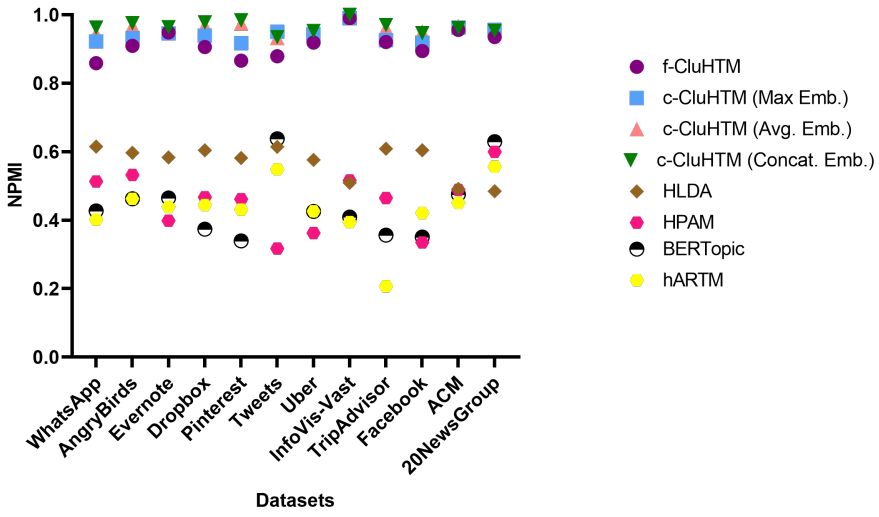


Figure 5.2: NPMI Topic Evaluation. c-CluHTM (Avg. emb.) and c-CluHTM (Concat. emb.) produce the best results in all evaluated datasets.

Considering the NPMI results, we can observe in Figure 5.2 that the c-CluHTM variants achieved the best results in all evaluated datasets. The three c-CluHTM variants (Max, Avg, and Concat) tied with each other in all but two datasets (Pinterest and TripAdvisor), in which c-CluHTM (Max emb.) was a bit worse than the other two variants. c-CluHTM (Concat. emb.) achieves the best NPMI results in seven out of 12 datasets, tying with f-CluHTM in the following datasets – Evernote, Uber, InfoVis-Vast, ACM, and 20News. Regardless of static or contextual embeddings, the NPMI results show the superiority of all CluHTM solutions over HPAM, HLDA, BERTopic, and hARMT. Indeed, for some datasets, such as InfoVis-Vas, ACM, and 20News, the c-CluHTM results are remarkable, reaching NPMI scores close to 1.0 (the maximum).

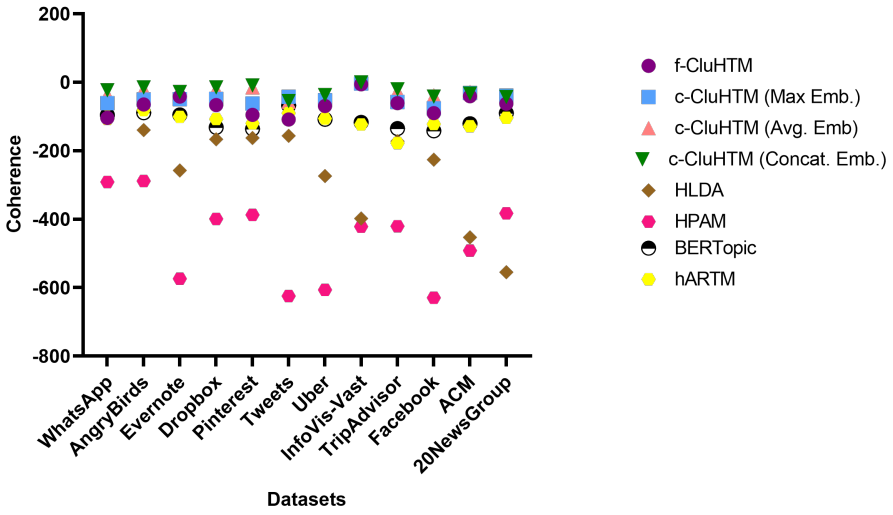


Figure 5.3: Coherence Topic Evaluation. c-CluHTM (Max emb.), c-CluHTM (Avg. emb.), and c-CluHTM (Concat. emb.) are the best solutions achieving the best results in all the datasets evaluated.

Figure 5.3 shows the results for the Coherence metrics. Similar behavior for the c-CluHTM variants can be observed as in the case of NPMI. The three c-CluHTM variants achieved the best results in all evaluated datasets. The c-CluHTM (Concat. emb.) produced the highest results, outperforming f-CluHTM in three out of 12 datasets, while the HLDA solution tied in the WhatsApp dataset.

Contrasting NPMI and Coherence metrics, we can see that the metrics captured different aspects of the topical quality. Both focus on measuring the quality of the topics based on the information shared among the words that compose the latent topics. Coherence has a higher sensitivity, generating variability between the topic scores. This is why, for example, for the AngryBirds dataset, c-CluHTM (Concat. emb.) is statistically equivalent to f-CluHTM. In addition, Coherence does not have an upper bound limit, which makes each topic evaluation unique. Despite these differences, both NPMI and Coherence are consistent regarding the best methods to build topics.

5.5 Extending the Hierarchical Topic Modeling Evaluation

We start by introducing the concept of Topology in the context of HTM, the basis of the explanation for the proposed metrics. We then present the proposed metrics for evaluating topics in HTM.

The proposed evaluation metrics require two *consecutive* levels of a (hierarchical) topology to compute the quality of the topics. Figure 5.4 illustrates two levels of a topic hierarchy built by an HTM method m. In the first level of the hierarchy, method m generates two topics, the blue square and the blue triangle. Each topic is represented by top t words, where t is defined as an input parameter of method m. In the second level, m builds eight topics, four topics are derived from the blue square, represented as the pink squares, while the last four topics are derived from the blue triangle, illustrated as the four pink triangles. Note that each of these topics is also represented as top t words, which are used to measure the quality of the topics. Topology t_i is defined by the set of topics originating from topic i, including itself. In Figure 5.4, the topologies demarcated in yellow illustrate topologies t_1 and t_2 of topics 1 (blue square) and 2 (blue triangle), respectively. The topics illustrated as squares belong to the topology t_1 since the pink squares were derived from the blue square. In contrast, the triangles belong to topology t_2 as the blue triangle is the topic at a hierarchy level above the pink triangles. It is important to stress that the topologies must have more than one (sub-)topic in the second and lower levels. So, hierarchical structures with only one topic at each level are not suitable for this type of evaluation.

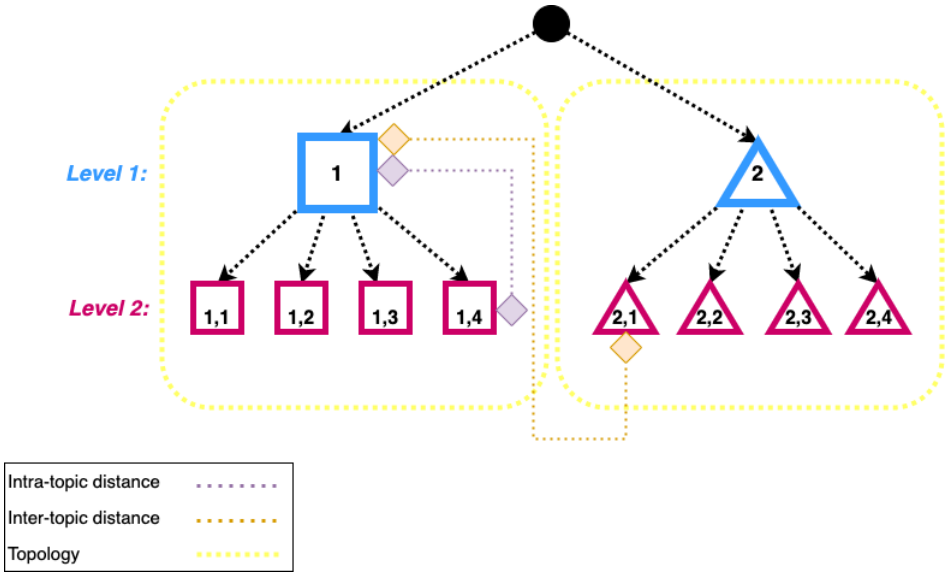


Figure 5.4: Example of Topic's topology built by an HTM method.

Now we can define our proposed quality evaluation metrics, which exploit concepts from clustering algorithms. Topics are seen as clusters composed of the top t words. These word clusters (topics) can be evaluated based on two perspectives: (i) *intra-group distances*, which measures the distance between objects of the same cluster, and (ii) *inter-group distances*, which measures the distance of objects from distinct clusters. We adapt these two concepts of *intra* and *inter-topic* distances to analyze topic information in a hierarchy. *Intra-topic distances* measure correlations between topics in the same hierarchy, that is, between topics in different levels of the same topology, as described in Section 5.5.1. *Inter-topic distances* measure correlations between topics from different topologies (described in Section 5.5.2).

Based on these two new concepts, we design two new topic quality metrics to assess topical quality, considering two aspects: (i) *Topic topological consistency (or redundancy)* and (ii) *Semantic hierarchical structure*. Before presenting the proposed metrics, let's introduce some notations we will use throughout this Section. Let $t_{i,j} \in \mathbb{T}_j$ be a latent topic built by the HTM method m, and $t_{i,j}$ be composed by the top κ words that best describe the topic $t_{i,j}$. $\mathbb{T}_j = \{t_1, t_2, t_3, \cdots, t_{|\mathbb{T}_j|}\}$ is the set of latent topics of the topology $\tau_j \in \tau$. The symbol $\tau = \{\tau_1, \tau_2, \cdots \tau_N\}$ defines the topologies created by the HTM method m. The two proposed quality metrics are described as follows:

Uniqueness: This metric evaluates whether HTM strategies can produce diverse topics without repetition of words. Uniqueness assesses the consistency of topological relationships regarding the information built by the strategy. If a strategy builds a hierarchy of similar topics without diversity between the words used to build the topics, the strategy cannot specialize within that topology. This metric is a straightforward computation of compound word singularity (uniqueness) in topics. *Uniqueness* is defined by the function $unique(t_{i,*}, t_{j,*})$, which returns the gross amount of unique words that compose both topics $t_{i,*}$ and $t_{j,*}$, where $*$ represent any topic in the topologies τ. The gross amount of unique words is divided by the total number of words regarding both topics ($t_{i,*}$ and $t_{j,*}$). Thus, Uniqueness captures the ratio of singularity in terms of words chosen to represent topics. The higher the value of this ratio, the more syntactically distinct the topics are.

$$unique(t_{i,*}, t_{j,*}) = \frac{(t_{i,*} \cup t_{j,*}) - (t_{i,*} \cap t_{j,*})}{(t_{i,*} \cup t_{j,*})} \tag{5.6}$$

The Uniqueness metric is defined in Equation 5.7, where $\omega(\cdot)$ computes the word count.

$$uniqueness(t_{i,*}, t_{j,*}) = \frac{unique(t_{i,*}, t_{j,*})}{\omega(t_{i,*}) + \omega(t_{j,*})} \tag{5.7}$$

SHS: This metric captures the semantic information in each topic. It uses word vectors (embeddings) that compose the topics to evaluate topic quality in terms of semantic

similarity. More specifically, SHS (Equation 5.8) is a variant of the cosine distance among the embedding representation of topics $t_{i,*}$ and $t_{j,*}$. The idea is to assess the diversity and consistency of the topological hierarchy τ created by HTM strategies. From this semantic perspective, the closer the word vectors are to the topics in the same topology, the more consistent the hierarchy is.

$$SHS(t_{i,*}, t_{j,*}) = 1.0 - \frac{\sqrt{\sum \zeta(t_{i,*}) * \zeta(t_{j,*})}}{\sqrt{\sum \zeta(t_{i,*})^2} * \sqrt{\sum \zeta(t_{j,*})^2}} \tag{5.8}$$

where $w_{i,x}$ and $w_{i,y}$ are the embedding representations for words $w_{i,x} \in t_x \in \mathbb{T}_*$ and $w_{i,y} \in t_y \in \mathbb{T}_*$. $*$ represents any topic in the topologies τ.

The embedding representation of the topic t is measured by the average pair-wise cosine distance of their top κ words. $\zeta(t)$ illustrates the mean embedding representation of the topic t.

$$\zeta(t) = \frac{\sum_{w_i \in t} \overrightarrow{w_i}}{\kappa} \tag{5.9}$$

The two proposed metrics are defined in the two mentioned contexts: by considering (i) Intra and (ii) Inter-topic distances, as described next.

5.5.1 Metrics Considering Intra-topic Distances

The *Intra-topic distances* measure correlations among topics $\mathbb{T}_z$ within the same topology $\tau_z \in \tau$. Equation 5.10 presents the intra-topic distance measured as the average of the $\sigma(\tau_i, \tau_j)$ between topics $\mathbb{T}_z \in \tau_z$. Note that Equation 5.10 is a template to build the two proposed quality metrics in terms of intra-topic distances. The function $\sigma(\cdot)$ is a template that can be replaced for Uniqueness or SHS.

$$intra = \frac{\sum_{t_i \in \mathbb{T}_z} \sum_{t_j \in \mathbb{T}_z, t_i \neq t_j} \sigma\left(t_{i,z}, t_{j,z}\right)}{|\mathbb{T}_z|} \tag{5.10}$$

where $|\tau|$ corresponds to the total number of topologies.

Uniqueness: In this metric, the distance function in Equation 5.10 is replaced by the $uniqueness(t_i, t_j)$ function. Uniqueness in terms of Intra-topic distances (DU_{intra}) is defined as:

$$DU_{intra} = \frac{\sum_{t_i \in \mathbb{T}_z} \sum_{t_j \in \mathbb{T}_z, t_i \neq t_j} uniqueness(t_i, t_j)}{|\mathbb{T}_z|} \tag{5.11}$$

where t_i e t_j belongs to the same topology and same level of the hierarchy.

SHS: To capture semantics, we use the SHS (Equation 5.8) of the embedding centroids measured by Equation 5.9. Thus, the SHS Intra-topic distance (DC_{intra}) is defined as:

$$DC_{intra}(\tau_k) = \frac{\sum_{t_i \in \mathbb{T}_z} \sum_{t_j \in \mathbb{T}_z, t_i \neq t_j} SHS(t_i, t_j)}{|\mathbb{T}_z|} \tag{5.12}$$

5.5.2 Metrics Considering Inter-topic Distances

Inter-topic distances measure correlations among topics from different topologies. The goal is to assess how far two different topologies τ_z and τ_y are. Equation 5.13 refers to the *Inter-topic distance* measured by the distance among the topics of a topology $\mathbb{T}_z \in \tau_z$ to distinct topologies $\mathbb{T}_y \in \tau_z$. Again, the function $\sigma(\cdot)$ is a template that can be replaced for Uniqueness or SHS metrics. We present the instantiation of two metrics that consider Inter-topic distances.

$$inter = \frac{\sum_{t_i \in \mathbb{T}_z} \sum_{t_j \in \mathbb{T}_y} \sigma(t_i, t_j)}{|\mathbb{T}_z| * |\mathbb{T}_y|} \tag{5.13}$$

where $|\mathbb{T}_z| * |\mathbb{T}_y|$ corresponds to the total number of topics evaluated.

Uniqueness: The function $\sigma(\cdot)$ (in Equation 5.13) is instantiated using the function $uniqueness(t_i, t_j)$ to define DU_{inter} as:

$$DU_{inter} = \frac{\sum_{t_i \in \mathbb{T}_z} \sum_{t_j \in \mathbb{T}_y} uniqueness(t_i, t_j)}{|\mathbb{T}_z| * |\mathbb{T}_y|} \tag{5.14}$$

SHS: The Cosine distance among the centroids (Equations 5.8 and 5.9) is exploited Equation 5.13. The semantics in terms of SHS Inter-topic distance metrics are defined (DC_{inter}) as:

$$DC_{inter} = \frac{\sum_{t_i \in \mathbb{T}_z} \sum_{t_j \in \mathbb{T}_y} SHS(t_i, t_j)}{|\mathbb{T}_z| * |\mathbb{T}_y|} \tag{5.15}$$

5.6 Experimental Evaluation of the Proposed HTM Quality Metrics

We consider for this evaluation c-CluHTM (Concat. emb.) - the best c-CluHTM variant in Section 5.4, f-CluHTM, HLDA, HPAM, and hARTM. BERTopic has a different way of building the topologies (as mentioned in Section 2.4). Since it did perform among the best methods in the previous evaluation, we did not consider it here. Regarding hARTM, it can only generate hierarchical structures for the ACM and 20NewsGroup, the largest datasets considered in our evaluation. This happens because hARTM requires a lot of data to associate a probability distribution with latent topics over a hierarchical structure. We kept this method in our new evaluation, comparing it against the other methods only in ACM and 20NewsGroup.

We assess the statistical significance of the results with a t-test with Holm-Bonferroni correction with 95% confidence. This test assures that the best results, marked with a green triangle (▲), are statistically superior to all others. Statistical ties are represented as a yellow dot (●).

5.6.1 Evaluation in terms of Uniqueness

Table 5.2 presents the *intra topic distance* in terms of the Uniqueness of topics for five and ten words. Contrasting these two scenarios, we can observe that increasing the number of words representing the latent topics increases the sensitivity of the Uniqueness score. We can also observe that most methods achieved a statistical tie in most datasets, but HLDA, performed poorly in some datasets. Focusing on the results of HLDA, we can observe that the number of words changed the quality of the method regarding the duplicity of words in the latent topics. This means this method starts to add duplicated words when it increases the number of words composing the latent topics. This interesting phenomenon can be captured by Uniqueness but not by traditional metrics (NPMI and Coherence).

Table 5.3 presents some topics generated by HLDA to show the behavior captured by the Uniqueness metric in the WhatsApp dataset, considering ten words. We can observe in the Table that there are several duplicated subtopics (in **bold**) in the topology with basically the same words. This justifies the low Uniqueness score obtained for this method according to Equation 5.11. In contrast, Table 5.4 shows some latent topics

Table 5.2: Uniqueness Intra Topic Analysis.

Datasets	5 Words					10 Words				
	c-CluHTM (Concat. Emb)	f-CluHTM	HLDA	HPAM	hARTM	c-CluHTM (Concat. Emb)	f-CluHTM	HLDA	HPAM	hARTM
WhatsApp	**0.9996**	**0.9984**	0.6826	**0.9947**	-	**0.9973**	**0.9983**	0.6041	**0.9914**	-
AngryBirds	**0.9983**	**0.9995**	0.8896	**0.9613**	-	**0.9949**	**0.9989**	0.8115	**0.9722**	-
Evernote	**1.0000**	**1.0000**	**0.9501**	**0.9525**	-	**1.0000**	**1.0000**	0.8934	**0.9551**	-
Dropbox	**0.9989**	**0.9958**	**0.9309**	**0.9784**	-	**0.9979**	**0.9927**	0.8748	**0.9827**	-
Pinterest	**0.9989**	**0.9979**	**0.9456**	**0.9804**	-	**0.9964**	**0.9958**	0.8690	**0.9844**	-
Tweets	**1.0000**	**0.9997**	**0.9118**	**0.9982**	-	**1.0000**	**0.9995**	0.7871	**0.9979**	
Uber	**1.0000**	**1.0000**	**0.9653**	**0.9247**	-	**0.9998**	**1.0000**	0.8990	**0.9426**	-
InfoVis-Vast	**0.9943**	**0.9996**	**1.0000**	**0.9800**	-	**0.9926**	**0.9976**	**0.9964**	**0.9792**	-
TripAdvisor	**0.9980**	**0.9999**	**0.9227**	**0.9676**	-	**0.9985**	**0.9998**	0.8457	**0.9710**	-
Facebook	**0.9995**	**0.9947**	0.9222	**0.9827**	-	**0.9989**	**0.9961**	0.8243	**0.9790**	
ACM	**1.0000**	**1.0000**	**0.9945**	**0.9838**	**0.9875**	**0.9988**	**1.0000**	**0.9771**	**0.9818**	**0.9865**
20NewsGroup	**0.9934**	**1.0000**	**0.9971**	**0.9316**	**0.9800**	**0.9913**	**0.9975**	**0.9971**	**0.9361**	**0.9800**

generated by the c-CluHTM (Concat. emb.). We can observe that there are no duplicated topics, justifying the high score of Uniqueness.

Traditional metrics assess the quality of the topics independently, and the final score is the average of the evaluated topics in isolation. These measures do not capture relationships among the subtopics based on their shared words, unable to capture phenomena such as duplicated sub-topics illustrated in Table 5.3. In the example shown in Table 5.3, the subtopic "*community region relative shift expanding bye news brought displays seamlessly*" is repeated five times and, for the sake of calculation of NPMI and Coherence, this duplicated topic is also considered five times in the final score. This duplication will unavoidably cause distortions in the metric's final value.

Table 5.4 shows some topics built by c-CluHTM (Concat. emb.) in contrast the topics built by HLDA. We can observe that the c-CluHTM (Concat. emb.) does not include duplications in the topics. It generated topics correlating meaning/semantics (underlined words) and an outlier subtopic (marked in italics).

Table 5.7 shows the Uniqueness results in terms of *inter topic distance*, regarding five and ten words. We observe a similar behavior as in Table 5.2 – most methods are tied against each other, and the results for ten words show a decrease in the HLDA performance.

We performed a comprehensive manual analysis of the generated topics by each method, and the only duplicated topics were built by HLDA. This showed that the Uniqueness method works, and most importantly, this method is capable of revealing issues regarding the generated topics. Indeed Uniqueness can also be seen as a bias indicator for other evaluation metrics, such as NPMI and Coherence. For instance, in WhatsApp, the latent topic built by HLDA, shown in Table 5.3 "community region relative shift expanding bye news brought seamlessly displays." has a score of around 0.58 for NPMI, and the average score of NPMI for HLDA in the WhatsApp dataset is around 0.61 (standard deviation of 0.03). This suggests that this latent topic biases the final method's score.

Table 5.3: Example of topic topology built for the WhatsApp dataset using HLDA.

Main topic	**Subtopics**
ultimate hour tool verification received community region relative shift expanding	bug community region relative shift expanding bye news brought seamlessly
	code constant reliable community region relative shift expanding bye news
	community region relative shift expanding bye news brought seamlessly displays
	community region relative shift expanding bye news brought seamlessly displays
access loads talk androids community region relative shift expanding bye	community region relative shift expanding bye news brought seamlessly displays
	documents service community region relative shift expanding bye news brought
	community region relative shift expanding bye news brought seamlessly displays
	community region relative shift expanding bye news brought seamlessly displays
	community region relative shift expanding bye news brought seamlessly displays

Table 5.4: Example of topic topology built for the WhatsApp dataset using c-CluHTM (Concat. emb.).

Main topic	**Subtopics**
displayed requested opens releases blinking managed generated alternatives designed issues	blank activate prompt breach insert shift blinking center react releases
	glaring generated revelations shuffled democratic tempo expects arrived hooked listened
	format speak recorder sound picture texts bubble wall papers compression
	entering custom internal usable default locally lightweight ubiquitous utilizing implemented
usable windows default virus implemented pack operating offer development bugs	massive cache stored storage bean menu prompt feed screen rights
	beats hip download label viral annual exceed chill compression recorder
	compulsory unable unknown fulfilling possibly including preferred enhanced tempo entered
	react breach prompt loads implemented enhance factory custom reset releases
	volleyball functions medicine mice ken implemented providers carrier nickel japan

5.6.2 Evaluation in terms of SHS

We present *intra* and *inter topic distance* analyses, in terms of semantic information, as described in Section 5.5. As mentioned, this topic evaluation exploits *word embedding* vectors to capture the semantic relationship among words through cosine distance. Table 5.8 presents the results of *intra topic distances* for five and ten words. Observing the results for five words, c-CluHTM (Concat. emb.) and f-CluHTM presented the best results, f-CluHTM achieved the best results in two datasets (Tweets and Facebook), meaning that the topics built by this method, considering the same topology, have higher

Table 5.5: Example of topic topology built for the Facebook dataset using c-CluHTM (Concat. emb.). The underline words show some examples of words that have semantic relationships.

Main Topic	**Subtopics**
cellular broadband micro fiber android proxy windows hardware wireless mobile	intended received reported actual noticed accompanied listed previously permanent written
	proxy worm port hosting authentication gateway windows administrators plug interfaces
	subway troll nexus toad sonata bike dong blackberry cord brother.
	mega bulk unit micro consumption hardware floppy tank cal chi
	pornographic stills recordings targeted photographs videos images artwork photography definition

Table 5.6: Example of topic topology built for the Facebook dataset using f-CluHTM. The underline words show some examples of words that have semantic relationships.

Main Topic	**Subtopics**
handset cellphone helpline telephone mobiles blackberry cell bugged bugging battery	helpline timer tel mobiles clock handset blackberry bugging cellphone cell
	ELS cycle restarts timer rewind pace calls shutdown multiples retrying
	reinstalled reinstalls reinstalling autostart unpin reinstall installs tamper installing liquidate
	listening typing replying phoned talking chatting replies busy contacting speaking
	queries threads replies responses requests messages comments notices notifications remarks

Table 5.7: Uniqueness Inter Topic Analysis.

Datasets	**5 Words**					**10 Words**				
	c-CluHTM (Concat. Emb)	**f-CluHTM**	**HLDA**	**HPAM**	**hARTM**	**c-CluHTM (Concat. Emb)**	**f-CluHTM**	**HLDA**	**HPAM**	**hARTM**
WhatsApp	**0.9937** •	**0.9969** •	0.6889	**0.9696** •	-	**0.9888** •	**0.9956** •	0.6074	**0.9665** •	-
AngryBirds	**0.9927** •	**0.9965** •	0.9077	**0.9382** •	-	**0.9874** •	**0.9945** •	0.8235	**0.9474** •	-
Evernote	**0.9964** •	**0.9950** •	**0.9544** •	**0.9327** •	-	**0.9942** •	**0.993** •	0.8977	**0.9360** •	-
Dropbox	**0.9945** •	**0.9976** •	**0.9375** •	**0.9546** •	-	**0.9914** •	**0.9954** •	0.8865	**0.9582** •	-
Pinterest	**0.9914** •	**0.9976** •	**0.9485** •	**0.9568** •	-	**0.9871** •	**0.9962** •	0.8792	**0.9605** •	-
Tweets	**0.9935** •	**0.9990** •	**0.9111** •	**0.9736** •	-	**0.9924** •	**0.9985** •	0.7863	**0.9729** •	-
Uber	**0.9975** •	**0.9954** •	**0.9656** •	**0.9020** •	-	**0.9947** •	**0.9945** •	0.8995	**0.9194** •	-
InfoVis-Vast	**0.9892** •	**0.9958** •	**0.9943** •	**0.9514** •	-	**0.9842** •	**0.9913** •	**0.9947** •	**0.9510** •	-
TripAdvisor	**0.9936** •	**0.9966** •	**0.9285** •	**0.9421** •	-	**0.9918** •	**0.9950** •	0.8506	**0.9448** •	-
Facebook	**0.9974** •	**0.9976** •	0.9298	**0.9560** •	-	**0.9958** •	**0.9963** •	0.8308	**0.9533** •	-
ACM	**0.9964** •	**0.9952** •	**0.9942** •	**0.9591** •	**0.9985** •	**0.9952** •	**0.9949** •	**0.9776** •	**0.9577** •	**0.9963** •
20NewsGroup	**0.9963** •	**0.9994** •	**0.9977** •	**0.9054** •	**0.9745** •	**0.9951** •	**0.9990** •	**0.9968** •	**0.9098** •	**0.9781** •

Table 5.8: SHS Intra Topic Analysis.

Datasets	5 Words					10 Words				
	c-CluHTM (Concat. Emb)	f-CluHTM	HLDA	HPAM	hARTM	c-CluHTM (Concat. Emb)	f-CluHTM	HLDA	HPAM	hARTM
WhatsApp	**0.3657** ●	**0.3919** ●	0.2696	0.2879	-	0.2538	**0.3207** ▲	0.1255	0.1955	-
AngryBirds	**0.3783** ●	**0.4082** ●	0.3017	0.3046	-	0.2674	**0.3379** ▲	0.1811	0.1912	-
Evernote	**0.3666** ●	**0.4410** ●	0.3287	0.2997	-	0.2760	**0.3892** ●	0.2012	0.2078	-
Dropbox	**0.3932** ●	**0.3629** ●	**0.3587** ●	0.3084	-	0.2845	**0.2961** ●	0.2099	0.2141	-
Pinterest	**0.3535** ●	**0.4012** ●	0.3135	0.3001	-	0.2462	**0.3257** ▲	0.1811	0.1872	-
Tweets	0.3886	**0.3950** ▲	0.3680	0.2987	-	0.2699	**0.3319** ▲	0.2375	0.2033	-
Uber	**0.3856** ●	**0.4344** ●	0.3219	0.3082	-	0.2903	**0.3837** ▲	0.1975	0.2163	-
InfoVis-Vast	**0.3355** ●	**0.3190** ●	**0.2933** ●	**0.2868** ●	-	**0.2392** ●	**0.2597** ●	0.1707	**0.2063** ●	-
TripAdvisor	**0.3805** ●	**0.4151** ●	0.3504	0.3072	-	**0.2732** ●	**0.3557** ●	0.2196	0.2025	-
Facebook	0.3756	**0.4532** ▲	0.3409	0.2994	-	0.2738	**0.3912** ▲	0.2109	0.2016	-
ACM	**0.3793** ●	**0.3990** ●	**0.3767** ●	**0.3267** ●	0.2497	**0.3068** ●	**0.3471** ●	0.2574	0.2378	0.1708
20NewsGroup	**0.3452** ●	**0.3932** ●	**0.3733** ●	**0.4183** ●	0.2728	**0.2770** ●	**0.3215** ●	0.2633	**0.2981** ●	0.2001

semantic distances than the topics constructed by the other HTM methods. Regarding the results for ten words, f-CluHTM presented the best results in six out of 12 datasets. Interestingly, the number of words composing the latent topics increases the semantic information of the topics for f-CluHTM. Another relevant point is that HPAM presented the worst results for NPMI and Coherence, but for InfoVis-Vast and 20NewsGroup datasets, HTM statistically tied with c-CluHTM (Concat. emb.) and f-CluHTM.

Table 5.5 shows another example of the topic topology produced by c-CluHTM (Concat. emb.) for the Facebook dataset. We can see that the words *cellular* and *mobile* derive a topic that has the words *nexus* and *blackbarry*, which are types of mobile phones. This topology also shows an interesting case, where the words *proxy* and *windows* are repeated in the main topic and second subtopic. However, the second subtopic is a more specific topic that "elaborates" why these words appeared in the main topic. Table 5.6 shows an example of the topic topology produced by f-CluHTM for the same Facebook dataset. Interestingly, the main topic refers to mobiles and bugs, and the second subtopics followed the same pattern by adding new words. For instance, the second and third topics are more related to bugs, while the others have more mobile-related words. We can also observe that words more semantically related to the main topics are at the bottom of the top words. For instance the first subtopic, the words *blackbarry*, *bug*, *cellphone* are in top-7, 8, 9, respectively. This explains why the results are better for ten words in Table 5.8.

Revising the topics of WhatsApp for HLDA (Table 5.3), we can also observe that the duplicated topics – *community region relative shift expanding bye news brought seamlessly displays* – impacted the results of SHS, as they have no semantic relationship with the *Main Topics* of Table 5.3.

Finally, Table 5.9 shows the results for *topic inter distance*. Again, f-CluHTM is the best method, tying with c-CluHTM (Concat. emb.) in two out of 12 datasets with five words. Similarly to what we infer from Table 5.8, we can say that the topics built by f-CluHTM are more dissimilar in different topologies regarding semantic information. This reflects a desired behavior of a "good" HTM method since different topologies should express distinct dataset concepts. f-CluHTM and c-CluHTM (Concat. emb.) were su-

Table 5.9: SHS Inter Topic Analysis.

Datasets	5 Words					10 Words				
	c-CluHTM (Concat. Emb)	f-CluHTM	HLDA	HPAM	hARTM	c-CluHTM (Concat. Emb)	f-CluHTM	HLDA	HPAM	hARTM
WhatsApp	0.3755	**0.5070** ▲	0.2786	0.2883	-	0.2697	**0.4430** ▲	0.1289	0.1944	-
AngryBirds	0.3993	**0.4983** ▲	0.3163	0.3030	-	0.2921	**0.4248** ▲	0.1881	0.1896	-
Evernote	0.4148	**0.5041** ▲	0.3328	0.3078	-	0.3264	**0.4560** ▲	0.2046	0.2144	-
Dropbox	0.4212	**0.4970** ▲	0.3787	0.3147	-	0.3119	**0.4273** ▲	0.2237	0.2187	-
Pinterest	0.3819	**0.5027** ▲	0.3159	0.3138	-	0.2684	**0.4297** ▲	0.1852	0.1944	-
Tweets	0.4354	**0.5471** ▲	0.3709	0.3005	-	0.3538	**0.4890** ▲	0.2387	0.2050	-
Uber	0.4564	**0.5155** ▲	0.3224	0.3201	-	0.3638	**0.4661** ▲	0.1975	0.2261	-
InfoVis-Vast	**0.3790** ●	**0.3671** ●	0.2950	0.2947	-	0.2805	**0.3033** ▲	0.1721	0.2128	-
TripAdvisor	0.4013	**0.5009** ▲	0.3597	0.3071	-	0.2920	**0.4413** ▲	0.2239	0.2060	-
Facebook	0.4295	**0.5510** ▲	0.3443	0.3138	-	0.3321	**0.4915** ▲	0.2126	0.2110	-
ACM	**0.4899** ●	**0.5047** ●	0.3831	0.3422	0.2794	0.4303	**0.4585** ▲	0.2620	0.2516	0.1914
20NewsGroup	0.4938	**0.5909** ▲	0.3939	0.4308	0.3318	0.4263	**0.5348** ▲	0.2818	0.3131	0.2501

perior to the other baselines by considerable margins. It is important to note that since SHS measures the semantic distance among word vectors, the pooling transformation applied to transform the BERT embeddings into a static version may have impacted the cosine distance measure compared to the fastText embeddings since pooling strategies are susceptible to loose information.

However, when contrasting the SHS inter-topic with the NPMI and Coherence results, we can observe that the pooling does not affect the quality of the topics regarding the co-occurrence information among words. Again, the SHS results show another perspective on analyzing the quality of topics that goes beyond the traditional metrics.

5.7 Chapter Summary

In this chapter, we introduce a novel unsupervised non-probabilistic method – CluHTM. Our new method exploits the global semantic information provided by the CluWords representation and an original application of a stability measure to define the "shape" of the hierarchy. We present two variants of the CluHTM method – (i) f-CluHTM, which exploits pre-trained static embeddings to build the CluWords representations and (ii) c-CluHTM, which exploits the Contextual word embeddings from BERT transformed into static version using pooling approaches to build the CluWords representations. CluHTM variants excelled, being around twice as effective as the strongest state-of-the-art baselines, considering all tested datasets and evaluation metrics. The overall gains over some of these strongest baselines are higher than 500% in some datasets. In addition, we present new evaluation metrics for evaluating hierarchical topic modeling methods. The newly proposed topic quality metrics assess aspects related to topological consistency (or redundancy) and the hierarchical semantic structure that are important to hierarchical methods. These are different and complementary aspects than those cap-

tured by traditional TM metrics such as NPMI and Coherence, which measure the quality of each topic in an isolated manner, disregarding topological relationships among topics. Our new proposed topic quality metrics capture distinct behaviors from the topics built, including duplicity of topics built by some HTM methods. Our results also show that c-CluHTM and f-CluHTM present the best results in building a hierarchical structure while avoiding redundancy.

Chapter 6

Extending the CluWords concept for Sentiment Classification

In this chapter, we discuss the linguistic justification for using semantic relationships extracted from word embedding in lexicon expansion. Lexicons are used in several applications in the context of sentiment analysis, mainly in unsupervised scenarios. They are generally used in sentence-level polarity detection. This task is particularly relevant in social media, such as the analysis of news [74]. We start by formalizing the hypotheses and providing extensive empirical evidence for them. We present a simple method for lexicon expansion that exploits the embedding information. Then, we perform an experimental evaluation, comparing our proposed method with several baselines. The experimental results of the proposed lexicon expansion present strong evidence that the semantic relationship can generate significant information for using CluWords in the context of Sentiment Analysis. Thus, we present the CluSent, a CluWords instantiation specially designed for Sentiment Analysis. We initially revisit the three steps – Clustering, Filtering, and Weighting, presented in Chapter 3. Then we present the new filtering and weighting building blocks. Next, we present the experimental setup – datasets, baselines, and evaluation metrics. Finally, we present the results compared with the main sentiment analysis methods in the literature.

6.1 A linguistic justification for the use of word embeddings in sentiment analysis.

The linguistic insight for our proposal traces back to the seminal work of Bréal, a linguist who already in the XIX century defended that the semantics of a word are related to the word's usage and that these semantics can be derived from the meaning

of the word itself and from the context in which the word is used[1] [11]. Several modern linguists deepened Bréal's visions on the meaning and sentiment of the word itself and how such meaning changes in a given context [102, 65, 66]. This linguistic school emphasizes the distinction between a word's (general) meaning and its meaning in a particular context (which, in our case, corresponds to affective meaning), as interactions between words may influence their meanings. A typical example is the necessity of considering intensifications to properly capture the semantics of a sentence.

The importance placed by Bréal's school on the context of a word as a means of deriving its (affective) meaning is, interestingly, akin to the importance placed by word embeddings on context as a means of providing (general) meaning. Since word embeddings use context to build their vector space and derive (general) meaning, it seems natural to wonder whether word vectors also capture information on the affective meaning and, more precisely, words' sentiment value. In the following, we explore this insight.

In this section, we formalize the hypotheses supporting our method for using semantic relationships extracted from word embeddings for lexicon expansion. To avoid over-generalization, we restrict ourselves to the scope of the **VADER lexicon**. VADER's lexicon is one of the most effective (according to [76]) and popular[2] lexicons available in the literature, and remains relevant in recent work (e.g., [54, 2]). More precisely, we here provide extensive empirical evidence that collectively guides our lexicon expansion's design and demonstrates that word embedding information may be effective for lexicon expansion.

6.1.1 Hypotheses

Following the key insight that word embeddings may carry relevant information about words' sentiment value, we laid down hypotheses that, if confirmed, could guide the design of a novel lexicon-expansion technique (currently based on VADER's lexicon). In this section, we formalize these hypotheses and present extensive empirical evidence in their favor.

Hypothesis (H1): Word-vectors *may* carry relevant information about words' sentiment values (polarity and intensity).

Given a lexicon embedded in a word-vector space, we say that two word-vectors are α-*neighbors* if a similarity metric (e.g., cosine) between them is at least a value $\alpha \in [0, 1]$.

[1] Although Edmund Burke (1729-1797) had defended some of these ideas before Bréal, he gave it a less formal treatment.

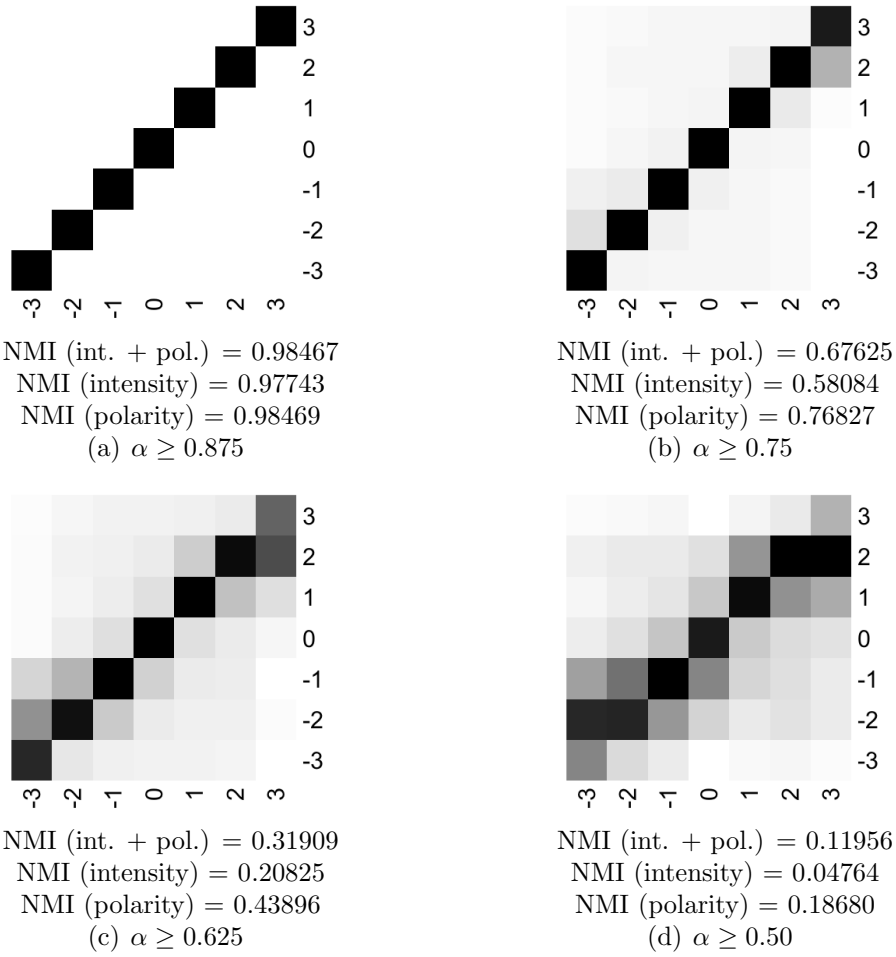
[2] The original Vader paper has currently more than 1200 references, according to Google Scholar.

We then define an *α-neighborhood multiset* [3] as follows. N_α consists of exactly all pairs (x, y) such that $x, y \in \{-3, -2, -1, 0, 1, 2, 3\}$ are the truncated sentiment values of word-vectors that are α-neighbors. For instance, if the two-word vectors corresponding to the words *fantastic* (sentiment value of +2.6) and *awesome* (sentiment value of +3.1) are 0.875-neighbors (i.e., their cosine similarity is at least $\alpha = 0.875$), then the pair $(+2, +3)$ will belong to the neighborhood multiset $N_{0.875}$. We then define a joint probability distribution p_α on random variables X, Y (corresponding to the first and second component of each ordered pair) derived from N_α in a straightforward way: $p_\alpha(x, y)$ is the ratio between the number of times the pair (x, y) appears in N_α and the total number of pairs in N_α.

As mentioned, we adopt the VADER lexicon [37], which contains the sentiment of 7,502 words, with sentiment values ranging in $(-4, 4)$. We embed each of its words into three embedding spaces: GoogleNews Word2Vec space [58], GLoVe space [71], and FastText [59]. We then compute the joint probability distribution p_α for α taking values 0.875, 0.75, 0.625, and 0.50. Figure 6.1 depicts a heat map for the GoogleNews Word2Vec space, for each p_α value: the x and y-axes indicate the sentiment values associated with X and Y. The darker the squares in the Figure, the higher the corresponding values of probability. A similar heat map can be observed in Figures 6.2 and 6.3 for the GLoVe and FastText spaces, respectively. Note that the distributions of Figures 6.1, 6.2, 6.3 are far from uniform: in fact, the highest probabilities lay on or around the diagonal. This pattern suggests that word vectors lying in the same neighborhood tend to have similar sentiment values. Moreover, the smaller the neighborhood (i.e., the greater the value of α), the more concentrated the probability is around the diagonal.

To properly quantify this empirically observed tendency, we computed the *normalized mutual information* $NMI(X;Y) = {}^{I(X;Y)}/_{H(X,Y)}$ among random variables X and Y, where $I(X;Y) = \sum_{x,y} p_\alpha(x, y) \log_2 {}^{p_\alpha(x,y)}/_{p_\alpha(x)p_\alpha(y)}$ is the *mutual information* between X and Y, and $H(X, Y) = -\sum_{x,y} p_\alpha(x, y) \log_2 p_\alpha(x, y)$ is their *joint entropy*. The normalized mutual information is a real value between 0 and 1, representing how much information about sentiment value (i.e., both intensity and polarity) is shared by word-vectors in the same neighborhood. More precisely, the closer the value of $NMI(X;Y)$ is to 1, the more information X and Y share. For the sake of comparison, we also computed the normalized mutual information w.r.t. sentiment intensity (the absolute value of the sentiment value) only, and w.r.t. polarity (the sign of the sentiment value) only. The results for Word2Vec space, also shown in Figure 6.1, strongly suggest that: (1) there is semantic information in the space of word vectors, and (2) this information degrades as the size of the neighborhood increases. Hence, the experiments substantiate our hypothesis that word vectors capture affective information in the form of polarity and intensity, even as we decrease the similarity threshold to as low as $\alpha = 0.50$. In Figures 6.2 and 6.3, similar

[3] A *multiset* (or a *bag*) is a generalization of the concept of a set that, unlike a set, allows multiple instances of the multiset's elements.



NMI (int. + pol.) = 0.98467
NMI (intensity) = 0.97743
NMI (polarity) = 0.98469
(a) $\alpha \geq 0.875$

NMI (int. + pol.) = 0.67625
NMI (intensity) = 0.58084
NMI (polarity) = 0.76827
(b) $\alpha \geq 0.75$

NMI (int. + pol.) = 0.31909
NMI (intensity) = 0.20825
NMI (polarity) = 0.43896
(c) $\alpha \geq 0.625$

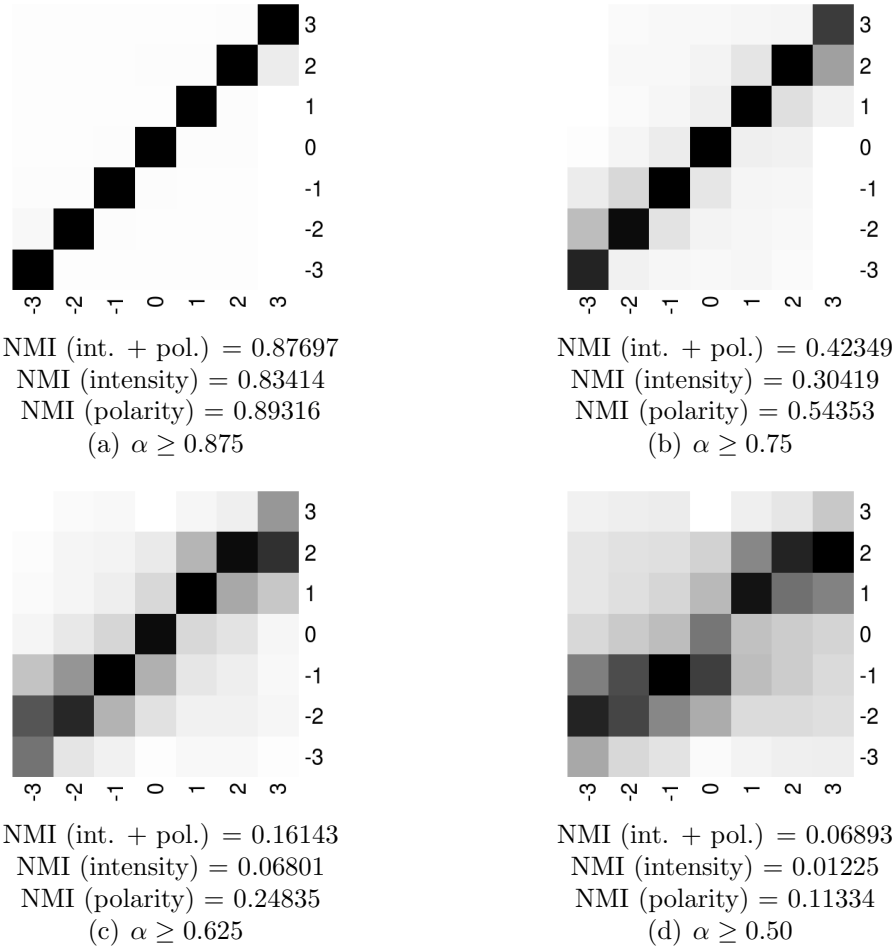
NMI (int. + pol.) = 0.11956
NMI (intensity) = 0.04764
NMI (polarity) = 0.18680
(d) $\alpha \geq 0.50$

Figure 6.1: Probability distribution of co-occurrence over the lexicons of VADER in the Word2Vec vector space, and their corresponding values of normalized mutual information (NMI) considering intensity, polarity, or both. Higher values of NMI (close to 1.0) indicate that the pair of words share similar information regarding intensity, polarity, or both.

behavior is observed for the GLoVe and FastText spaces, respectively, even though these word-vectors capture less information when compared to the Word2Vec space.

Interestingly, more than supporting hypothesis (H1), these experiments present strong empirical evidence that cases such as that of the pair "good" and "bad", in which opposite words have close embeddings, are indeed exceptions rather than the general rule in the VADER lexicon. Such examples have been taken to mean that word embeddings cannot be used for sentiment analysis, but our empirical analysis on the VADER lexicon is precisely showing that this is not necessarily the case. For instance, for polarity and intensity together, we have $NMI = 0.67625$ for words that are close by a factor of $\alpha \geq 0.75$ (which amounts to 7045 pairs of words). The effect is even more evident for words that are even closer to each other ($NMI = 0.98467$ for $\alpha \geq 0.875$, corresponding to 5913 words).

Moreover, these experiments also suggest that, as the distance between words increases (e.g., $\alpha \geq 0.625$), pairs sharing little information become more common (e.g.,



NMI (int. + pol.) = 0.87697
NMI (intensity) = 0.83414
NMI (polarity) = 0.89316
(a) $\alpha \geq 0.875$

NMI (int. + pol.) = 0.42349
NMI (intensity) = 0.30419
NMI (polarity) = 0.54353
(b) $\alpha \geq 0.75$

NMI (int. + pol.) = 0.16143
NMI (intensity) = 0.06801
NMI (polarity) = 0.24835
(c) $\alpha \geq 0.625$

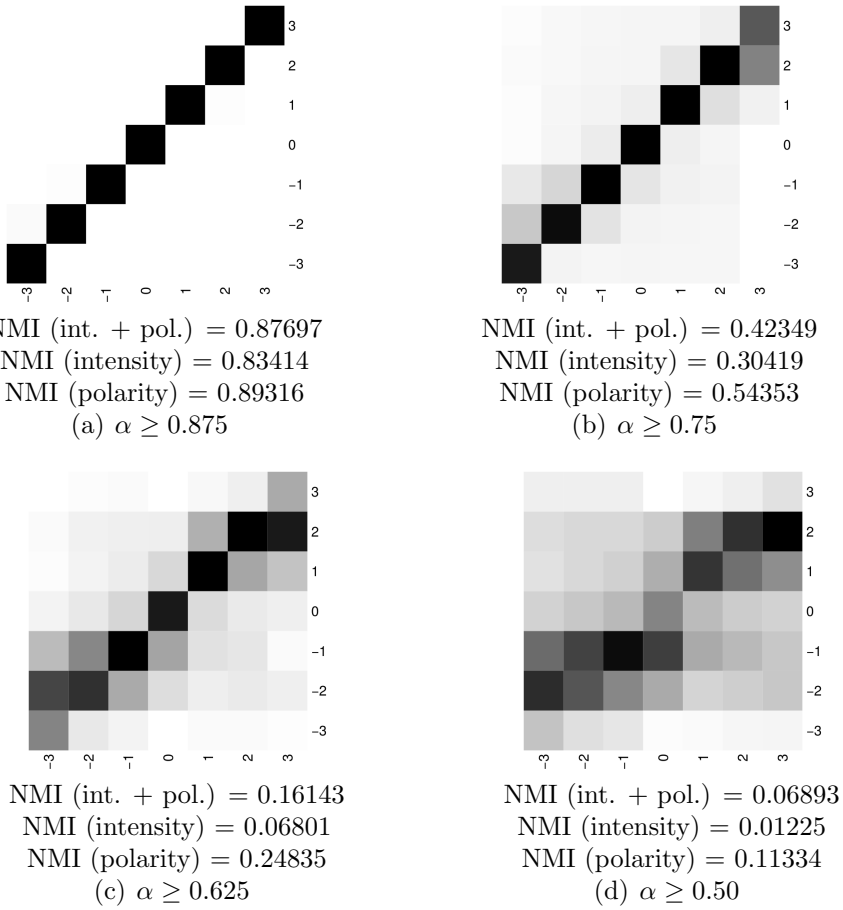
NMI (int. + pol.) = 0.06893
NMI (intensity) = 0.01225
NMI (polarity) = 0.11334
(d) $\alpha \geq 0.50$

Figure 6.2: Probability distribution of co-occurrence over the lexicons of VADER in the GLoVe vector space, and their corresponding values of normalized mutual information (NMI) considering intensity, polarity, or both. Higher values of NMI (close to 1.0) indicate that the pair of words share similar information regarding intensity, polarity, or both.

$NMI = 0.31909$), implying that it is not reliable to explore solely the closest similarities among word vectors directly to infer sentiment. It is necessary, hence, to use some approach to smooth out noisy pairs of word vectors (e.g. learning) if we want to fully exploit the information available in larger multisets (i.e., $\alpha \geq 0.625$).

Hypothesis (H1.1): The "closer together" two word-vectors (of the VADER lexicon) are, the more likely it is that they share "similar" polarities and values of intensity.

This refinement of hypothesis (H1) is already supported by the data in Figures 6.1, 6.2, and 6.3. We will explore it further in the case we incorporate into the lexicon *neutral words*, i.e., words with neither positive nor negative polarity (e.g., pronouns, numbers, and proper names). Our goal is to investigate whether it is possible to infer a word's sentiment value only by considering the word's vicinity in the word-vector space. The inclusion of neutral words makes the tests more robust and realistic since, in practice, texts do contain neutral words that can make the task of sentiment analysis significantly



NMI (int. + pol.) = 0.87697
NMI (intensity) = 0.83414
NMI (polarity) = 0.89316
(a) $\alpha \geq 0.875$

NMI (int. + pol.) = 0.42349
NMI (intensity) = 0.30419
NMI (polarity) = 0.54353
(b) $\alpha \geq 0.75$

NMI (int. + pol.) = 0.16143
NMI (intensity) = 0.06801
NMI (polarity) = 0.24835
(c) $\alpha \geq 0.625$

NMI (int. + pol.) = 0.06893
NMI (intensity) = 0.01225
NMI (polarity) = 0.11334
(d) $\alpha \geq 0.50$

Figure 6.3: Probability distribution of co-occurrence over the lexicons of VADER in the FastText vector space, and their corresponding values of normalized mutual information (NMI) considering intensity, polarity, or both. Higher values of NMI (close to 1.0) indicate that the pair of words share similar information regarding intensity, polarity, or both.

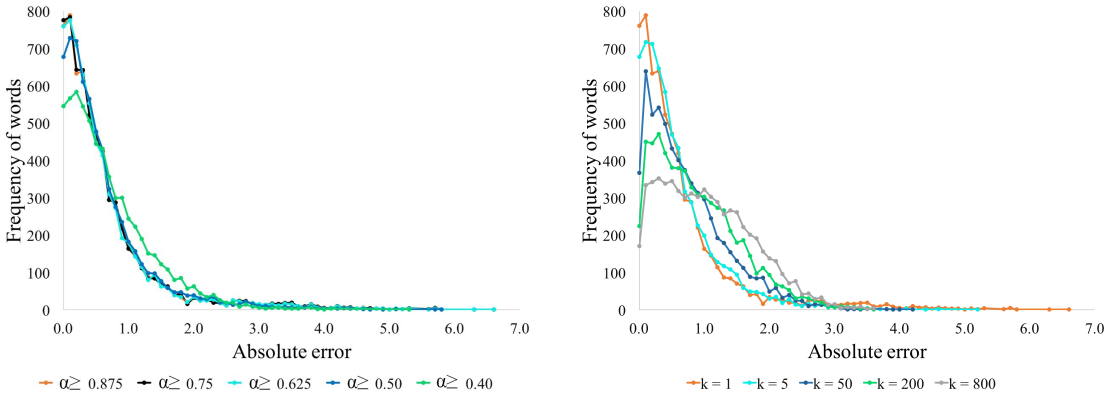
harder.

For that, we build an "artificial" lexicon $\mathcal{D}$ that incorporates into the VADER lexicon (which contains only words with positive or negative polarity) a new set of neutral words. As for the neutral words[4], we selected the 444 most common words without polarity from an external lexicon (SentiWordNet: http://sentiwordnet.isti.cnr.it/) related to pronouns, conjunction, numbers, names of places, and people. Thus, our expanded lexicon $\mathcal{D}$ has 3,331 positive words, 4,171 negative words, and 444 neutral words. For each word in $\mathcal{D}$, we estimated its sentiment value as the average sentiment value of all of its α-neighbors, for different values of α. We call this method *dynamic neighborhood* estimation since the number of neighbors used is variable. To measure the accuracy of the assessment, we computed the absolute error $|\hat{x} - x|$ of the estimated sentiment value $\hat{x}$ concerning the real sentiment value x for each word in $\mathcal{D}$. In the case of the neutral words, their "real sentiment value" would be 0 (zero), and we expect that the estimated

[4]A link to this word list will be available upon acceptance.

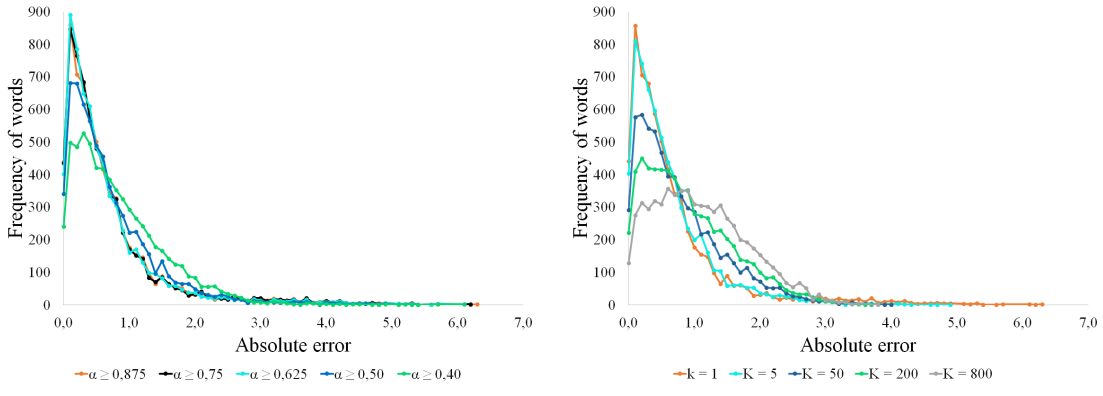
value for them, using their α-neighbors, would also be close to zero.

Figure 6.4(a) shows the histogram of absolute error values for Word2Vec space, using different sizes of dynamic neighborhoods, for all words in $\mathcal{D}$. Note that 82.89% of values of absolute errors do not exceed 1.0 (in a range that could go from 0 to 3.0), implying that the information about the word vectors also has some relation with the sentiment values assigned by humans (the VADER lexicon was quantified by humans). For the GLoVe and FastText spaces, depicted in Figures 6.5(a) and 6.6(a), 80.78% and 83.00% of values of absolute error are smaller or equal to 1.0, respectively. This suggests that it is possible to automatically infer sentiment values using semantic relationships derived from already cured words.



(a) Dynamic neighborhood (b) Static neighborhood

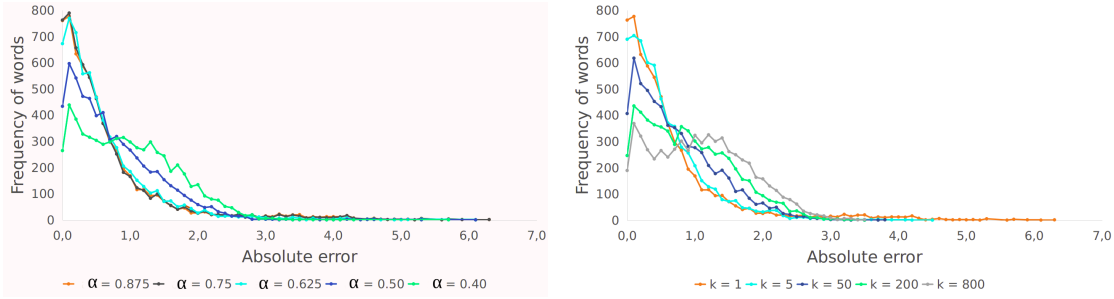
Figure 6.4: Histograms of the absolute error in sentiment-value prediction for various sizes of dynamic and static neighborhoods using Word2Vec vector space.



(a) Dynamic neighborhood (b) Static neighborhood

Figure 6.5: Histograms of the absolute error in sentiment-value prediction for various sizes of dynamic and static neighborhoods using GLoVe vector space.

For comparative purposes, we also test the accuracy of sentiment value estimation based on the combination of a fixed number of neighbors of each word vector. For this,



(a) Dynamic neighborhood

(b) Static neighborhood

Figure 6.6: Histograms of the absolute error in sentiment-value prediction for various sizes of dynamic and static neighborhoods using FastText vector space.

we define the *k-neighborhood* of a word vector as the set of the $k \geq 1$ closest other word vectors, according to cosine distance. As before, we employ different values of k and compute the absolute estimation error $|\hat{x} - x|$ for each word in $\mathcal{D}$, when $\hat{x}$ is computed as the average sentiment value of the word's k-neighborhood. Figures 6.4(b), 6.5(b), and 6.6(b) show the histogram of absolute error values for Word2Vec, GLoVe and FastText spaces, respectively, using different sizes of static neighborhoods, for all words in $\mathcal{D}$. For the three word-vectors spaces, about 71.0% of values of absolute error do not exceed 1.0, which indicates that static neighborhoods can also provide useful information about words' sentiment values.

In conclusion, the above experiments further support the hypothesis (H1.1). They also strongly suggest that predicting a sentiment score to a word using the nearest neighborhood approach attenuates the potential impact of words having opposite sentiments. In particular, the use of a dynamic neighborhood provides slightly better accuracy than that of a static neighborhood when predicting the actual values of the polarity/intensity of words. However, as we shall see in Section 6.3.2, for the task of sentence/document-level sentiment analysis it is better to be more *conservative* in the average prediction of most words (by using the static neighborhood). Indeed, large deviations of the actual value of an "aggressive" method may affect the overall sentiment estimation given the way most lexicon-based sentiment analysis methods work.

Next, we investigate how the information contained in both static and dynamic neighborhoods can be used for lexicon expansion.

6.2 Method for Lexicon Expansion

We now present our method for lexicon expansion using word embeddings, based on the hypotheses discussed in the previous section. The method, shown in Algorithm 2, takes as input three sets: $\mathcal{L}$ is the lexicon to be expanded, $\mathcal{V}$ is an external vocabulary whose words will be used to expand $\mathcal{L}$, and $\mathcal{W}$ is the collection of pairs $\langle t, w_t \rangle$ where t is a word in $\mathcal{V}$ and w_t is the word vector of t in $\mathcal{W}$. The output of the method is an expanded lexicon $\mathcal{L}_{ext}$ satisfying $\mathcal{L} \subset \mathcal{L}_{ext}$.

The method starts by obtaining two subsets $\mathcal{W}_\mathcal{L}$ and $\mathcal{W}_\mathcal{V}$ from $\mathcal{W}$ via function *WordEmbeddings* (lines 1 and 2 of Algorithm 2). Given a set of words $\mathcal{X}$ and the set $\mathcal{W}$ as parameters, function *WordEmbeddings* obtains the subset $\mathcal{W}_\mathcal{X}$ of $\mathcal{W}$ defined as $\mathcal{W}_\mathcal{X} = \{\langle t, w_t \rangle \mid t \in \mathcal{X} \wedge \langle t, w_t \rangle \in \mathcal{W}\}$. Next, the method initializes the expanded lexicon $\mathcal{L}_{ext}$ with the pairs $\langle t, v \rangle$ contained in lexicon $\mathcal{L}$ (line 3 of Algorithm 2).

Algorithm 2: LexiconExpansion

Input: $\mathcal{L}$ - A lexicon with sentiment values.
$\mathcal{V}$ - A vocabulary of words without sentiment values.
$\mathcal{W}$ - the set of word vectors for words in $\mathcal{L}$ and in $\mathcal{V}$.

Output: $\mathcal{L}_{ext}$ - The expanded lexicon.

1 $\mathcal{W}_\mathcal{L} \leftarrow$ **WordEmbeddings**$(\mathcal{L}, \mathcal{W})$;
2 $\mathcal{W}_\mathcal{V} \leftarrow$ **WordEmbeddings**$(\mathcal{V}, \mathcal{W})$;
3 $\mathcal{L}_{ext} \leftarrow \mathcal{L}$;
4 **foreach** $t_\mathcal{V} \in \mathcal{V}$ **do**
5 $\quad \mathcal{S} \leftarrow \emptyset$;
6 $\quad$ **foreach** $t_\mathcal{L} \in \mathcal{L}$ **do**
7 $\quad\quad \mathcal{S} \leftarrow \mathcal{S} \cup \{\langle t_\mathcal{L}, cosine(w_{t_\mathcal{V}}, w_{t_\mathcal{L}}) \rangle\}$,
8 $\quad\quad$ where $w_{t_\mathcal{V}} \in \mathcal{W}_\mathcal{V}$ and $w_{t_\mathcal{L}} \in \mathcal{W}_\mathcal{L}$;
9 $\quad \mathcal{N}_{t_\mathcal{V}} \leftarrow$ **ChooseNearestNeighbors**$(t_\mathcal{V}, \mathcal{S})$;
10 $\quad v_{t_\mathcal{V}} \leftarrow$ **k-NNRegression**$(t_\mathcal{V}, \mathcal{N}_{t_\mathcal{V}})$;
11 $\quad \mathcal{L}_{ext} \leftarrow \mathcal{L}_{ext} \cup \{\langle t_\mathcal{V}, v_{t_\mathcal{V}} \rangle\}$;

In order to expand the lexicon with words in $\mathcal{V}$, the method computes for each word $t_\mathcal{V}$ in $\mathcal{V}$ the cosine similarity measure between the word vector $w_{t_\mathcal{V}}$ of $t_\mathcal{V}$ and the word vector $w_{t_\mathcal{L}}$ of each word $t_\mathcal{L}$ in the lexicon $\mathcal{L}$ (lines 7 and 8 of Algorithm 2). After computing the set $\mathcal{S}$ of pairs containing words in $\mathcal{L}$ and their respective distances to term $t_\mathcal{V}$, the method obtains from $\mathcal{S}$ the words that are the "nearest neighbors" of $t_\mathcal{V}$ in $\mathcal{S}$, keeping these words in set $\mathcal{N}_{t_\mathcal{V}}$ (line 9). Function *ChooseNearestNeighbors* is used to obtain the set of the nearest neighbors of $t_\mathcal{V}$.

In line 10, the method uses the set $\mathcal{N}_{t_\mathcal{V}}$ of nearest neighbors of the word $t_\mathcal{V}$ to compute the sentiment value v of $t_\mathcal{V}$. We employ the function *k-NNRegression* to compute

this sentiment value, which is a regression method that uses the mean of the sentiment values of the words in the nearest neighbors set $\mathcal{N}_{t_\mathcal{V}}$ of term $t_\mathcal{V}$ as an estimation of the sentiment value v (i.e., $median(\mathcal{N}_{t_\mathcal{V}})$). In line 11 the pair $\langle t_\mathcal{V}, v \rangle$ is inserted in lexicon $\mathcal{L}_{ext}$.

Our method is affected by how the nearest neighbors are computed in Algorithm 2. In Section 6.1.1 we presented two alternatives for the implementation of function *ChooseNearestNeighbors*, namely, what we call "dynamic" and the "static neighborhoods". The first chooses the nearest neighbors by imposing a threshold α on the similarity between a word $t_\mathcal{V}$ and the words in the lexicon. The second uses a pre-defined parameter k and obtains the nearest neighbors of $t_\mathcal{V}$ as the words in the lexicon that are the k-nearest words to $t_\mathcal{V}$. Since we are using pre-trained word embeddings, the complexity of our method is dominated by the complexity of k-NN regression, which is quadratic on the number of words. However, notice that to generate an expanded lexicon we only need to run this procedure once, and this can be done in batch. Moreover, the incorporation of any new word into the lexicon can the done incrementally, i.e., only for this new word.

6.3 Evaluation of sentiment analysis using expanded lexicons

6.3.1 Experimental Setup

6.3.1.1 Textual Datasets

To evaluate polarity detection methods, we used a publicly available benchmark [76] with twenty real-world textual datasets gathered from various sources. Table 6.1 shows some characteristics of these datasets. Each column depicts, respectively, the dataset's name, number of *messages*—i.e., units of text whose sentiment value is to be determined—, number of features (words), the average number of words (density) in each message, and number of positive and of negative messages.

Dataset	#Msgs	#Feat	Density	#Pos	#Neg
aisopos_ntua	278	1,493	13.0	159	119
debate	1,979	3,360	11.5	730	1,249
english_dailabor	1,227	3,508	11.3	739	488
nikolaos_ted	727	1,635	11.7	318	409
pang_movie	10,662	12,432	13.9	5,331	5,331
sanders	1,091	3,102	13.5	519	572
sarcasm	71	107	6,78	33	38
sentistrength_bbc	752	5,655	40.3	99	653
sentistrength_digg	782	4,015	22.2	210	572
sentistrength_myspace	834	2,639	14.7	702	132
sentistrength_rw	705	4,595	43.3	484	221
sentistrength_twitter	2,289	7,777	13.8	1,340	949
sentistrength_youtube	2,432	6,275	12.2	1,665	767
stanford_tweets	359	1,620	12.0	182	177
tweet_semevaltest	3,060	9,087	16.4	2,223	837
vader_amazon	3,610	3,678	11.9	2,128	1,482
vader_movie	10,568	11,980	14.0	5,242	5,326
vader_nyt	4,946	8,756	13.0	2,204	2,742
vader_twitter	4,196	7,346	11.2	2,897	1,299
yelp_review	5,000	19,398	71.5	2,500	2,500

Table 6.1: Dataset characteristics

6.3.1.2 Evaluation, Algorithms, and Procedures

As already mentioned, our goal is to infer the polarity of sentences using lexicons expanded with our method, having VADER as the base lexicon. More specifically, we use as the input lexicon $\mathcal{L}$ for Algorithm 2 the set $\mathcal{D}$ consisting of the VADER lexicon expanded with a set of neutral words, as explained in Section 6.1.1. We also explore the VADER's shell to infer the sentiment value at the sentence level. The VADER shell [37] is a method that implements four general rules incorporating grammatical and syntactic conventions (for the English language) to express and emphasize the intensity of sentiments. The shell exploits these rules, along with the lexicon, to compute a sentiment value for a sentence. The VADER shell rules are as follows:

1. Punctuation (exclamation point - !) increases the sentiment intensity without modifying its semantic orientation.

2. Capitalization (ALL-CAPS) emphasizes a sentiment-relevant word in the presence of other non-capitalized words. It increases sentiment intensity without affecting its semantic orientation.

3. Degree modifiers (e.g., intensifiers, booster words, or degree adverbs) increase or decrease sentiment intensity.

4. The contrasting conjunction "but" signals a shift in sentiment polarity, with the sentiment of the text following the conjunction being dominant.

We compare our method with sixteen polarity detection methods investigated in [76]. These methods can be divided into two groups according to the techniques they employ: (i) pure lexicon-based methods, and (ii) lexicon-based methods combined with Machine Learning techniques.

Pure lexicon-based methods usually combine the lexicon with some processing of sentence characteristics to determine its polarity. These approaches make use of a series of intensifiers, punctuation transformations, emoticons, and other heuristics, as exemplified above with the VADER's shell. Other strategies use only a lexicon built especially for a particular domain and do not exploit a shell (i.e., a set o grammar rules). For these methods, we evaluated a combination of the specific lexicon with the VADER's shell to detect the sentence polarity. For these combinations, we employ the naming convention of referring to (pure lexicon + VADER shell) as $V\_$ for VADER, along with the prefix of the lexicon (e.g., V_AFINN). The pure lexicon-based strategies we considered in our experiments include Umigon [76], Opinion Lexicon [76], AFINN [67], SO-CAL [76], Pattern.en [76], PANAS-t[76], ANEW_SUB [101], VADER [37], Sentiment140 Lexicon [28], NRC VAD [64], and SenticNet 5 [13].

We also consider lexicon-based hybrid methods that use machine learning techniques in the process of expanding a lexicon. All such methods used in our study employed models pre-trained by their authors, namely Opinion Finder (MPQA) [76], SentiWordNet [25], and SentiStrength [88].

To thoroughly evaluate the performance of our proposal for lexicon expansion based on word embeddings, we contrasted it against five other methods for automatically expanding lexicons. For all these baseline techniques, with the exception of Synset-exp, we use the publicly available pre-computed word embeddings trained on the Google News Corpus and the VADER shell to detect sentence polarity. All variations use the same initial lexicon (set $\mathcal{D}$) and the VADER's implementation to predict the polarity of a sentence. The difference between each method resides in how they expand the original lexicon, as explained below.

RF Regression is a variation of our method in which we substitute our proposed k-NN Regression function (in line 10 of Algorithm 2) with the traditional non-linear regression that uses Random Forest [12] to predict the sentiment value from the vectorial representation of words. We experimented with this method to reinforce the suitability of k-NN Regression for the task at hand.

Synset-exp is a lexicon expansion method that uses WordNet synsets [60]. Synsets consist of nouns, verbs, adjectives, and adverbs grouped into sets of cognitive synonyms, and are interconnected by semantic concepts and lexical relations. This strategy first finds synonyms for each word w_l in lexicon $\mathcal{D}$ using synsets and the

semantic relationships between them, according to [103]. The same sentiment score of a word w_l is attributed to their synonyms.

RoWE is a method that uses linear regression and a word-embedding space to build new lexicons (affective dimensions for words) [49]. According to its creators, a word may have more than one affective dimension related to sentiment. In our experiments, each word has only two dimensions (positive and negative dimensions). To standardize lexicons in only one dimension, we select as the intensity of a word the dimension (positive or negative) that has the highest value. We assume that the intensity of a word is the dimension with the largest value. To generate the sentiment intensity, a seed of words of the same polarity (e.g., positive sentiment) is used as a training set. As the linear regression method, we employ Bayesian Ridge regression as suggested in [49], with parameters tuned via cross-validation.

SENTPRO is a framework to learn sentiment domain-specific lexicons [33]. It generates word embeddings adapted to the domain of interest and uses the similarity between the word vectors to represent relationships between words. These relationships are discovered using random walks based on a small set of context words selected as seeds. In our experiments, we evaluate this approach by combining lexicon $\mathcal{D}$ with the publicly available lexicons provided by the authors[5]. The latter is generated from large corpora crawled from different domains (e.g., news, movies, politics, and reviews).

SSWE is a lexicon-expansion method that uses information from SSWE [85] to select a cluster of the nearest words for each word w_l of the original lexicon. The clusters of nearest words are selected using cosine similarity with a threshold of $\alpha \geq 0.60$, set according to the best possible choice for a given dataset. The sentiment score for each word inside a cluster is computed in the same way as in **RF Regression**, as suggested in [85].

All methods were compared using two standard text categorization measures: micro averaged F1 (MicroF1) and macro averaged F1 (MacroF1) [46]. MicroF1 measures the overall classification effectiveness of sentiment polarity, naturally giving more weight to the majority category. On the other hand, MacroF1 measures the classification effectiveness for each category and averages them, giving the same importance to both sentiment categories (positive and negative) in unbalanced datasets. To further evaluate our experimental results, we also use the macro averaged precision (MacroPrecision) and macro averaged recall (MacroRecall), whose harmonic mean results in MacroF1. It is important to note that the VADER shell can assign polarity only to documents whose vocabulary is included in the lexical dictionary. Thus, we consider as classification errors the cases in

[5]https://nlp.stanford.edu/projects/socialsent/

which the VADER shell can not assign any polarity. We also evaluate the Coverage [76] of the sentiment analysis methods defined as the proportion of documents in a collection for which the method is able to detect polarity.

We adopt a 5-fold cross-validation experimental setup. When experimenting with our method and the five other expansion alternatives described above, we used the words appearing in documents in the four training set folds in each round as the vocabulary to expand lexicon $\mathcal{D}$[6].

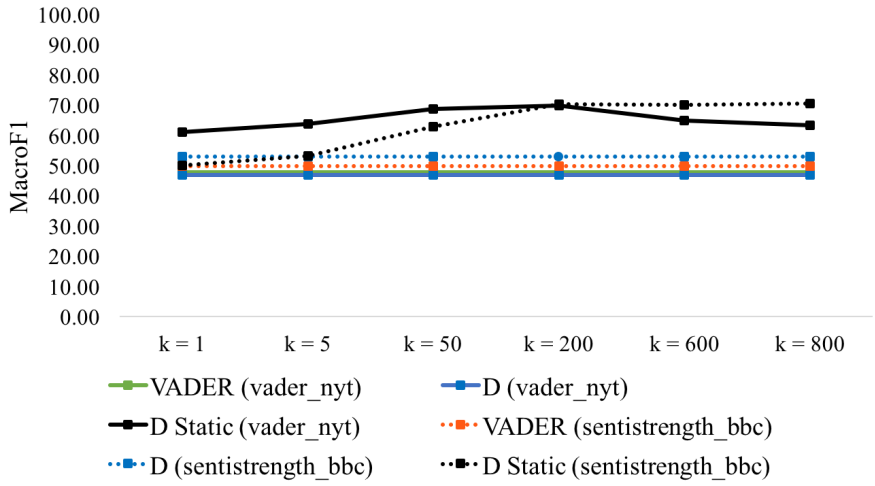
All free parameters of the methods were tuned using cross-validation in the respective training sets. We assess the statistical significance of our results by means of a paired t-test with 95% confidence and Holm correction to account for multiple tests. This test assures that the best results, marked with a green triangle (▲), are statistically superior to others. Statistical ties are represented as a gray dot (●), while losses are represented as red downward triangles (▼). The results obtained (and their standard deviations) are described in Section 6.3.4. To compare the proposed approach with all polarity detection methods, we use the average effectiveness obtained in the sentiment datasets. We also analyze results in each individual dataset. To properly compare different unsupervised approaches with statistical tests executing on the same dataset, we split each dataset into five disjoint folds, evaluating the effectiveness of the same method in different samples of the same dataset.

6.3.2 Choosing the Semantic Neighborhood

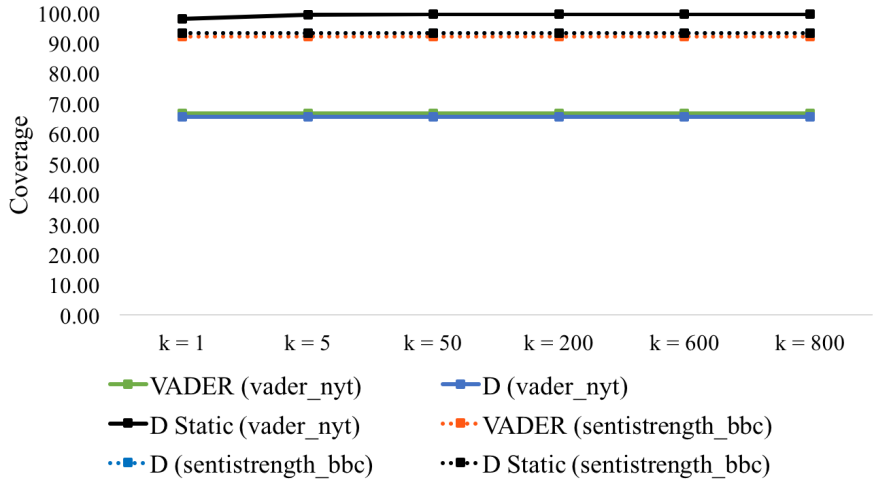
Figures 6.7 and 6.8 present the average results of Algorithm 2, considering two datasets (sentistrength_bbc and vader_nyt), with our expanded dictionaries: original VADER and VADER with neutral words (i.e., $\mathcal{D}$). These datasets were chosen to illustrate our arguments because some of our best experimental results were obtained with them. However, similar patterns were found in basically all tested datasets.

Figure 6.7(a) shows the results of MacroF1 using the static neighborhood. In most cases, we can observe that expanding a lexicon using semantic relationships results in superior sentiment detection when compared to VADER lexicons without expansion. The best MacroF1 results for the expanded dictionary were obtained with $k = 200$ in both datasets. As we can observe in Figure 6.7(b), the original VADER lexicon was not able to cover all texts in dataset vader_nyt, while the dictionary expanded with our method could classify nearly 100% of texts in that collection. In sentistrength_bbc, both VADER and $\mathcal{D}$ Lexicons were sufficient to achieve a coverage of more than 90%, and the expanded

[6]In our method, this vocabulary is used as $\mathcal{V}$ in Algorithm 2.



(a) MacroF1



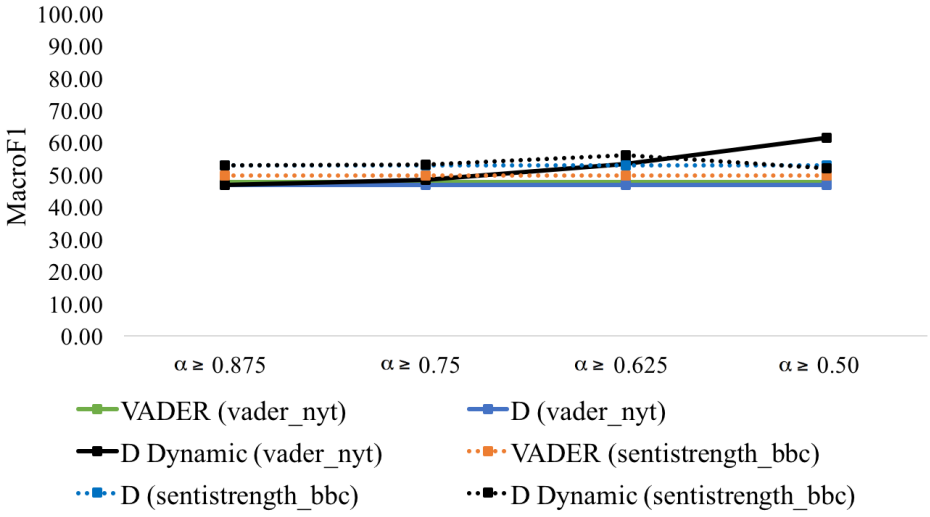
(b) Coverage

Figure 6.7: Static Semantic Neighborhood

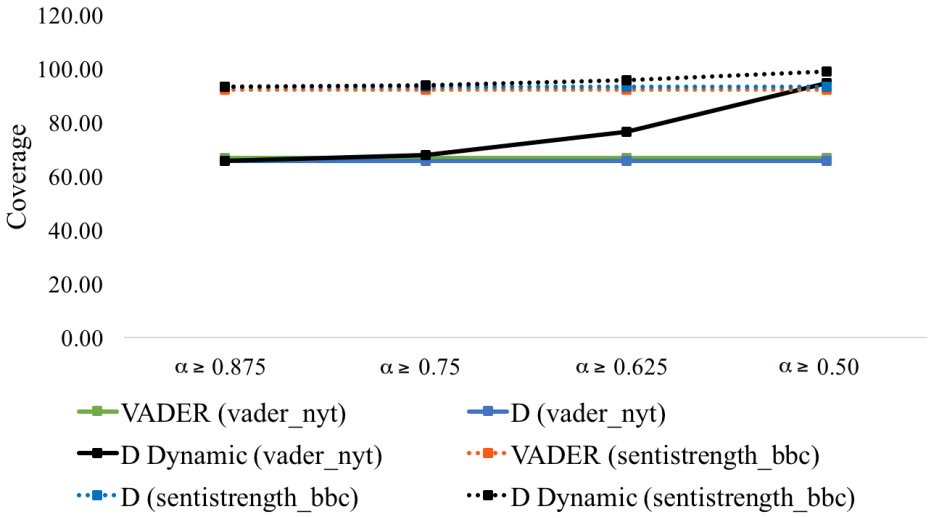
dictionary could only obtain a marginal gain.

Figure 6.8 shows the results using dynamic neighborhood. We can observe that the MacroF1 results (Figure 6.8(a)) obtained with the expanded dictionaries are only enhanced with a relaxed value of the similarity threshold α. High values of α impose a too rigorous criterion on which words are included in the expanded lexicon. In this case, only a few words are included, resulting in an expanded dictionary only slightly larger than the original one. This also explains why the expanded dictionaries were not able to significantly increase the coverage in both datasets (see Figure 6.8(b)).

Figures 6.7(a) and 6.8(a) show that for the task of sentence-level sentiment analysis, the use of static neighborhoods yielded, in general, better results (e.g., $k = 200$) than dynamic neighborhoods. We explain these results by the aforementioned more "conservative" nature of static neighborhood prediction when compared to the dynamic one. We conducted an experiment in the same dictionary used before (dictionary $\mathcal{D}$ from



(a) MacroF1



(b) Coverage

Figure 6.8: Dynamic Semantic Neighborhood

Algorithm 2) to reinforce this argument. In Figure 6.9, we measured the Root-mean-squared-error (RSME) for the static neighborhood by varying k from 1 to 800. For each k, we estimated the RMSE of each word's neighbors and then computed the median of the RMSE values. (We use median since it is a more robust measure of data variation.) The results in Figure 6.9 (related to Figure 6.4(b)) show that the error tends to be smaller for larger values of k than for smaller neighborhoods, stabilizing around $k = 200$. However, *more importantly*, we can see that the prediction value of each lexicon can be considered as a conservative prediction since the RMSE scores lie in the interval $[0, 2]$. We argue that for sentence-level sentiment analysis, this may be a better option for the following reasons.

- Since the polarity of a sentence is specified based on the individual polarity of its

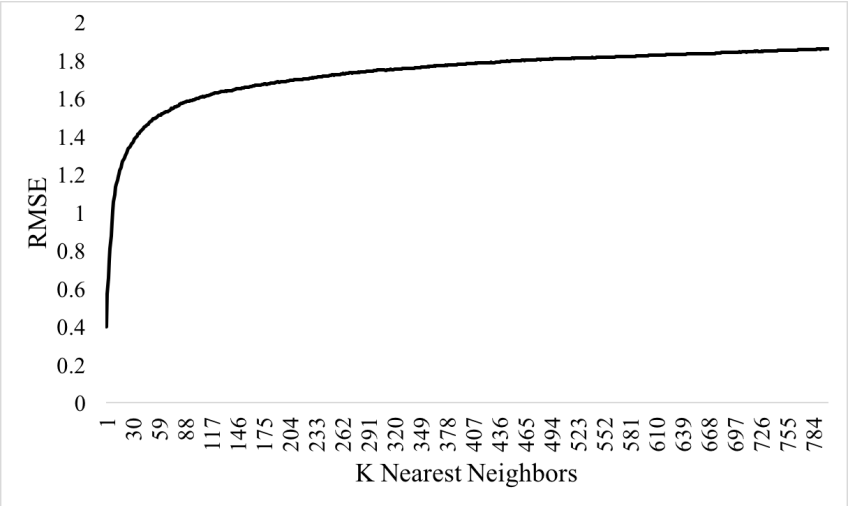


Figure 6.9: RMSE varying the number of static semantic neighbors

constituent words, large error estimates for a few words may cause higher deviations in polarity prediction.

- As we have no control over which words are indeed negative, positive, or neutral, as well as over their respective intensities, it is better to give more weight to curated (lexicon) words than to automatic predictions.
- Higher k values (e.g., around 200 neighbors) can smooth the impact of sentiment-opposite words (e.g., cases such as "good" and "bad").

For the reasons above, we will adopt only static neighborhoods in the following experiments.

6.3.3 k-NN Regression Expansion: Choosing the Lexicon

In this section, we evaluate our lexicon expansion solution considering two alternatives bases for expansion: (i) the dictionary $\mathcal{D}$ composed of the VADER lexicon (7,501 words) together with 444 neutral words, and (ii) NRC VAD, a proposed lexicon composed of more than 20,000 words [64]. In NRC VAD, each word has negative and positive scores, within the range $[0, 1]$, representing aspects related to Valence, Arousal, and Dominance. To expand NRC VAD, and use the VADER shell along with the expansion, we had to apply a Min-Max Scaling to normalize the scores of NRC VAD within the range $[-4, 4]$ since the VADER shell considers sentiment values in this range. In these experiments, we

use the NRC Valence scores since in preliminary experiments it produced better results than those of NRC Arousal and NRC Dominance scores.

Table 6.2 shows the results regarding the effectiveness (averaged over all datasets described in Table 1) of our proposed expansion strategy (line 1) compared to the original lexicon (line 2). As we can see, our k-NN Regression Expansion using lexicon $\mathcal{D}$ outperforms VADER, considering all measures (MacroF1, MicroF1, and coverage).

On the other hand, the proposed expansion strategy had little impact on expanding NRC VAD. This is clearly seen when comparing k-NN Regression (NRC VAD) with V_NRC VAD (3rd and 4th lines of Table 2, respectively): all results in all metrics are statistically tied for both approaches. We believe that our expansion approach did not significantly impact this lexicon because the vocabularies of the evaluated datasets already present high coverage (99% on average) in the original NRC VAD, leaving little room for improvement.

Another important point to stress is that both VADER and NRC VAD have similar effectiveness in MacroF1 and MicroF1, which suggests that they are good lexicons for detecting sentiment at the sentence level. However, the NRC VAD coverage is much higher, leaving little room for improvements by any expansion technique. This could explain why our expanded version of VADER using our k-NN Regression excelled, producing the best overall results in basically all metrics – a combination of the quality of the original VADER lexicon itself (reflected in the sentiment scores) with an improved coverage due to our technique. Consequently, we will adopt the k-NN regression with the lexicon $\mathcal{D}$ in the next experiments and analyses.

Methods	MacroPrec.	MacroRec.	MicroF1	MacroF1	Coverage
k-NN regression ($\mathcal{D}$)	74.71 (6.53) ▲	76.01 (6.38) ▲	76.65 (6.31) ▲	74.56 (6.51) ▲	99.51 (0.66) ▲
VADER	64.41 (10.67)	64.19 (10.08)	65.15 (11.19)	62.37 (11.00)	84.30 (7.83)
k-NN regression (NRC VAD)	67.21 (7.60) •	57.66 (4.75) •	59.47 (13.93) •	50.46 (10.77) •	99.63 (0.53) •
V_NRC VAD	66.52 (6.94)	60.70 (4.86)	62.21 (12.09)	56.15 (9.80)	97.29 (2.28)

Table 6.2: Average results of the lexicon-based methods compared with its lexicon expansion. Our strategy with the VADER lexicon outperforms or ties in all metrics when compared to the standard VADER lexicon.

6.3.4 Comparative Experimental Results

We now present the results of experiments evaluating the effectiveness of our expanded lexicons with respect to polarity detection tasks. We start by comparing the effectiveness of our expanded dictionary with traditional polarity detection methods and then move on to compare several different automatic expansion strategies.

6.3.4.1 Evaluation of Polarity Detection Methods

Table 6.3 contrasts the results of proposed expanded lexicons with a large representative sample of unsupervised polarity detection methods from the literature, considering different evaluation metrics. As before, results correspond to the average in all previously described datasets.

As we can see, our proposed strategy always achieves superior results considering all measures (MacroF1, MicroF1, and coverage). In fact, when comparing k-NN regression with the best alternative, V_Sentiment140 (considering the average of all datasets), it achieves substantial gains of 14% and 13.7% in MacroF1 and MicroF1, respectively. Moreover, on average, the MacroPrecision and MacroRecall are substantially better than any alternative, achieving gains of 12.7% and 12.8%, respectively, over the second-best baseline (Sentiment140 L). This shows that our extended lexicon can improve the number of correctly classified sentences among all samples in the datasets. In fact, our automatically extended lexicon provides sentiment evidence for the classification of sentences using words that were never manually associated with any sentiment polarity score. The high coverage of our method (99.51%) indicates that basically, all messages contain words automatically associated with sentiment scores.

The high result for MacroPrecision presents indirect evidence towards the quality of the sentiment scores we predict for words. More specifically, it provides some insight into polarities that were incorrectly inferred before the expansion but that we correctly inferred after expansion, mainly when compared to the original VADER method. The MacroPrecision result of the proposed strategy is statistically superior to those of all other methods. This is an important observation as the compared methods have manually labeled lexicons for specific domains as their main contributions.

Table 6.4 provides additional information about the effectiveness of the k-NN Regression. More precisely, it shows, in a per dataset basis, MacroPrecision and MacroRecall results of k-NN Regression compared to the three best baseline methods: VADER, Sentiment140 L., and V. Sentiment140. In terms of MacroPrecision, our proposed method has 13 wins, 5 ties, and only 2 losses (in sentistrengh_rw and yelp_review, both for Sentiment140 L). As for MacroRecall, the proposed method outperforms the other baselines in 5 cases, ties with them in 9, and loses in 6. Overall, these results show that k-NN Regression is consistent in assigning correct sentiment scores for words in the majority of cases.

Finally, Table 6.5 presents examples where our proposed expanded lexicon produced a correct sentiment value for a sentence for which the original VADER lexicon could not. The examples include (i) the original sentence; (ii) the words covered by the original lexicon; (iii) the value of the VADER shell given to the sentence using the original

lexicon; (iv) the words covered by the expanded version; (v) the value of the shell given to the sentence using the expanded version (correct polarity); and (vi) the correct class of the sentence.

Methods	MacroPrec.	MacroRec.	MicroF1	MacroF1	Coverage
k-NN regression-VADER	**74.71(6.53)** ▲	**76.01(6.38)** ▲	**76.65(6.31)** ▲	**74.56(6.51)** ▲	**99.51(0.66)** ▲
VADER	64.41 (10.67)	64.19 (10.08)	65.15 (11.19)	62.37 (11.00)	84.30 (7.83)
SentiStrength	56.03 (10.55)	56.44 (10.67)	58.19 (11.60)	55.45 (10.54)	70.71 (8.39)
Umigon	50.99 (15.18)	50.94 (15.60)	51.02 (15.56)	49.41 (15.54)	65.77 (16.22)
AFINN	47.18 (9.16)	46.90 (8.27)	47.97 (9.39)	45.26 (8.31)	63.11 (9.47)
PANAS-t	8.65 (5.65)	6.76 (4.19)	7.02 (4.29)	6.87 (4.12)	9.49 (5.83)
Opinion Finder	34.95 (7.37)	33.95 (7.41)	34.42 (8.16)	32.94 (7.00)	50.17 (14.87)
SO-CAL	57.20 (8.65)	57.70 (8.50)	58.16 (8.49)	56.22 (8.96)	73.81 (9.21)
Sentiment140 L.	63.20 (6.20)	62.34 (5.74)	62.33 (7.27)	59.05 (6.44)	94.39 (2.88)
Pattern.en	57.42 (11.35)	56.68 (10.49)	57.57 (12.89)	54.31 (12.28)	78.86 (9.50)
Opinion Lexicon	44.13 (7.97)	43.30 (7.54)	43.79 (7.84)	42.07 (7.46)	57.46 (10.66)
V_PANAS-t	5.46 (4.60)	3.91 (2.51)	4.12 (2.76)	4.05 (2.67)	5.51 (3.75)
V_AFINN	56.72 (9.59)	57.09 (9.41)	57.98 (10.15)	55.36 (9.57)	74.37 (8.72)
V_SentiWordNet	42.67 (8.29)	42.34 (7.76)	43.14 (8.51)	40.89 (7.58)	63.62 (10.07)
V_OpinionLexicon	55.21 (7.85)	55.49 (7.57)	56.07 (7.73)	53.90 (7.72)	70.73 (8.27)
V_ANEW_SUB	62.70 (6.01)	57.29 (3.66)	59.20 (11.88)	51.72 (8.64)	94.61 (2.78)
V_Sentiment140	66.30 (8.12)	67.36 (7.98)	67.42 (8.82)	65.43 (9.19)	93.78 (2.71)
V_NRC VAD	66.52 (6.94)	60.70 (4.86)	62.21 (12.09)	56.15 (9.80)	97.29 (2.28)
V_SenticNet 5	61.56 (5.02)	56.11 (3.47)	57.49 (13.59)	49.21 (10.13)	98.47 (1.56)

Table 6.3: Average results of the lexicon-based methods. Our strategy outperforms or ties in all metrics. It is able to classify more instances (MacroRec.) while preservig the accuracy (MicroPrec.) when compared to other lexicon-based strategies.

Datasets	k-NN Regression		Vader		Sentiment140 L.		V Sentiment140	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
aisopos_ntua	**88.85(4.03)** ●	**89.56(4.61)** ●	76.84 (4.35)	**87.52(6.62)**	**90.12(2.64)**	57.11 (7.06)	**86.45(4.07)**	76.53 (5.32)
debate	**61.55(5.95)** ▲	52.31 (1.47) ▼	37.38 (4.01)	59.12 (3.00)	48.70 (4.44)	47.42 (4.84)	46.52 (3.60)	**68.45(3.45)**
english_dailabor	**89.38(0.93)** ▲	**85.28(1.77)** ▲	78.44 (1.49)	82.05 (1.44)	87.10 (3.48)	58.09 (3.98)	84.61 (3.32)	77.24 (3.41)
nikolaos_ted	**66.40(3.81)** ●	**79.22(6.38)** ●	54.72(1.58)	**74.64(4.55)**	61.43 (4.94)	46.04 (3.63)	49.84 (3.05)	**68.86(5.63)**
pang_movie	**66.29(0.80)** ▲	73.51 (0.26) ▼	53.71 (1.08)	69.46 (0.76)	59.27 (1.04)	48.47 (1.26)	56.49 (0.53)	**79.05(1.13)**
sanders	**68.41(5.70)** ●	**78.83(4.66)** ▲	57.01 (5.29)	71.72(5.35)	**69.29(3.15)**	49.65 (6.49)	**67.32(3.94)**	64.44 (6.88)
sarcasm	**63.35(12.31)** ●	**78.45(23.18)** ●	**52.50(6.26)**	**88.81(9.81)**	**53.67(27.25)**	**43.71(27.31)**	**68.10(18.03)**	**64.05(30.25)**
sentistrength_bbc	**42.24(7.31)** ▲	**62.07(4.77)** ●	19.41 (5.32)	**69.91(10.59)**	25.01 (12.48)	20.69 (5.75)	17.62 (3.53)	66.42 (8.99)
sentistrength_digg	**56.10(4.56)** ▲	**71.88(3.22)** ●	35.18 (4.84)	**71.02(10.45)**	41.30 (2.91)	36.53 (5.54)	36.50 (4.14)	62.03 (5.62)
sentistrength_myspace	**96.19(1.17)** ▲	70.06 (6.05) ▼	88.32 (4.06)	76.51 (3.92)	89.99 (2.70)	59.25 (2.94)	91.04 (3.04)	**80.24(3.85)**
sentistrength_rw	85.55 (4.06) ▼	83.42 (2.98) ▼	76.93 (3.14)	**87.59(4.07)**	**93.56(5.80)**	26.79 (3.14)	85.34 (3.08)	63.48 (3.36)
sentistrength_twitter	**82.55(2.49)** ▲	**75.46(3.47)** ●	67.75 (2.16)	**76.75(1.96)**	80.80 (0.99)	51.94 (2.77)	76.28 (1.68)	77.49 (2.19)
sentistrength_youtube	**88.68(1.12)** ▲	**78.58(1.94)** ●	77.82 (2.35)	**78.18(1.12)**	81.96 (2.45)	49.87 (1.70)	78.96 (1.80)	73.02 (1.65)
stanford_tweets	**82.32(8.46)** ▲	**82.81(4.42)** ▲	69.20 (8.43)	78.55 (3.02)	76.98 (7.10)	55.55 (8.86)	76.04 (8.05)	71.93 (5.48)
tweet_semevaltest	**90.19(1.50)** ▲	77.57 (1.46) ▼	83.92 (1.36)	**79.39(0.56)**	88.90 (1.29)	54.79 (2.34)	85.15 (1.39)	74.72 (0.75)
vader_amazon	72.82 (1.61) ▲	79.75 (0.98) ▲	61.28 (1.59)	66.77 (2.02)	69.59 (3.54)	29.96 (2.20)	68.60 (2.15)	48.90 (2.89)
vader_movie	**67.12(1.58)** ▲	**73.84(0.97)** ●	54.33 (1.94)	71.19 (1.37)	57.95 (0.86)	46.66 (1.39)	55.74 (1.72)	**74.29(0.67)**
vader_nyt	**68.15(3.57)** ▲	**63.23(1.08)** ▲	42.95 (2.61)	52.36 (2.54)	48.91 (2.56)	39.29 (1.65)	51.05 (1.40)	59.19 (2.47)
vader_twitter	**96.31(0.99)** ●	89.16 (1.01) ▼	**96.52(0.58)**	**94.00(1.15)**	85.81 (1.93)	51.90 (2.25)	85.27 (1.91)	73.28 (0.53)
yelp_reviews	70.07 (2.16) ▼	**96.95(0.73)** ●	66.67 (1.83)	**97.45(0.46)**	**90.65(3.43)**	18.45 (1.41)	83.61 (0.68)	84.99 (1.64)

Table 6.4: Precision and Recall Comparison for the three best lexicon-based baselines. Our strategy outperforms or ties with them in most cases.

6.3.4.2 Evaluation of Automatically Generated Lexicons

In this section, we compare the proposed k-NN regression with alternative methods to automatically expand the original manually-labeled VADER lexicon, as well as the traditional SentWordNet (in case of Synset-exp), for the task of sentence-level sentiment analysis. Table 6.6 presents results regarding MacroF1, showing that our method is the

Sentence	Covered words VADER lexicon	VADER shell given VADER lexicon	Covered words expanded lexicon	VADER shell given expanded lexicon	Ground truth
"I'm going to go out and get another gun... and lots of bullets."	gun	0.0	gun, bullets, out, lots	-0.317	Negative
"Leona Helmsley went to jail for tax evasion. Others get Cabinet positions. Hah! "	-	0.0	jail, tax, evasion, cabinet, positions	-0.6607	Negative
"Makes little difference? Look at homerun records. Obviously, being able to practice for much long periods (due to drug use) has an effect on reaction time and coordination."	-	0.0	difference, homerun, records, practice, periods, drug	-0.4106	Negative
"Dugg for the title."	-	0.0	dugg, title	+0.166	Positive
"I'm going to make a bundle on $0.99 downloads of my pet rock iPhone app."	-	0.0	bundle, pet, rock, app	+0.0822	Positive
"If people are willing to pay then good on them. Market forces, supply, and demand and all that"	good, demand	+0.25	good, demand, forces, supply, market	+0.13	Positive
"Err, it was a joke wasn't it? One mans joke is anothers Spit in the face"	joke	+0.5267	err, joke, spit, face	-0.6322	Negative

Table 6.5: Examples of sentences illustrating the coverage of original VADER lexicon and our proposed one. In all the examples, the VADER shell with the expanded lexicon could assign the the correct sentiment value to the sentence.

best strategy in 17 out of the 20 evaluated datasets. The highest gains of our method were 46.3%, 64%, 26.2%, 8.2%, 66%, and 38% over original VADER-lexicon, RoWE, Synset-exp, RF Regression, SENTPRO, and SSWE, respectively. We believe that the superiority of our strategy comes from the combination of several factors, including:

- The *higher quality and specificity* of the domain-specific VADER lexicons when compared to generic SentiWordnet used in Synset-exp.
- The advantages of using k-NN *local optimization*, when compared to the global optimization of RF Regression. More precisely, the k-NN Regression tries to optimize the overall regression by maximizing the accuracy of prediction for each individual word, instead of a global loss function such as RMSE or NDCG (in the case of ranking). Note, however, that RF regression was the runner-up, which shows that other aspects of our solution (e.g., the construction of the neighborhood) besides the regression method are also relevant to the observed difference.
- The *simplicity* and *capability of better filtering out noise* of our approach when compared to RoWE, SENTPRO, and SSWE. By combining the "opinion" of many neighbors,[7] we can deal more effectively with outliers (i.e., values that deviate from the general mean) and noisy word embeddings, which may occur due to the low representativeness of certain words in the datasets used to generate them. This mainly affects strategies that:

[7]Remind that, as seen in Section 6.3.2, better results are obtained with larger values of k.

– do not provide means for dealing with such noise (e.g., Synset-exp); or

– use very specific information – such as RoWE's seed with only one type of polarity, SENTPRO's random walk with small seeds, or SSWE's cluttered clusters.

Datasets	VADER-Lexicons	RoWE	Synset-exp	RF Regression	SENTPRO	SSWE	k-NN regression
aisopos_ntua	76.31(2.05)	82.54(4.26)	78.84(5.39)	82.03(2.97)	75.10(5.00)	75.75(3.44)	**86.97(2.89)** ▲
debate	48.19(2.12)	62.49(1.39)	57.12(2.18)	61.27(2.87)	58.78(2.13)	52.19(2.97)	**66.96(1.90)** ▲
english_dailabor	74.01(3.00)	74.87(3.81)	75.73(4.54)	82.65(2.46)	75.41(4.16)	74.68(3.36)	**84.46(1.76)** ▲
nikolaos_ted	61.75(1.59)	68.97(3.65)	61.94(3.40)	68.08(3.05)	65.63(4.50)	61.22(2.32)	**73.30(2.75)** ▲
pang_movie	53.80(0.94)	59.12(1.20)	60.79(1.00)	63.99(0.56)	61.93(1.12)	60.32(1.07)	**67.97(0.33)** ▲
sanders	60.50(3.81)	66.99(1.97)	64.67(3.82)	68.37(2.29)	63.32(4.40)	60.63(3.02)	**72.55(3.32)** ▲
sarcasm	50.99(13.41)	**65.47(14.48)**	55.44(9.24)	**74.10(8.96)**	53.10(12.74)	53.28(12.70)	**69.28(12.99)** ●
sentistrength_bbc	49.68(4.00)	63.66(9.60)	52.05(4.78)	61.65(3.82)	62.76(7.79)	50.77(4.47)	**70.25(3.24)** ▲
sentistrength_digg	55.61(3.35)	70.58(4.30)	58.09(4.03)	67.92(3.29)	67.13(5.58)	55.12(2.23)	**73.30(2.05)** ▲
sentistrength_myspace	58.42(3.69)	54.48(2.53)	65.37(3.11)	68.92(3.57)	67.01(1.24)	63.18(5.50)	**65.28(3.67)** ▲
sentistrength_rw	66.07(2.63)	46.34(1.23)	65.41(4.34)	71.28(4.83)	53.58(5.88)	60.78(4.84)	**76.02(3.76)** ▲
sentistrength_twitter	62.64(1.80)	67.16(1.74)	66.93(1.37)	72.65(1.44)	65.04(1.16)	64.30(1.56)	**75.88(2.75)** ▲
sentistrength_youtube	64.97(1.87)	70.22(0.99)	67.55(1.76)	**76.12(0.81)**	70.14(1.59)	66.67(2.37)	**76.45(1.72)** ●
stanford_tweets	70.92(4.84)	72.15(4.97)	73.66(4.17)	78.40(4.93)	72.32(6.70)	70.76(3.96)	**82.30(5.81)** ▲
tweet_semevaltest	68.58(0.97)	53.34(2.84)	70.21(1.83)	73.80(0.77)	69.25(1.71)	69.68(1.03)	**74.45(1.48)** ▲
vader_amazon	52.92(1.96)	57.53(1.05)	60.96(2.50)	66.75(1.89)	57.17(1.82)	56.83(1.69)	**68.76(1.55)** ▲
vader_movie	55.07(1.07)	59.80(0.44)	61.66(1.16)	64.43(1.14)	62.26(0.89)	61.21(0.87)	**69.02(0.89)** ▲
vader_nyt	47.70(1.48)	51.45(0.52)	58.82(1.34)	68.39(2.45)	61.40(0.43)	56.43(1.90)	**69.79(1.57)** ▲
vader_twitter	**92.54(0.83)**	75.85(1.17)	86.57(1.14)	89.04(0.73)	88.08(1.28)	88.95(1.28)	88.96(0.77) ▼
yelp_reviews	72.90(1.43)	69.63(1.07)	70.92(1.52)	71.04(0.73)	46.33(1.29)	67.33(1.23)	**76.92(1.40)** ▲

Table 6.6: Average MacroF1 results (and their standard deviations) of methods that automatically expand the $\mathcal{D}$ dictionary and the original VADER strategy. The proposed k-NN regression strategy outperforms or ties with the alternatives in eighteen of twenty datasets.

6.4 Cluwords Concept for Sentiment Analysis, named CluSent

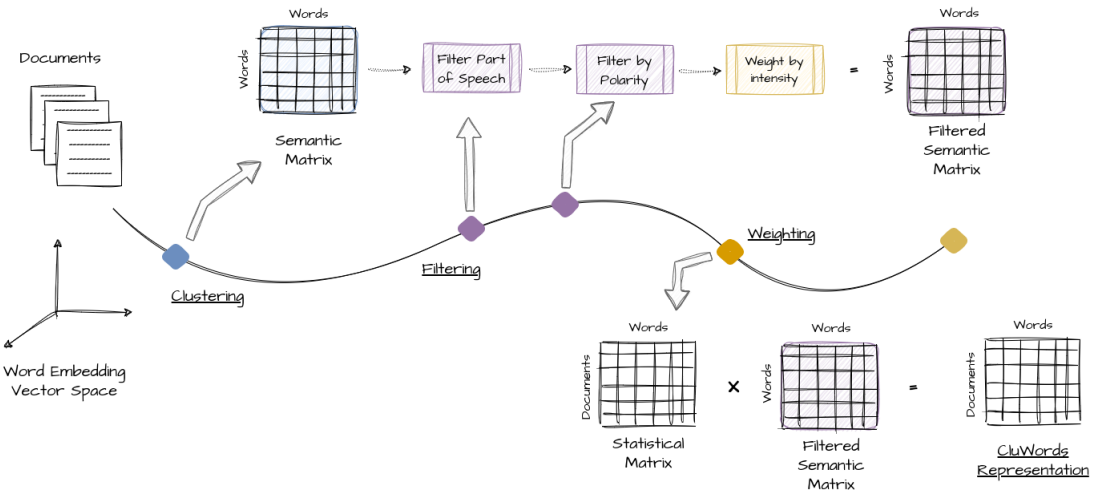


Figure 6.10: Diagram showing the steps for building the CluSent representation.

Conceptually, CluSent is built by applying three generic steps to a given source text representation: clustering, filtering, and weighting to build a richer (more informative) representation for a textual collection. Figure 6.10 illustrates how CluSent representations are instantiated for a given collection. Each dot in the Figure represents an instantiation of a method applied to compose the CluSent representation. In a nutshell, CluSent exploits clusters of semantically related word embeddings [59] built through the application of distance functions (first blue dot in Figure 6.10) and filtering mechanisms (second and third dot in Figure 6.10). More than simple groups of (filtered) related words, CluSent is coupled with specific weighting schemes[8] used to capture their importance to sentiment analysis tasks (purple dots in Figure 6.10). In Section 6.4.1, we present the clustering solution. Next, we describe (Section 6.4.2) the CluSent's part-of-speech filtering method followed by (Section 6.4.3) the filtering and weighting steps that exploit sentiment information and are used to build the document representation (Section 6.4.4).

6.4.1 Clustering

Let $\mathcal{W}$ be the set of vectors representing each word t in the dataset vocabulary (represented as $\mathcal{V}$). Each word $t \in \mathcal{V}$ has a corresponding vector $u \in \mathcal{W}$. The semantic matrix in the Figure 6.10 is defined as $C \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$, where each dimension has the size of the vocabulary ($|\mathcal{V}|$), t' represents the rows of C while t represents the columns. Finally, each index $C_{t',t}$ is computed according to Eq. 6.1.

$$C_{t',t} = \begin{cases} \omega(u_{t'}, u_t) & \text{if } \omega(u_{t'}, u_t) \geq \alpha \\ 0 & \text{otherwise,} \end{cases} \tag{6.1}$$

Where $\omega(u_{t'}, u_t)$ is the cosine similarity defined in Eq. 6.2 and *alpha* is a similarity threshold that acts as a regularizer for the representation. Larger values of α lead to sparser representations. In this notation, each column t of the semantic matrix C will form a CluWord t. Each value of the matrix $C_{t',t}$ will receive the cosine similarity between the vectors $u_{t'}$ and u_t in the embedding space $\mathcal{W}$ if it is greater than or equal to α. Otherwise, $C_{t',t}$ receives zero, according to the Eq. 6.1.

$$\omega(u_{t'}, u_t) = \frac{\sum_i^l u_{t'i} \cdot u_{ti}}{\sqrt{\sum_i^l u_{t'i}^2} \cdot \sqrt{\sum_i^l u_{ti}^2}} \tag{6.2}$$

[8]These weighting schemes combine the raw document representation with relevant information, such as semantic and/or lexicon information.

The vector $\overrightarrow{C_{,t}}$ represents the semantic information of a *cluster of words* (aka CluWord) t, and the α value filters potential noisy words (i.e., words that do not have a significant relationship with t). Since threshold α is a cosine similarity value, it is contained within the interval $[0, 1]$. If $\alpha = 0$, the similarities of every term in $\mathcal{V}_T$ are included in the CluWord t. If $\alpha = 1$ only the similarity of t to itself (i.e. $\omega(u_{t'}, u_t)$) is included in CluWord t. Thus, selecting a parameter α value is important for generating "good" CluWord t. Moreover, α controls the sparsity of the resulting document representation. With high α values, only a few CluWord terms relate to a document. This representation is similar to the traditional BoW representation, where the occurrence of a word in a document determines whether that word will be used in the document representation. With low α values, more CluWord terms tend to be related to the document, reducing the document representation's sparsity. Once we select an appropriate value for α, each CluWord t keeps the values of similarities of the terms most similar to t according to the criteria (e.g., context, co-occurrence) established by the word embeddings.

6.4.2 Part-of-Speech Filtering

Here we describe the part-of-speech filtering mechanism used to smooth noise in the semantic matrix $C \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$. This filter removes pairs of words that do not belong to the same grammatical group. Thus, this filter keeps in a neighborhood of a CluWord t ($\overrightarrow{C_{,t}}$) only terms (t') that have a semantic similarity and share the same grammatical group. The intuition is that, for the sake of sentiment analysis, we want to keep adjectives that are semantically similar to other adjectives, verbs that are semantically similar to other verbs, same for adverbs, and so on. We will analyze the impact of this very conservative filter in our experiments.

Formally, the Part-of-Speech (PoS) filtering method uses a function $pos(.)$ to filter each term t' of $\overrightarrow{C_{,t}}$ that does not belong to the same part-of-speech category of term t (Equation 6.3). We exploit the Spacy[9] part-of-speech tagger available for the English language to build function $pos(.)$.

$$C_{t',t} = \begin{cases} C_{t',t} & \text{if } pos(t) = pos(t') \\ 0 & \text{otherwise,} \end{cases} \tag{6.3}$$

Since the semantic matrix C is sparse, the search terms in a sparse matrix ($\mathbb{R}^{|\mathcal{D}|\times|\mathcal{V}|}$) representation and the complexity of those searches are $\mathcal{O}(NNZ)$, where NNZ represents

[9] https://spacy.io

the non-zero values.

6.4.3 Sentiment Filtering and Weighting

Many sentiment analysis approaches to a sentence or document use a lexicon dictionary. A lexicon is formed by a set of words tagged with their respective *sentiment value*, consisting of a number (within a defined range) that expresses both: the words' polarity (given by the number's sign) and intensity (given by number's absolute value). The intuition for the CluSent is to use information from a lexicon dictionary as another filter to remove semantic noise that can affect the quality of the representation, especially in the sentiment analysis scenario. Words with opposite polarities may be co-located in the same neighborhood of a CluWord t since the semantic similarity of embeddings correlated with positional, contextual, and co-occurrence information does not consider the polarity of a word. Thus, words of opposite polarities may belong to the same CluWord. Indeed this phenomenon has been observed in the literature [93]. We use this filter to keep *polarity consistency* within a CluWord. We also exploit the lexicon's word intensity as a weighting scheme to enhance the semantic information within a CluWord.

More formally, the lexicon dictionary is represented as $\mathcal{L} = \{\langle w_1, v_1, \rangle, \cdots, \langle w_{|\mathcal{L}|}, v_{|\mathcal{L}|}\rangle\}$, where w_i is a word and v_i is the sentiment value of word w_i, $1 \leq i \leq |\mathcal{L}|$. The sentiment value v_i of a word w_i expresses both word's polarity and intensity. The sentiment absolute values may vary according to the lexicon used. In *CluSent*, we use an expanded version of the VADER [37] lexical dictionary [93], described in Section 6.3.3, where the sentiment absolute values range between $(-4, 4)$. Given the semantic matrix, C, the method exploits Equation 6.4 to filter terms t' of $\overrightarrow{C_{,t}}$ that does not share the same polarity as term t. In addition, the sentiment value of the term t' is used to weight the semantic value $C_{t',t}$.

$$C_{t',t} = \begin{cases} C_{t',t} \times v_{t'} & \text{if } sign(v_t) = sign(v_{t'}) \\ 0 & \text{otherwise,} \end{cases} \tag{6.4}$$

Since the semantic matrix C is sparse, the search terms in a sparse matrix ($\mathbb{R}^{|\mathcal{D}|\times|\mathcal{V}|}$) representation, and the complexity of those searches are $\mathcal{O}(NNZ)$, where NNZ represents the non-zero values.

6.4.4 Building the CluSent Representation

This step is responsible for building the CluSent representation (the last yellow dot in Figure 6.10) is defined as the product between the term-frequency matrix and semantic matrix C. The term-frequency matrix (TF) can be represented as a $TF \in \mathbb{R}^{|\mathcal{D}|\times|\mathcal{V}|}$, where each position $TF_{d,t}$ relates to the frequency of a word t in document d. Thus, given a CluSent (CS) term t for a document d, its data representation corresponds to

$$CS_{d,t} = \overrightarrow{TF_d} \times \overrightarrow{C_{,t}} \tag{6.5}$$

where $\overrightarrow{TF_d}$ has the term-frequencies of document d, and $\overrightarrow{C_{,t}}$ is the semantic scores for the term t.

6.5 Experimental Setup

6.5.1 Textual datasets

To evaluate the quality of the proposed methods, we adopt nineteen real-world textual datasets gathered from various sources, such as the highly popular SEMEVAL (semeval_tw) [77], stanford_tw [28] and Stanford Sentiment Treebank v2 (SST-2)[10] datasets. Besides those, we exploit 16 other datasets with various news, reviews, and social media domains with different characteristics, such as class distribution, density, etc. These datasets have high relevance for sentiment analysis, used, for instance, in a very popular benchmark of unsupervised methods [76] as well as in highly cited papers such as [37], that proposed the VADER lexicon.

Table 6.7 shows some characteristics of these 19 datasets. Each column depicts the dataset's name, number of *messages*, number of words, the average number of words (density) in each message, and the number of positive and negative messages. As we can see, most datasets are highly imbalanced, i.e., they have a skewed distribution increasing the bias toward the largest class.

[10] https://www.kaggle.com/atulanandjha/stanford-sentiment-treebank-v2-sst2

Dataset	#msgs	#feat	density	#pos	#neg
aisopos_tw	278	1,586	83.60	159	119
debate	1,979	4,179	86.49	730	1,249
narr_tw	1,227	4,002	74.76	739	488
pappas_ted	727	1,886	92.16	318	409
sanders_tw	1,091	3,601	97.08	519	572
ss_bbc	752	7,674	396.82	99	653
ss_digg	782	5,164	188.49	210	572
ss_myspace	834	2,914	104.26	702	132
ss_rw	705	5,643	345.02	484	221
ss_twitter	2,289	8,835	94.19	1,340	949
ss_youtube	2,432	7,534	90.04	1,665	767
stanford_tw	359	1,746	81.62	182	177
semeval_tw	3,060	10,507	115.99	2,223	837
vader_amzn	3,610	5,039	88.54	2,128	1,482
vader_movie	10,568	17,759	111.67	5,242	5,326
vader_nyt	4,946	12,932	105.42	2,204	2,742
vader_tw	4,196	9,046	79.69	2,897	1,299
yelp_review	5,000	25,494	681.46	2,500	2,500
SST-2	68,221	14,583	53.17	38,013	30,208

Table 6.7: Dataset characteristics

6.5.2 Evaluation, Algorithms, and Procedures

The effectiveness of the experiments was evaluated using two standard text categorization measures: $MicroF1$ and $MacroF1$ [47]. While $MicroF1$ measures the classification effectiveness of overall decisions, $MacroF1$ measures the classification effectiveness for each class and averages them. $MacroF1$ is very suitable for datasets with high imbalance as all classes have the same importance in the measure.

All experiments were executed using a 5-fold cross-validation procedure. All tuning parameters for the baselines and our methods were discovered in the validation partitions. At the same time, the reported results correspond to the average on the 5 test sets of the folded cross-validation procedure.

We use as baselines popular and SOTA methods such as RNTN, NB-weighted-BON+dv-cosine, kNN Regression Expansion, and L-MIXED, based on their performance on public benchmarks. In one of these benchmarks [53], L-MIXED produced the best-known results in the literature in some of the tested datasets, considered a SOTA baseline in the field. We also consider BERT a solid baseline since it was surpassed only marginally (without statistical significance) by another recent SOTA baseline (SentiBERT), which could not be used in our experiments due to a lack of code and reproducibility information in the original paper. Finally, we also adopted the kNN Regression Expansion, a recent and effective sentiment analysis SOTA baseline especially designed for short-text datasets, as is the case of most experimented datasets [93].

For BERT, we configured hyperparameters as suggested by the authors [21]. We performed a search for the best hyperparameters following a trial-and-error process, and

the best set for the remaining ones was chosen with fine-tuning using nested cross-validation within the training sets (batch size: 32, initial learning rate: 5e-5, max sequence length: 150 tokens, max patience: 5 epochs). For other baselines, we performed fine-tuning according to the appropriate author's scripts in the source code. For RNTN, the hyperparameter word vector size, learning, and mini-batch size are adjusted with the AdaGrad algorithm, while the activation function is hyperbolic tangent. For NB-weighted-BON+dv-cosine and L-MIXED, we used grid search to optimize the number of iterations, learning rate, and regularization force. For kNN Regression Expansion, we exploited the pre-trained FastText embedding and performed fine-tuning of neighbors according to the author's script in the source code.

For CluSent, we consider the pre-trained FastText embedding [11] to build the semantic matrix, described in Section 6.4. FastText is essentially an extension of the Word2Vec model, which treats each word as composed of character n-grams, allowing to (i) generate better word embeddings for rare words and (ii) construct word vectors for a word that does not appear in the training corpus. Both improvements are not implemented in GloVE.

The α parameter (in Eq. 6.1 Section 6.4.1) is strictly sensitive to the embedding space, being responsible for controlling the CluSent's density. The smaller the alpha value, the greater the CluSent representation's density. A small alpha may increase the noise in the CluSent representation, while a large alpha may impoverish it. We adopted a percentile-based strategy to select the 5% of word pairs with the highest cosine similarity scores in the embedding space. This process was performed empirically over the FastText embeddings.

We run nested cross-validation over the training set to select the best CluSent instantiation for each dataset. In other words, whether to use the PosTagging filtering and the TF-AL weighting and filtering mechanisms are determined per dataset with nested-cross validation in the training set. We exploit the Linear SVM classifier in the CluSent, a top-notch method for text classification that is even superior to neural architectures such as BERT when faced with information shortage [18]. The regularization parameter was chosen among eleven values from 2^{-5} to 2^{15} using 5-fold nested cross-validation within the training set.

We assess the statistical significance of our results by exploiting a Two-way ANOVA test with 95% confidence. This test assures that the best results, marked with a green triangle (▲), are statistically superior to all others. Statistical ties are represented as a yellow dot (●), while losses are represented as red downward triangles (▼).

[11] https://dl.fbaipublicfiles.com/fasttext/vectors-english/wiki-news-300d-1M.vec.zip

6.6 Experimental Results

Dataset	BERT	NB-weighted-BON + dv-cosine	RNTN	L-MIXED	kNN Regression Expansion	CluSent
aisopos_tw	**86.73**	**84.74**	63.63	83.58	82.95	**87.74**●
debate	73.79	66.42	62.4	**77.41**	61.53	**75.13**●
narr_tw	79.71	63.42	74.12	**82.48**	**83.46**	**86.50**●
pappas_ted	73.52	74.85	63.42	**77.64**	65.43	**78.82**●
sanders	**78.07**	76.29	68.02	**80.47**	69.81	**80.37**●
ss_bbc	55.99	46.48	55.55	51.28	60.36	**68.94**▲
ss_digg	65.68	43.20	66.05	55.87	65.55	**71.07**▲
ss_myspace	61.02	45.67	62.47	49.88	**75.35**	**73.35**●
ss_rw	70.56	42.12	62.90	57.72	67.53	**75.62**▲
ss_twitter	72.21	55.99	68.17	**74.81**	**73.94**	**75.44**●
ss_youtube	76.55	54.40	71.31	**79.69**	**77.09**	**79.02**●
stanford_tw	75.70	72.88	77.52	**79.54**	**81.41**	77.07▼
semeval_tw	**74.09**	48.60	68.92	68.37	**75.52**	**76.51**●
vader_amzn	**71.48**	62.85	69.33	**73.89**	62.49	**71.94**●
vader_movie	78.09	76.59	75.31	**82.63**	64.59	75.11▼
vader_nyt	**65.56**	53.19	60.92	**66.92**	**66.00**	**65.56**●
vader_tw	81.92	61.23	71.67	82.53	**89.25**	**89.63**●
yelp_review	**94.08**	**93.30**	74.33	**94.59**	62.46	**92.36**●
SST-2	**94.39**	86.87	82.75	**93.13**	55.11	89.02 ▼

Table 6.8: MacroF1 results. CluSent is the best method (winning or tying) in 16 out of 19 datasets.

Table 6.8 shows the MacroF1 effectiveness results. Best results in all datasets (including ties) are marked in **bold**. As we can see, CluSent is the best overall method – it outperforms the baselines with three overall wins (statistically superior results over all others ▲) and 13 ties in first (best) place (●), considering the 19 datasets. In other words, CluSent was the best method in 16 out of 19 cases. L-MIXED was the strongest baseline, with 12 ties, five losses, and only two wins when directly compared with CluSent. Remind that L-MIXED is considered a solid SOTA baseline in public benchmarks. BERT and kNN Regression Expansion lost to CluSent in most cases (9 and 10 losses), with nine and eight ties, respectively. BERT only surpassed Clusent in *SST-2*, tying with L-MIXED, while KNN Regression outperformed CluSent only in Stanford_tw. In cases in which CluSent outperformed the *best baseline in each dataset*, it did by large margins, such as in *ss_bbc* with gains of 14.21% over KNN Regression, 7.60% in ss_digg over RNTN, and 7.17% in *ss_rw* over BERT. Among the three CluSent's losses, one was only against L-MIXED (in *vader_movie*), *stanford_tw* against L-MIXED and kNN Regression Expansion and *SST-2* against BERT and L-MIXED.

Figure 6.11 shows the effectiveness of the results in terms of MicroF1. In this scenario, CluSent tied for first place in 14 out of 19 cases, twelve with L-MIXED, the strongest baseline in terms of MicroF1. This result makes CluSent the best overall method along with L_MIXED, as detailed in Table 3. The slightly better CluSent's MacroF1 results when compared to MicroF1 may be due to the high skewness (class imbalance) of some datasets (e.g., *debate*, *ss_bbc*, *ss_myspace*). When faced with an information

shortage, there is a tendency to increase the classifier's natural bias towards the largest class. The CluSent semantic expansion helps counterbalance this natural bias, making the classification fairer to the minority class. This fact is better reflected in the MacroF1 scores. However, further investigation of this hypothesis is necessary to confirm it.

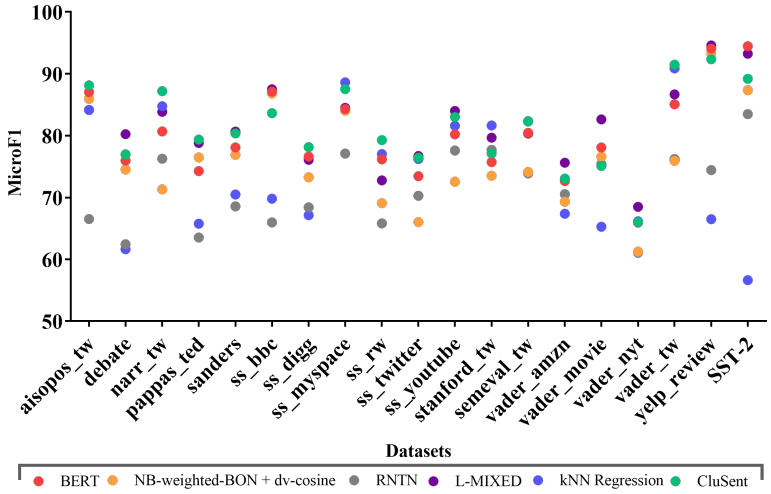


Figure 6.11: MicroF1 results. CluSent is the runner-up method (winning or tying) in 14 out of 19 datasets.

To summarize the results of Table 6.8, we analyze using *Fractional rankings* to determine the most effective overall method across the multiple datasets. In Fractional rankings, items that perform equally (i..e, statistical ties) receive the same ranking number, the mean of the ranking they would receive under ordinal rankings considering the ties. In our scenario, we rank each method for each dataset based on the MacroF1 score and the statistical tests. As mentioned, ties receive the same rank position. Table 6.9 shows the fractional ranking for the MacroF1 results, and the last row, called Aggregated (Aggr.) ranking, is the ranking summation of all datasets' rankings for each method. For instance, in *ss_bbc, ss _digg and ss_rw* where CluSent is the sole best method with no tie, it receives a ranking of 1 while in *narr_tw, pappas_ted, ss_myspace, and vader_tw*, where Clusent ties as the best method with another baseline, it receives a ranking of 1.5 (Rank: 1.5, 1.5, 3, ...).

As can be seen in the Aggregated Ranking, CluSent is by far the best overall method (lowest aggregated ranking: 41.5) considering the 19 datasets, with BERT coming in a distant second place (Aggr. ranking: 52.0). This analysis emphasizes CluSent's consistency across many different domains, captured by the different datasets.

Dataset	BERT	NB-weighted-BON + dv-cosine	RNTN	L-MIXED	kNN Regression Expansion	CluSent
aisopos_tw	2.0	2.0	5.5	4.0	5.5	2.0
debate	2.0	5.0	6.0	2.0	4.0	2.0
narr_tw	4.0	6.0	5.0	3.0	1.5	1.5
pappas_ted	3.0	4.0	6.0	1.5	5.0	1.5
sanders	2.0	4.0	6.0	2.0	5.0	2.0
ss_bbc	3.0	6.0	4.0	5.0	2.0	1.0
ss_digg	3.0	6.0	2.0	5.0	4.0	1.0
ss_myspace	4.0	6.0	3.0	5.0	1.5	1.5
ss_rw	2.0	6.0	4.0	5.0	3.0	1.0
ss_twitter	4.0	6.0	5.0	2.0	2.0	2.0
ss_youtube	2.5	6.0	5.0	2.5	2.5	2.5
stanford_tw	3.0	6.0	4.0	1.5	1.5	5.0
semeval_tw	2.0	6.0	4.0	5.0	2.0	2.0
vader_amzn	2.0	5.0	4.0	2.0	6.0	2.0
vader_movie	2.0	3.0	4.0	1.0	6.0	5.0
vader_nyt	2.5	6.0	5.0	2.5	2.5	2.5
vader_tw	4.0	6.0	5.0	3.0	1.5	1.5
yelp_review	2.5	2.5	5.0	2.5	6.0	2.5
SST-2	1.5	4.0	5.0	1.5	6.0	3.0
Aggr. Ranking	**52.0**	**95.5**	**87.5**	**56.0**	**67.5**	**41.5**

Table 6.9: Fractional Rank for MacroF1 results. CluSent is the best overall method in the Aggregated Ranking.

6.6.1 Difficult cases solved by CluSent

As an example of a problematic case that CluSent can handle and other methods can not, in ss_bbc, the raw negative document "that's why the meeting may well be just a joke" has been misclassified by CluSent's base classifier (Linear SVM). CluSent expanded the original document representation into a vector with 47 non-zero new dimensions related to the semantic neighborhood, including new words such as "silly" and "apology". This information and the weighting step allowed it to correct the misclassification.

Another example in the same dataset is the document "Science once again ignored by the mainstream so they can continue to collect dollars with the marketing of the green business agenda.". Comparing the CluSent with Linear SVM, we observe that CluSent added more negative information, such as "abandoned", "blinded", and "blurred". The filters also removed positive words in the same neighborhood, i.e., no positive words were added. Both actions helped to correct SVM's misclassification.

6.6.2 CluSent Instantiation

This section presents the experimental setup and how we instantiate the CluSent using the approaches described in Section 6.4.

We experiment to observe the impact of varying the instantiation methods, described in Section 6.4 in the CluSent representation. The intuition of this experiment is to evaluate different forms for instantiating the method and how they are impacted in terms of effective results in the evaluated datasets. Since the effectiveness of filtering and weighting methods may differ on distinct datasets or samplings, we propose to present an instantiation version of CluSent, in which the specific form of instantiation is chosen automatically through tuning, using nested cross-validation in the training set. In other words, in CluSent, the filtering/weighting steps that will be turned on/off are automatically chosen, potentially different for each dataset.

Each evaluated instantiation is seen in Table 6.10, where the effectiveness of the results was performed over nested cross-validation over the training (a.k.a effectiveness results of the grid search). Each column in the Table represents a different instantiation described as follows: (i) CW – is the core of the CluSent representation, corresponding to the instantiation of the clusterization method (Section 6.4.1), and the merge between Term Frequency and Semantic information (Section 6.4.4); (ii) CW + PoS – adds to the previous CluSent instantiation CW, the Part-of-Speech filtering method (Section 6.4.2); (iii) CW + TF-AL – builds the CluSent representation adding the core methods (Sections 6.4.1 and 6.4.4), and the sentiment filtering and weighting technique (Section 6.4.3); CW + PoS + TF-AL – turns on all components to build the representation; (iv) CluSent – is the automatic instantiation of the Part-of-Speech and Sentiment filtering and weighting approaches that chooses the best components to turn on/off for each individual dataset based on tuning performed by means of nested cross-validation over the training set.

Table 6.10 shows the effective results regarding MacroF1. Besides the explained marks for the statistical tests (▲, ●,▼) best results in all datasets (including ties) are also marked in **bold** in the Table. The results showed that CluSent ties with the best manual CluSent instantiation in each dataset, the only instantiation to obtain the best effectiveness in all 19 experimented datasets. In the next sections, we will adopt CluSent as the method of choice to compare against the baselines.

Finally, for analysis purposes, we added in Table 6.10 the instantiated components (marked with ×) selected in CluSent's tunning process. We can see that, when the PosTagging component is turned on, the TF-AL filtering/weighting is also selected. A few cases in which only the TF-AL component is selected, such as *narr_tw, stanford_tw, and vader_tw*. In these datasets, the PosTagging filtering, which is very conservative, tends to be detrimental. Finally, both components are turned off in a few other datasets, such as pappas_ted and sanders. In these datasets, these components tend not to have much impact.

Dataset	CW	CW + PoS	CW + TF-AL	CW + PoS + TF-AL	CluSent		
					MacroF1	Instantiations	
						PoS	TF-AL
aisopos_tw	**86.95**	83.61	**87.71**	**89.38**	**87.74** •	×	×
debate	**74.50**	**75.23**	**75.73**	**75.76**	**75.13** •	×	×
narr_tw	**84.77**	82.67	**86.51**	**85.15**	**86.50** •		×
pappas_ted	**78.35**	**77.75**	**77.53**	**77.86**	**78.82** •		
sanders	**81.61**	**80.80**	**81.36**	**80.06**	**80.37** •		
ss_bbc	**68.19**	64.12	**67.35**	**66.03**	**68.94** •	×	×
ss_digg	71.73	**66.80**	71.86	71.85	**71.07** •	×	×
ss_myspace	**74.76**	70.71	**71.86**	70.47	**73.35** •	×	×
ss_rw	76.78	**70.81**	76.41	77.03	**75.62** •	×	×
ss_twitter	**76.73**	**75.93**	**77.16**	**76.49**	**75.44** •		
ss_youtube	**82.14**	**80.36**	**79.94**	**79.23**	**79.02** •	×	×
stanford_tw	75.93	75.93	**79.79**	**79.89**	**77.07** •		×
semeval_tw	**76.29**	**75.80**	**76.99**	**76.94**	**76.51** •		
vader_amzn	**69.26**	**68.99**	**71.64**	**72.22**	**71.94** •	×	×
vader_movie	**75.03**	**75.47**	**73.59**	**74.65**	**75.11** •	×	
vader_nyt	**66.15**	**65.69**	**66.56**	**66.41**	**65.56** •		×
vader_tw	86.62	86.41	**89.64**	**90.05**	**89.63** •	×	×
yelp_review	**92.72**	**92.54**	**92.10**	**92.14**	**92.36** •		
SST-2	**88.49**	**88.49**	**88.22**	**88.25**	**89.02** •		

Table 6.10: MacroF1 results of different CluWords instantiations combined with the linear SVM classifier. CluSent corresponds to the results of an automatic instantiation of the CluWords performed by nested cross-validation over the training set.

Dataset	Density				CluSent Density	
	CW	CW + PoS	CW + TF-AL	CW + PoS + TF-AL	Positive	Negative
ss_bbc	3,916	2,608	1,957	1,192	484	708
ss_digg	1,964	1,167	910	488	242	246
ss_myspace	1,126	661	510	268	204	64
ss_rw	2,745	1,758	1,328	770	519	251
vader_movie	4,338	2,164	1,649	766	-	-

Table 6.11: Density analysis of the CluWord Instantiations

6.6.3 Density of CluSent Representation

To better explain these results, we observe in Table 6.7 that the datasets in which CluSent outperforms all baselines (*ss_bbc, ss_digg, ss_myspace, and ss_rw)*, some by large margins, are small and have high skewness (class imbalance higher than 50%). Also, in these datasets, the number of words is much higher than the number of documents, suggesting that the words are infrequent within the documents (see column density in Table 6.7). on its turn, Table 6.11 shows the document density for some CluSent instantiations. We can see that the density increases considerably, regardless of the instantiation. For instance, in *ss_myspace*, the density of documents increases by at least 1822%. This is a direct consequence of CluSent's semantic expansion.

Observing the execution logs, we notice also that in the CW + TF, CW + Pos + TF-AL and in the CluSent instantiations, the sentiment-based filtering and weighting mechanism (Section 6.4.3) was added (turned on) in *ss_bbc, ss_digg, ss_myspace, and ss_rw.* In Table 6.11, the document density of the CluSent instantiation was further broken down into two sentiment polarities (positive and negative). The TF-AL instantiation makes it possible to identify the polarity of words based on the lexical dictionary used by the

method. Thus, we explore this information to measure the average density of documents according to the polarity. Notice that the positive and negative densities add up to the density of CW + PoS + TF-AL, the most complete instantiation. As we can see, the density was reduced in these cases due to the *noise filtering* mechanism, but it is still much higher than in the original representation. This justifies the robustness of CluSent in capturing the best document representation – semantically expanded, with noise removal and sentiment-based weighting, ultimately justifying its effectiveness.

Table 6.11 also includes information on the dataset in which CluSent did not surpass the baseline (*vader_movie*). We can see in Table 6.7 that these datasets are larger, more balanced, and have the number of documents closer to the number of words, on average. In other words, they suffer less from information shortage problems. When looking at the CluSent instantiations, we observed that in both datasets, there was a similar increase in average density due to expansion. However, observing again the execution logs, the sentiment-based filtering/weighting was turned off in the CluSent instantiation for the dataset – *vader_movie* (indicated as '-' in the Table 6.11). This may suggest that the CluWord expansion in these datasets produced a less noisy representation, in which the noise filtering mechanism by polarity did not produce much impact. However, this expansion did not surpass the baselines, which took advantage of this dataset's larger information and less sparsity. This indicates room for further improvements in expansion and filtering strategies in the CluSent.

6.7 Chapter Summary

Firstly, this chapter provides formal hypotheses supported by strong empirical and experimental evidence showing promising insights to explore CluWords in Sentiment Analysis. In addition, we proposed a new, simple, yet very effective technique for expanding human-built lexicons. Our method can use the general representation of words provided by word embeddings and their relationships (captured by simple distance computations) to produce **high coverage** lexicons that significantly improve accuracy. Using the extended lexicon generated with our method to predict the sentiment of sentences, we achieved gains in MacroF1 of up to 26% and 38% against RoWE and SENTPRO, two of the closest baselines.

Secondly, this chapter presents a new instantiation of the CluWords for sentiment analysis – CluSent – that exploits semantic expansion and tackles information shortage and noise issues. CluSent representation is built by a dynamic pipeline of instantiations to build dataset-oriented document representations. It combines supervised and unsu-

pervised solutions, using external information from word embeddings and unsupervised lexicons. In our experiments, CluSent outperformed the evaluated baselines in 30 out of 38 possibilities, excelling in a Fractional Ranking aggregated analysis, with gains of more than 14% against some of the strongest baselines.

We envision plenty of future work ahead of us. The CluSent instantiation showed promising insights in terms of how well the CluWords concept can deal with designing filtering and weighting schemes to mitigate noise from the data representation. However, we don't believe that we take the full potential of the semantic de-noise that the CluWords can take from the data representation. Class information can be explored by selecting the most discriminative words of each class that will compose the Cluwords, filtering out potential noise, and making a per-class expansion of the vocabulary. CluWords can also take advantage of contextual word embeddings (i.e., BERT), however, these models generate several embeddings for each pair (word, sentence), and we must investigate the best ways to pool these embeddings in the context of each Cluword. However, pooling embeddings can introduce certain drawbacks, such as loss of information, and difficulty in handling noisy data.

Chapter 7

Conclusion

In this chapter, we summarize the research contributions of this Ph.D. thesis and point out some directions for further investigation.

7.1 Topic Modeling

We introduced the CluWords concept, presenting its three main steps - clustering, filtering, and weighting. We introduced the application of its CluWords in topic modeling. The designed CluWords exploit large word embedding spaces in this domain to cluster semantic relationships exploiting cosine similarity. It also exploits a filtering strategy to conjugate into a single representation syntactic and semantic information. Finally, it exploits an adaptation of the TF-IDF weighting scheme to measure (i.e., weight) the importance of a given CluWord to express the topics of a document.

Our thorough experimental evaluation (12 datasets, eight baselines, two evaluation metrics, three topic lengths) in this domain showed that we outperform the best (state-of-the-art) methods for topic modeling known in the literature, with a much smaller variability regarding the quality of the produced topics. A test for equality of variances concerning the NPMI scores of the best methods, CluWords and SeaNMF, showed that both methods have different variances. Thus, we can answer our first research question *(i) Can we exploit the CluWords to enhance document representation for topic modeling?* Both experimental evaluation and equality experiments show that the CluWords can generate more cohesive topics. Revising the first research question regarding the Topic Modeling context – *(i) Can we exploit the CluWords to enhance document representation for topic modeling?* – Our experimental evaluation showed great evidence that the CluWords could enrich the document representation.

7.2 Hierarchical Topic Modeling

We introduced a novel unsupervised non-probabilistic method – CluHTM. Our new method exploits the global semantic information provided by the CluWords representation and an original application of a stability measure to define the "shape" of the hierarchy. We present two variants of the CluHTM method – (i) f-CluHTM, which exploits pre-trained static embeddings to build the CluWords representations, and (ii) c-CluHTM, which exploits the Contextual word embeddings from BERT-transformed into static version using pooling approaches to build the CluWords representations. CluHTM variants excelled, being around twice as effective as the strongest state-of-the-art baselines, considering all tested datasets and evaluation metrics. The overall gains over some of these strongest baselines are higher than 500% in some datasets. In addition, we present new evaluation metrics for evaluating hierarchical topic modeling methods. The newly proposed topic quality metrics assess aspects related to topological consistency (or redundancy) and the hierarchical semantic structure that are important to hierarchical methods. These are different and complementary aspects than those captured by traditional TM metrics such as NPMI and Coherence, which measure the quality of each topic in an isolated manner, disregarding topological relationships among topics. Our new proposed topic quality metrics capture distinct behaviors from the topics built, including duplicity of topics built by some HTM methods. Our results also show that c-CluHTM and f-CluHTM present the best results in building a hierarchical structure while avoiding redundancy.

Regarding the experimental results, we may say that we have positive evidence regarding the second research question about topic modeling: *(ii) Can CluWords add more information to hierarchical topic modeling models at deeper levels of the hierarchy?* Our experimental results show that the CluWords can improve the coherence among topics of the same hierarchy level. In addition, our proposed evaluation metrics showed that in a different perspective – *Topic topological consistency (or redundancy)* and *Semantic hierarchical structure* the CluHTM variants achieve better results when compared to the HTM baselines.

7.3 Sentiment Analysis

We provided a formal hypothesis, supported by strong empirical and experimental evidence, that shows promising insights to explore CluWords in the context of Sentiment Analysis. In addition, we proposed a new, simple, yet very effective technique for expanding human-built lexicons. Our method can use the general representation of words provided by word embeddings and their relationships (captured by simple distance computations) to produce **high coverage** lexicons that significantly improve accuracy. Using the extended lexicon generated with our method to predict the sentiment of sentences, we achieved gains in MacroF1 of up to 26% and 38% against RoWE and SENTPRO, two of the closest baselines.

We also present a new instantiation of the CluWords for sentiment analysis – CluSent – that exploits semantic expansion and tackles information shortage and noise issues. CluSent representation is built by a dynamic pipeline of instantiations to build dataset-oriented document representations. It combines supervised and unsupervised solutions, using external information from word embeddings and unsupervised lexicons. In our experiments, CluSent outperformed the evaluated baselines in 30 out of 38 possibilities, excelling in a Fractional Ranking aggregated analysis, with gains of more than 14% against some of the strongest baselines.

Our experimental results showed evidence to answer the research questions – **Sentiment Analysis:** *(i) Can the CluWords be used to overcome issues of lack of information in sentiment analysis tasks? (ii) Can polarity/intensity and Part-of-Speech (PoS) be used to filter out words from CluWords for sentiment analysis?*. Indeed, the filtering and weighting blocks that exploit PoS and the sentiment value can enrich the data representations. However, we don't believe we take the full potential of the semantic de-noise that the CluWords can take from the data representation. Class information can be explored by selecting the most discriminative words of each class that will compose the Cluwords, filtering out potential noise, and expanding the vocabulary per class. CluWords can also use contextual word embeddings (i.e., BERT). However, these models generate several embeddings for each pair (word, sentence), and we must investigate the best ways to pool these embeddings in the context of each Cluword. However, pooling embeddings can introduce drawbacks, such as information loss and difficulty handling noisy data.

7.4 Future Work

As for future work, in the context of topic modeling and hierarchical topic modeling, there is room to test new instantiations of the CluWords (i.e., filters similar to the ones designed for CluSent) and new embedding representations, especially multi-lingual embedding representations. We believe that CluWords has the potential to deal with multi-language domains. Also, we have started the study and the design of new ways of evaluating hierarchical topic modeling metrics, but there is more to be exploited in this field. We believe that existing metrics, such as those used for hierarchical clusters, could potentially be used for evaluating hierarchical topic modeling strategies.

In the context of sentiment analysis, we believe that exploiting contextual embedding representations can significantly improve the CluWords representation, but since CluWords requires static embeddings as the input. One possible way would be to extract meaning embedding representation through contextual embeddings since one word can have more than one meaning.

Finally, we also believe that CluWords has the potential to be exploited in other domains, such as recommendation systems and information retrieval. Regarding Recommendation Systems, there is plenty of matrix-based strategies, such as Factorization Machines and Context-Aware Recommender Systems (CARS) that easily can be combined to the CluWords representation. Regarding information retrieval, we believe that CluWords could be used as query expansion to improve the quality of search queries or even reduce the zero results in a search engine. In addition, we also believe that CluWords could be exploited by Representation-based Similarity strategies.

7.5 Summary

In sum, regarding the main research question in Chapter *(i) Can CluWords be effectively exploited to advance the state-of-the-art in NLP and Information Retrieval tasks?* Our experimental evaluation in topic modeling and hierarchical topic modeling suggests that the CluWords concept has the potential to advance the state-of-the-art in NLP. Concerning the second main research question *(ii) Are task-specific filtering and weighting effective mechanisms capable of adapting the CluWords to different NLP application scenarios?* the CluSent instantiation shows that it is possible to adapt the CluWords for a specific scenario, as well as, to build instantiation blocks that might mitigate the noise

of the data representation.

Bibliography

[1] Mohamad Alissa, Issa Haddad, Jonathan Meyer, Jade Obeid, Kostis Vilaetis, Nicolas Wiecek, and Sukrit Wongariyakavee. Sentiment analysis for open domain conversational agent, 2021.

[2] Matheus Araújo, Adriano Pereira, and Fabrício Benevenuto. A comparative study of machine translation for multilingual sentence-level sentiment analysis. *Information Sciences*, 512:1078 – 1102, 2020.

[3] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Senti wordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC'10*, 2010.

[4] Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *ACL'14*, 2014.

[5] M. S. Bartlett. Properties of sufficiency and statistical tests. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 160, 1937.

[6] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[7] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[8] Jelke Bloem, Antske Fokkens, and Aurélie Herbelot. Evaluating the consistency of word embeddings from small data. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 132–141, Varna, Bulgaria, September 2019. INCOMA Ltd.

[9] Johan Bollen, Bruno Gonçalves, Guangchen Ruan, and Huina Mao. Happiness is assortative in online social networks. *CoRR*, abs/1103.0784, 2011.

[10] M. M. Bradley and P. J. Lang. Affective norms for English words (ANEW): Stimuli, instruction manual, and affective ratings. Technical report, Center for Research in Psychophysiology, University of Florida, 1999.

[11] M. Bréal. *Essai de sémantique: science des significations./Michel Bréal.* Slatkine Reprints, 1924.

[12] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, October 2001.

[13] Erik Cambria, Soujanya Poria, Devamanyu Hazarika, and Kenneth Kwok. Senticnet 5: Discovering conceptual primitives for sentiment analysis by means of context embeddings. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1795–1802. AAAI Press, 2018.

[14] Zhiyuan Chen and Bing Liu. Topic modeling using topics from many domains, lifelong learning and big data. In *ICML'14*, pages II–703–II–711, 2014.

[15] X. Cheng, X. Yan, Y. Lan, and J. Guo. Btm: Topic modeling over short texts. *IEEE TKDE*, 26(12):2928–2941, Dec 2014.

[16] Rob Churchill and Lisa Singh. The evolution of topic modeling. *ACM Comput. Surv.*, 54(10s), nov 2022.

[17] Washington Cunha, Sérgio Canuto, Felipe Viegas, Thiago Salles, Christian Gomes, Vitor Mangaravite, Elaine Resende, Thierson Rosa, Marcos André Gonçalves, and Leonardo Rocha. Extended pre-processing pipeline for text classification: On the role of meta-feature representations, sparsification and selective sampling. *Information Processing and Management*, 57(4):102263, 2020.

[18] Washington Cunha, Vítor Mangaravite, Christian Gomes, Sérgio Canuto, Elaine Resende, Cecilia Nascimento, Felipe Viegas, Celso França, Wellington Santos Martins, Jussara M. Almeida, Thierson Rosa, Leonardo Rocha, and Marcos André Gonçalves. On the cost-effectiveness of neural and non-neural approaches and representations for text classification: A comprehensive comparative study. *IP&M*, 58(3):102481, 2021.

[19] Rajarshi Das, Manzil Zaheer, and Chris Dyer. Gaussian lda for topic models with word embeddings. In *ACL (1)*, pages 795–804, 2015.

[20] Claudio M.V. de Andrade, Fabiano M. Belém, Washington Cunha, Celso França, Felipe Viegas, Leonardo Rocha, and Marcos André Gonçalves. On the class separability of contextual embeddings representations – or "the classifier does not matter when the (text) representation is so good!". *Information Processing & Management*, 60(4):103336, 2023.

[21] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[22] Eduard Dragut, Hong Wang, Clement Yu, Prasad Sistla, and Weiyi Meng. Polarity consistency checking for sentiment dictionaries. In *ACL'12*, pages 997–1005, 2012.

[23] Eduard C. Dragut, Clement Yu, Prasad Sistla, and Weiyi Meng. Construction of a sentimental word dictionary. In *CIKM '10*, pages 1761–1764, 2010.

[24] Philipp Dufter, Nora Kassner, and Hinrich Schütze. Static embeddings as efficient knowledge bases?, 2021.

[25] Andrea Esuli and Fabrizio Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *LREC'06*, pages 417–422, 2006.

[26] A. Fahad, N. Alshatri, Z. Tari, A. Alamri, I. Khalil, A. Y. Zomaya, S. Foufou, and A. Bouras. A survey of clustering algorithms for big data: Taxonomy and empirical analysis. *IEEE Transactions on Emerging Topics in Computing*, 2(03):267–279, jul 2014.

[27] Fábio Figueiredo, Leonardo Rocha, Thierson Couto, Thiago Salles, Marcos André Gonçalves, and Wagner Meira Jr. Word co-occurrence features for text classification. *Information Systems*, 36(5):843–858, 2011.

[28] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *Processing*, pages 1–6, 2009.

[29] Derek Greene, Derek O'Callaghan, and Pádraig Cunningham. How many topics? stability analysis for topic models. *CoRR*, 2014.

[30] Thomas L Griffiths, Michael I Jordan, Joshua B Tenenbaum, and David M Blei. Hierarchical topic models and the nested chinese restaurant process. In *Advances in neural information processing systems*, pages 17–24, 2004.

[31] Maarten Grootendorst. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*, 2022.

[32] Emitza Guzman and Walid Maalej. How do users like this feature? a fine grained sentiment analysis of app reviews. In *Requirements Engineering*, pages 153–162. IEEE Computer Society, 2014.

[33] William L. Hamilton, Kevin Clark, Jure Leskovec, and Dan Jurafsky. Inducing domain-specific sentiment lexicons from unlabeled corpora. *CoRR*, abs/1606.02820, 2016.

[34] Zellig S Harris. Distributional structure. *Word*, 1954.

[35] Thomas Hofmann. Probabilistic latent semantic indexing. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 50–57, 1999.

[36] Qi Huang, Zhanghao Chen, Zijie Lu, and Yuan Ye. Analysis of bag-of-n-grams representation's properties based on textual reconstruction. *CoRR*, abs/1809.06502, 2018.

[37] Clayton J. Hutto and Eric Gilbert. VADER: A parsimonious rule-based model for sentiment analysis of social media text. In *ICWSM'14*, 2014.

[38] Zhigang Jin, Xiaofang Zhao, and Yuhong Liu. Heterogeneous graph network embedding for sentiment analysis on social media. *Cognitive Computation*, 13(1):81–95, Jan 2021.

[39] Adam D I Kramer, Jamie E Guillory, and Jeffrey T Hancock. Experimental evidence of massive-scale emotional contagion through social networks. *PNAS*, 111(24):8788–90, 2014.

[40] Harold W. Kuhn. The hungarian method for the assignment problem. In *50 Years of Integer Programming*, 2010.

[41] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[42] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[43] Guy Lev, Benjamin Klein, and Lior Wolf. In defense of word embedding for generic text representation. In *NLDB'15*, pages 35–50, 2015.

[44] Howard Levene. Robust tests for equality of variances. 1960.

[45] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 2177–2185, Cambridge, MA, USA, 2014. MIT Press.

[46] David D. Lewis. Evaluating text categorization. In *Proceedings of the Workshop on Speech and Natural Language*, HLT '91, pages 312–318, Stroudsburg, PA, USA, 1991. Association for Computational Linguistics.

[47] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *JMLR.*, 5:361–397, 2004.

[48] Chenliang Li, Yu Duan, Haoran Wang, Zhiqian Zhang, Aixin Sun, and Zongyang Ma. Enhancing topic modeling for short texts with auxiliary word embeddings. *ACM TOIS*, 2017.

[49] M. Li, Q. Lu, Y. Long, and L. Gui. Inferring affective meanings of words from word embedding. *IEEE Transactions on Affective Computing*, 8(4):443–456, Oct.-Dec. 2017.

[50] Quanzhi Li, Sameena Shah, Xiaomo Liu, Armineh Nourbakhsh, and Rui Fang. Tweetsift: Tweet topic classification based on entity knowledge base and topic enhanced word embedding. In *CIKM'16*, pages 2429–2432, 2016.

[51] Wei Li and Andrew McCallum. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd international conference on Machine learning*, pages 577–584. ACM, 2006.

[52] Rui Liu, Xingguang Wang, Deqing Wang, Yuan Zuo, He Zhang, and Xianzhu Zheng. Topic splitting: a hierarchical topic model based on non-negative matrix factorization. *Journal of Systems Science and Systems Engineering*, 27(4):479–496, 2018.

[53] Alhassan Mabrouk, Rebeca P. Díaz Redondo, and Mohammed Kayed. Deep learning-based sentiment classification: A comparative survey. *IEEE Access*, 8:85616–85638, 2020.

[54] Anbazhagan Mahadevan and Michael Arock. Integrated topic modeling and sentiment analysis: a review rating prediction approach for recommender systems. *Turkish Journal of Electrical Engineering and Computer Sciences*, 28:107–123, 01 2020.

[55] Yu A Malkov and Dmitry A Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836, 2018.

[56] Jon D Mcauliffe and David M Blei. Supervised topic models. In *Advances in neural information processing systems*, pages 121–128, 2008.

[57] Philipe F. Melo, Daniel Hasan Dalip, Manoel M. Junior, Marcos André Gonçalves, and Fabrício Benevenuto. 10sent: A stable sentiment analysis method based on the combination of off-the-shelf approaches. *J. Assoc. Inf. Sci. Technol.*, 70(3):242–255, 2019.

[58] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[59] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *LREC'18*, 2018.

[60] George A. Miller. Wordnet: A lexical database for english. *CACM*, 38(11):39–41, 1995.

[61] Glenn W. Milligan and Martha C. Cooper. Methodology review: Clustering methods. *Applied Psychological Measurement*, 11(4):329–354, 1987.

[62] David Mimno, Wei Li, and Andrew McCallum. Mixtures of hierarchical topics with pachinko allocation. In *Proceedings of the 24th ICML*, pages 633–640. ACM, 2007.

[63] David M. Mimno and Laure Thompson. The strange geometry of skip-gram with negative sampling. In *EMNLP*, 2017.

[64] Saif Mohammad. Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 English words. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 174–184, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[65] Brigitte Nerlich. Michel bréal: The beginnings of semantics. edited and translated by george wolf. *Historiographia Linguistica*, 18(2-3):369–375, 1991.

[66] Brigitte Nerlich and David D. Clarke. Language, action and context linguistic pragmatics in europe and america (1800–1950). *Journal of Pragmatics*, 22(5):439 – 463, 1994.

[67] Finn Årup Nielsen. A new ANEW: evaluation of a word list for sentiment analysis in microblogs. *CoRR*, abs/1103.2903, 2011.

[68] Sergey I Nikolenko. Topic quality metrics based on distributed word representations. In *SIGIR'16*, pages 1029–1032, 2016.

[69] Sergey I Nikolenko, Sergei Koltcov, and Olessia Koltsova. Topic modelling for qualitative studies. *Journal of Information Science*, 43(1):88–102, 2017.

[70] Farhad Nooralahzadeh, Lilja Øvrelid, and Jan Tore Lønning. Evaluation of Domain-specific Word Embeddings using Knowledge Resources. In Nicoletta Calzolari (Conference chair), Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga, editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 7-12, 2018 2018. European Language Resources Association (ELRA).

[71] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[72] Adler J Perotte, Frank Wood, Noemie Elhadad, and Nicholas Bartlett. Hierarchically supervised latent dirichlet allocation. In *Advances in neural information processing systems*, pages 2609–2617, 2011.

[73] Jipeng Qiang, Ping Chen, Tong Wang, and Xindong Wu. Topic modeling over short texts by incorporating word embeddings. In *PAKDD*. Springer, 2017.

[74] Julio Reis, Pollyanna Goncalves, Pedro Vaz de Melo, Raquel Prates, and Fabricio Benevenuto. Magnet news: You choose the polarity of what you read. In *ICSWM'14*, 2014.

[75] Philip Resnik, William Armstrong, Leonardo Claudino, Thang Nguyen, Viet-An Nguyen, and Jordan Boyd-Graber. Beyond lda: exploring supervised topic modeling for depression-related language in twitter. In *Proc. of the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, pages 99–107, 2015.

[76] Filipe N Ribeiro, Matheus Araújo, Pollyanna Gonçalves, Marcos André Gonçalves, and Fabrício Benevenuto. Sentibench: A benchmark comparison of state-of-the-practice sentiment analysis methods. *EPJ Data Science*, 5(1):1–29, 2016.

[77] Sara Rosenthal, Noura Farra, and Preslav Nakov. Semeval-2017 task 4: Sentiment analysis in twitter. *CoRR*, abs/1912.00741, 2019.

[78] Sascha Rothe, Sebastian Ebert, and Hinrich Schütze. Ultradense word embeddings by orthogonal transformation. *CoRR*, abs/1602.07572, 2016.

[79] Devendra Singh Sachan, Manzil Zaheer, and Ruslan Salakhutdinov. Revisiting lstm networks for semi-supervised text classification via mixed objective function. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6940–6948, Jul. 2019.

[80] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523, 1988.

[81] Bei Shi, Wai Lam, Shoaib Jameel, Steven Schockaert, and Kwun Ping Lai. Jointly learning word embeddings and latent topics. In *SIGIR'17*, pages 375–384, 2017.

[82] Tian Shi, Kyeongpil Kang, Jaegul Choo, and Chandan K. Reddy. Short-text topic modeling via non-negative matrix factorization enriched with local word-context correlations. In *WWW '18*, pages 1105–1114, 2018.

[83] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP'19*, pages 1631–1642, Seattle, Washington, USA, October 2013. ACL.

[84] Duyu Tang, Furu Wei, Bing Qin, Ming Zhou, and Ting Liu. Building large-scale twitter-specific sentiment lexicon : A representation learning approach. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 172–182, 2014.

[85] Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. Learning sentiment-specific word embedding for twitter sentiment classification. In *ACL (1)*, pages 1555–1565. The Association for Computer Linguistics, 2014.

[86] Jian Tang, Zhaoshi Meng, XuanLong Nguyen, Qiaozhu Mei, and Ming Zhang. Understanding the limiting factors of topic modeling via posterior contraction analysis. In *Proceedings of the 31st ICML'14*, pages I–190–I–198. JMLR.org, 2014.

[87] Yla R. Tausczik and James W. Pennebaker. The psychological meaning of words: Liwc and computerized text analysis methods. *J. of Lang. and Soc. Psych.*, 29(1):24–54, 2010.

[88] Mike Thelwall. Heart and soul: Sentiment strength detection in the social web with sentistrength. http://sentistrength.wlv.ac.uk/documentation/SentiStrengthChapter.pdf, 2013.

[89] Mike Thelwall, Kevan Buckley, Georgios Paltoglou, Di Cai, and Arvid Kappas. Sentiment in short strength detection informal text. *JASIST*, 61(12):2544–2558, 2010.

[90] Tan Thongtan and Tanasanee Phienthrakul. Sentiment classification using document embeddings trained with cosine similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 407–414, Florence, Italy, July 2019. Association for Computational Linguistics.

[91] Ike Vayansky and Sathish A.P. Kumar. A review of topic modeling methods. *Information Systems*, 94:101582, 2020.

[92] Leonid Velikovich, Sasha Blair-Goldensohn, Kerry Hannan, and Ryan McDonald. The viability of web-derived polarity lexicons. In *HLT '10*, pages 777–785, 2010.

[93] Felipe Viegas, Mário S. Alvim, Sérgio Canuto, Thierson Rosa, Marcos André Gonçalves, and Leonardo Rocha. Exploiting semantic relationships for unsupervised expansion of sentiment lexicons. *Information Systems*, 94:101606, 2020.

[94] Felipe Viegas, Sergio Canuto, Washington Cunha, Celso França, Claudio Valiense, Leonardo Rocha, and Marcos André Gonçalves. Clusent – combining semantic expansion and de-noising for dataset-oriented sentiment analysis of short texts. In *Proceedings of the 29th Brazilian Symposium on Multimedia and the Web*, WebMedia '23, page 110–118, New York, NY, USA, 2023. Association for Computing Machinery.

[95] Felipe Viegas, Sérgio Canuto, Christian Gomes, Washington Luiz, Thierson Rosa, Sabir Ribas, Leonardo Rocha, and Marcos André Gonçalves. Cluwords: Exploiting semantic word clustering representation for enhanced topic modeling. In *Proceedings of WSDM '19*, pages 753–761, 2019.

[96] Felipe Viegas, Washington Cunha, Christian Gomes, Antonio Pereira, Leonardo Rocha, and Marcos André Gonçalves. Cluhtm: Semantic hierarchical topic modeling based on cluwords. In *The 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

[97] Felipe Viegas, Marcos André Gonçalves, Wellington Martins, and Leonardo C. da Rocha. Parallel lazy semi-naive bayes strategies for effective and efficient document classification. In *CIKM'15*, pages 1071–1080, 2015.

[98] Felipe Viegas, Washington Luiz, Christian Gomes, Amir Khatibi, Sérgio Canuto, Fernando Mourão, Thiago Salles, Leonardo Rocha, and Marcos André Gonçalves. Semantically-enhanced topic modeling. In *CIKM '18*, 2018.

[99] Konstantin Vorontsov, Oleksandr Frei, Murat Apishev, Peter Romov, and Marina Dudarenko. Bigartm: Open source library for regularized multimodal topic modeling of large collections. In Mikhail Yu. Khachay, Natalia Konstantinova, Alexander Panchenko, Dmitry Ignatov, and Valeri G. Labunets, editors, *Analysis of Images, Social Networks and Texts*, pages 370–381, Cham, 2015. Springer International Publishing.

[100] Konstantin Vorontsov and Anna Potapenko. Additive regularization of topic models. *Mach. Learn.*, 101(1-3):303–323, 2015.

[101] Amy Beth Warriner, Victor Kuperman, and Marc Brysbaert. Norms of valence, arousal, and dominance for 13,915 english lemmas. *Behavior Research Methods*, 45(4):1191–1207, 2013.

[102] A. S. Wilkins. Bréal's semantics - semantics: Studies in the science of meaning. by bréal michel. translated by cust henry mrs, with a preface by j. p. postgate. london: William heinemann. 8vo. pp. lxvi, 342. 7s. 6d. net. *The Classical Review*, 15(2):127–128, 1901.

[103] Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *ACL '94*, pages 133–138, 1994.

[104] Yueshen Xu, Jianwei Yin, Jianbin Huang, and Yuyu Yin. Hierarchical topic modeling with automatic knowledge mining. *Expert Systems with Applications*, 103:106–117, 2018.

[105] Da Yin, Tao Meng, and Kai-Wei Chang. SentiBERT: A transferable transformer-based architecture for compositional sentiment semantics. In *Proceedings of the 58th Conference of the Association for Computational Linguistics, ACL 2020, Seattle, USA*, 2020.

[106] Liang-Chih Yu, Jin Wang, K. Robert Lai, and Xuejie Zhang. Refining word embeddings for sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 534–539, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.