National Technical University of Athens
School of Electrical and Computer Engineering
Division of Signals, Control and Robotics

Sentence-Level Sentiment Analysis using Topic Modeling

Diploma Thesis

of

Efstathia Christopoulou

Supervisor: Alexandros Potamianos
Associate Professor

Computer Vision, Speech Communication and Signal Processing Laboratory
Athens, July 2016

National Technical University of Athens
School of Electrical and Computer Engineering
Division of Signals, Control and Robotics

Sentence-Level Sentiment Analysis using Topic Modeling

Diploma Thesis

of

Efstathia Christopoulou

Supervisor: Alexandros Potamianos
Associate Professor

Approved by the committee on July 13th 2016.

(Signature)	*(Signature)*	*(Signature)*
....	....	....
Alexandros Potamianos	Petros Maragos	Anastasia Krithara
Associate Professor	Professor	Research Associate
N.T.U.A	N.T.U.A	N.C.S.R. "Demokritos"

Athens, July 2016

....
Efstathia Christopoulou
Electrical and Computer Engineer, N.T.U.A.

Acknowledgments

This Diploma Thesis signals the end of my student life in the School of Electrical and Computer Engineering at the National Technical University of Athens.

First and foremost, I would like to deeply thank my supervisor Alexandros Potamianos for introducing me to the field of Machine Learning and Natural Language Processing through the course of Pattern Recognition. His approach not only played a major role in my decision to turn to him when searching for my diploma thesis subject, but also to continue my studies in this area. I cannot but thank him for the trust he placed on me, the knowledge, guidance and new opportunities he offered me for my academic career along with the encouragement he never seized to provide me to aim higher throughout this year.

Additionally, I should thank Dr. George Paliouras and Dr. Anastasia Krithara from N.C.S.R. "Demokritos" for their support and contribution to a large part of this Diploma Thesis, as well as for the fruitful discussions and ideas we shared. I cannot overlook the help and support of all my colleagues, members of the NTUA and TUC group, especially Dr. Elias Iosif and Elisavet Palogiannidi with which I started working since the beginning, including Sasa Kolovou and Filippos Kokkinos that were part of the 'Tweester' team that participated in the SemEval-2016 competition.

Last but not least, I would like to express my love and gratitude to all my friends and family that stayed by my side during these years, believed in me, helped me overcome the difficulties I came across and become the person I am today. I would specially like to thank Christos for his patience and support all this time.

This Diploma thesis is dedicated to my grandparents Persefoni and Stavros.

Efstathia Christopoulou
July 13th, 2016

Περίληψη

Η συναισθηματική ανάλυση κειμένου, είναι ένας κλάδος της Μηχανικής Μάθησης που αποσκοπεί στην αυτόματη αναγνώριση υποκειμενικής πληροφορίας σε γραπτές πηγές. Λόγω της ανάπτυξης του διαδικτύου και συνεπώς του όγκου πληροφοριών που διανέμονται στα κοινωνικά δίκτυα, πολλαπλές εφαρμογές χρησιμοποιούν αυτή την τεχνολογία για να βελτιόσουν την λήψη των αποφάσεών τους.

Σκοπός αυτής της Διπλωματικής εργασίας είναι η διερεύνηση μιας μεθόδου προσαμοργής για την βελτίωση της αυτόματης, δυαδικής αναγνώρισης συναισθήματος σε προτάσεις (θετική/αρνητική), με τη βοήθεια τεχνικών Θεματικής Μοντελοποίησης. Σε σύγκριση με τις κλασσικές μεθόδους που χρησιμοποιούν μια ποκιλια λεξιλογικών χαρακτηριστικών και λεξικών σαν είσοδο σε ταξινομητές, τα θεματικά μοντέλα έχουν την ικανότητα να συλλαμβάνουν το νοηματικό πλαίσιο στο οποίο ανήκει μια πρόταση και έτσι να εκτιμούν τη σημασία της με μεγαλύτερη λεπτομέρεια.

Το πρώτο μέρος της Διπλωματικής εργασίας, ασχολείται με τις τεχνικές Θεματικής μοντελοποίησης οι οποίες διευκολύνουν την εξαγωγή πληροφορίας από αδόμητες συλλογές κειμένων. Περιγράφεται η Latent Dirichlet Allocation (LDA) τεχνική, που προσπαθεί να ανακατασκευάσει την διαδικασία συγγραφής ενός κειμένου και συνεπώς να παράγει τις θεματικές ενότητες από τις οποίες αποτελείται. Στο δεύτερο μέρος, τα μοντέλα διανυσματικού χώρου διερευνόνται, τα οποία στοχεύουν στο να εκφράσουν τη σημασιολογική ομοιότητα μεταξύ λέξεων, κωδικοποιώντας τη γλώσσα ως μια μαθηματική κατανομή και συνεπώς σχηματίζοντας ένα σημασιολογικό χώρο. Επιπλέον, παρουσιάζονται μοντέλα στο χώρο των συναισθημάτων, τα οποία κωδικοποιούν το συναίσθημα των λέξεων, και συγκεκριμένα πώς είναι εφικτό το πέρασμα από το σημασιολογικό στον συναισθηματικό χώρο μέσω ενός μοντέλου αντιστοίχισης, χρησιμοποιώντας ένα λεξικό, ώστε να εκτιμηθεί το συναίσθημα νέων λέξεων.

Ο αλγόριθμος που παρουσιάζεται επικεντρώνεται κυρίως στην προσαρμογή του Σημασιολογικού χώρου, και ονομάζεται Σημασιολογικό Μοντέλο Προσαρμογής (ΣΜΠ). Στην προσέγγιση αυτή, ο σημασιολογικός χώρος μιας πρότασης αναπαρίσταται ως ένα ζυγισμένο μείγμα από διαφορετικά θεματικά-σημασιολογικά μοντέλα με βάση την εκτίμηση ενός εκπαιδευμένου πιθανοτικού θεματικού μοντέλου. Η τελική εκτίμηση του συναισθήματος μια πρότασης είναι το αποτέλεσμα ενός σημασιολογικού-συναισθηματικού μοντέλου που ενώνει τον σημασιολογικό και συναισθηματικό χώρο μέσω μιας αντιστοίχισης και ενός υπάρχοντος συναισθηματικού λεξικού.

Η απόδοση του ΣΜΠ μοντέλου μπορεί να εκτιμηθεί τόσο σε εφαρμογές σημασιολογικής ομοιότητας λέξεων αλλά και σε αναγνώριση συναισθήματος προτάσεων, δείχνοτας μια βελτίωση στη μέτρηση της συσχέτισης με τιμές ομοιότητας δοσμένες από ανθρώπους, παράλληλα με μια αύξηση της ακρίβειας ταξινόμησης για προτάσεις, σε δεδομένα γενικού περιεχομένου αλλά και από το Twitter σε σύγκριση με ένα σύστημα που δεν χρησιμοποιεί θεματικές ενότητες.

Λέξεις Κλειδιά

συναισθηματική αναλύση προτάσεων, θεματική μοντελοποίηση, LDA, σημασιολογικά μοντέλα, word2vec, συναισθηματικά μοντέλα, προσαρμογή

Abstract

Affective Text Analysis, is a field of Machine Learning that aims to automatically detect subjective information in textual data. Due to the development of the World Wide Web and consequently the amount of opinions distributed in social media, numerous applications use this technology to improve their decision making.

The scope of this Diploma Thesis is to explore an adaptation algorithm for improving the automatic binary sentiment classification of sentences in text (positive/negative), with the help of Topic modeling techniques. Compared to classic approaches that use a variety of lexical features and existing annotated lexicons as input to classifiers, topic models are able to capture the context of a sentence and thus estimate its meaning in greater detail.

The first part of the thesis, is devoted to Topic Modeling techniques that facilitate the extraction of meaningful information from unstructured collections of documents. The Latent Dirichlet Allocation (LDA) method is described which tries to reconstruct the process of writing a document and generate the topics that it consists of. In the second part, different Vector Space Models are investigated, which aim to express how similar a word is to another, by representing the language as a distribution, forming a semantic space. Additionally, Affective Space Models, that measure the sentiment of words are explored, and in particular, how one can create a mapping from the semantic to the affective space with the help of a small lexicon in order to estimate the sentiment of new words.

The presented algorithm focuses on the adaptation of the Semantic space, named Semantic Model Adaptation (SMA). In the SMA approach, the semantic space of each sentence is represented as a weighted mixture of different topic-semantic models according to a trained probabilistic topic model. The final estimation of a sentence's score is the result of a semantic-affective model, that connects the semantic with the affective space through a mapping and an existing annotated affective lexicon.

The proposed model can be evaluated on both pair-wise semantic similarity and sentence affective estimation tasks, showing a significant improvement of correlation with human annotations, as well as an important increase in the classification accuracy for sentence-level binary sentiment detection in generic and Twitter data, compared to a baseline method that does not use topic models.

Keywords

affective analysis, sentence-level sentiment analysis, topic modeling, LDA, distributional semantic models, word2vec, adaptation

Contents

List of Figures

List of Tables

Abbreviations

BOW	Bag-of-words
LDA	Latent Dirichlet Allocation
LSA	Latent Semantic Analysis
LSI	Latent Semantic Indexing
pLSA	Probabilistic Latent Semantic Analysis
pLSI	Probabilistic Latent Semantic Indexing
CTM	Correlated Topic Model
PMI	Point-wise Mutual Information
TF-IDF	Term frequency-Inverse Topic Frequency
SAM	Semantic-Affective Model
POS	Part-of-Speech
SVM	Support Vector Machine
NB	Naive Bayes
LSE	Least Squares Estimation
RR	Ridge Regression
ANEW	Affective Norms for English Words
CBOW	Continuous Bag-of-words
W2V	Word2Vec
SVD	Singular Value Decomposition
VSM	Vector Space Model
DSM	Distributional Semantic Model
ME	Maximum Entropy
SVR	Support Vector Regression
OVA	One-versus-all
EM	Expectation Maximization
HDP	Hierarchical Dirichlet Process
HMM	Hidden Markov Model
JST	Joint Sentiment Topic
PMF	Probability Mass Function
MCMC	Monte Carlo Markov Chain
KL	Kullback-Leibler

Chapter 1

Introduction

1.1 Motivation

Opinions play a major role in human activities as they influence people's behaviors. The growth of the World Wide Web and the transformation of information into digital form, has enabled users to access abundant amounts of data as well as share their opinions and feelings about their surroundings instantly. A lot of actions show this kind of influence in people's lives, from researching movie reviews to medicine, through the internet.

However, the time and effort of analyzing such amounts of data is typically proportional to their size, making their processing extremely difficult for an average human. For a lot of years, surveys, polls and focus groups were conducted with an objective to gather as much information as possible in order to capture the general preference or dissatisfaction of the public about products, services or other contemporary topics. Therefore, the field of computer science is involved with Affective Text Analysis, which means the automated detection of subjective information, alternatively known as emotion recognition, in textual sources, aiming to solve this problem. Yet, the rise of the social media, such as Facebook and Twitter has turned the attention of this research to small pieces of text usually sentences, as the main way of expression in social websites.

There are many ways to estimate the expressed opinion of a sentence, as is analyzed in Chapter 2. However, a known problem in this task is the large variety of topics detected in sentences and how they influence their interpretation. The meaning of words changes not only in different domains but also when using different writing style (e.g. formal, sarcastic) resulting multiple word senses and meanings. The majority of labeled data available are domain-specific and thus fail to provide a solution for tasks in different domains. One popular way of overcoming this problem is the Topic Modeling of general purpose data. These methods try to represent sentences as a probabilistic mixture of different topic-specific models and can lead to a more accurate estimation of the polarity of a sentence (positive/negative in the simplest of cases). However, in order to finally determine if a sentence is positive or negative, different scientific areas need to be combined such as Distributional Semantic Models and Affective Models.

1.2 Affective Text Analysis

The analysis of the emotional content of text is known as Affective Text Analysis. "Affect" is an umbrella-term that describes feelings, emotions, moods and temperaments and is used as often as the term "Emotion" and "Sentiment". A task in affective analysis can be determined by the type of emotion to detect (e.g. anger, happiness) or the level of analysis to perform (words, sentences, paragraphs, documents, collections). The main goal of this field is to identify the opinion of the author or the reader, in smaller or bigger sections of a document. It is a wide area that is strongly connected with the field of Natural Language Processing (NLP) and Distributional Semantic Models.

Generally, the domain which is involved with the development of intelligent systems that are able to perceive and encode sentiment is known as Affective Computing. In order to build systems as such, we need to represent human emotions in a scale understandable by computers.

1.2.1 Emotion Representation

Emotions can be conceptualized either in discrete or in dimensional view. Paul Ekman determined the six basic emotions that are common among people in all cultures: anger, happiness, surprise, disgust, sadness and fear. [1].

For both theoretical and practical reasons researchers define emotions according to one or more dimensions. Wilhelm Max Wundt, the father of modern psychology, proposed that emotions can be described by three dimensions: "pleasurable versus unpleasurable", "arousing or subduing" and "strain or relaxation" [2]. Additionally, Harold Schlosberg named three dimensions of emotion: "pleasantness-unpleasantness", "attention-rejection" and "level of activation" (Schlosberg (1954)). The basic emotional dimensions are described below:

Valence is a subjective feeling of pleasantness or unpleasantness. Ranges from highly positive to highly negative. It is also called Pleasure-Displeasure. For instance, both 'anger' and 'fear' are unpleasant emotions, and score high on the displeasure scale. However 'joy' is a pleasant emotion.

Arousal is a subjective state of feeling activated or deactivated. Ranges from calming or soothing to exciting or agitating. Alternatively it can be found as Arousal-Activation, which is the dimension of emotion that expresses in what degree a person is ready-to-act/aroused or relaxed. Activation/Arousal measures the intensity of the emotion. For instance, while both 'anger' and 'rage' are unpleasant emotions, 'rage' has a higher intensity or a higher arousal state. However 'boredom', which is also an unpleasant state, has a low arousal value.

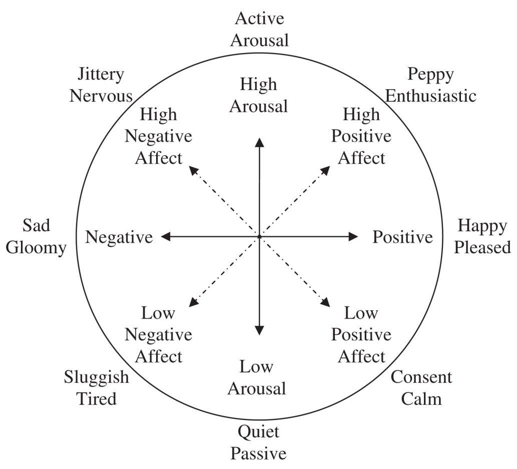
Dominance represents the controlling and dominant nature of the emotion. For instance while both 'fear' and 'anger' are unpleasant emotions, 'anger' is a dominant emotion, while 'fear' is a submissive emotion.

Consequently, dimensional models of emotion attempt to conceptualize human emotions by defining where they lie in two or three dimensions. Most dimensional models incorporate valence and arousal while they may not use intensity dimensions as dominance is strongly correlated with arousal. The two models that are most prominent are a) the circumplex model and b) the Plutchik's model.

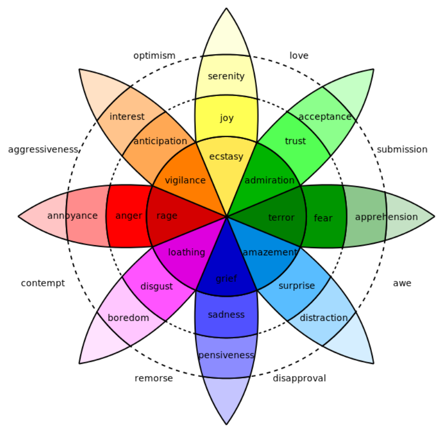
The circumplex model of emotion was developed by James Russell (Russell (1980)). This model suggests that emotions are distributed in a two-dimensional circular space, containing arousal and valence dimensions. Arousal represents the vertical axis and valence

[1] https://www.paulekman.com/wp-content/uploads/2013/07/Basic-Emotions.pdf
[2] http://psychclassics.yorku.ca/Wundt/Outlines/

represents the horizontal axis, while the center of the circle represents a neutral valence and a medium level of arousal. In this model, emotional states can be represented at any level of valence and arousal, or at a neutral level of one or both of these factors.



(a) Circumplex model representation.



(b) Plutchik's wheel of emotions.

Figure 1.1: Different representations of emotion.

Robert Plutchik offers a three-dimensional model that is a hybrid of both basic-complex categories and dimensional theories (Plutchik (2001)). It arranges emotions in concentric circles where inner circles are more basic and outer circles more complex. Notably, outer circles are also formed by blending the inner circle emotions. Plutchik's model, as Russell's, emanates from a circumplex representation, where emotional words were plotted based on similarity.

Based on these affective spaces, all emotions can be represented. In the simplest case, the estimation can be made using only the valence dimension, for two classes, positive (high valence) and negative (low valence), three classes, if the neutral class (zero valence) is included (which does not contain any sentiment), or five classes and more, according to the application and how much the affective scale needs to be extended.

Finally, subjectivity or objectivity of a text section is another category of 'sentiment'. However, in the case of sentences, as it is quite often to contain more than one sentiment, sometimes an Aspect-Based Sentiment Analysis is more appropriate, which detects the expressed opinion towards a specific aspect or person contained in the sentence.

1.2.2 Topic Models

Topic Models, also called Statistical Topic Models, is a class of mathematical models, that embody a set of assumptions that contribute to the discovery of the hidden thematic structure of documents, using probability distributions. More specifically, the basic assumption of topic modeling techniques is that each document can be viewed as a mixture of multiple topics which are represented as probability distributions. The term "Topic" usually describes large thematic areas such as "computers", "health", "movies" etc, or can characterize more specific domains such as 'price', 'quality' etc. However, when studying Topic Models, the same term describes a topic as a distribution over a fixed vocabulary of words. Alternatively, a topic consists of a cluster of words that frequently occur together in this domain. Topic modeling is a frequently used text-mining tool for analyzing large collections of text. The main advantage of these models is that they are unsupervised and consequently do not require labeled data, which are hard to find.

Using the context of words in documents, topic models can identify and connect words with similar meanings as well as, distinguish between uses of words with multiple meanings (word disambiguation). Topic Models combined with Affective analysis can be used to improve the estimation of words affective scores. The most well-known approaches include learning topic-based sentiment labels when capturing simultaneously sentiments and topics in document text collections. Additionally, aspect-based sentiment analysis can benefit from topic modeling techniques that are able to distinguish the different aspects a review addresses.

1.2.3 Distributional Semantics

Distributional Semantics is a research area that develops and studies theories and methods for quantifying and categorizing semantic similarities between linguistic items based on their distributional properties in large samples of language data. The basic idea of distributional semantics can be summed up in the so-called Distributional hypothesis: “Linguistic items with similar distributions have similar meanings” or in other words, “Words that tend to occur together in pieces of text, tend to have similar meaning”.

Distributional Semantic models (DSMs), i.e. models that try to encode the meaning of words and larger sections of text into a mathematical representation of a feature-vector, are closely related with Affective Text analysis. The DSMs can be combined with affective text analysis methods in order to capture the sentiment of words. The representation of words as semantic feature vectors can be used, not only as additional features in sentiment classification, but also as a means to convey sentiment through the estimation of similarity, relatedness, synonymy and multiple other relations between words, with the help of linear algebra, making it a strong tool for the polarity recognition task.

1.3 Thesis Scope & Contribution

The main objective of this Thesis is to explore an adaptation of Distributional Semantic Models algorithm for identifying the polarity of sentences between two classes, positive or negative.

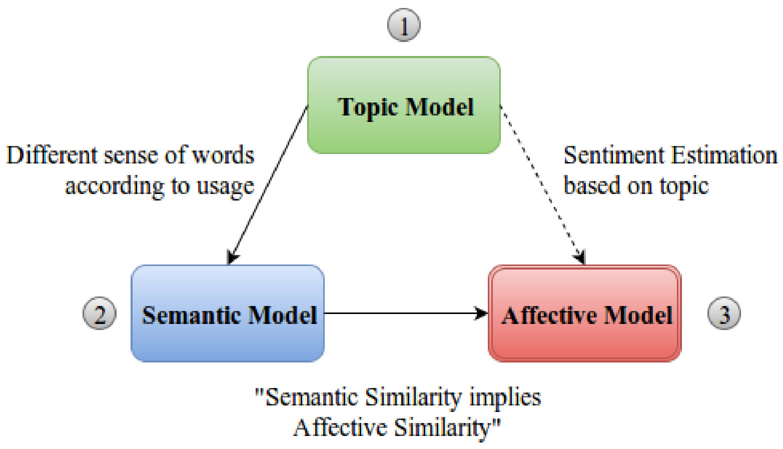

1
Topic Model
Different sense of words according to usage
Sentiment Estimation based on topic
2
Semantic Model
Affective Model
3
"Semantic Similarity implies Affective Similarity"

Figure 1.2: Thesis main research areas.

There are three models that are combined in the proposed approach, Topic Models, Semantic Models and Affective Models, using some basic assumptions as shown in figure 1.2. The pair one-two is directly connected by the fact that words can have multiple meanings in different domains, the pair two-three is directly connected with the assumption that "Semantic similarity implies affective similarity" and the pair one-three is indirectly connected through the assumption that before the estimation of a sentence's polarity we can firstly identify the topics it contains.

The presented technique uses the Latent Dirichlet Allocation (LDA) Topic Modeling method that is able to extract meaningful information from unstructured text, by splitting collections of documents into the topics they may contain. Furthermore, the semantic representation of words is explored in each topic, in order to extract comprehensible information for the relations between words, and in particular identify their similarity. The proposed approach is based on adapting the semantic space of each sentence in topic-level, aiming to better describe its meaning. Finally in order to identify sentiment in sentences, a mapping from the semantic to the affective space is examined, by exploiting the assumption that "semantic similarity implies affective similarity".

The contribution of this Thesis lies in the improvement of binary classification in sentences. In particular, we aim not only to improve the classification accuracy but also the correlation in terms of valence score in sentence-level. The main difference between topic models and lexicon-based ones, is that they can achieve word disambiguation by identifying a sentence's context. On the contrary, lexicon-based classification methods, fail to capture the context of words included in a sentence and may produce misleading results. The evaluation on word-pair similarities and sentence-level polarity classification, shows that classification accuracy and correlation improve with the integration of topic-trained semantic models.

This topic-semantic models adaptation scheme was also part of the system submitted to the SemEval-2016 competition, Task 4: Sentiment Analysis in Twitter, in subtasks A: Message Polarity Classification and B: Tweet classification according to a two-point scale, ranking first in Task B (Palogiannidi et al. (2016)).

1.4 Thesis Organization

This diploma thesis is organized in seven Chapters and one Appendix.

In Chapter 2 a large description of the bibliography is provided, divided in four sections, according to the fields that the thesis addresses. The first section, describes different sentiment analysis approaches that were investigated with a view to detect the polarity (usually three class, positive-negative-neutral) of sentences and reviews (small paragraphs). The second section, introduces the Distributional Semantic Models that are widely used to discover word similarities. The third section, describes the evolution of topic modeling techniques over time and the state-of-the-art algorithms that are used nowadays. Finally, the fourth section is a literature review on Sentiment Analysis techniques that incorporate Topic Modeling as a comparison with the one proposed in this Thesis.

In Chapter 3 the Topic Modeling technique used as part of the algorithm is described. The functionality of the Latent Dirichlet Allocation (LDA) technique is broadly explained in a theoretical and practical level, along with its statistical structure.

In chapter 4 two different Natural Language Processing models are described. In the first part, Semantic models, which aim to represent the meaning of words as mathematical structures of vector spaces are analyzed along with the Word2Vec model that was used for the experiments. In the second part, an existing Affective model, that tries to capture the emotional information of words is analyzed. This model takes advantage of an existing affective lexicon in order to predict the sentiment of unknown words by creating a mapping from the semantic to the affective space.

Chapter 5 is the most important part of this work, as it outlines the novel algorithm that aims to improve sentiment classification in sentences, using the aforementioned tools. The idea of the Semantic Model Adaptation (SMA) is illustrated and in particular how the semantic space of a sentence can be represented as a mixture of multiple topic-semantic models.

Chapter 6 includes all the results and the experimental procedure for the different experiments that were conducted, on different datasets. Firstly, the topic models that were created are evaluated with different analysis methods. Secondly, the performance of SMA is examined on pair-wise semantic similarity and on sentence-level polarity classification tasks.

Chapter 7 includes a summary of this work, as well as conclusions that resulted from the different experiments. Additionally, an improved version of the SMA algorithm is presented as on-going words. Finally, an on-going work idea and directions for possible future work are illustrated.

In appendix A the Gensim toolbox is described and more specifically two of its basic implementations that were used in this thesis, the LDA algorithm for Topic Modeling and the Word2Vec semantic model. The necessary commands for data pre-processing and models construction are presented along with some outputs during the execution.

Chapter 2

Related Work

2.1 Sentiment Analysis in Reviews and Sentences

Most of the published works, are focused on detecting sentiment in review-paragraph level, as an effort to identify whether the overall expressed opinion is positive, negative or neutral. However, the immediate rise of Social Media has led researchers to sentence-level sentiment extraction. Techniques for both levels are explored here, as there are closely related.

The approaches followed in sentiment analysis, can be classified into 3 major categories: a) Machine learning approaches, supervised (using labeled data) and unsupervised (with little (semi-supervised) or no labeled data) that make explicit use of classifiers, b) Semantic orientation approaches, which are based on positive and negative sentiment words contained in the evaluation text and c) Lexicon-based approaches that use existing annotated sentiment lexicons of words and small phrases. The second category can be divided into i) corpus-based techniques that try to identify word relations based on their co-occurrence in text, and ii) dictionary-based techniques which use synonyms, antonyms and word senses stored in lexical databases that help determine word sentiments. Additionally, combinations and hybrid approaches of the above techniques are commonly used.

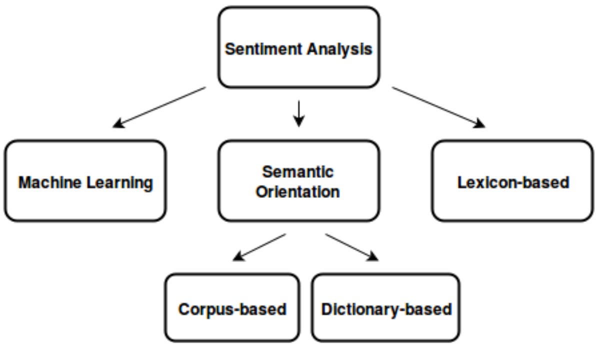


Figure 2.1: Sentiment Analysis categories.

2.1.1 Machine Learning approaches

Starting with the Machine Learning approaches, their objective is to estimate the polarity of sentences or reviews in usually a two or three-class classification problem. The general procedure steps are the following: Firstly obtain a corpus of training data, secondly represent each document/sentence as a feature vector, learn a model from these data by training a classifier and then classify the unseen/test data into a category using the trained classifier.

The original work for sentiment detection in reviews, was presented by Pang et al. (2002). In their approach, they experimented with three different classifiers, on a Movie Reviews dataset, for a binary classification problem (positive-negative): Naive Bayes (NB), Maximum Entropy (ME) and Support Vector Machines (SVM). The features that were

Classifier	Formula
Naive Bayes	$P_{NB}(c\|d) := \frac{P(c)\cdot(\prod_{i=1}^{m} P(f_i\|c)_i^n(d))}{P(d)}$
Maximum Entropy	$P_{ME}(c\|d) := \frac{1}{Z(d)} exp(\sum_{i=1} \lambda_{i,c} F_{i,c}(d,c))$ $F_{i,c}(d,c') = \begin{cases} 1 & \text{if } n_i(d) > 0 \text{ and } c = c' \\ 0 & \text{other} \end{cases}$
SVM	$\overrightarrow{w} = \sum_j a_j c_j \overrightarrow{d_j}, a_j \geq 0$

Table 2.1: Basic classifiers tested in Pang et al. (2002).

used varied from unigrams (single words) to adjectives and multiple combinations, as well as the representation of the feature vector differed from binary notation (existence - non existence of a word-feature) to frequency (of word-feature in the review). SVM proved to give the best performance compared to NB, but overall, they concluded that these classifiers perform better on topic-based categorization rather than sentiment classification. Afterwards, Pang and Lee (2004) used subjectivity detection as a prior step to improve performance (figure 2.2). In particular, they trained a subjectivity detector to classify sentences into subjective and objective, discard the subjective and pass the objective to a classifier. They showed that the results of subjectivity, work as a better and cleaner input to the classifier than the original document, as they contain more sentimental (subjective) information.

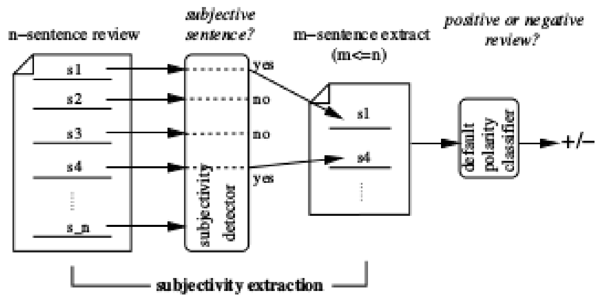


Figure 2.2: Polarity classification via subjectivity detection Pang and Lee (2004).

In order to extend sentiment analysis, Pang and Lee (2005) investigated the problem of five scale polarity of a review (as Schler (2005)), using a multi-class SVM (one-versus-all (OVA)), Regression (SVR) and metric labeling (relation between items and labels) algorithms. As an extension Goldberg and Zhu (2006) represented documents in a graph of labeled and unlabeled data, by connecting them using a distance measure from the document features. The objective of this approach was to use unlabeled data for training, as labeled ones are few, which proved to give better performance than ignoring unlabeled data. In order to identify sentiment for the different aspects contained in a review, Shimada and Endo (2008) experimented with SVMs, SVR, Maximum Entropy and cosine similarity measure algorithms along with various feature extraction methods.

2.1.2 Feature Space in ML

Moreover, another important factor in these models is the type of features used. As Machine Leaning approaches basically use classifiers in order to estimate sentiment in textual sources, the data they need must be encoded in the form of a vector. Generally, these vectors can contain lexical or non lexical information as well as numerical values.

Features that belong to the first category, can be derived from any textual source. Assuming that a vocabulary is extracted from the training dataset, the words it contains can be used as features. More specifically, words that express sentiment, prepositions, very long words, the number of a words possible senses (Malandrakis et al. (2013)), even stop-words (words like 'I', 'the') can be encoded in a bag-of-words (BOW) format and serve as features that represent a document. Concerning the second category, non-lexical features can be Part-of-Speech (POS) tags (adjectives, verbs, adverbs), punctuation, emoticons, urls or references, mainly derived from social media sources like Twitter. Finally, numerical features include: binary representation (presence or not of this term in the document), occurrence frequency (frequency of this term in the documents), TF-IDF scores (term frequency - inverse document frequency), statistical values of affective scores (min, max, mean, std), cluster labels and values produced by Distributional Semantic Models.

Strongly related with the lexicon-based category, multiple machine learning techniques use sentiment lexicons to improve their performance. Osherenko and André (2007) proposed to reduce the dimensionality of word feature vectors and use only small sets of affective words as features obtained from a lexicon. This technique, evaluated with SVM, showed no improvement in recognition performance but it can be concluded that feature reduction can be achieved without losing important information. Finally, Lin et al. (2012) investigated the use of different features in classification tasks.

2.1.3 Semantic Orientation Approaches

Semantic orientation approaches have proved to improve performance, as they are based on the relations between words in text. Generally, the state-of-the-art methods are based on the observation that similar opinion words tend to appear together in a corpus. Consequently, one can assume that if they appear frequently in the same context they will express the same sentiment. This is a statistical approach that shows that frequent occurrence of words in a document implies their closeness in meaning. The first idea was proposed by Turney (2002). In their work, the Point-wise Mutual Information (PMI) criterion was used to express the statistical dependence between two words. In particular, they measured the sentiment polarity of a word w as the difference between the the PMI value of word w with word "excellent" and the PMI value of word w with the word "poor" (equation 2.2). Additionally in Turney and Littman (2003) a set of positive and negative words is used instead of only 'excellent' and 'poor'. Therefore, the difference of the PMI sum is calculated (equation 2.3). As an extension to this work, Chaovalit and Zhou (2005) used the Google search engine and measured the number of hits returned for a pair of words query, in order to better estimate their relation. Moreover, Read and Carroll (2009) extended this approach by employing Semantic Spaces and Distributional Similarity as weakly-supervised methods. They require only a large collection of unlabeled data resulting a less domain-dependent performance. In the same context, Baroni and Lenci (2010) presented in their paper how information about the relations of words can be extracted from a corpus as word-link-word tuples arranged into a third-order tensor. This type of representation allows information to be used in multiple tasks such as (as mentioned in the paper) modeling word similarity judgments, discovering synonyms, concept categorization, predicting selectional preferences of verbs, solving analogy problems, classifying relations between word pairs, harvesting quality structures with patterns or example pairs, predicting the typical properties of concepts, and classifying verbs into alternation classes. More information about Semantic models can be found on Chapter 2, Section 2.2 and Chapter 4.

$$PMI(word_1, word_2) = log_2\Big(\frac{P(word_1 \bigcap word_2}{P(word_1)P(word_2)}\Big) \tag{2.1}$$

$$SO(phrase) = PMI(phrase, \text{“}excellent\text{”}) - PMI(phrase, \text{“}poor\text{”}) \tag{2.2}$$

$$SO - PMI(word) = \sum_{\substack{pword\\ \in Pwords}} PMI(word, pword) - \sum_{\substack{nword\\ \in Nwords}} PMI(word, nword) \tag{2.3}$$

As far as the second sub-category of semantic orientation tasks is concerned, the most well-known tool is WordNet (Fellbaum (1998)). It is a large database that provides different kinds of information about the relation between words, as well as their multiple senses, which can help in sentiment prediction. The possibility to disambiguate the senses of words with WordNet leads to the identification of opinions in text. Kamps et al. (2004) proposed to use the relative shortest path distance of the “synonym” relation, whereas Kim and Hovy (2004) recommended to obtain a list of sentiment words by expanding an initial set with synonyms and antonyms of these words from WordNet. In their work, the polarity of the word is determined by the relative count of positive and negative synonyms of this word. In **?** a Sentiment-Analyzer tries to extract the sentiment about a subject from online documents. They do not detect the overall sentiment in a review but determine the sentiment about an exact aspect mentioned in the review, by extracting features about topics, features about sentiment and relations between subjects and sentiments with the help of a dictionary and a sentiment pattern database. Also, Andreevskaia and Bergler (2006) presented a method for extracting sentiment-bearing adjectives from WordNet using the Sentiment Tag Extraction Program (STEP).

2.1.4 Lexicon-based approaches

Finally, dictionary approaches use existing lexicons with opinion polarities for words and small phrases. Some of the most known lexica are presented in the table below. These affective lexicons can be used to predict the sentiment expressed in a document or a sentence, by identifying the opinionative words they contain and compute the final sentiment with the help of simple rule-based algorithms (Zhu et al. (2009)). In their work Taboada et al. (2011) use the Semantic Orientation CALculator (SO-CAL) which exploits dictionaries of words annotated with their semantic orientation (polarity and

Affective Lexica	Source
General Inquirer	http://www.wjh.harvard.edu/~inquirer/
WordNet-Affect	http://wndomains.fbk.eu/wnaffect.html
Senti-WordNet	Esuli and Sebastiani (2006)
Affective Norms for English Words	Bradley and Lang (1999)
Dictionary of Affect of Language	http://www.hdcus.com/
Bing Liu Opinion Lexicon	http://www.cs.uic.edu/~liub/FBS/
AFINN	Nielsen (2011)

Table 2.2: List of most known existing affective dictionaries

strength), and incorporates intensification and negation in order to predict sentiment in text. Generally, all approaches tend to use sentiment lexicons as features to extract new polarity scores for unseen words. The approach of Malandrakis et al. (2011a), which uses an existing lexicon not only to predict words scores but also to determined the polarity of sentences is further analyzed in chapter 4. Lastly, Jijkoun et al. (2010) present a method for automatic generation of topic-specific subjectivity lexicons from a general purpose polarity lexicon that allow users to pin-point subjective on-topic information in a set of relevant documents.

2.2 Distributional Semantic Modeling

Harris was the first one to question if it is possible to model the language as a distribution (Harris (1954)). In his paper, he underlines that as language contains different objects (e.g. words, phrases) with specific properties (features) it can be encoded in a distributional structure, by measuring the occurrence of some objects with others. Therefore, the distribution of an element can be represented as the sum of the other elements surrounding it, i.e. as a vector whose each position corresponds to how many times these elements where witnessed together (in the same environment). These are known as vector space models (Schiitze (1993), Sahlgren (2006)). One can assume, that if we model words as vectors these representations can express their meanings. However, this representation is a property of language whereas the meaning of a word is the interpretation of a person, and can potentially change.

Distributional semantics favor the use of linear algebra as computational tool and representational framework. The basic approach is to collect distributional information in high-dimensional vectors, and to define distributional/semantic similarity in terms of vector similarity. As mentioned before, the main assumption of DSMs is that “Similarity of context implies similarity of meaning”. By exploiting the properties of linear algebra, multiple operations can be applied on or between vectors in order to extract different types of information.

There is a rich variety of computational models implementing distributional semantics. In the survey of Verma et al. (2011) Vector Space Models are examined for a general textual semantic processing, by directly extracting linguistic features from a text collection. In Latent Semantic Indexing (analyzed in the next section 2.3) one of the most simple DSMs is constructed by forming an occurence term-document matrix (number of times a word appears in a document, Deerwester et al. (1990)). The measure of similarity between two words can be calculated through the similarity of two vectors (rows) in the matrix. Baroni and Lenci (2010) describe in detail the construction of distributional semantic models as a distributional memory framework that is able to extract all semantic information from a corpus and store it in third order tensors. They show that this type of information can be used in different tasks and the evaluation results compare DSMs with state-of-the-art methods for specific tasks. In the same direction, Iosif and Potamianos (2010) illustrate a method to compute the semantic similarity between words in an unsupervised way with the help of web-harvested data. Additionally, different types of features are investigated, showing that counting the number of co-occurrences of words in a window (few words before and after the word of interest - context based method) provide better performance when measuring the correlation with human annotations, compared to counting in larger sections (co-occurrence based method).

The last years, more and more different methods are introduced for distributional models construction. Apart from the most common count-based context methods, the most popular use neural networks, such as the Word2Vec model (Mikolov et al. (2013)) and other approaches based on neural networks (Zeng et al. (2014), Mnih and Kavukcuoglu (2013)) for evaluating similarity and other relation between words. The Glove model (Pennington et al. (2014)), is a log-bilinear regression model that combines the advantages of the two major model families in the literature: global matrix factorization and local context window methods.

2.3 The Methods of Topic Modeling

In the fields of machine learning and natural language processing, a topic model (TM) is considered as a type of statistical-generative model that helps in discovering abstract “topics” that occur in a large collection of documents. In the bibliography, a generative model is a Bayesian network which provides an description of an observed phenomenon. It describes the conditional dependencies between random variables in a network along with the relation of cause and effect. Topic models are probabilistic generative models. They try to model a textual source and usually the generation process of this source by means of a Bayesian network. Intuitively, given that a document is about a particular topic, one would expect domain-specific words to appear in the document more or less frequently. On the other hand, it is obvious that each document consists of multiple topics in different proportions. Thus, topic modeling, tries to capture the information included in large text collections with the help of statistics. The most known of it’s techniques, are based the assumption of exchange-ability (Aldous (1985)): The order of words and documents can be neglected. These techniques that are also included in the paper of Blei (2012) and are presented below.

The first model to be studied was the Latent Semantic Indexing (LSI), also known as Latent Semantic Analysis (LSA). It was analyzed by Papadimitriou et al. (1998) and was firstly used in information extraction tasks. It aims at creating document vectors in order to represent their semantic context and more specifically, to identify hidden relations between documents and words. The steps of this algorithm are the following: To begin with, a 2-dimensional matrix is created, where the rows correspond to words in a collection of documents (a vocabulary) and columns correspond to the documents of this collection. Each element in the matrix has a frequency value that represents the number of times this word exists in each document. This representation of documents (as vectors of word frequencies) is called Bag-of-words model (BOW). The assumption where LSA is based on is that words that are semantically related, tend to co-occur in similar pieces of text.

Subsequently, a dimension reducing linear projection, the Singular Value decomposition (SVD) technique, is applied to lower the dimension of rows or columns, by preserving the other dimension. It is supposed that similarity between documents or words is better estimated in the latent space than the original, as when word or document vectors are grouped together a hidden concept is implied. Logically, "documents that share frequently co-occurring terms will have a similar representation in the latent space even if they have no terms in common" as explicitly mentioned in the publication. In other words, LSA manages to reduce the noise in a text and detect words that refer to the same topic.

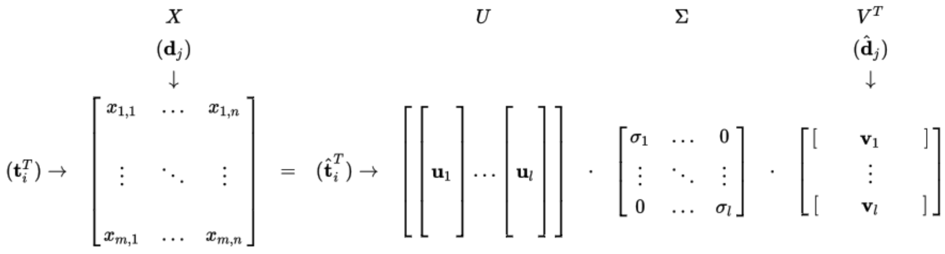


Figure 2.3: Singular Value Decomposition technique used in LSI for dimensionality reduction.

Although this technique was successful in domains such as indexing ((Deerwester et al., 1990)) it faced problems due to its unstable statistical background. As a result, it was later improved to Probabilistic Latent Semantic Indexing (pLSI) or Probabilistic Latent Semantic Analysis (pLSA), by Hofmann (1999). It's basic idea is that a document can be viewed as a mixture model, where the mixture components are multinomial random variables and can be interpreted as topics. Hence, each word can be drawn from a topic and different words in a document can be generated from different topics. It is generally a method that may be observed by two different perspectives. Firstly, as a Latent Variable Model, whose structure is based on a statistical model, called the aspect model. The hidden variables of this model correspond to topics or concepts and are associated with the observed variables that represent the documents and words in a text collection. Each document is represented as a probabilistic mixture of multinomial variables and consequently expressed as a distribution over topics. Secondly, as a Matrix factorization technique. Similarly to LSA, it aims to reduce the dimensionality of the co-occurrence matrix between words (rows) and documents (columns), called the document-term matrix. However, pLSA is based on a more solid statistical and probabilistic interpretation than LSA, that only uses a mathematical formula. In figure 2.4 the structure of the aspect-model of pLSA is shown. In summary, pLSA is a generative process for documents

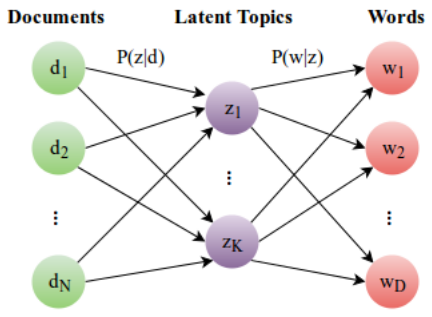


Figure 2.4: The structure of pLSA model.

that firstly selects a document (from the collection), with a certain probability, and for each word in this document it selects 1) a topic from a multinomial distribution given the document and 2) a word from a multinomial distribution given the topic. The model's parameters can be estimated by finding the values that maximize the predictive probability for the observed word occurrences via the Expectation-Maximization (EM) algorithm. On the other hand, if the second perspective is preferred, in order to reduce the size of the co-occurrence document-term matrix (NxM), a decomposition technique is applied into three different matrices: U matrix with word vectors, D diagonal matrix with singular values and V document vectors.

However, pLSA provides no probabilistic model at the document level. This means that no generative process takes place in order to extract the mixture proportions that constitute the document (e.g. a document is 50% about animals, 30% about food, 20% about cars). As a result, it suffers from 2 major problems: a) the number of parameters grows linearly with the corpus size, leading to overfitting problems and b) there is no indication on how to assign a probability to new-unseen documents.

For this reason, the most well known technique for Topic Modeling was introduced by Blei et al. (2003), the Latent Dirichlet Allocation (LDA). LDA tries to overcome the problems of pLSA by transforming the models variables into hidden random variables. In this way, these variables are not linked with the trained set and consequently not restricted. Additionally, there is a change in the distributions that are now Dirichlet. Therefore, LDA achieves better generalization and avoids overfitting. For a more detailed review of LDA see Chapter 3.

An extension of LDA came four years later again by Blei and Lafferty (2007), the Correlated Topic Model (CTM) and Pachinko Allocation machine (Li and McCallum (2006)). This model aims to improve LDA which does not model the correlation of the occurrence between topics. Although LDA assumes that topics are specified before the generation of documents, and there is no correlation between them, the intuition behind CTM is that one latent topic may be correlated with the presence of another. As an example, a topic about "vacations" is more likely to be also about "restaurants" than about "genetics". An important difference between CTM and LDA is that topic proportions are drawn from a logistic normal prior and not a Dirichlet. As the Dirichlet is a distribution that assumes that components are nearly independent, the logistic normal can model the dependence between components. Under this score, the CTM is able to capture all possible topics that are not independent and as a result it supports more topics.

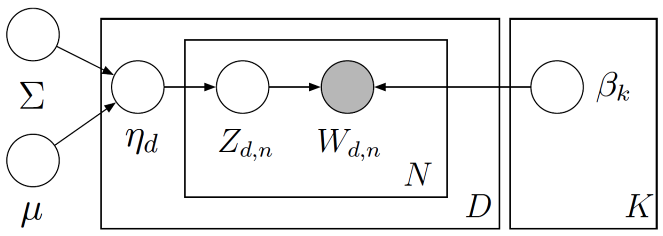


Figure 2.5: Graphical model representation of the CTM (from Blei and Lafferty (2007)).

Other extensions of LDA have been provided through the years. In tasks where the order of words matter in a document (such as language generation) the assumption of negligible word order is not appropriate. A possible solution was introduced by Griffiths et al. (2004) as a model that switches between LDA and a standard Hidden Markov Model (HMM). Although the parameter space is larger, the language modeling performance improves. Additionally, when long-running collections need to be analyzed, where topics change over time, a dynamic topic model was introduced by Blei and Lafferty (2006). The topic is now a sequence of distributions over words, and it is easy to track how it changes over time. For the case where the number of topics needs to be discovered (not assumed known and fixed from the start as in simple LDA), a Bayesian non-parametric topic model was formed by Teh et al. (2006), the Hierarchical Dirichlet Process (HDP). In this model, the number of topics is determined by the data during posterior inference and new documents can exhibit previously unseen topics. In order to relax the assumption that every word is likely in any topic, a spherical topic model allows words to be unlikely in a topic (Reisinger and Mooney (2010)).

In many text settings that additional meta-data need to be included, other techniques have been introduced. In order for each word in a document to influence the probability that the word appears in the document, "bursty" topic models have been developed by Doyle and Elkan (2009). Sparse topic models that enforce the structure in the topic distributions (Wang and Blei (2009)), Relational Topic models (Chang and Blei (2010)) assume that each document is modeled as in LDA and that the links between documents depend on the distance between their proportions and finally, Author-topic models (Rosen-Zvi et al. (2004)) allow inference about authors as well as documents.

2.4 Sentiment Analysis using Topic Models

The previously mentioned approaches for sentiment analysis in sentence level, especially those that use affective lexica, do not include information about the context where the words of a sentence appear. In order to overcome this problem research was focused on how topics and topic models can be used to help estimate the expressed sentiment of a sentence by combining the two.

In the work of Eguchi and Lavrenko (2006) sentiment and topic are detected through user queries. They combine sentiment relevance models and topic relevance models with model parameters estimated from training data, considering the topic dependence of the sentiment.

The most well-known approach that was able to detect not only topics but also sentiments in reviews and was introduced by Mei et al. (2007). In their approach they try to identify the topics in an article, associate each topic with sentiment polarities, and model each topic with its corresponding sentiments by using a topic-sentiment-mixture model. They are based on the assumption that a document can contain different topics and each topic consists of different sentiments using multinomial distributions. In more detail, it divides the words in the document into two major classes: common English words and topic words, while the former can be positive, neutral or negative. By building on this generative process the algorithm is able to 1) learn general sentiment models (one for positive and one for negative opinions), 2) Extract topic models and sentiment coverages, 3) model the topic life cycle and sentiment dynamics.

In the work of Lin and He (2009) a joint model of sentiment and topics (JST) is proposed. In this model there are distinct topic and sentiment labels. With the help of Dirichlet distributions, two additional latent variables are integrated and the generative process is as follows: for each document choose a distribution with parameter 'γ', for each sentiment label choose a distribution with parameter 'α', for each word in the document, choose a sentiment label, a topic and a word from the distribution over words defined by the topic and the sentiment label. This model is completely unsupervised and does not require labeled data.

Another model proposed by Jo and Oh (2011) is an aspect-sentiment unification model. It analyzes the problem of how sentiments for different aspects are expressed. Firstly a Sentence-LDA probabilistic generative model is introduced that assumes all words in a single sentence are generated from one aspect. Then an extension of this model is performed, the Aspect-Sentiment unification model (ASUM) which incorporates aspect and sentiment together to model sentiments towards different aspects. It is similar to the JST model, but it constrains words to come from the same language model.

Rao et al. (2014), propose two sentiment topic models to associate latent topics with evoked emotions of readers. The first model which is an extension of the existing Supervised Topic Model, generates a set of topics from words, followed by sampling emotions from each topic. The second model generates topics from social emotions directly.

Finally, Xiang et al. (2014) is the most similar paper to this work, as they also use a mixture model. The main differences are 1) that in our work, a similarity mixture model is used (not an affective one) and then the sentiment is predicted, using an existing affective lexicon, 2) The clustering is based on LDA and not on K-means and 3) we trained different semantic models and not SVMs which implies that our approach is unsupervised, until the use of the lexicon.

Chapter 3

Latent Dirichlet Allocation (LDA)

3.1 Theoretical Explanation

As the main goal of this Thesis is not to explore in depth how LDA works, but basically understand they way it works for our task, we will describe it's intuition and cognitive motivation firstly [1].

LDA is an algorithm proposed by Blei et al. (2003) that can extract hidden topics in large collections of text. Let's consider the following example.

Example 3.1. *Assume the we have the following sentences:*
1. "I like to eat fish and vegetables"
2. "Cats eat fish"
3. "Fish are pets"

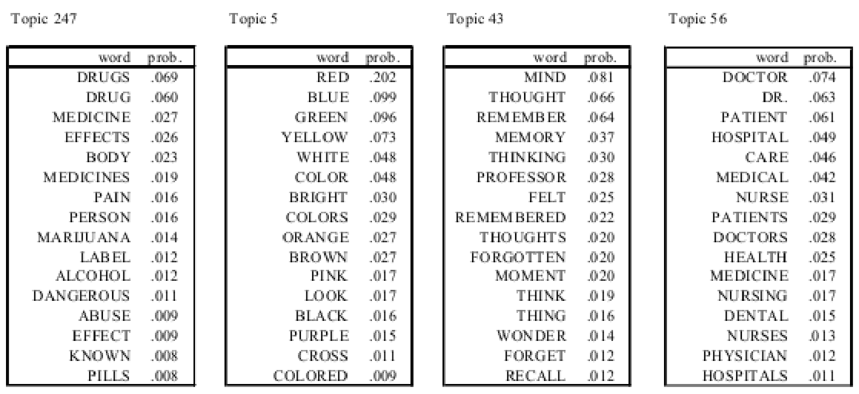
LDA is able to classify the words in these sentence to topics. One can notice that the green words talk about 'food' and the blue words are about 'pets'.

So, overall these sentences contain the following topics:
Sentence 1: 100% food
Sentence 2: 66.7% pets, 33.3% food
Sentence 3: 100% pets

And each topic can contain the following words:
Topic 'food': 40% eat, 40% fish, 20% vegetables
Topic 'pets': 80% fish, 20% cats

Now in a theoretical way, we can describe how this process works. Let's imagine that we have a big collection of documents and we want to identify the different topics, themes or concepts that it contains. Assume that each document discusses multiple topics (like 'sentence 2' in the previous example) and that each topic contains words with certain probabilities, according to their appearance in the topic (larger probability indicates more frequent/common word in the topic). In the previous example, topic 1 talks about pets and so the word 'fish' will be very common in all the documents that contain this topic and consequently this topic will produce the word 'fish' with a large probability. Additionally, the same words can belong to more than one topic ('fish' again according to the example, belongs to both topics).

[1] `https://tedunderwood.com/2012/04/07/topic-modeling-made-just-simple-enough/`


Topic 247

word	prob.
DRUGS	.069
DRUG	.060
MEDICINE	.027
EFFECTS	.026
BODY	.023
MEDICINES	.019
PAIN	.016
PERSON	.016
MARIJUANA	.014
LABEL	.012
ALCOHOL	.012
DANGEROUS	.011
ABUSE	.009
EFFECT	.009
KNOWN	.008
PILLS	.008

Topic 5

word	prob.
RED	.202
BLUE	.099
GREEN	.096
YELLOW	.073
WHITE	.048
COLOR	.048
BRIGHT	.030
COLORS	.029
ORANGE	.027
BROWN	.027
PINK	.017
LOOK	.017
BLACK	.016
PURPLE	.015
CROSS	.011
COLORED	.009

Topic 43

word	prob.
MIND	.081
THOUGHT	.066
REMEMBER	.064
MEMORY	.037
THINKING	.030
PROFESSOR	.028
FELT	.025
REMEMBERED	.022
THOUGHTS	.020
FORGOTTEN	.020
MOMENT	.020
THINK	.019
THING	.016
WONDER	.014
FORGET	.012
RECALL	.012

Topic 56

word	prob.
DOCTOR	.074
DR.	.063
PATIENT	.061
HOSPITAL	.049
CARE	.046
MEDICAL	.042
NURSE	.031
PATIENTS	.029
DOCTORS	.028
HEALTH	.025
MEDICINE	.017
NURSING	.017
DENTAL	.015
NURSES	.013
PHYSICIAN	.012
HOSPITALS	.011

Figure 3.1: Example of Topics produced by LDA from Steyvers and Griffiths (2007).

Of course it is impossible to know from the beginning which are these topics, in the case of thousands of documents. The goal of LDA is to backtrack the procedure of writing a document and find the topics that generated this document. For example, consider figure 3.1. If one gives equal probability to the first two topics, they would create document the talks about a person whose color perception was influenced by the large consumption of drugs. The assumption is that the author has already pre-decided the topics he is going to write about. However, the unknown factors are too many to extract these topics directly. In order to overcome this problem, we assume that we know which topic produces all the words except from a random word w in document d. The only thing that is left to answer is, how can we decide to which topic this word belongs.

In order to answer this question, we can consider a) the frequency of this word in a topic T (words can occur frequently in more than one topic) and b) how common is this topic T to the document. For example, if the word 'bank' is very common in documents that talk about 'money' (topic T), there is a chance that 'bank' belongs to the 'money topic'. However, 'bank' can also occur frequently in a topic about 'water'. So it is very important to select the 'money topic' if the document we took the word 'bank' from talks more about 'money' than 'water'.

Summing all these considerations together, a logical approach is the following: For each possible topic T, multiply the frequency of this word w in T by the number of words in document d that already belong to T (without considering word w). Finally, divide by the total number of words in document d in order to obtain a probability. The result will represent the probability that this word came from T. The mathematical formula is described below:

$$P(T,w|d) = \frac{\text{freq of word w in T} + \beta_w}{\text{total tokens in T} + \beta} \cdot \{\text{\# of words in d that belong to T} + \alpha\} \tag{3.1}$$

In order to make this formula more understandable, in fact, the second term in the product, $\{\text{\# of words in d that belong to T}\}$ describes the proportion of topic T in document d (e.g. document d is about topic T 20%). This way, both the strength of topic T in the document d and of word w in topic T are considered. The parameters β_w, β and α are called hyperparameters and ensure that there is some chance that the word w belongs to topic T even if there is no connection between them. This means that there will be no zero probabilities, only a very small one in such a case.

In the end, we need to make this process generative. Using the formula 3.1 we need some initial variables. This step is done randomly by simply guessing where each word belongs. Then, go through all documents of the collection, word by word and reassign to each word a new probability for each topic. During this procedure, we will notice that words that were common in topics will be grouped together, as well as topics that were common in documents. As a result, topics of specific words (most probable ones) will be formed and become more and more consistent.

3.2 Statistical Explanation

Apart from the the simple-theoretical approach described in the previous section, a more formal one is needed. LDA is method that can create documents, given specific topics and unstructured text. Choosing the example proposed in Blei (2012) (figure 3.2), an article with title "Seeking Life's Bare (Genetic) Necessities" is about using data analysis to determine the number of genes an organism needs to survive. By looking this article closely, one can identify words about 'data analysis', 'evolutionary biology', 'genetics'. As it is observed this document contains multiple topics in different proportions and the fact the it talks about these topics could help us classify it in bigger collections. LDA tries to capture this intuition and generate documents. In statistical language, each document is represented as a distribution over topics and each topic as a distribution over words. Naturally, the topic about 'genetics' contains words related to genetics with a high probability as well as the topic about 'evolutionary biology' contains related words with high probability. One basic assumption is made: *The number of topics is decided before the data has been generated.* Or in other words, that the author of the collection has decided which the topics will be before he started writing. Additionally, all documents in the collection share all topics but they exhibit them in different proportions.

By parsing all documents in the collection, the word generation is achieved in 2 stages. For each document:

- Randomly choose a distribution over topics

For each word in the document:

- Randomly choose a topic from the distribution over topics
- Randomly choose a word from the corresponding distribution over the vocabulary

The goal of LDA is to discover these topics in a collection in an automated way. The observations in the model are the documents, while the topics and more specifically their distributions (document-topic and topic-word) are hidden variables. Consequently, the problem we need to solve is how to predict the hidden distribution based on the observations. Or in a more abstract manner "What is the hidden structure that likely generated the observed collection?". This is a classic problem of inference.

In probabilistic topic models the data are treated as they originated from a generative process that include hidden variables. This process defines a joint probability distribution over both the observed and the hidden random variables. This probability distribution is used to infer the latent variables and create a conditional (or posterior) distribution. Thus, the inference problem is a computational problem of finding this conditional distribution of the topics.

3.2.1 Notation and Terminology

Here we define basic terminology in order to explain the steps of the algorithm.

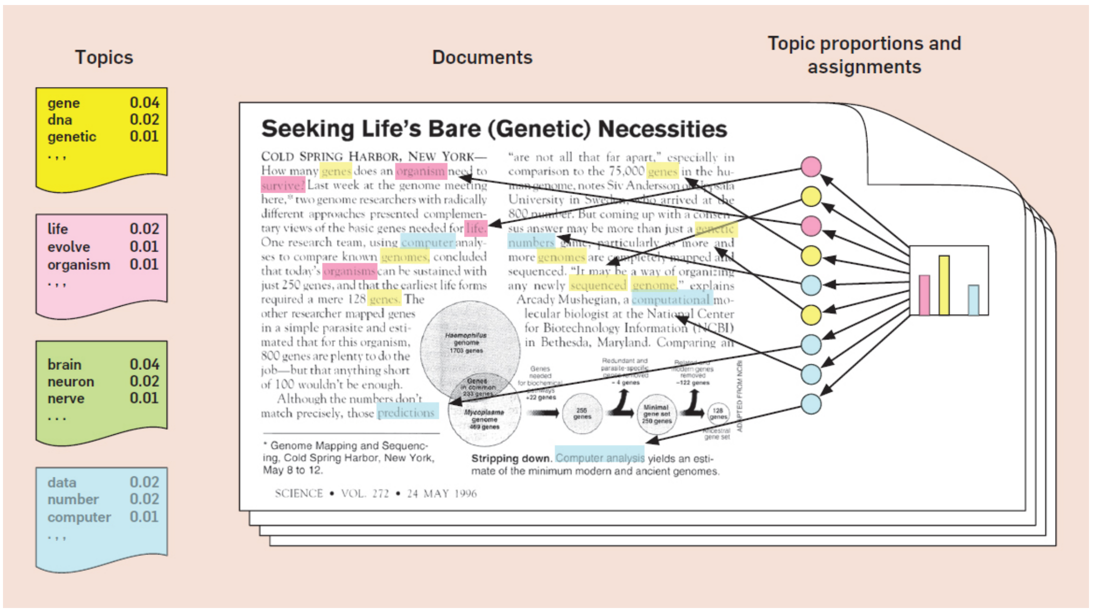


Figure 3.2: Explanation of the intuition behind LDA, be Blei (2012).

- One word is defined as a unigram from a certain vocabulary of size V
- A word is represented as a vector of zero elements, with 1 only to the position of this given word (e.g. $w_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & \dots & 0 \end{bmatrix}$)
- We define a document as a group of N words $\mathbf{d} = \{w_1, w_2, ..., w_N\}$
- We define a corpus as a collection of M documents $\mathbf{C} = \{d_1, d_2, ..., d_M\}$
- A topic β_k is a distribution over the vocabulary
- θ_d is defined as the distribution of topics for document d and $\theta_{k,d}$ is the topic proportion of topic β_k in document d
- z_d is the topic-word distribution for document d and $z_{d,n}$ is the topic assignment for word w_n in document d

3.2.2 Algorithm

In order to formally describe the algorithm, the generative process that LDA is based on is the following procedure:
For each document **d** in a corpus **C**,

1. Choose N $\sim$ Poisson(ξ)
2. Choose $\theta \sim$ Dir(α)
3. For each of the N words in the document:
   (a) Choose a topic $z_i \sim$ Multinomial(θ)
   (b) Choose a word w_n from $p(w_n|z_n, \beta)$, a Multinomial probability conditioned on the topic z_n

The joint distribution of the hidden and observed variables is described below:

$$p(t_{1:T}, \theta_{1:D}, z_{1:D}, w_{1:D}) = \prod_{i=1}^{K} p(t_i) \prod_{d=1}^{D} p(\theta_d) \Big(\prod_{n=1}^{N} p(z_{d,n}|\theta_d) p(w_{d,n}|t_{1:K}, z_{d,n}) \Big) \tag{3.2}$$

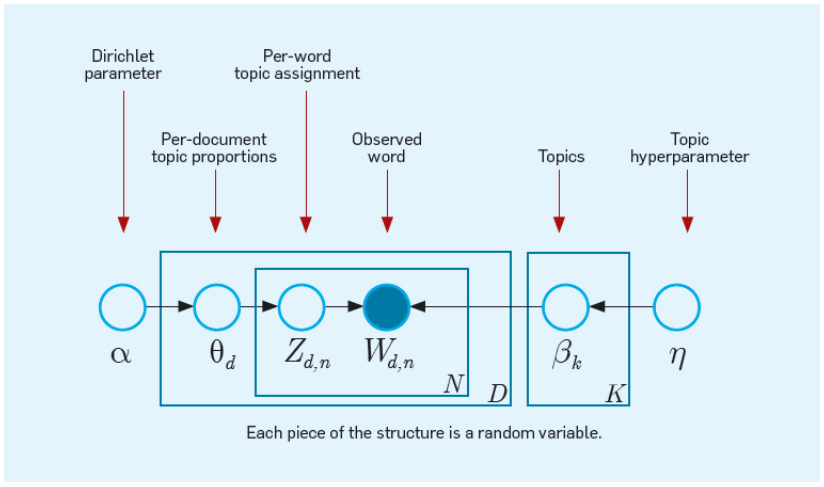


Figure 3.3: Plate notation of LDA.

According to the equation above, multiple dependencies are observed. The $z_{d,n}$ depends on the document-topic distribution θ_d, the word $w_{d,n}$ depends on the topic assignment $z_{d,n}$ and all the topics β. These dependencies are better illustrated in figure 3.3.

Distribution	Probability Mass Function (PMF)
Dirichlet	$\frac{1}{B(\alpha)} \prod_{i=1}^{M} x_i^{(\alpha_i - 1)}, B(\alpha) = \frac{\prod_{i=1}^{M} \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{M} \alpha_i)}$
Poisson	$\exp^{-\lambda} \sum_{i=0}^{M} \frac{\lambda^i}{i!}$
Multinomial	$\frac{\Gamma(\sum_i x_i + 1)}{\prod_i \Gamma(x_i + 1)} \prod_{i=1}^{M} p_i^{x_i}$

Table 3.1: Distributions used in LDA table.

3.2.3 Inference Methods

In order to compute the posterior distribution of the topic structure the Bayes Theorem is used:

$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D}|w_{1:D}) = \frac{p(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D})}{p(w_{1:D})} \tag{3.3}$$

The numerator is the joint distribution of all random variables and the denominator is the marginal probability of the observations, which is the probability of seeing the observed collection of documents under any topic structure. However, the possible number of topic structures is exponentially large thus making it difficult to infer the words related to a given topic and the topics being discussed in a given document.

Topic modeling algorithms try to find ways to estimate the desired posterior probability, they so-called *approximate inference methods* (figure 3.4). They generally fall into two categories: 1) sampling-based algorithms and 2) variational algorithms.

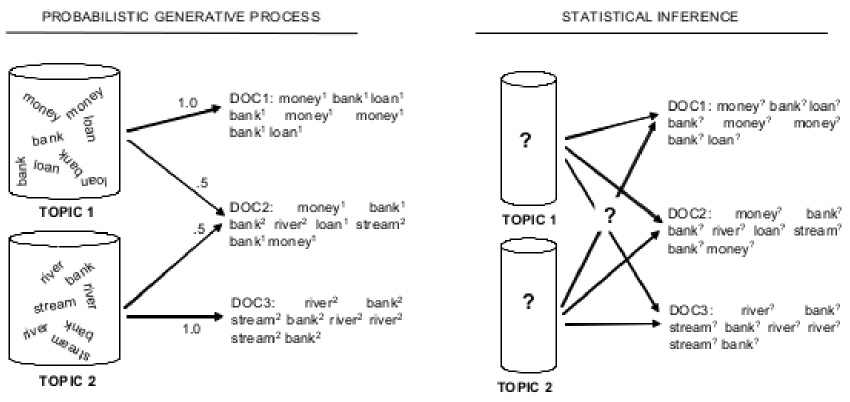

PROBABILISTIC GENERATIVE PROCESS
STATISTICAL INFERENCE
TOPIC 1
TOPIC 2
1.0
.5
.5
1.0
DOC1: money¹ bank¹ loan¹ bank¹ money¹ money¹ bank¹ loan¹
DOC2: money¹ bank¹ bank² river² loan¹ stream² bank¹ money¹
DOC3: river² bank² stream² bank² river² river² stream² bank²
?
DOC1: money? bank? loan? bank? money? money? bank? loan?
DOC2: money? bank? bank? river? loan? stream? bank? money?
DOC3: river? bank? stream? bank? river? river? stream? bank?

Figure 3.4: Generative process along with the problem of inference (source: Steyvers and Griffiths (2007))

Sampling-based algorithms attempt to collect samples from the posterior in order to construct an approximation as an empirical distribution. The most commonly used sampling algorithm for topic modeling is Gibbs sampling. It is a Markov Chain Monte Carlo (MCMC) algorithm that helps obtain a sequence of variables from a probability distribution when direct sampling is difficult. It's main characteristic is the it uses random numbers and is an alternative to the deterministic method of Expectation-Maximization (EM). In Gibbs sampling, a Markov chain is constructed (a sequence of random variables, dependent on the previous) whose limiting distribution is the posterior. If we run this algorithm for enough time we will collect many samples and it will be able to construct a probability approximation.

Variational methods on the other hand, rather than approximating the posterior with samples, assume that a family of distributions exists over the hidden structure (lower-bounds on the log likelihood) and find which member is closer to the desired posterior. Thus, the inference problem is now an optimization problem. In order to obtain these families, one can simplify the plate-notation model of figure 3.3 by removing edges between θ, z and w. If the w nodes are removed as well, the resulting model will have free variational parameters. The optimal values are found by minimizing the Kullback-Leibler (KL) divergence.

Chapter 4

Semantic and Affective Models

4.1 Distributional Semantic Models

The ideas of formal semantics, also discussed in Chapter 2, include the representation of phrases, sentences and words in terms of set-theoretic models. The main intuition behind semantics is that the world consists of objects with properties and relations exist between them. Models that capture this intuition have been developed and proved to be useful for computational semantics. There is a tendency to use vector spaces in order to represent the meaning of sentences, as they provide a natural mechanism for calculating similarity and distance. In this way the step of comparing words and larger lexical units becomes very efficient.

4.1.1 Definition

Models like the ones mentioned before, are considered a kind of knowledge models. These models, that have been developed for decades, are also known as vector spaces, semantic spaces or word spaces. As mentioned in Baroni and Lenci (2010) Distributional Semantic Models (DSMs) rely on some version of the distributional hypothesis (Harris (1954), Miller and Charles (1991), Turney et al. (2010)): "Words that occur in similar contexts tend to have similar meanings". In other words, "the degree of semantic similarity between two words can be modeled as a function of the degree of overlap among their linguistic contexts". Vector space models (VSMs) represent (embed) words in a continuous vector space where semantically similar words are mapped to nearby points ('are embedded nearby each other').

Usually, the bag-of-words (BOW) format is used for vector representation. This means that the order of the words in a document does not matter, only their occurrence frequency. Obviously, this is a vague assumption since the meaning of a word is strongly connected with the order of the words surrounding it. However, this simple approach seems to have good results as it generally captures the context of the word. At the end of the vectors construction we will be able to compare these vectors, using a distance metric, cosine or euclidean for example, and estimate their similarity.

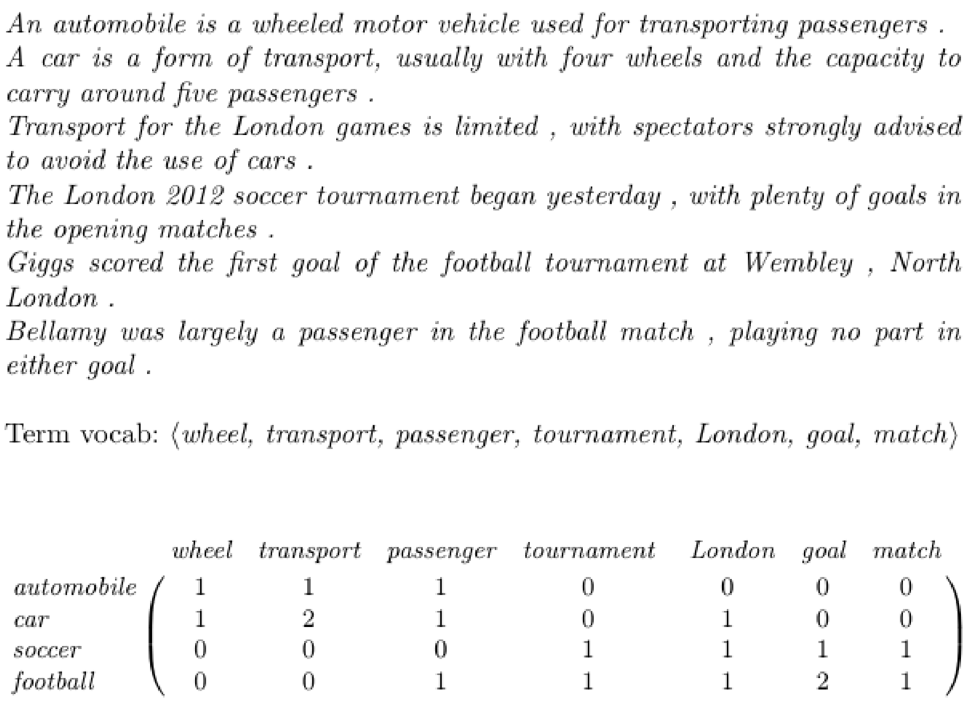
Distributional semantic models differ primarily with respect to the following parameters:

- Context type (text regions vs. linguistic items)
- Context window (size, extension, etc.)

- Frequency weighting (e.g. entropy, point-wise mutual information, etc.)
- Dimension reduction (e.g. random indexing, singular value decomposition, etc.)
- Similarity measure (e.g. cosine similarity, Jaccard, etc.)

4.1.2 Functionality

In general, in order to construct these vectors, a) a corpus and b) a vocabulary are needed. If the vocabulary is not provided one can be extracted from the existing corpus. As a first step, one could think to represent each word as a vector with the frequencies of co-occurrence with all the words in the vocabulary. This means that if the vocabulary contains 1000 words, each word will be represented as a 1000D vector. Each element of this vector will contain the frequency of co-occurrence of these 2 words (word of interest (row) and element-word (column)) in the whole corpus. However, this is a very broad notion of word similarity (these vectors would seem to be produced randomly), and for this reason we have to narrow down this search in order to find a *topical* similarity. To achieve that, we consider only the co-occurrence with a few words either side of the target word, also known as a 'window'.



An automobile is a wheeled motor vehicle used for transporting passengers .
A car is a form of transport, usually with four wheels and the capacity to carry around five passengers .
Transport for the London games is limited , with spectators strongly advised to avoid the use of cars .
The London 2012 soccer tournament began yesterday , with plenty of goals in the opening matches .
Giggs scored the first goal of the football tournament at Wembley , North London .
Bellamy was largely a passenger in the football match , playing no part in either goal .

Term vocab: ⟨*wheel, transport, passenger, tournament, London, goal, match*⟩

$$\begin{array}{r|ccccccc} & \textit{wheel} & \textit{transport} & \textit{passenger} & \textit{tournament} & \textit{London} & \textit{goal} & \textit{match} \\ \textit{automobile} & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ \textit{car} & 1 & 2 & 1 & 0 & 1 & 0 & 0 \\ \textit{soccer} & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ \textit{football} & 0 & 0 & 1 & 1 & 1 & 2 & 1 \end{array}$$

Figure 4.1: Example of term-term BOW matrix, using term co-occurrence frequency and window = 1. Image from Clark (2012).

Figure 4.1 show an example of this construction from a toy corpus using a window of size 1 (only consider 1 word from the left and 1 from the right of the target word). The rows represent the target words and the columns are the vocabulary words that are used as context-features. It is not obligatory for the target words to be included in the vocabulary, only to exist in the corpus. As a next step, the similarity between these vectors can be calculated using the cosine angle between them.

$$sim(\overrightarrow{v_1}, \overrightarrow{v_2}) = \frac{\overrightarrow{v_1} \cdot \overrightarrow{v_2}}{\|\overrightarrow{v_1}\| \cdot \|\overrightarrow{v_2}\|} \tag{4.1}$$

As a result, for the example above, we obtain:
sim(automobile, car) $= 4/\sqrt{21} = 0.87$
sim(automobile, soccer) $= 0/\sqrt{12} = 0$
sim(automobile, football) $= 1/\sqrt{24} = 0.2$
sim(car, soccer) $= 1/\sqrt{28} = 0.19$
sim(car, football) $= 2/\sqrt{56} = 0.27$
sim(soccer, football) $= 5/\sqrt{32} = 0.88$

The term-term co-occurrence frequency matrix can be constructed with different values, such as Point-wise Mutual Information (PMI) scores ($pmi(x, y) = log\frac{p(x,y)}{p(x)p(y)}$), TF-IDF scores, binary (1-0) for presence or not of the feature word in the context etc. Reduction techniques can also be applied in case the vocabulary size is too large. One disadvantage of the SVD technique (the most common one) is that the induced hidden dimensions are difficult to interpret, whereas basis vectors, can be related to conceptual properties and given psycho linguistic interpretation as mention in Baroni and Lenci (2010). Also, except from the cosine similarity, multiple types of other similarity metrics can be used, such as the Pearson coefficient, Jaccard coefficient, Tanimoto coefficient and the Euclidean. These coefficients are summarized in the following table.

Similarity Type	Formula
Cosine	$cos(\theta) = \frac{\overrightarrow{v_1} \cdot \overrightarrow{v_2}}{\|\overrightarrow{v_1}\| \cdot \|\overrightarrow{v_2}\|}$
Euclidean	$d(v_1, v_2) = \sqrt{\sum_{i=1}^{N} (v_{1i} - v_{2i})^2}$
Pearson	$\rho_{v_1,v_2} = \frac{cov(v_1,v_2)}{\sigma_{v_1}\sigma_{v_2}}$
Jaccard	$J = \frac{M_{11}}{M_{01}+M_{10}+M_{11}}$
Tanimoto	$T(\overrightarrow{v_1}, \overrightarrow{v_2}) = \frac{\overrightarrow{v_1} \cdot \overrightarrow{v_2}}{\|\overrightarrow{v_1}\|^2 + \|\overrightarrow{v_2}\|^2 - \overrightarrow{v_1} \cdot \overrightarrow{v_2}}$

Table 4.1: List of similarity metrics for word-vectors comparison.

Vector space models, can be divided into two categories, a) count-based and b) predictive methods. As stated in Baroni and Lenci (2010), 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 in terms of learning small, dense embedding vectors (considered parameters of the model). Previously, we described the first category and we continue with the most well known model from the second one.

4.1.3 Word2Vec

Word2vec is a particularly computationally-efficient predictive model for learning high-quality word embeddings from raw text and is the one we used in this thesis. It was introduced by Mikolov et al. (2013) and in their paper they show how neural network based language models significantly outperform N-gram models. The former, can also be trained on larger amounts of data and millions of words in the vocabulary. As they additionally mention, it was found that similarity of word representations goes beyond simple syntactic regularities. Using a word offset technique where simple algebraic operations are performed on the word vectors, it was shown for example that $vector(\text{“}King\text{”}) - vector(\text{“}Man\text{”}) + vector(\text{“}Woman\text{”})$ results in a vector that is closest to the vector representation of the word “*Queen*”.

The general structure is based on the probabilistic feedforward neural network language model, which has been proposed in Bengio et al. (2006). It consists of input, projection, hidden and output layers. At the input layer, N previous words are encoded in a one-hot coding, where V is the size of the vocabulary. This means that each word is represented as a vector of zeros with one only in the position of this word. The input layer is then projected to a projection layer P that has a dimentionality NxD, using a shared projection matrix. In Word2Vec hierarchical softmax can also be used, where the vocabulary is represented as a Huffman binary tree. There are 2 possible architectures, the Skip-gram Model and the Continuous Bag-of-Words (CBOW). The parameters of Word2Vec are explained in detail in Rong (2014).

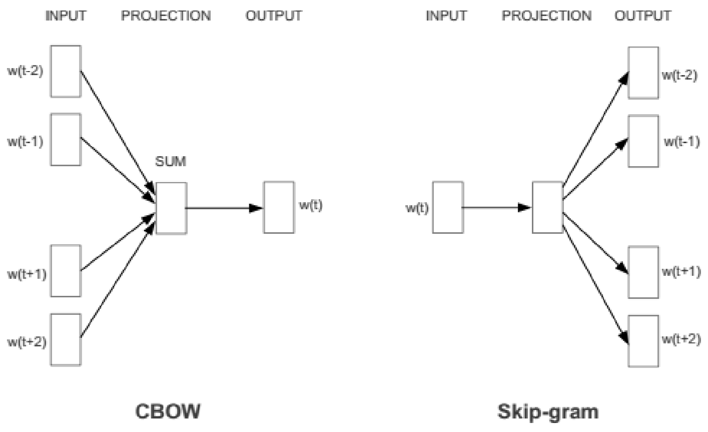


Figure 4.2: CBOW and Skipgram architectures of Word2Vec.

Skip-Gram

The skipgram model tries to predict the context of a target word, given this word. In fact, it learns two separate embeddings for each word w, the word embedding and the context embedding. They are encoded in two matrices, the word matrix W and the context matrix C. The rows of the first are the word vectors with dimension d (1×d) (for a word in the vocabulary) and the columns of the second are the context vectors (d×1). The skipgram model, predicts each neighboring word in a context window L, $win = [w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}]$. In the end, we want to compute $P(w_k|w_j)$ (w_j is the target, w_k is the context). The procedure is the following:

1. The input vector x is a one-hot for the target word w_j
2. Predict the probability for each for the 2L words in win
   (a) Select the vector from W and multiply it with x to produce the projection matrix. The projection layer will be the embedding for w_j (v_j)
   (b) Compute the dot product $v_j \cdot c_k$ which gives the output scores for each vocabulary word
   (c) Normalize the dot products into probabilities using a soft-max function

CBOW

CBOW is a mirror image of skipgram, as it predicts the target word given the context words. The main difference is in the computation of the projection layer. Again there are 2 d-dimensional embeddings, for the words and the contexts that form the matrices W and C (transposed compared to skipgram ones). Now we want to predict $P(w_k, w_j)$ (w_k is the target, w_j is the context). The input layer is connected to the projection layer by a $C = |V|xd$ context matrix. This matrix is repeated between each one-hot input vector and the projection layer. The multiplication of vector x and matrix C will give 2 context vectors, of which we take the average and form the projection layer. Afterwards, the projection layer is multiplied with the word matrix W resulting the V scores for each word in the vocabulary. Finally, these scores are transformed into a probability using a soft-max layer.

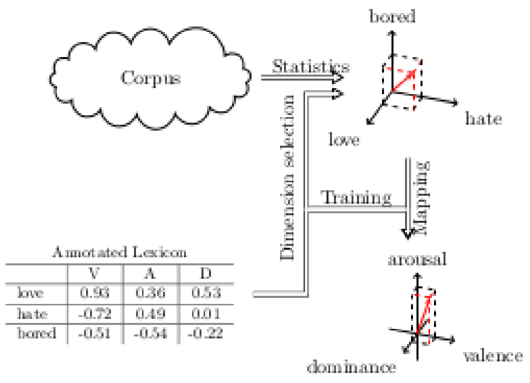
4.2 Affective Model

The model proposed by Malandrakis et al. (2011a) uses an existing affective lexicon that defines the semantic orientation of words (unigrams) in continuous affective scores. The main assumption of this model is that "Semantic similarity can be translated to affective similarity" and thus a mapping from the semantic to the affective space is built. As a result, given the semantic similarity between two words, the affective rating of the one given the other can be determined. The idea was based on the work of Turney and Littman (2002) that used a set of words with known affective ratings, known as *seed words*. Then the semantic similarity of a seed word and a new word is computed with the help a distributional semantic model and used to estimate the affective score of the new word. The initial seed words can be labels of affective categories ("anger", "happiness"), words with continuous scores etc. The final step is to combine the scores of the seed words in order to extract the score of the unseen word.

4.2.1 From Semantic to Affective Space

Generally, semantic models can be seen as a cognitive process of analyzing the human behavior when it comes to language understanding. Therefore, appears a need for a model that can capture sentiment. The Semantic-Affective Model (SAM)(Malandrakis et al. (2011a),Turney and Littman (2002)) aims to bridge the gap between the semantic and the affective space, as well as to transfer the knowledge from one domain to the other. The assumption on which it is based can be summed up in the phrase "Semantic similarity implies affective similarity". The concept is presented in figure 4.3.

The model assumes the existence of two spaces a) a semantic space, where all words are somehow connected to each other by their semantic meaning and b) an affective space,



Annotated Lexicon

	V	A	D
love	0.93	0.36	0.53
hate	-0.72	0.49	0.01
bored	-0.51	-0.54	-0.22

Figure 4.3: Semantic-Affective model as proposed in Malandrakis et al. (2014).

where words are represented by their affective meaning as points in 3 dimensional space of valence, arousal and dominance (described in 1). A large generic corpus is used to built the semantic model from which similarities between pairs of words will be extracted and used in the SAM. The goal is to find the appropriate mapping, in order to estimate affective scores for unseen words using both spaces. This procedure is outlined in figure 4.4.

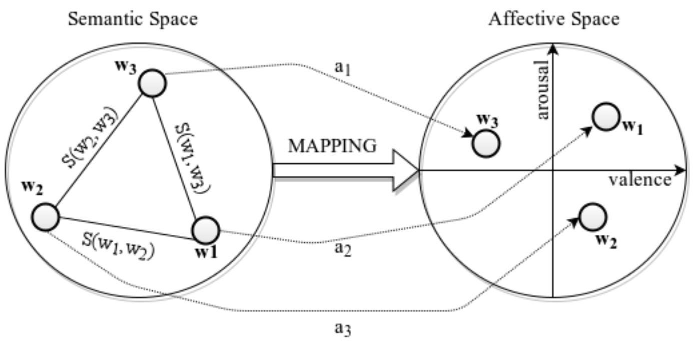


Figure 4.4: Semantic-Affective Mapping concept.

4.2.2 Word-level tagging

In order to estimate the score of a word, the model uses an existing annotated lexicon from where only a dataset is actually used (seed words). The affective scores usually are in the valence dimension in a range of $[-1, 1]$ (from very positive to very negative), *from the reader perspective*. Additionally, the semantic model built using a corpus is used to extract word-pairs similarity scores. Finally, the semantic-affective mapping is calculated using both the lexicon and the similarity scores in a training process. The ratings of the new word, are estimated as the linear combination of the ratings of seed words. The following equation summarizes the above:

$$v(w_j) = \alpha_0 + \sum_{n=1}^{N} \alpha_i \cdot v(w_i) \cdot d(w_i, w_j) \tag{4.2}$$

where w_j is the word we mean to characterize, $w_1, ..., w_N$ are the seed words, $v(w_i)$

is the valence rating for seed word w_i, α_i is the weight corresponding to word w_i and $d(w_i, w_j)$ is a measure of semantic similarity between words w_i and w_j.

The weights α_i are basically the semantic-affective mapping we described before and can be learned by solving a linear system. If we have a corpus with K annotated words, and choose N of them as seeds ($N < K$), we can estimate the weights α_i of the model. We can use equation 4.2 to create K linear equations with $N+1$ unknown variables. These α parameters can obtain their optimal value using Least Mean Squares (LMS) or a regularized version known as Ridge Regression (RR). The LSE method is described in equation 4.3.

$$\mathbf{X} \cdot \alpha = \mathbf{y}, \text{ where } \hat{\alpha}_{lse} = argmin_{\alpha_{lse}} ||\mathbf{y} - \mathbf{X}\alpha_{lse}||^2 \tag{4.3}$$

On the other hand, for the RR method, an additional parameter λ is introduced that works as a regularization factor which used tries to best fit a curve into the data (lexicon affective scores) as shown in equation 4.4.

$$\hat{\alpha}_{rr} = argmin_{\alpha_{rr}} \Big(||\mathbf{y} - \mathbf{X}\alpha_{rr}||^2 + \lambda ||\alpha_{rr}|| \Big) \tag{4.4}$$

$\mathbf{X}$ is a matrix containing all similarity pairs between the seed words and the whole affective lexicon of size $K \times (N+1)$, α is the coefficients (mapping) vector of size $((N+1) \times 1)$ and y is the vector with the affective scores for the whole lexicon of size $(K+1) \times 1$.

$$\begin{bmatrix} 1 & d(w_1, w_1)v(w_1) & \cdots & d(w_1, w_N)v(w_N) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & d(w_K, w_1)v(w_1) & \cdots & d(w_K, w_N)v(w_N) \end{bmatrix} \cdot \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_N \end{bmatrix} = \begin{bmatrix} 1 \\ v(w_1) \\ \vdots \\ v(w_K) \end{bmatrix} \tag{4.5}$$

where $\alpha_1, ..., \alpha_N$ are the N seed word weights and α_0 is an additional parameter that works as an affective bias in the seed word set.

The main motivation behind the idea of training these parameters, is the fact that semantic similarity does not fully capture the relevance of a seed word for valence computation, as mentioned in Malandrakis et al. (2011a). Illustrating an example, assume that the word 'love' is the word we want to estimate the affect for and there are two seed words, for example 'apple', 'life' that have the same semantic similarity with 'love'. Using the assumption that "semantic similarity implies affective similarity", both the scores of these words will be included in the sum with the same weight. However, not both words should be weighted the same as they may express totally different contexts or are unambiguous. This would result a misleading general sentiment for the word of interest. The LMS machine learning method enables us to capture exactly this intuition.

In order to select the seeds words (N) that will be used to estimate an unknown word, a feature selection method is performed. In particular, we aim that the subset that is collected is balanced, which means that the sum of the scores of each individual word should be close to zero.

The semantic model can be any distributional semantic model (like the ones described in the previous section and Chapter 2) and the similarity metric that is used in the model can be any similarity measure between vectors of two words. Additionally, kernels can be applied to this metric for better performance as proposed in Malandrakis et al. (2011b).

Generally, the affective lexicon that is used for this method is the Affective Norms for English Words (ANEW) proposed by Bradley and Lang (1999) which contains 1034 English words with valence, arousal and dominance scores between [-1,1]. Additionally,

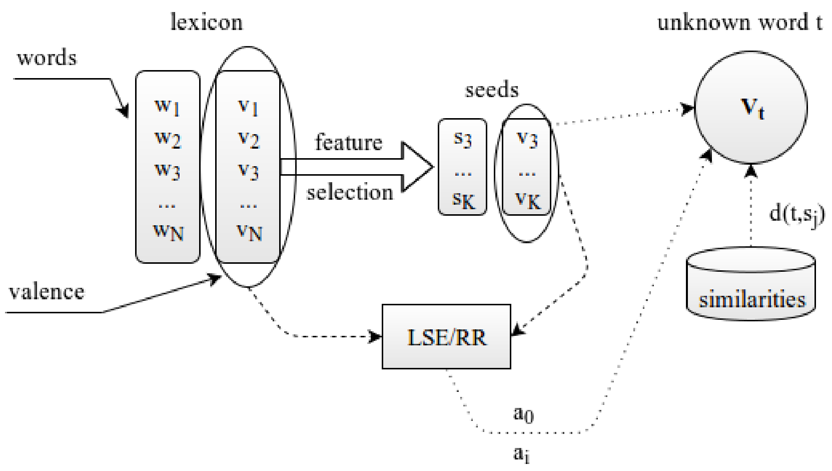


Figure 4.5: Unseen word affective score estimation procedure using SAM.

the five basic emotions are also included in this lexicon. An example of how the training is done is shown below:

Order	w_i	$v(w_i)$	α_i	$v(w_i)x\alpha_i$
1	triumphant	0.96	0.34	0.33
2	love	0.93	0.78	0.73
3	paradise	0.93	2.07	1.93
4	loved	0.91	1.72	1.57
5	joy	0.9	2.57	2.31
6	rape	-0.94	2.56	-0.09
7	suicide	-0.94	2.56	-2.5
8	funeral	-0.9	0.09	-0.08
9	cancer	-0.88	0.77	-0.68
10	rejected	-0.88	0.83	-0.73
-	(offset)	1	-0.11	-0.11

Table 4.2: Example of training with 10 seed words from ANEW, using LSE.

In summary, the aforementioned model estimates a new-unseen word using scores of existing words in different proportions, based on their measure or relatedness and a trainable weight factor. Generally, it can compute generic affective scores very well, or adapt them in a specific context by training the coefficient weights with another corpus (Malandrakis et al. (2011b)).

4.2.3 Sentence-level tagging

Fusion

Moving on to the sentence level, the principle of compositionality (Pelletier (1994)) states that the meaning of a sentence is the sum of the meaning of its parts. So in order to estimate the affective score of a sentence using the affective scores of its words can be achieved using three fusion models, as proposed in Malandrakis et al. (2011a).

1. Linear Fusion
$$v_1(s) = \frac{1}{N}\sum_{i=1}^{N} v(w_i) \tag{4.6}$$
In this fusion scheme, simply the average of a sentences words scores will give the overall score for this sentence. This equally weights all words and tends to give lower scores to sentences that contain many neutral words.

2. Weighted Fusion
$$v_2(s) = \frac{1}{\sum_{i=1}^{N} |v(w_i)|}\sum_{i=1}^{N} v(w_i)^2 \cdot sign(v(w_i)) \tag{4.7}$$
This is a normalized weighted average, in which words with higher affective scores are weighted more.

3. Non-linear Max Fusion
$$v_3(s) = max_i(|v(w_i)|) \cdot sign(v(w_z)), \text{ where } z = argmax_i(|v(w_i)|) \tag{4.8}$$
Finally, a non-linear fusion that characterizes the sentence with the same score as the highest word in the sentence.

Classification

Additionally, in order to predict the affective score for a sentence the state-of-the art method includes the use of a classifier. More specifically, if the affective scores for each word in a sentence are obtained, some statistics are extracted and work as features. In this way, each sentence is represented as a feature vector, with values the statistics from its words scores.

The main difference between this approach and the fusion is that the first requires training data in order to train the classifier, while the latter is completely unsupervised as far as the evaluation is concerned.

The statistics that can be used, as proposed in Malandrakis et al. (2011b) are:

- length (cardinality)
- max
- min
- mean
- variance
- standard deviation
- range (max - min)
- amplitude
- sum

Normalized version of these features can be created, according to Part-of-Speech Tagging (POS Tagging). For example, feature max over all nouns divided by the max over all tokens. If we consider nouns, adverbs, adjectives and verbs as POS tags of interest, we end up with 27 features, which means a 27D feature vector for each sentence. In a more simple case, we can extract simple features and normalized them by the length of the sentence, leading to a 18D vector.

A classifier can be then trained using some labeled sentences and used to predict the affective scores of new sentences. Naive Bayes Tree, Random Forest, SVM are the most common cases. The classification is usually binary, between positive and negative sentences but the 3-class problem can be also performed, using the neutral class additionally.

Chapter 5

Sentence-level Adaptation

5.1 Motivation

As simple lexicon based models fail to capture the context where a sentence belongs, topic modeling methods try to incorporate topic knowledge with a view to lead to a more specific detection of the expressed sentiment in a sentence. This can be viewed as two-step classification process, where firstly the sentence is classified to a topic and secondly in a positive or negative sentiment class. Based on the fact that "semantic similarity implies affective similarity" the idea is to split a large collection of documents into the topics it consists of and train multiple semantic models in these specific domains. The plan is to combine a different subset of these models in a mixture model, for each test sentence. Actually, the semantic similarities of the words in a domain are recalculated (for each topic) and then merged to create new semantic similarities, unique for each sentence. After adapting the semantic space to each topic, we are able to use the Semantic-Affective Model described in Chapter 4 to connect the semantic and the affective space, estimate the affective scores of a sentence's words and then estimate the affective score of the whole sentence.

This idea was motivated by the fact that through this procedure a form of word disambiguation can be achieved. For example, if a sentence talks about computers it is likely to contain the word 'apple', but this can also be the case if it is about food. After training a topic model, it will be able to distinguish that a sentence with the word 'apple' is about fruit and classify it to the topic that is related to food (as we saw in Chapter 3). For a new sentence, the topics that is likely to be about will be selected and the similarities that are needed in equation 4.2 will be obtained from the semantic models of these topics.

In summary, we can use multiple semantic models from the different domains that a sentence can possibly contain, in order to construct more accurately its semantic space.

5.2 Topic Modeling

5.2.1 Latent Model training

The data required for this task is a large corpus. It can be a generic one (downloaded from the web (e.g. Wikipedia)) or a collection formed from reviews (e.g. Movie reviews).

We will need two versions of this corpus. One that contains documents (from paragraphs to larger lexical units) and another the contains sentences. The first corpus will be used as input to the Latent Dirichlet allocation algorithm as in its default form assumes that each document contains multiple topics. In the case of sentences this assumption cannot stand (how many topics can a single sentence contain?), so documents must be the input. The sentences corpus will be used in order to construct the topics. The main intuition behind this idea is that smaller pieces of document (sentences) will possibly talk about one topic, so the classification can be strict (1 sentence → 1 topic).

So as to create topics, we give as input to the LDA the document corpus. The toolbox that enabled us to work with large amounts of data is the Gensim Toolbox, created by Řehůřek and Sojka (2010). More information about this toolbox is included in Appendix A.

The inputs of LDA are:
- A vocabulary
- A corpus in bag-of-words (BOW) format

We extract a vocabulary from the corpus. If it is too large infrequent terms can be trimmed. The BOW format is constructed based on this vocabulary, for each document in the corpus. This means, that each document is represented as a V-dimensional vector with the frequency of each element (word in the vocabulary) as value.

The outputs of LDA are 2 distributions,
- distribution of document-topics
- distribution of topic-words

The first, includes the topics that each document is likely to contain and their probability. For example 40% Topic 1 (e.g. food), 30% Topic 5 (e.g. animals) and 30% Topic 9 (e.g. music). The second, includes the words that each topic is likely to contain, by assigning them a probability. An example is shown in table 5.1 similar to figure 3.1.

Topic 1		Topic 2		Topic 3		Topic 4	
space	0.06	recipe	0.07	music	0.06	medical	0.09
astronomy	0.05	recipes	0.07	guitar	0.06	treatment	0.08
moon	0.05	food	0.06	piano	0.05	health	0.07
astrology	0.04	cooking	0.06	classical	0.06	disease	0.07
star	0.04	chicken	0.06	jazz	0.045	cancer	0.06
solar	0.03	chocolate	0.06	sheet	0.04	symptoms	0.06
NASA	0.03	kitchen	0.06	dance	0.03	pain	0.06
earth	0.02	cheese	0.05	free	0.02	causes	0.04
mars	0.02	cake	0.02	opera	0.01	surgery	0.04
news	0.02	home	0.01	band	0.01	answers	0.03

Table 5.1: Example of Topic Modeling output, distribution words in a topic, using the top 10 most probable words.

Using these two distributions, we can then classify each *sentence* of the large collection (sentences corpus) into the topics that we chose to generate. The concept is to apply the trained LDA model to each corpus-sentence, and receive in return the topics that it may belong as well as their probabilities.

Firstly, the sentence needs to be represented as a bag-of-words (BOW) vector, with the words of the vocabulary.

An example is shown below:

Example 5.2. *Assume that our sentence of interest is:*
s = "Test to predict breast cancer relapse is approved."
After applying a trained LDA model on 60 topics, the result is:

Topic 1: 0.687, Topic 5: 0.174
This means that this sentence, has 68.7% chance to belong to topic 1 and 17.4% chance to belong to topic 5. The contents of these topics, in terms of the 10 most probable words they contain, are:
Topic #1: health, medical, cancer, treatment, disease, symptoms, pain, information, medicine, causes
Topic #5: drug, health, online, drugs, marijuana, alcohol, effects, smoking, addiction, sleep

We can notice that both topics talk about health, the first as a general domain and the second as more specific on drugs. Given the sentence, it seems that LDA correctly classifies it into these topics, giving more weight to the first one.

5.2.2 Topics Construction

The next step is to collect the appropriate sentences from the corpus collection and build topic-clusters. We can perform a strict clustering, which means to classify each sentence only to one topic or to perform soft clustering, which means classifying each sentence to multiple topics. In the first case, the sentence is classified to the topic that gives the maximum posterior probability. In the second one, the sentence is classified to all topics that have a posterior above a threshold.

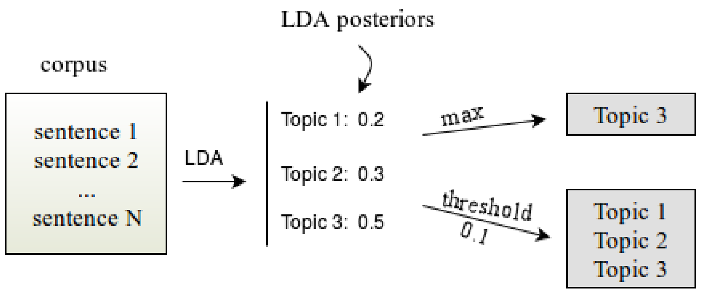


Figure 5.1: Topic construction diagram.

If a sentence belongs to all the topics with the same probability it is not classified anywhere as it is considered very ambiguous. Additionally, if LDA fails to return a possible topics for a sentence this sentence is also not classified anywhere. Figure 5.1 outlines the topics construction procedure.

After the topic-clusters construction, a separate semantic model is trained for each topic. For this purpose different DSMs can be trained, but we used the Word2Vec tool as a state-of-the art model. We tune its parameters (Chapter 6) and estimate the similarities between words for each topic using the sub-corpora created in the previous step.

5.3 Semantic Model Adaptation (SMA)

For the final step, we need to combine the semantic models trained for each topic so as to adapt the semantic space of a test-sentence.

5.3.1 Mixture of Models

The procedure can be described through the algorithm below.

1. Transform a test sentence in BOW format
2. Apply a trained LDA model to this sentence
3. Receive topics along with their posterior probabilities for this sentence
4. For each word in the sentence and a seed word (with known affective score), extract the similarity between the two from all the Topics that the sentence can be classified
5. Multiply each similarity with the topic-posterior and calculate the average over all topics

The combination of these models is summarized in the equation below:

$$sim(w_j, seed_i) = \frac{\sum_{t=1}^{T} p(top_t|s) \cdot sim_t(w_j, seed_i)}{\sum_{t=1}^{T} p(top_t|s)} \tag{5.1}$$

where $s = \{w_0, w_1, ..., w_j, ..., w_n\}]$ as a test sentence, w_j is a word in the sentence, $seed_i$ is a seed word, with known affective score, $p(top_t|s)$ is the topic posterior probability (probability for sentence s to contain topic t) and $sim_t(w_j, seed_i)$ is the similarity obtained from topic t between the words w_j and $seed_i$.

Overall, the model weights each similarity by the topic posterior and in the end produces a new similarity for a given pair.

Example 5.3. *Assume that the sentence of interest is the one in example 5.1 and that we use as seed words, the ones shown in table 4.2 from Chapter 4. In order to estimate the new-adapted similarity between the word 'breast' (word in the sentence) and 'cancer' (seed word) we extract the similarity between these words from topic 1 and topic 5.*
$sim_1(breast, cancer) = 0.32, sim_5(breast, cancer) = 0.25$
$p(t = 1|s) = 0.687, p(t = 1|s) = 0.174$
$sim_{final}(breast, cancer) = (0.32 * 0.687 + 0.25 * 0.174)/(0.687 + 0.174) = 0.31$

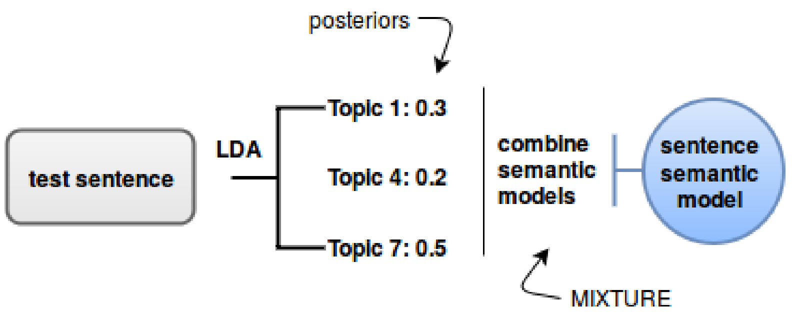


Figure 5.2: Mixture of Semantic models diagram.

In a similar way, we calculate the similarities between all the words of a sentence and the seed words that we chose.

5.3.2 Sentiment Prediction

Finally, the semantic-affective model 4.2 is used to predict the affective score for each word in the sentence. The sentence-level score is estimated using either the fusion or the classification scheme as described in Chapter 4.

More analytically, the seed selection procedure is used to extract a number of seed words that will be used to estimate the affective score of a sentence's words, using the adapted semantic similarity that was the result of the mixture.

$$v_{adapted}(w_j) = \alpha_0 + \sum_{n=1}^{N} \alpha_i \cdot v(w_i) \cdot d_{adapted}(w_i, w_j) \tag{5.2}$$

where $d_{adapted}(w_i, w_j)$ is equal to the $sim(w_j, seed_i)$ from 5.1.

Chapter 6

Experimental Procedure & Results

6.1 Datasets Desciption

Corpus

The main corpus that was used for the experiments was a generic, web-harvested corpus downloaded from the web. In order to create the corpus, firstly a vocabulary was built. The vocabulary of English packaged in the Aspell spellchecker for English, containing 135433 words was used to pose a query for each word to the Yahoo! search engine and collect snippets (short representative excerpts of the document shown under the result) of the top 500 results. Each snippet is usually composed of 2 sentences: title and content. The final corpus contains approximately 116 million sentences.

This corpus was pre-processed:

1. Tokenized: Split corpus to tokens
2. Numbers, stop-words (words like 'i', 'to', 'the' etc), punctuation removed
3. Duplicate lines removed

The final corpus (after processing) contains unique sentences. In order to create a corpus of documents, sentences of the corpus were concatenated sequentially in teams of 100. This means that the first 100 collected sentences, would be considered as the first document, the second 100 sentences would be the second document and so on. Because the corpus was constructed with multiple queries of words by collecting the first 500 results for each query, these 100 sentences will probably contain multiple examples of a word's different senses and thus can form a document about this query. In the end we have 900 thousand documents.

Words

As far as the words lexica that are used, as the affective human-annotated lexicon we use the Affective Norms for English words (ANEW) dataset (Bradley and Lang (1999)), which contains 1034 word scores in continuous dimensions of valence, arousal and dominance in [-1,1]. This dataset was created as part of a psychological experiment were the subjects (students) were asked to rate how they felt while reading each word.

Lexicon	# words	Dimension	Scores Range
ANEW	1034	Valence Arousal Dominance	[-1,1]

Table 6.1: ANEW affective lexicon.

In addition, in order to estimate the similarity between words, they exist a lot of pair-wise semantic similarity datasets. We use two of the most well-know datasets described in the table 6.2:

Dataset	Reference	Number of pairs	Scores range
WS-353	Finkelstein et al. (2001)	353	[0,10]
MEN	Bruni et al. (2014)	3000	[0,50]

Table 6.2: Word-pairs similarity datasets.

Sentences

In order to estimate the affective scores for sentences, we use two main dataset, a generic one and a Twitter one. The first is the SemEval 2007 News headlines dataset. It contains 250 sentences as training data and 1000 sentences as testing data, with continuous valence scores in [-100,100] (re-scaled in [-1,1] for out experiments). Additionally, the same dataset is annotated for the six basic emotions (anger, disgust, fear, joy, sadness, surprise) in a [0,100]. For our experiments we use only the valence scores of the dataset. The second dataset is from the SemEval 2016 Sentiment Analysis in Twitter task 4, and more specifically from the Task B: “Tweet classification according to a two-point scale”. It consisted of 3938 tweets as training set and 10551 tweets as testing set, with binary labels (1 for positive and -1 for negative).

Dataset	# Sentences	Dimension	Ratings
News Headlines SemEval-2007	250 train 1000 test	Valence Emotion	[-1,1] [0,100]
Twitter SemEval-2016 Task B	3938 train 10551 test	Binary	-1, 1

Table 6.3: Sentences datasets for polarity classification.

6.2 Topics Evaluation

In order to evaluate the topics that we constructed with the help of LDA, an analysis should take place. The following sections describe the results of each step of the algorithm before the mixture model.

LDA Training

Using the documents corpus as input we train the LDA model of the Gensim Toolbox with the following parameters:

- Number of topics from 10 to 100
- Number of iterations = 200

Firstly, we need to evaluate the quality of the topics that LDA produces. One of the outputs is the topic-word distribution for each topic. Using the top $n = 10$ most common words in each topic, as returned from the LDA $t = \{w_1, w_2, ..., w_{10}\}$, we can measure the intra-topic coherence as the average similarity between all pairs (45 pairs if we have 10 words from each topic) as proposed by Newman et al. (2010) and average over all topics $t \in T$ (equation 6.1). Word vectors that will help in the calculation of semantic similarity can be obtained by training word2vec on the whole web-harvested corpus described above (90M sentences).

$$Coh_{groupT} = \frac{1}{T}\sum_{t=1}^{T}\frac{\sum_{\substack{1\leq i\leq n\\ i+1\leq j\leq n}} sim(w_i,w_j)}{\binom{n}{2}} \tag{6.1}$$

This measure describes how coherent a topic can be considered, based on the relations between the words that characterize it. A higher average similarity score will indicate that the most topics in this group are well defined by LDA, whereas low similarity score will mean that too general topics were produced. The number of topics that give the highest average similarity can be considered as the best possible number of topics to extract from the corpus as the words that represent it are the most closely related compared to other topics.

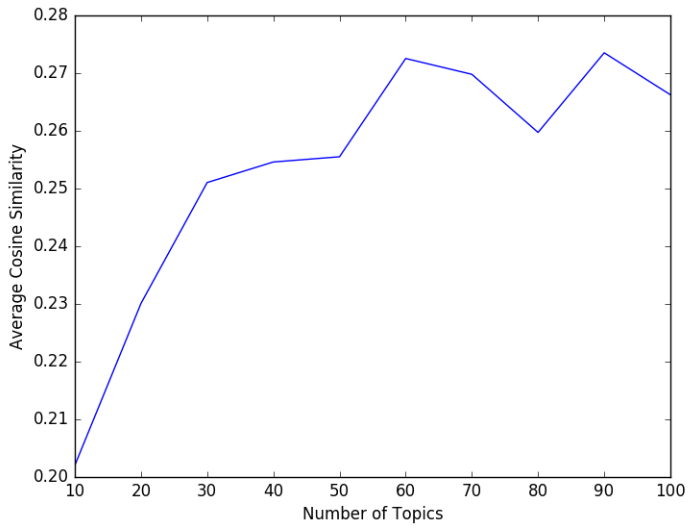


Figure 6.1: Intra-topic coherence measured after LDA training.

Observing figure 6.1, we can notice that generally 60 and 90 topics seem to be the most coherent, as according to the previous theorem, the words that characterize them have high similarity between them. Although we may expect this number of topics to achieve the best performance in the following experiments, this is something that depends on the number of data collected for each topic in the group.

Another metric that can be used is to measure the inter-topic coherence, which means the actual distance between these topics. Higher distance indicates that the topics are independent and unique while low distance shows limited power of topics distinction from

LDA. However, if one topic is a subtopic of another they will be very close and consequently their distance will be very low. The best way to actually measure the inter-topic coherence is to visualize these models. In order to to do that, we use the LDAviz toolkit [1] which is able to detect not only the marginal topic distributions but also the saliency of the vocabulary in each topic.

For reasons of completeness we can introduce the meaning of saliecy and distinctiveness (Chuang et al. (2012)). For a given word w, we need to compute its conditional probability $P(T|w)$, the likelihood that observed word w was generated by latent topic T. We also compute the marginal probability $P(T)$ as the total probability of topic T in the collection. The distinctiveness of a word is defined as:

$$Distinctiveness(w) = \sum_{T} P(T|w) log \frac{P(T|w)}{P(T)} = KL(P(T|w)||P(T)) \tag{6.2}$$

which is the Kullback-Leibler (KL) divergence (Kullback and Leibler (1951)) between, the topic distribution $P(T|w)$ given the word w, and the marginal topic distribution $P(T)$, the likelihood that any random word has been drawn from topic T. The word distinctiveness measures how informative is a word for the topic compared to a randomly selected word.

Then, we can define the word saliency:

$$Saliency(w) = P(w) \cdot Distinctiveness(w) \tag{6.3}$$

of a word w by weighting its frequency by its distinctiveness. Compared to the ranking by frequency $P(w)$, the ranking by saliency will penalize the words shared across several topics, as they will have a low distinctiveness, and boost words that are good predictors of one topic, as they will have a high distinctiveness.

An example for a possible classification is shown in figures 6.2, 6.3, along with a hyperlink for an interactive visual.

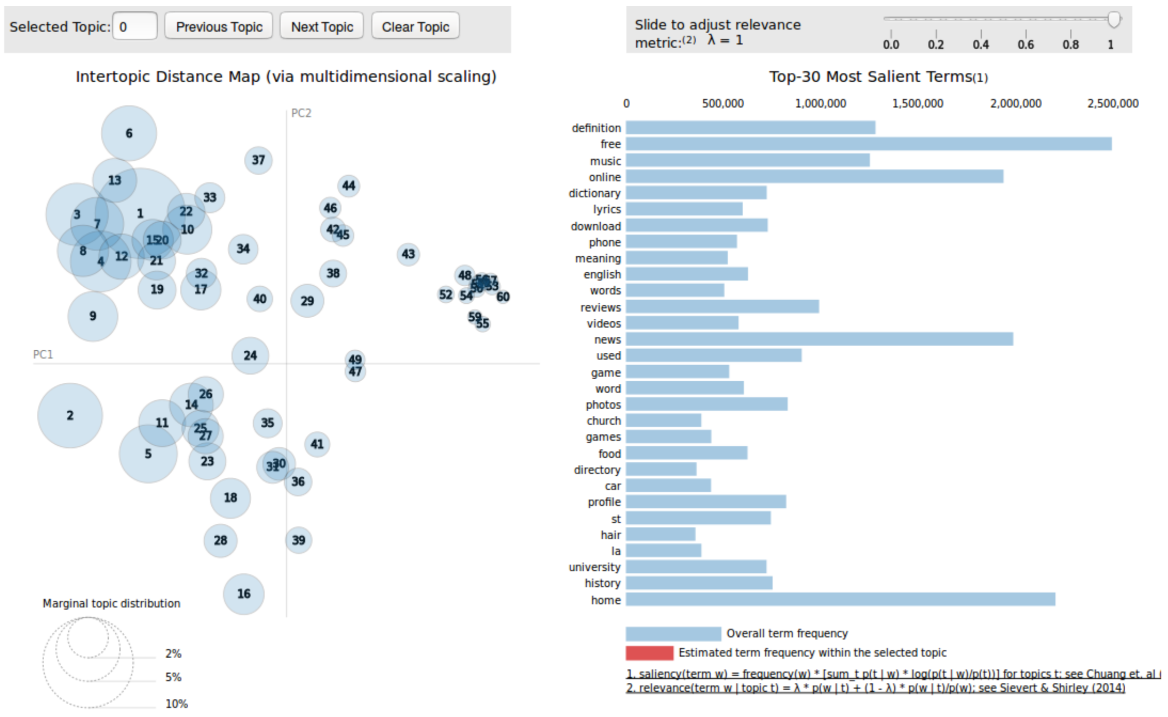
Topic-based corpora evaluation

The next step, is to construct each topic by classifying sentences from the corpus to different topic clusters. The topics can be classified using a posterior threshold or to be classified to only one topic, the topic with the maximum posterior probability. This means that a sentence is classified to more than one topics if the probability to belong in this topic is $> thres$ or it is classified only to the most probable topic. The number of sentences collected with each method per topic are shown in table 6.4 using a posterior threshold of 0.1.

As we can observe the number of sentences collected with the maximum probability is very low, especially for large number of topics. The size of the topics is an important factor for the training of the topic-DSMs. For this reason, for the following experiments we use the posterior threshold of 0.1, which enables us to perform a smooth clustering as we are able to collect approximately twice the number of sentences when using the maximum posterior.

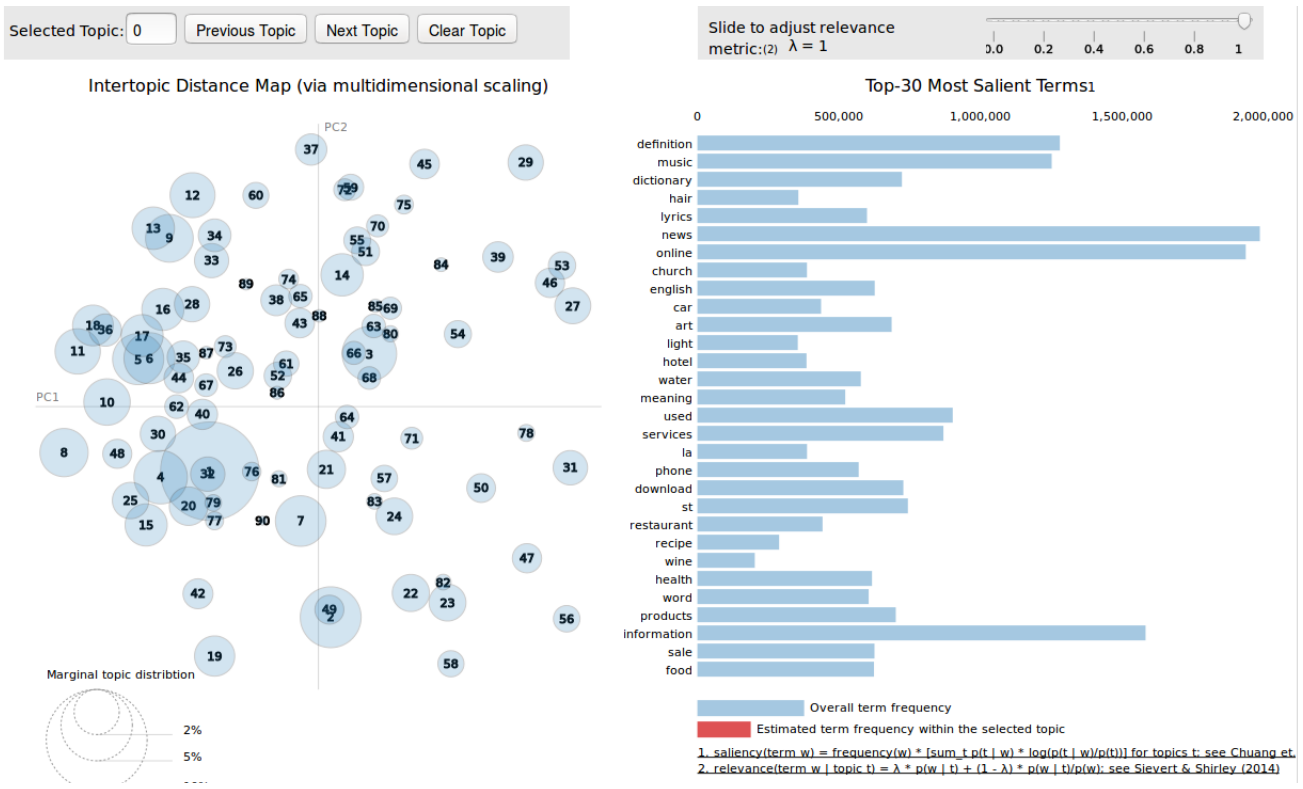
Based on the assumption that if a sentence is classified to all topics with the same probability is too ambiguous to be classified in a topic, there will be sentences that will not be classified anywhere. As we can observe from table 6.5, the number of “junk” sentences remains almost intact through all numbers of topics, showing that they are

[1] https://cran.r-project.org/web/packages/LDAvis/index.html



1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Figure 6.2: Example of visualization for 60 topics (*60 Topics Example Link*).



1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et.
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Figure 6.3: Example of visualization for 90 topics (*90 Topics Example Link*).

possibly some standard noise sentences, collected during the corpus construction process that need to be ignored.

	0.1 Posterior			Max Posterior		
Topics	Min	Max	**Mean**	Min	Max	**Mean**
10	10.8 M	20.7 M	14.6 M	5.2 M	12.9 M	8.9 M
20	4.9 M	15 M	7.7 M	2.3 M	10.1 M	4.4 M
30	2.4 M	11.9 M	5.4 M	830 K	7.4 M	2.9 M
40	2.3 M	8.4 M	4.3 M	692 K	5 M	2.2 M
50	1.2 M	7.7 M	3.6 M	338 K	4.8 M	1.7 M
60	1.2 M	9.4 M	3.1 M	292 K	5.2 M	1.4 M
70	604 K	5.3 M	2.7 M	99 K	3.3 M	1.2 M
80	297 K	6.2 M	2.4 M	44 K	3.4 M	1.1 M
90	412 K	6.4 M	2.2 M	56 K	3.6 M	987 K
100	277 K	7.4 M	2 M	42 K	4.6 M	888 K

Table 6.4: Statistics for number of sentences collected per topic using a posterior probability threshold = 0.1 and the max posterior probability.

Topics	# sentences
10	828360
20	828360
30	828360
40	828362
50	828361
60	828370
70	828382
80	828382
90	828416
100	828430

Table 6.5: Number of sentences not classified.

It is useful now to observe how many sentences are classified to one, two, three or T topics. Figure 6.4 shows different bar plots with the number of topics that a sentence can be classified along with these sentences count.

As far as the number of topics a sentence can be classified is concerned, only for 10 and approximately 20 topics most sentences can be classified to only one topic, (they give a very high posterior probability for this topic). Generally, this is the desired behavior, as it means that LDA is able to detect with great certainty the specific topic a sentence can belong. As the number of topics increases, it is logical that topics become more and more specific. In the end most sentences belong to two topics, while in the worst case a small amount can belong up to eight topics (for 100 topics). An explanation is that when a topic becomes more specific, subcategories of it are created. For example, when using only 20T it is possible to have a topic about health. When the topics become 60, there will be three topics about health, one about drugs, another about dental care and one about skin care. In this case, if the sentence is a general one that talks about cancer, LDA will classify it into all three topics with a posterior weight that is likely to be above 0.1 for all, and this is the reason that figure 6.4 illustrates the two topics as the number of topics most sentences are classified. On the other hand, if a sentence does not contain any of the topics that LDA generated is will be very difficult to classify it to only one topic.

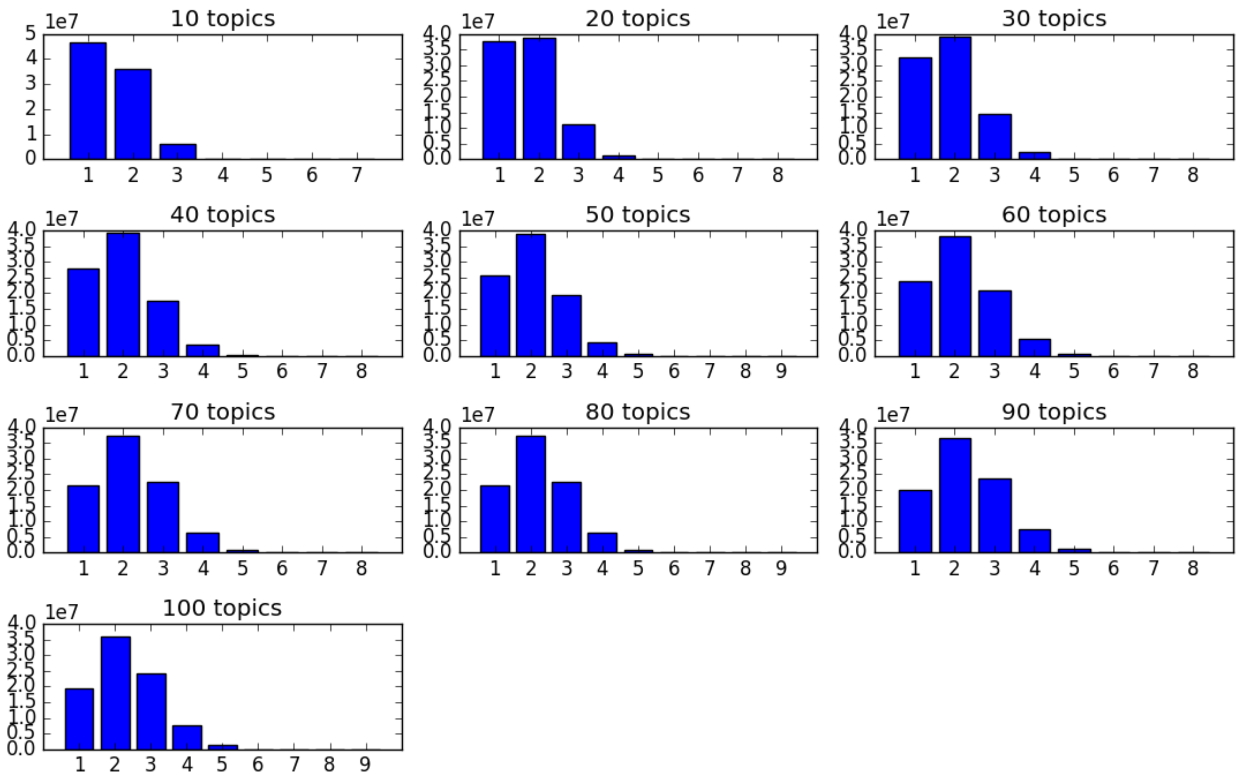


Figure 6.4: Number of sentences classified per number of topics.

Topic-Based DSM evaluation

In this step, a different semantic model is trained for each topic. In order decide which are the best parameters for training the semantic models, we conduct an experiment with different corpora sizes, vector dimensions and window sizes, using the CBOW architecture of Word2Vec and 5 iterations. In particular, we extract random sentences from the generic, web-harvested corpus and form sub-corpora. As this procedure is random, we do the same experiment 5 times and take the average Spearman Correlation score when evaluating on the MEN dataset.

By observing the results (figure 6.5), we can see that for 1M and 5M sentences the dimension of 300 and a window size of 5 give the best possible results. So these are the parameters that we will use in the topic-semantic models.

6.3 Semantic Similarity

Now we can move on to the evaluation of the mixture model, that performs the semantic space adaptation of each sentence. As we work in the space of meaning, we can also test the performance of this model on a semantic similarity task on word level, i.e. to measure the similarity between pairs of words.

In order to apply the trained LDA model to these datasets, we consider as a sentence a simple pair, $sentence = w_1 w_1$. Other possible solutions can be examined which are presented in Chapter 7.

Again, we measure the number of topics each pair is classified (figure 6.6) and it seems that most pairs are classified in one topic, while far less are classified to two topics and

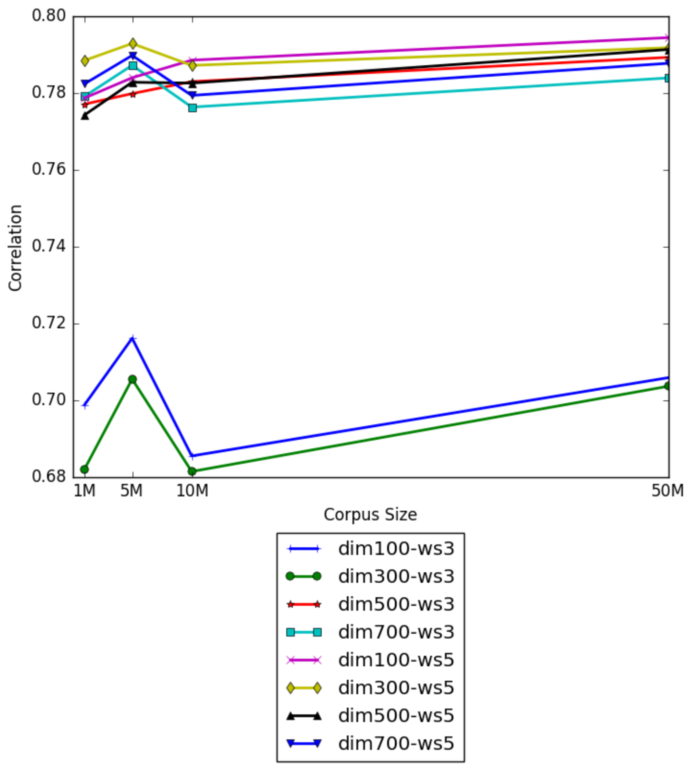
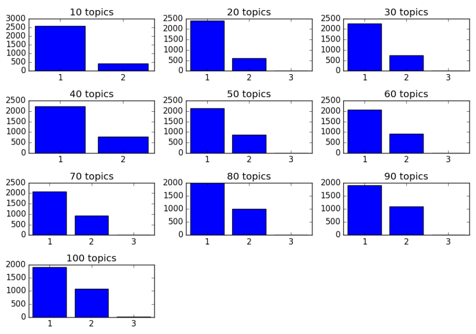
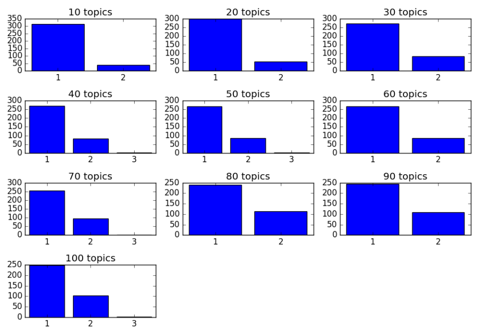


Figure 6.5: Experiment for different corpus size, vector dimension and window size for CBOW Word2Vec architecture.

three topics. This is the desired result, as it is reasonable that it is very difficult a pair of words to express more than one sense. Consequently, we can assume that words that are not so similar can be classified to more than one topic as LDA cannot estimate a specific topic for them.



(a) Number of pairs per number of topics for MEN dataset.



(b) Number of pairs per number of topics for WS-353 dataset.

Figure 6.6: Number of pairs per topic for MEN and WS datasets.

The tables 6.6, present the Spearman correlation results with human ratings, for different ranges of semantic similarity. These ranges were taken arbitrarily for evaluation purposes and can be altered (6.6).

For both datasets, the same pattern is observed. For the MEN dataset, we can notice an improvement of approximately 10% for high similarity pairs, whereas for low and median

Dataset	# High Sims (≥ 0.75)	# Median Sims ($[0.25, 0.75)$)	# Low Sims (< 0.25)
WS-353	94	227	32
MEN	595	1764	641

Table 6.6: Number of pairs for different similarity ranges for each dataset.

Topics	High Sims (> 0.75)	Median Sims	Low Sims (< 0.25)	Overall
10	0.2185	**0.6341**	0.2993	**0.7977**
20	0.2394	0.6227	0.2788	0.7972
30	0.2279	0.6109	**0.3183**	0.7881
40	**0.2448**	0.5578	0.3098	0.7502
50	0.2105	0.5598	0.2979	0.7512
60	0.2355	0.5522	0.2836	0.7492
70	0.2233	0.5312	0.2856	0.7398
80	0.2144	0.5341	0.2871	0.7337
90	0.2075	0.5388	0.2685	0.7368
100	0.2034	0.5173	0.2917	0.7151
No Topics	**0.1568**	0.5998	**0.2459**	**0.7728**

Table 6.7: Spearman correlation for MEN dataset, using different similarity ranges.

Topics	High Sims (> 0.75)	Median Sims	Low Sims (< 0.25)	Overall
10	0.5030	**0.5256**	0.0632	**0.7212**
20	0.5188	0.5021	0.2570	**0.7248**
30	**0.5536**	0.4920	0.1858	**0.7279**
40	0.4940	0.4807	0.2413	0.6673
50	0.4697	0.4460	0.0013	0.6590
60	0.4624	0.3950	0.0545	0.6332
70	0.4938	0.4612	0.3419	0.6538
80	0.4837	0.4750	0.2313	0.6913
90	0.5056	0.4603	0.2189	0.6729
100	0.4916	0.3877	0.2910	0.6428
No Topics	0.5014	0.4990	-0.0951	**0.7030**

Table 6.8: Spearman correlation for WordSim353 dataset, using different similarity ranges.

ones the improvement is of the order of 3% for median similarities and little topics and 5-7% for low similarities. This is an expected behavior. In particular, we know that two words already form a context. LDA will estimate the most probable topics (context) that a pair of words can belong. If these words are closely related (high similarity) we will be able to identify the exact topic in which they co-occur with a high probability. As a specific semantic model was trained for this topic, the similarity between these words will be very high. As a result, the final pair similarity will be better in this domain (although it may not be higher), compared to the baseline model that uses a semantic model trained on a generic corpus and is influenced by multiple other domains in the corpus. Additionally, if two words are totally dissimilar, the LDA will have trouble classifying them to a topic containing both. For this reason, the most possible scenario is to return that they belong to all topics with the same probability. In every topic, these words will have a very low similarity score, as they are dissimilar, and the mixture will assign them a value between the worst and the best case of their similarity, leading to a more precise calculation of their actual similarity. For the WS-353 dataset, the low similarities are very few and thus we cannot explain these results.

	High Sims		Median Sims		Low Sims	
Topics	# pairs	Posteriors Std	# pairs	Posteriors Std	# pairs	Posteriors Std
10	595	0.1976	1764	0.193	641	0.1751
20	595	0.1437	1764	0.1386	641	0.1212
30	595	0.1171	1764	0.1124	641	0.0984
40	42	0.0379	354	0.0287	404	0.0156
50	48	0.0487	420	0.0319	409	0.0166
60	36	0.0432	458	0.0252	435	0.0132
70	55	0.0425	454	0.0262	422	0.0116
80	70	0.0491	507	0.0291	429	0.0122
90	68	0.049	595	0.0297	436	0.0093
100	74	0.0402	574	0.0261	449	0.0111

Table 6.9: Number of pairs for MEN dataset, that are classified to more than one topic along with the standard deviation of their posterior probabilities for the topics they belong.

In order to enhance this explanation, table 6.9 shows the number of pairs that were classified to more than one topic for the different ranges of similarity. It is observed that for high similarities (595 pairs) only a few are classified to more than one topic. For the low similarities, approximately all are classified to more than one topic (as expected) and for the median the result depends on the connections between the two words and if they can belong to an existing topic. It is also important to notice that for 10, 20 and 30 topics, all pairs are classified to all topics. This is reasonable because these topics are quite generic and even terms that are highly related will obtain a probability of 0.1 for the rest of the topics except from the one they can more probably belong. This is also shown in figure 6.7 as the histograms for the MEN dataset probabilities during LDA classification. The highest posterior probability a pair can obtain is 0.67 and one can assume that the LDA model is biased. However this happens only for the pairs dataset as it will be shown thereafter.

Finally, considering the number of topics in the overall performance, the improvement is noticed for 30 and 40 topics, something that contradicts the coherence of these models, as measured in the previous section. This is something that is strongly related with the number of data collected for these topics. As illustrated in tables 6.11, 6.10 by increasing the number of iterations in the construction of the distributional semantic model seems to improve the high similarities only for 60 topics (but not the overall score) in WS-353 dataset while in MEN dataset an improvement is observed for the median similarities and consequently the overall score. At this point, it is important to underline that the use iterations in word2vec models makes it prone to overfitting. The values for 10 and 15 iterations that drop the performance indicate that overfitting makes the topic extremely specific to the data it contains and reduces it's generalization ability.

For a small number of topics (usually 10-20), the overall performance can outperform the baseline (as it increases the median similarities estimation). This can be explained if we consider the following: Each of the 10 topics, is large enough (15M sentences) so all the semantic models can be trained adequately. Additionally, the clustering of the corpus into 10 vague topics, by removing 'junk' phrases, reduces the noise in the corpus. In fact,

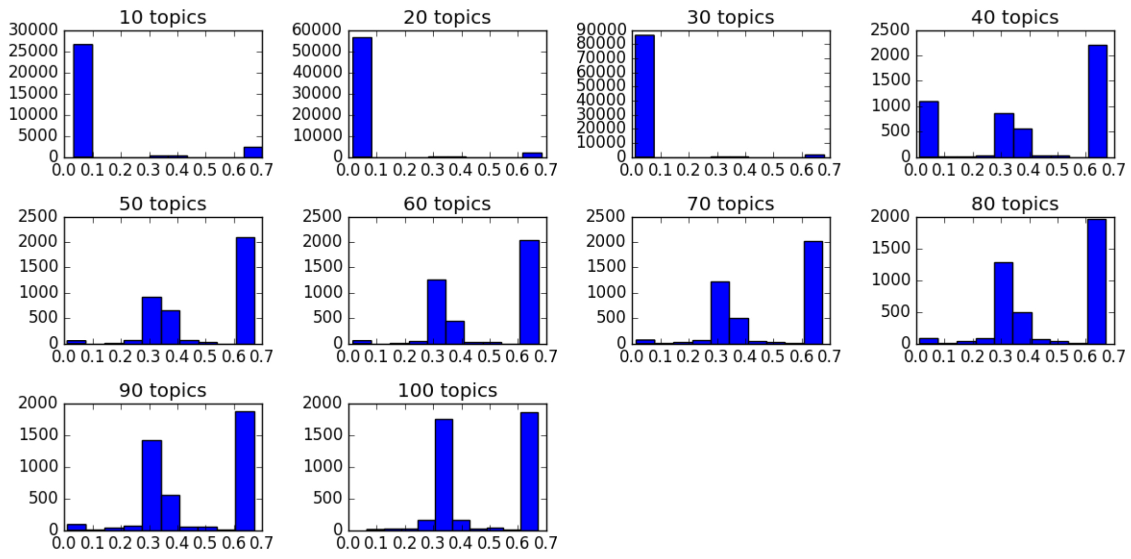


Figure 6.7: Posteriors histogram for MEN dataset.

some kind of boosting is performed. The vague topics can estimate well the similarity between all kinds of words (very similar, not so similar, dissimilar) and the combination of different similarities results a better estimation of the whole dataset.

	Iterations	High Sims	Median Sims	Low Sims	Overall
30 topics	1	0.5529	0.4430	0.1555	0.7020
	5	0.5536	0.4920	0.1858	0.7279
	10	0.5430	0.4622	0.1092	0.7087
	15	0.5439	0.4538	0.1678	0.7003
60 topics	1	0.4464	0.3437	0.0707	0.5957
	5	0.4624	0.3950	0.0545	0.6332
	10	0.4996	0.3747	0.0907	0.6158
	15	0.5019	0.3790	0.0683	0.6129
90 topics	1	0.4111	0.4297	0.1329	0.6078
	5	0.5056	0.4603	0.2189	0.6729
	10	0.4746	0.4562	0.1963	0.6468
	15	0.4722	0.4747	0.1423	0.6659

Table 6.10: Different number of iterations when trained the semantic model for 30, 60 and 90 topics for WS-353 dataset.

6.4 Sentiment Estimation

6.4.1 Baseline Training

The whole corpus (90M sentences) is trained with a CBOW model for 5 iterations, window size = 5 and vector dimension = 300. Firstly, we extract a similarity matrix for the ANEW lexicon (size 1034 × 1034). By using Word2Vec, we create feature vectors of dimension 300 for each word in our corpus. However these vectors can contain negative

	Iterations	High Sims	Median Sims	Low Sims	Overall
30 topics	1	0.1684	0.5837	0.2750	0.7596
	5	0.2279	0.6109	0.3183	0.7881
	10	0.2109	0.6036	0.2951	0.7792
	15	0.1977	0.5986	0.2991	0.7738
60 topics	1	0.1688	0.4873	0.2204	0.6678
	5	0.2355	0.5522	0.2836	0.7492
	10	0.2186	0.5602	0.2566	0.7521
	15	0.2135	0.5602	0.2335	0.7512
90 topics	1	0.1531	0.4308	0.1665	0.6305
	5	0.2075	0.5388	0.2685	0.7368
	10	0.1954	0.5457	0.2627	0.7451
	15	0.2017	0.5546	0.2660	0.7440

Table 6.11: Different number of iterations when trained the semantic model for 30, 60 and 90 topics for MEN dataset.

values and when measuring the cosine similarity between two vectors we can obtain a score in [-1,1]. Because the SAM (equation 4.2) does not work with negative similarities, we apply a type of 'rectification', by grounding (making equal to zero) all the negative similarities. Due to this action, the similarity matrix becomes quite sparse. For this reason, we need to use the regularized LSE, known as Ridge Regression in order to obtain good results for more than 100 seed words. By tuning the λ parameters on the ANEW, we obtain an optimal value of $\lambda = 0.35$ which gives a 91.17% accuracy and 0.879 correlation when applying a 10-fold cross validation on the ANEW dataset for 600 seeds (figure 6.8).

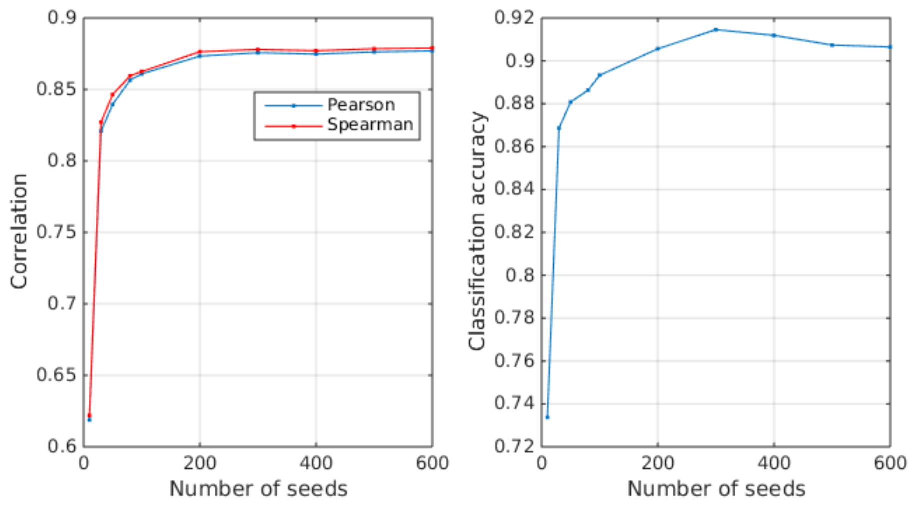


Figure 6.8: 10-fold Cross Validation on ANEW dataset using Ridge Regression and $\lambda = 0.35$.

For the SMA, the coefficients α are trained on the whole ANEW lexicon, with the λ parameter and remain the same for the baseline system (which does not incorporate topics) and the SMA, as in this model we adapt *only* the semantic space and not the mapping between the semantic and the affective space.

6.4.2 Semantic Model Adaptation

News Headlines

The posteriors histogram for the 1000 sentences is shown in figure 6.9. We can observe that for 10 and 20 topics, the majority of posteriors obtain very small values, whereas when the topics augment all possible ranges are covered. The behavior is expected as when the topics are very abstract and general, most sentences could be classified to all of them and consequently all topics obtain a common low probability (0.1) except from the most probable topic. On the other hand, when topics become more specific, LDA can distinguish them better and classify the sentences more accurately leading to larger posterior probabilities.

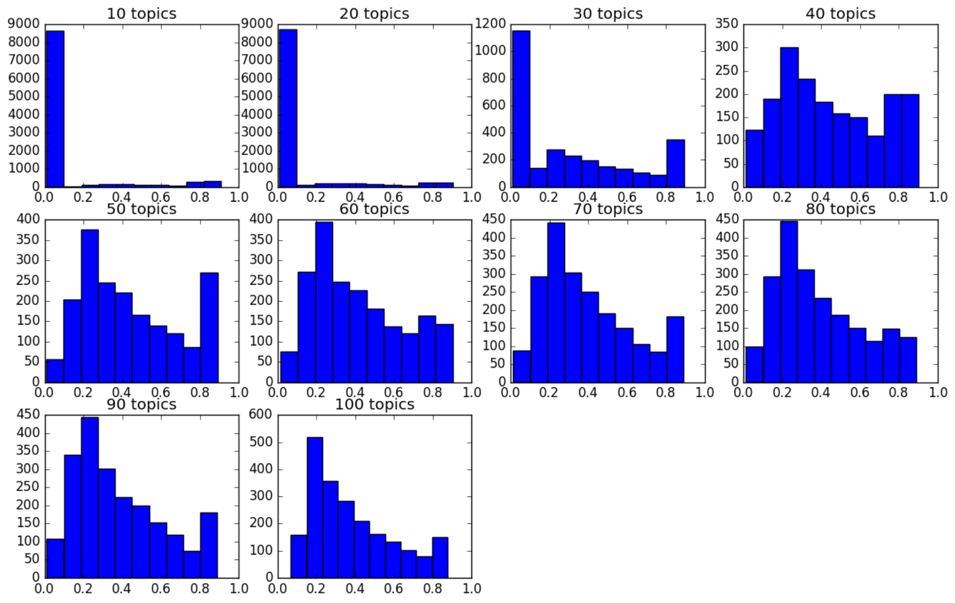


Figure 6.9: Posteriors histogram for 1000 news headlines sentences.

The results obtained with the fusion method are presented below, were we measure the classification accuracy (percentage of correctly classified samples) and the Spearman correlation between the correct (human annotated) and the estimated samples.

The results presented, show a significant improvement compared to the baseline system (no topics) that uses only one generic semantic model. It is important to notice that both accuracy and correlation take their peak values when using 80 seed words. This implies that maybe a seed selection procedure should take place in order to detect seed words that are topic-specific. In fact, when we apply the seed selection method (as described in Chapter 4) we choose words with high absolute valence scores. Generally, we can reasonably assume, that words with very high valence scores are unambiguous, which means that they cannot have more than one sense and are likely to exist in all topics (of course there are exceptions like ‘cancer’: -0.8 valence scores which can be the disease or the horoscope sign). By observing these results it is likely that 80 seed words is the best number of unambiguous collected words that can be used to estimate a unknown word. This resembles the work of Turney (2002) which used the words “excellent” and “poor” as

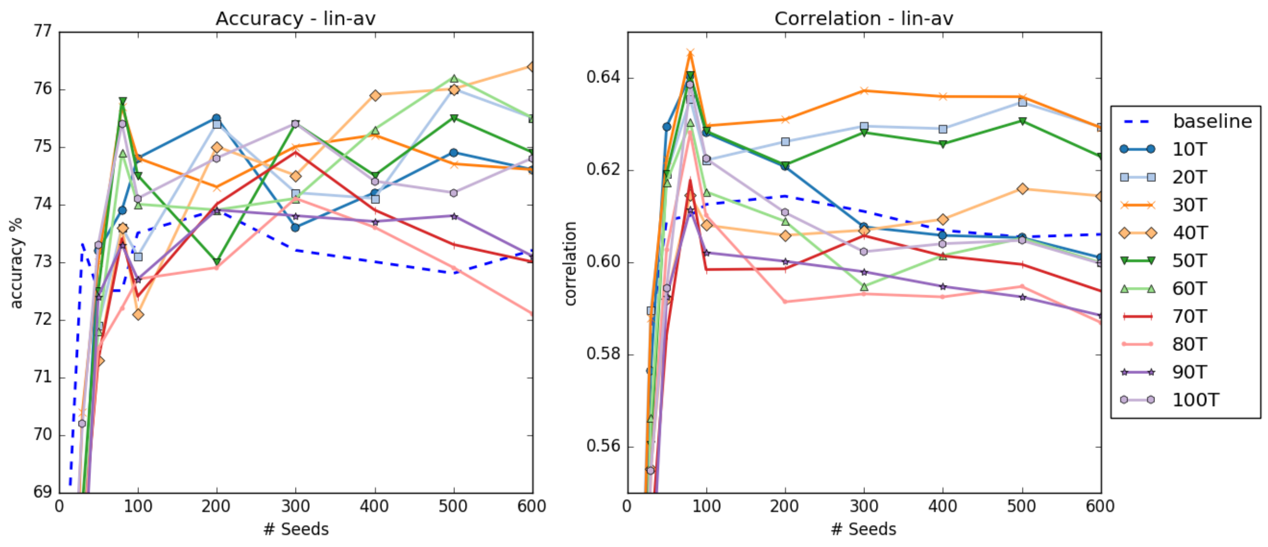


Figure 6.10: Accuracy and Spearman Correlation using linear fusion.

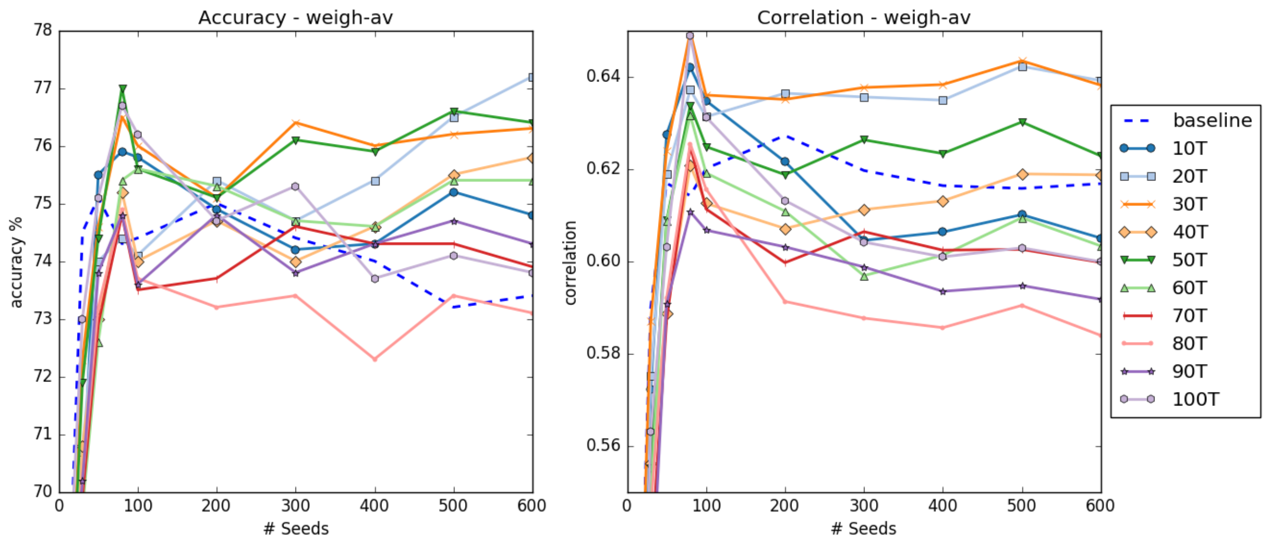


Figure 6.11: Accuracy and Spearman Correlation using weighted fusion.

reference in order to estimate the sentiment of a review. In summary, a better approach would include as seed words topic-specific words that are able to characterize a new word in a specific domain. The seed words histograms are shown in figure 6.13.

Moreover, again the 30 topics seem to perform well for all fusion schemes, while the best score is obtained for 20 topics and 600 seed words. The weighted fusion scheme generally gives the best results as it weights more the "important" words in a corpus, those that have a higher absolute valence score. As a result it ignores words with low scores (like stop-words or names) and focuses on those that give more affective information.

Concerning the correlation, the max fusion scheme gives the worst scores as News headlines are general purpose sentences that contain striking words with high valence score but they do not always represent the actual meaning of the sentence. Again, the reason that small topics seem to perform well can be a) that this is the optimal distinct number of topics our corpus may contain and b) the noise reduction that is performed.

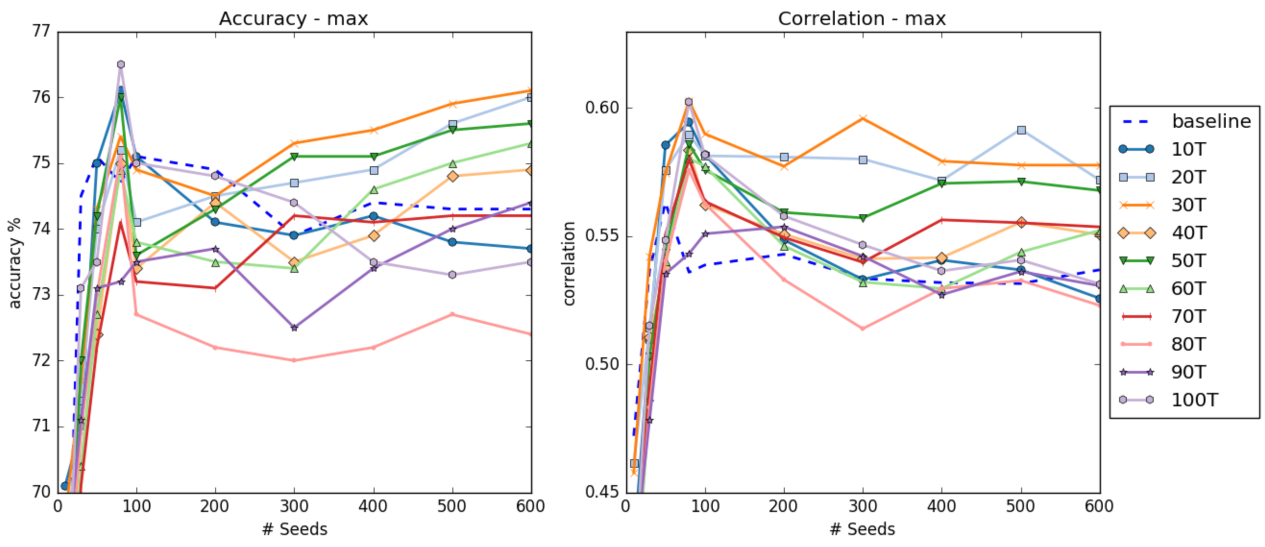


Figure 6.12: Accuracy and Spearman Correlation using non-linear max fusion.

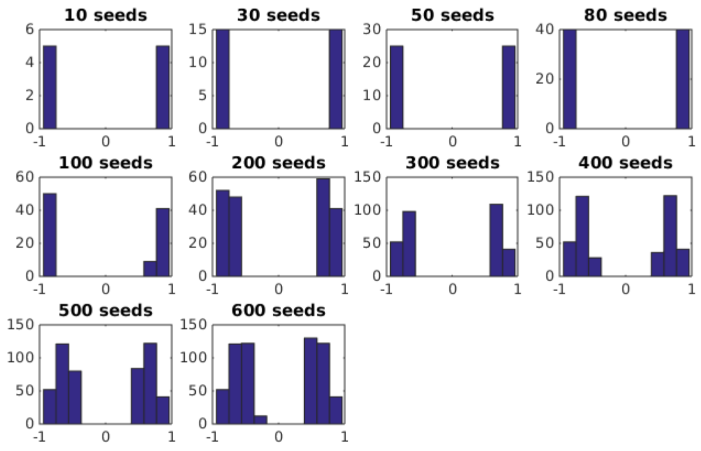


Figure 6.13: Histograms for seed words valence scores obtain from ANEW.

Table 6.12 summarizes the best results presented in the figures.

The results using the classification method are presented in figures 6.14, 6.15. The classifier that was used was a Linear Logistic Regression and a Naive Bayes classifier from the Weka Toolbox. The features extracted from each sentence's scores are the following: min, max, mean, std, var, length, extremum (larger absolute value in the vector), sum, range and normalized versions of these features according to the length of the sentence.

Based on these results, again we can notice that 20 topics seem to give the best performance along with 40 topics. However, if we train the model using a Naive Bayed classifier, we see that 50 topics perform best and the actual improvement is less than in the fusion scheme. Possibly, the classifier is unable to detect the adaptation scheme as it. Again in both classifiers, we observe the pick in 80 seed words.

		Linear Fusion		Weighted Fusion		Max Fusion	
Topics	Seeds	CA (%)	CC	CA (%)	CC	CA (%)	CC
10	80	74.8	0.6370	76.1	0.5945	76.1	0.5631
20	600	75.2	0.6257	**77.2**	0.6391	76	0.5719
30	80	75.7	**0.6455**	76.5	**0.6496**	75.4	0.6028
40	600	76.4	0.6143	75.9	0.6173	74.9	0.5506
50	80	**75.8**	0.6406	**77**	0.6337	76	0.5859
60	500	76.2	0.6051	75.5	0.6078	75	0.5438
100	80	75.6	0.6366	76.7	0.6489	76.5	0.6024
No Topics	200	73.9	0.6142	75.1	0.6272	74.9	0.5429

Table 6.12: Results summary (CA: Classification Accuracy, CC: Classification Correlation (Spearman)) for sentence polarity detection using SMA.

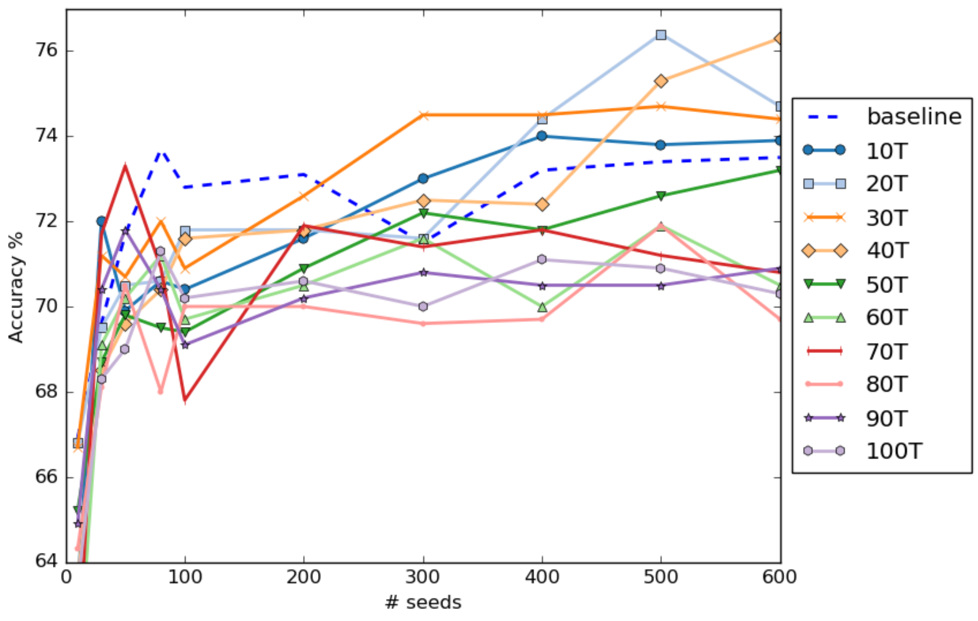


Figure 6.14: Classification accuracy using Linear Logistic Regression classifier.

Twitter

Additionally, we apply this procedure on the Twitter dataset, and evaluate using the same features as before with a Naive Bayes Tree classifier (which gives the best performance). We measure the average precision, recall, F1 score along with the classification accuracy.

$$recall = \frac{TP}{TP+FN}, precision = \frac{TP}{TP+FP}, F1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{6.4}$$

The results obtained for the twitter dataset using 600 seeds are summarized in the table 6.13 along with the results of the Tweester system (Palogiannidi et al. (2016))that was submitted in the task, raking in the first place.

As we can observe the classification accuracy is generally high and for 20 Topics it surpasses the baseline (No Topics). In fact, the dataset is biased towards positive tweets as shown in the table 6.14.

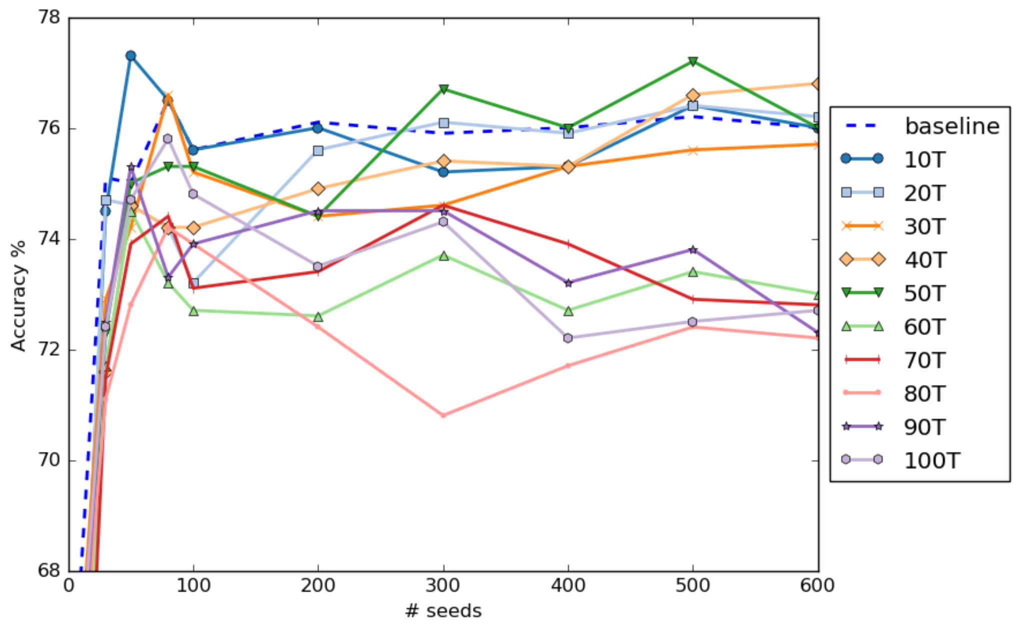


Figure 6.15: Classification accuracy using Naive Bayes classifier.

Topics	Accuracy (%)	Av. Recall	Av. Precision	Av. F1 score
10	82.98	0.672	0.7795	0.70005
20	**84.72**	0.734	0.791	**0.755**
30	83.41	0.673	**0.7955**	0.703
40	82.85	0.689	0.7645	0.713
50	83.42	0.678	0.79	0.708
60	82.75	0.6625	0.7795	0.6905
70	83.94	0.72	0.7785	0.742
80	82.51	0.71	0.7495	0.726
90	83.07	0.6635	0.792	0.693
100	76.44	**0.754**	0.6955	0.708
No Topics	83.27	0.6925	0.7745	0.7185
Tweester	0.862	0.797	-	0.799

Table 6.13: Classification Results for SemEval-2016 Twitter Task B dataset.

	# Total Tweets	# Positive	# Negative
Train	3938	3260	678
Test	10551	8212	2339

Table 6.14: Number of positive and negative tweets in the SemEval-2016 Task 4, Subtask B dataset.

Considering this, the model performs poorly in the detection of negative sentences and thus resulting low average recall and average F1 score values.

Generally, the Twitter datasets are considered the most difficult to classify, due to their informal language, unusual expressions and diverse vocabulary. Consequently, when training an LDA model to a generic corpus that contains “normal” sentences and not

Tweets, a large amount of information is lost, as in the evaluation process we do not consider punctuation, emoticons, elongated words etc, features that play an important role in the social media datasets. Additionally, the corpus and the twitter dataset were created in totally different time periods. For example the movie "Ant-Man" is not contained as a movie name in our corpus. This also makes the classification of the tweets dataset more difficult.

However, the performance of the model is very encouraging despite all these obstacles described above, showing that it can have potential when using the appropriate data for training the LDA model.

Chapter 7

Conclusion

7.1 Conclusions

This thesis discussed the sentiment analysis of sentences in a binary classification problem using continuous Affective Spaces, Distributional Semantic Models and Topic Modeling techniques. We proposed a model that aims to improve the classification accuracy on sentence-level polarity prediction and the correlation of word-pairs similarity with human annotations. The main mechanisms that are incorporated in this system, are a Topic Modeling approach, that can help in the distinction of specific domains or 'themes' included in a large collection of documents, the Distributional Semantic Models, that try to represent the meaning of words in the form of mathematical vectors, by exploiting the properties of language, and an affective model based on the assumption that the "semantic similarity implies affective similarity", which functions as a bridge between the space of meaning and the space of sentiment.

In more detail, in this work, a novel adaptation approach was introduced which incorporates topic knowledge with purpose to adapt the semantic space for each individual sentence. The model is based on the idea that in order to estimate the affect or the meaning of a sentence, we first need to identify the context (topic) where it belongs. In this way we will be able to overcome the problem of word disambiguation when estimating the affective score of words, as a word can express a different meaning and consequently sentiment, depending on its use in a larger lexical unit.

The basic steps of the algorithm are, a) topics extraction from a web-harvested corpus using the LDA technique, b) topic-clusters creation by classifying the corpus sentences into the extracted topics using the trained LDA model and c) a different semantic model training on each topic sub-corpus. In the final step, the semantic space of a sentence is adapted, i.e. the similarities between the sentence's words and an existing annotated sentiment lexicon. The final similarity matrix for a sentence is produced using a weighted mixture of the semantic models of the topics it can be classified, which is decided by applying a trained LDA model to that sentence.

The results obtained from testing on pair-wise semantic similarity between words, reveal the robustness of the algorithm especially when measuring the similarity of very related pairs. We showed that a well-trained LDA model can identify with low uncertainty even the relevant topics for pairs of words. Additionally, it is concluded that the overall performance can be improved when the topics contain enough data for the good training of the semantic model, although in general the performance of each individual model depends on the models parameters. When using predictive models like Word2Vec, the number of iterations is the main factor that leads to more or less adaptation, but it also introduces a form of overfitting that can lead to poor performance.

The performance of the semantic space adaptation mixture model on sentence sentiment classification tasks prove that this procedure of two-step classification, i.e. identifying the topic of interest and then measuring the meaning of a sentence can improve not only the polarity recognition but also estimate more accurately the actual valence score assigned to the sentence. Although a great improvement is achieved when evaluating on general datasets (e.g. News headlines), social media messages (e.g. Twitter) is a much more difficult problem and a special training on specific data is needed in order to obtain better results. However, this generic model shows a decent behavior in this task despite the loss of important information and the ignorance of stylistic features.

In summary, topic models combined with affective text analysis methods look very promising in the sentiment analysis of sentences, as they offer a completely unsupervised way of adapting a semantic model to multiple tasks.

7.2 On-going Work

7.2.1 Semantic-Affective Model Adaptation (SAMA)

An improvement of the algorithm (SMA) is proposed here, that could lead to better affective results. Although there are no evaluation results yet for this algorithm, its motivation and steps are described as part of on-going work.

Motivation

In this approach, we try to adapt both the affective space and the semantic space. This means that the affective scores are now recalculated additionally to the semantic similarities as in the SMA. The motivation behind this idea is that the previous algorithm did not incorporate any knowledge about affect when using topics and thus could lead to better semantic similarity performance than affective. As shown in the literature, models that capture sentiment and topic at the same time lead to better results. So the main idea, is not only to extract different seed words for a topic, but also to adapt the affective scores of these seed words according to the topic. Then, we can use them in the Semantic-Affective model (equation 4.2), to estimate multiple topic-affective scores for the words of a sentence. The final word score is produced using a mixture model of topic-affective scores. Again the sentence's score estimation is based on a fusion or classification scheme.

Algorithm

Generally, the topic Modeling procedure up to the step that a different semantic model is built on each topic-cluster is the same as in the Semantic Model Adaptation (SMA). The principle change is the affective scores adaptation and the mixture model.

The steps of the algorithm are described below:

1. For each topic:
   (a) Select Topic-Seeds according to two criteria
      - High frequency in topic
      - Balanced set (sum of valence score approximately zero)

   (b) Create a similarity matrix of size $N \times N$ that contains the similarity pairs between the topic seeds

2. While iter $< k$:

   (a) Train α coefficients from semantic-affective model (equation 4.2) for the selected topic-seeds

   (b) Recalculate valence scores for topic-seeds using the semantic-affective model:

$$v_{topic-adapted}(seed_j) = \alpha_{0,topic} + \sum_{i=1}^{N} \alpha_{i,topic} v_{anew}(w_i) sim_{topic}(seed_j, w_i)$$

3. Apply LDA to each sentence and check the possible topics

4. For each word in the sentence ($w_{sentence}$):

   - Use the semantic similarity between this word and a topic-seed $d(w_{sentence}, w_{topic-seed})$ from a generic DSM (trained on a large corpus)
   - Estimate the word's valence score with SAM for each topic:

$$v_{topic}(w_{sentence}) = a_{0,topic} + \sum_{i=1}^{N} a_{i,topic} v_{topic-adapted}(w_{topic-seed}) d(w_{sentence}, w_{topic-seed})$$

5. Fuse the different valence scores for this word, using topic-posteriors to obtain the final valence score of the word

$$v_{final}(w_{sentence}) = \frac{\sum_{t=1}^{T} p(t_i|sentence) \cdot v_t(w_{sentence})}{\sum_{t=1}^{T} p(t_i|sentence)}$$

7.3 Directions for Future Work

The model proposed in this thesis, contains a plethora of different parameters that must be tuned in order to improve the general performance of the model. In all the experiments described in Chapter 6, standard implementations for Topic Modeling and Semantic Models were used due to the huge amount of data produced by these models, for all possible topics.

Some possible ideas for future work on this domain are described below.

Topic Modeling

- The topic modeling method (LDA) can be easily altered, in order to be applied directly to sentences and not documents. Different implementations can be found in the web according to the data one needs to analyze ([1], [2]).

- The simple topic construction method that we used can be improved. An example is to classify each sentence to all topics it may belong and keep the information of the posterior probability for use in the semantic model. This is a possible "weighted" soft clustering. Alternatively, no topic modeling can be used, and other classification methods (such as K-Means) can take its place.

[1] https://github.com/datquocnguyen/LFTM/
[2] https://github.com/minghui/Twitter-LDA

Semantic Models

- Other models can replace Word2Vec, such as classic context-based approaches using PPMI features or the Glove package [3].
- For all topics, the parameters of the semantic model were fixed. A possible extension would be to use different parameters for each topic (although it may be difficult for large number of topics). Performance may increase by tuning the following important parameters of Word2Vec are:
  - vector dimension
  - window size
  - number of iterations
  - algorithm (CBOW, SkipGram)
- Additionally, as far as the normalization of feature vectors extracted by the semantic model is concerned, except from the norm-1 normalization (unit vector) that we used, z-score or min-max may improve performance.

Semantic Model Adaptation

Concerning the first model, that tries to adapt the semantic space of each sentence as a mixture of topic-semantic spaces the following can be examined:

- A seed selection approach as it is observed that the model performs best when the number of seed words is low.
- A different mixture model, that can weight more the high similarities between pairs.
- Another idea is to only adapt the feature vectors of topic words. This means the all other words vectors (seeds or not) that are 'generic' can be trained using a general corpus and not a topic-specific corpus.
- An idea concerning the task of semantic similarity between words is to give sentences as input to the LDA and not just two words. This can be achieved with the help of WordNet. For each word of interest, one can obtain the definition of the multiple senses of this word. All possible combinations of definitions can be created between two terms. The final similarity will be the maximum among all results according to the 'maximum sense hypothesis': "The similarity of two words is the maximum similarity among their senses".

[3] `http://nlp.stanford.edu/projects/glove/`

Appendix A

Gensim Toolbox

The Gensim Toolbox [1] created by Řehůřek and Sojka (2010) is a collection of scripts written in Python. It's name comes from the "Generate Similar" as it was firstly used to collect the most similar articles for a given article. In this thesis, we used both the Topic Modeling tools, as well as the implementation of Word2Vec that offers in python. These two tools are described below, with examples from the page's tutorials.

A.1 Topic Modeling

In order to use topic modeling in gensim, we firstly need to transform the input collection of documents. The first step is the construction of a dictionary:

```python
import gensim, logging
from gensim import corpora, models, similarities

logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=logging.INFO)

# load a list with known stopwords
stoplist = [i.strip() for i in open('my_en_stopwords.txt', 'r')]

# collect statistics about all tokens (reads file line by line)
dictionary = corpora.Dictionary((line.lower().split() for line in open('mycorpus.txt','r')), prune_at=None)

# remove stop words and words that appear only once
stop_ids = [dictionary.token2id[stopword] for stopword in stoplist if stopword in dictionary.token2id]

# this method, removes stopwords from dictionary
dictionary.filter_tokens(bad_ids=stop_ids)

# Remove too infrequent and frequent words
dictionary.filter_extremes(no_below=3, no_above=0.8)

# remove gaps in id sequence after words that were removed
dictionary.compactify()

# store the dictionary, for future reference
dictionary.save('corpus.dict')
dictionary.save_as_text('corpus.dict_txt',sort_by_word=True)
```

[1] https://radimrehurek.com/gensim/

If we try to print the dictionary we can see:

```
print(dictionary)
print(dictionary.token2id)
```

```
Dictionary(72 unique tokens: [u'heavy', u'amish', u'says', u'deal', u'porn']...)
{u'heavy': 0, u'amish': 1, u'says': 2, u'deal': 3, u'porn': 4, u'insists': 60, u'lebanon': 6, u'dead': 7, u'two': 26, u'b
rain': 9, u'say': 10, u'set': 11, u'kill': 12, u'ex': 13, u'satellites': 71, u'madonna': 14, u'ship': 15, u'linked': 16,
u'marathon': 17, u'yankee': 18, u'brazil': 19, u'plot': 20, u'police': 21, u'cancer': 22, u'record': 64, u'tv': 23, u'pho
tographer': 24, u'vegetables': 25, u'three': 8, u'tropical': 27, u'iraqi': 28, u'attack': 29, u'adoption': 30, u'storm':
31, u'scientists': 32, u'terror': 33, u'women': 65, u'happy': 36, u'viking': 37, u'life': 38, u'good': 39, u'crash': 40,
u'sinks': 41, u'missing': 42, u'confusion': 43, u'troops': 55, u'gaza': 35, u'internet': 45, u'admits': 46, u'dies': 47,
u'china': 48, u'child': 49, u'russian': 50, u'drinking': 51, u'iraq': 52, u'killed': 53, u'vaccine': 54, u'man': 34, u'ba
ghdad': 61, u'mars': 56, u'prize': 57, u'wins': 58, u'study': 59, u'flu': 5, u"men's": 44, u"alzheimer's": 62, u'victory'
: 66, u'uk': 67, u'lose': 68, u'report': 69, u'found': 70, u'finds': 63}
```

For a new sentence, the transformation to BOW using the vocabulary is shown below:

```
new_doc = "life on mars"
new_vec = dictionary.doc2bow(new_doc.lower().split()) # 'on' is a stoword and so removed
print(new_vec)
```

```
[(38, 1), (56, 1)]
```

Gensim, is able to handle large collections, by loading only one line at a time, avoiding the large memory usage. This can be achieved with the use of an iterator, that converts each line into BOW format, using the dictionary created above. We can call this iterator, and save our corpus in Matrix Market format (other formats are used as well).

```
class MyCorpus(object):
    def __iter__(self):
        for line in open('mycorpus.txt'):
            # assume there's one document per line, tokens separated by whitespace
            yield dictionary.doc2bow(line.lower().split())

corpus = MyCorpus()
print(corpus)

corpora.MmCorpus.serialize('corpus.mm', corpus)
```

```
<__main__.MyCorpus object at 0x7f71a7f8f7d0>
```

```
print list(corpus)[25] # print line 26 of the corpus
```

```
[(8, 1), (15, 1), (42, 1), (50, 1), (69, 1), (70, 1)]
```

The next step is to load the corpus and the dictionary and apply a transformation. In our case we used the LDA transformation and more specifically the LDAMulticore implementation that is able to run in parallel:

```
# load dictionary and corpus
dictionary = corpora.Dictionary.load('corpus.dict')
corpus = corpora.MmCorpus('corpus.mm')

# apply LDA transformation to the BOW corpus
model = gensim.models.ldamulticore.LdaMulticore(corpus, num_topics=2, id2word=dictionary, workers=2, iterations=100)

# once the model is ready, save it
model.save('corpus_2T.lda')
```

Finally, after the model is trained, for each new sentence we can check where the trained model classifies them as follows:

In general, the parameters of LDA are the following.

Parameters

num_topics: the number of topics to be created
id2word: the dictionary that will be used for the topic construction

```
# load model
model = models.LdaModel.load('corpus_2T.lda',mmap='r')

# transform new sentence
sentence = "life on mars"
transf_sent = dictionary.doc2bow(sentence.lower().split())
print transf_sent
```

```
[(38, 1), (56, 1)]
```

```
lda_sent = model[transf_sent]
print lda_sent
```

```
[(0, 0.19657055121821196), (1, 0.80342944878178812)]
```

aplha, eta: hyperparameters of document-topic and topic-word distributions. The default values are 1/number of topics.
minimum_probability: Topics that give posterior probability below this number are ignored (default is 0.001).
iterations: number of passes for each chunk of documents

A.2 Word2Vec

In the same way, word2vec can load one line at a time and then appying the word2vec algorithm. There are many parameters that can be tuned, as they are described below.

```
from gensim import corpora, models, similarities, logging
from gensim.models.word2vec import Word2Vec

logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=logging.INFO)

class MySentences(object):
    def __iter__(self):
        for line in open('mycorpus.txt'):
            yield line.lower().split()

sentences = MySentences() # a memory-friendly iterator

model = Word2Vec(sentences, size=300, window=5, workers=7, sg=0, hs=0, negative=5,
                 cbow_mean=1, iter=5, min_count=5, max_vocab_size=1000000)
model.init_sims(replace=False) # cannot be trained after this temp!

model.save('corpus_w2v.model')
```

Parameters

size: Dimension of the feature vector
window: Context window around the term of interest (w words left, w words right)
workers: number of threads to be used
sg: if 1 skip-gram algorithm is used, if 0 cbow algorithm is used
hs: if 1 hierarchical softmax is used as inference method, if 0 and negative is non-zero, negative sampling is used
negative: if > 0, negative sampling is used and the number is equal to the "noise words" that should be drawn (usually between 5-20)
cbow_mean: if 0, the sum of the context word vectors will be used, if 1 the mean will be used
iter: the number of epochs over the corpus
min_count: minimum frequency of words. Words with frequency below this value will be ignored.
max_vocab_size: maximum vocabulary size created during training. The infrequent words are cropped.

Bibliography

David J Aldous. *Exchangeability and related topics.* Springer, 1985.

Alina Andreevskaia and Sabine Bergler. Mining wordnet for a fuzzy sentiment: Sentiment tag extraction from wordnet glosses. In *EACL*, volume 6, pages 209–216, 2006.

Marco Baroni and Alessandro Lenci. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721, 2010.

Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer, 2006.

David M Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

David M Blei and John D Lafferty. Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning*, pages 113–120. ACM, 2006.

David M Blei and John D Lafferty. A correlated topic model of science. *The Annals of Applied Statistics*, pages 17–35, 2007.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

Margaret M Bradley and Peter J Lang. Affective norms for english words (anew): Instruction manual and affective ratings. Technical report, Technical report C-1, the center for research in psychophysiology, University of Florida, 1999.

Elia Bruni, Nam-Khanh Tran, and Marco Baroni. Multimodal distributional semantics. *J. Artif. Intell. Res.(JAIR)*, 49(1-47), 2014.

Jonathan Chang and David M Blei. Hierarchical relational models for document networks. *The Annals of Applied Statistics*, pages 124–150, 2010.

Pimwadee Chaovalit and Lina Zhou. Movie review mining: A comparison between supervised and unsupervised classification approaches. In *System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on*, pages 112c–112c. IEEE, 2005.

Jason Chuang, Christopher D Manning, and Jeffrey Heer. Termite: Visualization techniques for assessing textual topic models. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pages 74–77. ACM, 2012.

Stephen Clark. Vector space models of lexical meaning. *Handbook of Contemporary Semantics, Wiley-Blackwell, à paraître*, 2012.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391, 1990.

Gabriel Doyle and Charles Elkan. Accounting for burstiness in topic models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 281–288. ACM, 2009.

Koji Eguchi and Victor Lavrenko. Sentiment retrieval using generative models. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 345–354. Association for Computational Linguistics, 2006.

Andrea Esuli and Fabrizio Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, volume 6, pages 417–422. Citeseer, 2006.

Christiane Fellbaum. *WordNet*. Wiley Online Library, 1998.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414. ACM, 2001.

Andrew B Goldberg and Xiaojin Zhu. Seeing stars when there aren't many stars: graph-based semi-supervised learning for sentiment categorization. In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing*, pages 45–52. Association for Computational Linguistics, 2006.

Thomas L Griffiths, Mark Steyvers, David M Blei, and Joshua B Tenenbaum. Integrating topics and syntax. In *NIPS*, volume 4, pages 537–544, 2004.

Zellig S Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

Elias Iosif and Alexandros Potamianos. Unsupervised semantic similarity computation between terms using web documents. *IEEE Transactions on Knowledge and Data Engineering*, 22(11):1637–1647, 2010.

Valentin Jijkoun, Maarten de Rijke, and Wouter Weerkamp. Generating focused topic-specific sentiment lexicons. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 585–594. Association for Computational Linguistics, 2010.

Yohan Jo and Alice H Oh. Aspect and sentiment unification model for online review analysis. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 815–824. ACM, 2011.

Jaap Kamps, MJ Marx, Robert J Mokken, M de Rijke, et al. Using wordnet to measure semantic orientations of adjectives. 2004.

Soo-Min Kim and Eduard Hovy. Determining the sentiment of opinions. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1367. Association for Computational Linguistics, 2004.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

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.

Chenghua Lin and Yulan He. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 375–384. ACM, 2009.

Yuming Lin, Jingwei Zhang, Xiaoling Wang, and Aoying Zhou. Sentiment classification via integrating multiple feature presentations. In *Proceedings of the 21st International Conference on World Wide Web*, pages 569–570. ACM, 2012.

Nikolaos Malandrakis, Alexandros Potamianos, Elias Iosif, and Shrikanth Narayanan. Distributional semantic models for affective text analysis. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(11):2379–2392, 2013.

Nikolaos Malandrakis, Alexandros Potamianos, Kean J Hsu, Kalina N Babeva, Michelle C Feng, Gerald C Davison, and Shrikanth Narayanan. Affective language model adaptation via corpus selection. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4838–4842. IEEE, 2014.

Nikos Malandrakis, Alexandros Potamianos, Elias Iosif, and Shrikanth Narayanan. Emotiword: Affective lexicon creation with application to interaction and multimedia data. In *Computational Intelligence for Multimedia Understanding*, pages 30–41. Springer, 2011a.

Nikos Malandrakis, Alexandros Potamianos, Elias Iosif, and Shrikanth S Narayanan. Kernel models for affective lexicon creation. In *In Proceedings of INTERSPEECH*, pages 2977–2980, 2011b.

Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. Topic sentiment mixture: modeling facets and opinions in weblogs. In *Proceedings of the 16th international conference on World Wide Web*, pages 171–180. ACM, 2007.

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

George A Miller and Walter G Charles. Contextual correlates of semantic similarity. *Language and cognitive processes*, 6(1):1–28, 1991.

Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in Neural Information Processing Systems*, pages 2265–2273, 2013.

David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. Automatic evaluation of topic coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 100–108. Association for Computational Linguistics, 2010.

F. Å. Nielsen. Afinn, 2011.

Alexander Osherenko and Elisabeth André. Lexical affect sensing: Are affect dictionaries necessary to analyze affect? In *Affective Computing and Intelligent Interaction*, pages 230–241. Springer, 2007.

Elisavet Palogiannidi, Athanasia Kolovou, Fenia Christopoulou, Filippos Kokkinos, Elias Iosif, Nikolaos Malandrakis, Harris Papageorgiou, Shrikanth Narayanan, and Alexandros Potamianos. Tweester at semeval-2016 task 4: Sentiment analysis in twitter using semantic-affective model adaptation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval 2016), San Diego, US*, 2016.

Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics, 2004.

Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 115–124. Association for Computational Linguistics, 2005.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

Christos H Papadimitriou, Hisao Tamaki, Prabhakar Raghavan, and Santosh Vempala. Latent semantic indexing: A probabilistic analysis. In *Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 159–168. ACM, 1998.

Francis Jeffry Pelletier. The principle of semantic compositionality. *Topoi*, 13(1):11–24, 1994.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–43, 2014.

Robert Plutchik. The nature of emotions human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice. *American Scientist*, 89(4):344–350, 2001.

Yanghui Rao, Qing Li, Xudong Mao, and Liu Wenyin. Sentiment topic models for social emotion mining. *Information Sciences*, 266:90–100, 2014.

Jonathon Read and John Carroll. Weakly supervised techniques for domain-independent sentiment classification. In *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*, pages 45–52. ACM, 2009.

Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50. ELRA, 2010.

Joseph Reisinger and Raymond Mooney. A mixture model with sharing for lexical semantics. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182. Association for Computational Linguistics, 2010.

Xin Rong. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*, 2014.

Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. The author-topic model for authors and documents. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 487–494. AUAI Press, 2004.

James A Russell. A circumplex model of affect. *Journal of Personality and Social Psychology*, 39(6):1161–1178, 1980.

Magnus Sahlgren. *The Word-space model.* PhD thesis, Citeseer, 2006.

Hinrich Schiitze. Word space. *Advances in neural information processing systems*, 5: 895–902, 1993.

Jonathan Schler. The importance of neutral examples for learning sentiment. In *In Workshop on the Analysis of Informal and Formal Information Exchange during Negotiations (FINEXIN*. Citeseer, 2005.

Harold Schlosberg. Three dimensions of emotion. *Psychological review*, 61(2):81, 1954.

Kazutaka Shimada and Tsutomu Endo. Seeing several stars: A rating inference task for a document containing several evaluation criteria. In *Advances in Knowledge Discovery and Data Mining*, pages 1006–1014. Springer, 2008.

Mark Steyvers and Tom Griffiths. Probabilistic topic models. *Handbook of latent semantic analysis*, 427(7):424–440, 2007.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2):267–307, 2011.

Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet Processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

Peter Turney and Michael L Littman. Unsupervised learning of semantic orientation from a hundred-billion-word corpus. 2002.

Peter D Turney. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 417–424. Association for Computational Linguistics, 2002.

Peter D Turney and Michael L Littman. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems (TOIS)*, 21(4):315–346, 2003.

Peter D Turney, Patrick Pantel, et al. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37(1):141–188, 2010.

Sudha Verma, Sarah Vieweg, William J Corvey, Leysia Palen, James H Martin, Martha Palmer, Aaron Schram, and Kenneth Mark Anderson. Natural Language Processing to the Rescue? Extracting" Situational Awareness" Tweets During Mass Emergency. In *ICWSM*. Citeseer, 2011.

Chong Wang and David M Blei. Decoupling sparsity and smoothness in the discrete hierarchical dirichlet process. In *Advances in neural information processing systems*, pages 1982–1989, 2009.

Bing Xiang, Liang Zhou, and Thomson Reuters. Improving twitter sentiment analysis with topic-based mixture modeling and semi-supervised training. In *ACL (2)*, pages 434–439, 2014.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. Relation classification via convolutional deep neural network. In *COLING*, pages 2335–2344, 2014.

Jingbo Zhu, Muhua Zhu, Huizhen Wang, and Benjamin K Tsou. Aspect-based sentence segmentation for sentiment summarization. In *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*, pages 65–72. ACM, 2009.