Sentic LDA: Improving on LDA with Semantic Similarity for Aspect-Based Sentiment Analysis

Soujanya Poria, Iti Chaturvedi, and Erik Cambria
School of Computer Science and Engineering
Nanyang Technological University, Singapore
Email: {sporia,iti,cambria}@ntu.edu.sg

Federica Bisio
DITEN
University of Genoa, Italy
Email: federica.bisio@edu.unige.it

***Abstract*—The advent of the Social Web has provided netizens with new tools for creating and sharing, in a time- and cost-efficient way, their contents, ideas, and opinions with virtually the millions of people connected to the World Wide Web. This huge amount of information, however, is mainly unstructured as specifically produced for human consumption and, hence, it is not directly machine-processable. In order to enable a more efficient passage from unstructured information to structured data, aspect-based opinion mining models the relations between opinion targets contained in a document and the polarity values associated with these. Because aspects are often implicit, however, spotting them and calculating their respective polarity is an extremely difficult task, which is closer to natural language understanding rather than natural language processing. To this end, Sentic LDA exploits common-sense reasoning to shift LDA clustering from a syntactic to a semantic level. Rather than looking at word co-occurrence frequencies, Sentic LDA leverages on the semantics associated with words and multi-word expressions to improve clustering and, hence, outperform state-of-the-art techniques for aspect extraction.**

I. Introduction

The opportunity to capture the opinions of the general public about social events, political movements, company strategies, marketing campaigns, and product preferences has raised increasing interest both in the scientific community, for the exciting open challenges, and in the business world, for the remarkable fallouts in marketing and financial market prediction. Today, sentiment analysis research finds applications in several different scenarios and there is a good number of companies, large and small, that include the analysis of opinions and sentiments as part of their mission [1].

Opinion mining techniques can be exploited for the creation and automated upkeep of review and opinion aggregation websites. Sentiment analysis has also a great potential as a sub-component technology for other systems. It can enhance the capabilities of customer relationship management and recommendation systems allowing, for example, to find out which features customers are particularly interested in or to exclude from the recommendations items that have received very negative feedbacks. Similarly, they can be used in social communication for troll filtering and to enhance anti-spam systems.

Business intelligence is also one of the main factors behind corporate interest in the field of sentiment analysis. Nowadays, companies invest an increasing amount of money in marketing strategies and they are constantly interested in both collecting and predicting the attitudes of the general public towards their products and brands. The design of automatic tools capable to crawl reviews and opinions over the Web in real-time and to create condensed versions of them represents one of the most active research and development area. The development of such systems, moreover, is not only important for commercial purposes, but also for government intelligence applications able to monitor increases in hostile communications or to model cyber-issue diffusion.

Several commercial and academic tools track public viewpoints on a large-scale by offering graphical summarizations of trends and opinions in the blogosphere. Nevertheless, most commercial off-the-shelf (COTS) tools are limited to the extraction of a polarity value associated to the whole document, rather than distinguishing into single aspects. In addition, such methods mainly rely on parts of text in which emotional states are explicitly expressed and, hence, they are unable to capture opinions and sentiments that are expressed implicitly. Because they are mainly based on statistical properties associated with words [2], in fact, COTS tools are easily tricked by linguistic operators such as negation and disjunction.

In this work, we present a novel framework, termed Sentic LDA, which integrates common-sense computing [3] in the calculation of word distributions in the latent Dirichlet allocation (LDA) algorithm [4], thus enabling the shift from syntax to semantics in aspect-based sentiment analysis. Sentic LDA, in fact, goes beyond a mere statistical approach to aspect extraction by leveraging on the semantics associated with words and, hence, improves clustering.

The structure of the paper is as follows: Section II illustrates related work in the field of aspect-based sentiment analysis; Section III describes how the proposed framework enhances the standard LDA algorithm; Section IV explains how aspects are extracted from text; Section V proposes a comparative evaluation of Sentic LDA with state-of-the art techniques for aspect extraction; finally, Section VI sets out conclusions.

978-1-5090-0620-5/16/$31.00

II. Related work

Aspect extraction from opinions was first studied by Hu and Liu [5]. They introduced the distinction between explicit and implicit aspects. However, the authors only dealt with explicit aspects. They used a set of rules based on statistical observations. Hu and Liu's method was later improved by Popescu and Etzioni [6] and by Blair-Goldensonh [7]. [6] assumes the product class is known in advance. Their algorithm detects whether a noun or noun phrase is a product feature by computing PMI between the noun phrase and the product class. Scaffidi et al. [8] presented a method that uses language model to identify product features. They assume that the product features are more frequent in product reviews than in a general natural language text. However, their method seems to be inaccurate in terms of precision, since the retrieved aspects are affected by noise.

Topic modeling has been widely used as a basis to perform extraction and grouping of aspects [9] [10]. In the literature, two models have been considered: pLSA [11] and LDA [12]. Both models introduce a latent variable 'topic' between the observed variables 'document' and 'word' to analyze the semantic topic distribution of documents. In topic models, each document is represented as a random mixture over latent topics, where each topic is characterized by a distribution over words. The LDA model defines a Dirichlet probabilistic generative process for document-topic distribution; in each document, a latent aspect is chosen according to a multinomial distribution, controlled by a Dirichlet prior α. Then, given an aspect, a word is extracted according to another multinomial distribution, controlled by another Dirichlet prior β.

Among the existing works employing these models, we can find the extraction of global aspects (such as the brand of a product) and local aspects (such as the property of a product) [13], the extraction of key phrases [14], the rating of multi-aspects [15] and the summarization of aspects and sentiments [16]. [17] employed Maximum-Entropy to train a switch variable based on POS tags of words and use it to separate aspect and sentiment words. [18] added user feedback into LDA as a response variable connected to each document. In [19], a semi-supervised model was proposed. DF-LDA [20] also represents a semi-supervised model, which allows the user to set must-link and cannot-link constraints. A must-link means that two terms must be in the same topic, while a cannot-link means that two terms cannot be in the same topic. [21] proposed two semi-supervised models for product aspect extraction, based on the use of seeding aspects. Inside the category of supervised methods, [22] employed seed words to guide topic models to learn topics of specific interest to a user, while [15] and [23] employed seeding words to extract related product aspects from product reviews.

We propose a novel LDA algorithm for clustering *aspect terms* according to their *aspect category*. This method, termed Sentic LDA from sentic computing [24], outperforms state-of-the-art algorithms by leveraging on the semantic similarity between two words for supervising the clustering process.

As a consequence, the words are not only clustered depending on the word frequency measure, but also considering the semantic similarity between each pair of words. Another key aspect of the proposed framework is that, unlike most state-of-the-art tools, it proposes an unsupervised aspect extraction, which makes it highly scalable. The ensemble application of the new LDA method and the unsupervised aspect term extraction algorithm allows the proposed framework to automatically find the categories of the aspect terms. To the best of our knowledge, there is no state-of-the-art work describing the problem of automatic retrieval of aspect terms and their categories without using any knowledge base.

III. Sentic LDA

The basic steps of the proposed framework can be summarized as follows:

- First, a training corpus is processed using the *Sentic LDA* framework. LDA produces a set of clusters where each cluster represents an aspect category.
- After the clustering process, we manually label each cluster by looking at the elements inside. The metric for labeling a cluster with a specific aspect category is a majority-based criterion, counting the aspect terms belonging to a certain aspect category. It has to be noted that the number of clusters is fixed a priori based on the training corpus.
- To extract the aspect terms from the input sentence, an unsupervised aspect term extraction process is employed.
- Each aspect term is searched among the clusters produced by LDA in Step 1. If the term is found in more than one cluster, then the cluster for which the aspect term has the highest probability to belong to is chosen and the corresponding aspect category is assigned.

A. Method Description

The proposed scheme for LDA is structured as follows. Let V be the vocabulary length with $[\mathbf{w}_1, ..., \mathbf{w}_W]$ words of the vocabulary. For each document $d \in \{1, ..., D\}$, S_d is the number of sentences; for each sentence $s_d \in \{1, ..., S_d\}$, $N_{(d,s)}$ is the number of words. We assume that each sentence talks about one aspect. Let C be the number of seed sets, with $[Q_1, ..., Q_C]$ seed sets (i.e., groups of semantically related terms). α represents the vector of Dirichlet priors related to the document-topic distribution; without loss of generality, we can assume same α values for all topics. β defines the vector of Dirichlet priors related to the topic-word distribution. For each word in the vocabulary, β is used to embed the common-sense similarity between the current word and each aspect (Fig. 1).

By common-sense knowledge, we mean obvious things people normally know and usually leave unstated. In AI, commonsense usually refers to background knowledge humans rely on for understanding and communication, e.g., the way objects relate to each other in the world, people's goals in their daily lives, and the emotional content of events or situations. We extract such knowledge from SenticNet [25] and its extensions [26], [27].

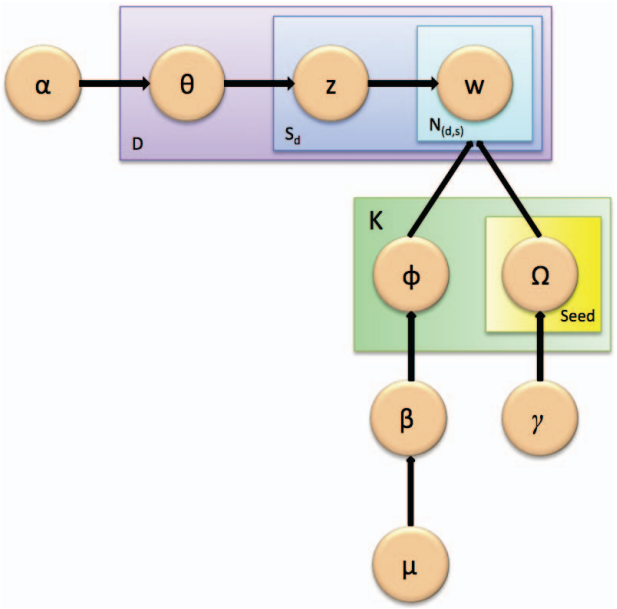

Figure 1. LDA framework

In particular, we indicate as μ the contribution of commonsense knowledge in determining β. To this end, we update at each iteration of the collapsed Gibbs sampling algorithm the values of β for each word, calculating the average similarity between the current word and the top 50 words contained in the cluster related to the current aspect.

We set the threshold to 50 words by trial and error to minimize noise in the clusters, while covering almost all major aspect terms in the corpus. Since high values of β refer to a topic-word distribution where many words per topic are present, then reasonably β values for seed words will be generally higher to their related aspects.

For example, if w_i is a seed word for an aspect t_k, then the correspondent β has a high value; if it is a general word then the value should be considerably lower. This is due to the fact that seeds actually guide the clustering LDA process modifying the probability distributions to obtain more reliable aspect categories, using the information given by commonsense knowledge. As a result, the β priors can be represented in a matrix structured as follows:

	t_1	t_2	...	t_K
w_1	β_{11}	β_{12}	...	β_{1K}
...				
w_i	β_{i1}	β_{i2}	...	β_{iK}
...				
w_N	β_{N1}	β_{N2}	...	β_{NK}

where t indicates a topic and K is the number of topics, w indicates a word and N is the number of words. For seed words β priors will be higher than β for general words for the related seeded aspect. γ represents the vector of Dirichlet priors related to the seed-word distribution.

The generation process is given as follows:

- $\forall k \in \{1,...,K\}$:
  - $\forall w_i \in [w_1,...,w_W]$:
    - $\phi_{k,i} \sim Dir(\beta_i)$ distribution of topics over words
    - $\forall l \in [Q_1,...,Q_C]$:
      - $\Omega_{k,l} \sim Dir(\gamma)$ distribution over seeds
- $\forall d \in \{1,...,D\}$:
  - $\theta_d \sim Dir(\alpha)$ distribution of documents over topics
  - $\forall s_d \in d$:
    - $z_{(d,s)} \sim Multi(\theta_d)$ draw an aspect
    - $\forall w_i \in s_d$:
      - $w_i \sim Multi(\phi_{z_{(d,s)}})$ draw a word
      - if w_i is general word: emit w_i
      - if w_i is seed word: $w_i \sim \Omega_{z_{(d,s)},l_{(d,s,i)}}$

We employ collapsed Gibbs sampling to estimate the posterior distribution of $P(\mathbf{z}/\mathbf{w};\alpha,\beta,\gamma)$. Since $P(\mathbf{z}/\mathbf{w};\alpha,\beta,\gamma) = \frac{P(\mathbf{z},\mathbf{w};\alpha,\beta,\gamma)}{\sum_z P(\mathbf{z},\mathbf{w};\alpha,\beta,\gamma)}$, Gibbs sampling equations can be derived from the joint likelihood of a word w with an aspect assignment z. Given the total probability of the model:

$$P(\mathbf{z},\mathbf{w},\theta,\phi,\Omega;\alpha,\beta,\gamma) = P(\phi;\beta)P(\theta;\alpha)P(\Omega;\gamma)\, P(\mathbf{z}/\theta)P(\mathbf{w}/\phi)P(\mathbf{w}/\Omega) \quad (1)$$

The joint likelihood of w and z is obtained integrating in θ, ϕ and Ω:

$$P(\mathbf{z},\mathbf{w};\alpha,\beta,\gamma) = \int P(\mathbf{z}/\theta)P(\theta;\alpha)\,d\theta \times \int P(\mathbf{w}/\phi)P(\phi;\beta)\,d\phi \times \int P(\mathbf{w}/\Omega)P(\Omega;\gamma)\,d\Omega \quad (2)$$

Since all terms are multinomials with Dirichlet priors and Dirichlet distribution is conjugate to the multinomial distribution, we can derive:

$$P(\mathbf{z},\mathbf{w};\alpha,\beta,\gamma) = \prod_d \frac{B(n^{sent}_{d,(.)}+\alpha)}{B(\alpha)} \prod_k \frac{B(n^{U}_{(.),k}+\beta)}{B(\beta)} \prod_l \frac{B(n^{S}_{(.),k,l}+\gamma)}{B(\gamma)} \quad (3)$$

where $n^{sent}_{d,k}$ is the number of sentences in document d assigned to aspect k, $n^{U}_{d,k}$ is the number of times general words in document d are assigned to aspect k, $n^{S}_{d,k,l}$ is the number of times seed words in seed set Q_l in document d are assigned to aspect k, and $B(\alpha) = \frac{\prod_k \Gamma(\alpha_k)}{\Gamma(\sum_k \alpha_k)}$, $B(\beta) = \frac{\prod_r \Gamma(\beta_r)}{\Gamma(\sum_r \beta_r)}$ ($\forall r$ word in the vocabulary V) and $B(\gamma) = \frac{\prod_r \Gamma(\gamma_r)}{\Gamma(\sum_r \gamma_r)}$ ($\forall r$ word in the vocabulary V_l) are the multinomial beta functions.

Given that, the posterior distribution at each step of the Markov chain can be obtained as:

$$P(\mathbf{z}_{(d,s,i)} = k/\mathbf{z}_{-(d,s,i)}, \mathbf{w}; \alpha, \beta, \gamma) = \\ = \frac{P(\mathbf{z}, \mathbf{w}; \alpha, \beta, \gamma)}{P(\mathbf{z}_{-(d,s,i)}, \mathbf{w}; \alpha, \beta, \gamma)} = \\ = \prod_d \frac{B(n^{sent}_{d,(.)} + \alpha)}{B(n^{sent,-(d,s,i)}_{d,(.)} + \alpha)} \\ \prod_k \frac{B(n^{U}_{(.),k} + \beta)}{B(n^{U,-(d,s,i)}_{(.),k} + \beta)} \\ \prod_l \frac{B(n^{S}_{(.),k,l} + \gamma)}{B(n^{S,-(d,s,i)}_{(.),k,l} + \gamma)} \tag{4}$$

where i is the current word, s the current sentence, d the current document and $-(d,s,i)$ means "leaving out the i-th word of sentence s in document d". The final equation is:

$$P(\mathbf{z}_{(d,s,i)} = k/\mathbf{z}_{-(d,s,i)}, \mathbf{w}; \alpha, \beta, \gamma) \propto \\ \propto (n^{sent,-(d,s,i)}_{d,k} + \alpha_k) \\ \frac{n^{U,-(d,s,i)}_{v,k} + \beta_v}{\sum_{r=1}^{V} n^{U,-(d,s,i)}_{r,k} + \beta_r} \\ \frac{n^{S,-(d,s,i)}_{v,k,l} + \gamma_l}{\sum_{m=1}^{V_l} n^{S,-(d,s,i)}_{m,k,l} + \gamma_m} \tag{5}$$

where v is the vocabulary position of the i-th word in V.

The LDA Collapsed Gibbs Sampling algorithm is then modified to add common-sense knowledge, as in the following:

- D number of documents, S_d number of sentences for each document $d \in \{1,...,D\}$, T number of iterations, $N_{(d,s)}$ number of words of sentence s in document d, K number of aspects, w_i i-th word of a sentence, $s_{i,k}$ similarity measure between the i-th word of a sentence and the cluster related to the corresponding aspect k.

- Randomly initialize $\mathbf{z}$
- $\forall s_d \in \{1,...,S_d\}$ of d where $d \in \{1,...,D\}$
- for $t = 1 \rightarrow T$
- for $i = 1 \rightarrow N_{d,s}$
  - $n_{d,s,z} -= 1$ $n_{i,z} -= 1$
  - for $k = 1 \rightarrow K$
    - Calculate $s_{i,k,t}$
    - if $(s_{i,k,t} > s_{i,k,t-1} + 0.1)$ increase β_v=$\beta_v + 0.001$
      - if $(\beta_v > 1)$ decrease β_v=$\beta_v - 0.001$
    - else if $(s_{i,k,t} < s_{i,k,t-1} - 0.1)$ decrease β_v=$\beta_v - 0.001$
      - if $(\beta_v < 0)$ increase β_v=$\beta_v + 0.001$
    - $P(\mathbf{z} = k/\mathbf{w}) = (n^{sent}_{d,k} + \alpha_k) \frac{n^{U}_{v,k} + \beta_v}{\sum_{r=1}^{V} n^{U}_{r,k} + \beta_r} \frac{n^{S}_{v,k,l} + \gamma_l}{\sum_{m=1}^{V_l} n^{S}_{m,k,l} + \gamma_m}$
  - Sample aspect from $P(\mathbf{z}/\mathbf{w})$
  - $n_{d,s,z} += 1$ $n_{i,z} += 1$

After an initialization of the variables, the algorithm is run over a certain number of iterations. In each loop, an aspect is sampled for each word of a sentence. Before building the distribution, it is necessary to remove the current assignment from the equation, decrementing the counts. If the word is a general word, the similarity measure s is calculated and β are updated accordingly.

Changing the value of β means modifying the probability of a word to belong to a topic. At each iteration for a word w_i, the average similarity is calculated between w_i and the top 50 words in the topic. The details of the similarity measure is described in next section. If the similarity value for the i-th word decreases by a certain value compared to the previous iteration, then also the corresponding β has to be decreased. Experimentally, we set the factor used to increase or decrease β to 0.001. We change β only when the similarity score for a word increases or decreases by at least 0.1 compared to the previous iteration. This value is also set experimentally.

The main hypothesis behind this algorithm is that common-sense knowledge can better match similarity between words (Fig. 2). While word co-occurrence frequencies only grasp similarity at syntactic level, in fact, semantic similarity can handle conceptual similarity between natural language concepts and, hence, can efficiently guide the creation of consistent word-topic probability distributions to obtain more reliable aspect terms per aspect category. The posterior probability is calculated using (5); the distribution is then sampled to obtain the aspect and the related counts are incremented.

B. Similarity Calculation

We calculate similarity between words and multi-word expressions by means of affective analogical reasoning [28]. In particular, we apply dimensionality-reduction techniques to a matrix of common-sense knowledge [29].

The objective of such a compression is to describe semantic features that could apply to known affective concepts by analogy: if a concept in the matrix has no value specified for a feature owned by many similar concepts, then by analogy the concept is likely to have that feature as well. In other words, concepts and features that point in similar directions and, therefore, have high dot products, are good candidates for analogies. The matrix is first normalized so that the dot products in the similarity matrix are simply the cosines of the angles between the corresponding vectors, creating a well-defined similarity scale that ranges from 1 (exactly similar) to -1 (exactly dissimilar).

We removed stop-words, punctuation and words which occur less than 5 times in the corpus. Then, we measured the semantic similarity between each pair of words and stored the information in a database. This database was then employed in the LDA process described in Section III-A to find the similarity between two words. In the LDA algorithm at each iteration, for a word w_i the similarity score between w_i and each of the top 50 words in the topic t_k is retrieved from the database and then the average of scores are calculated.

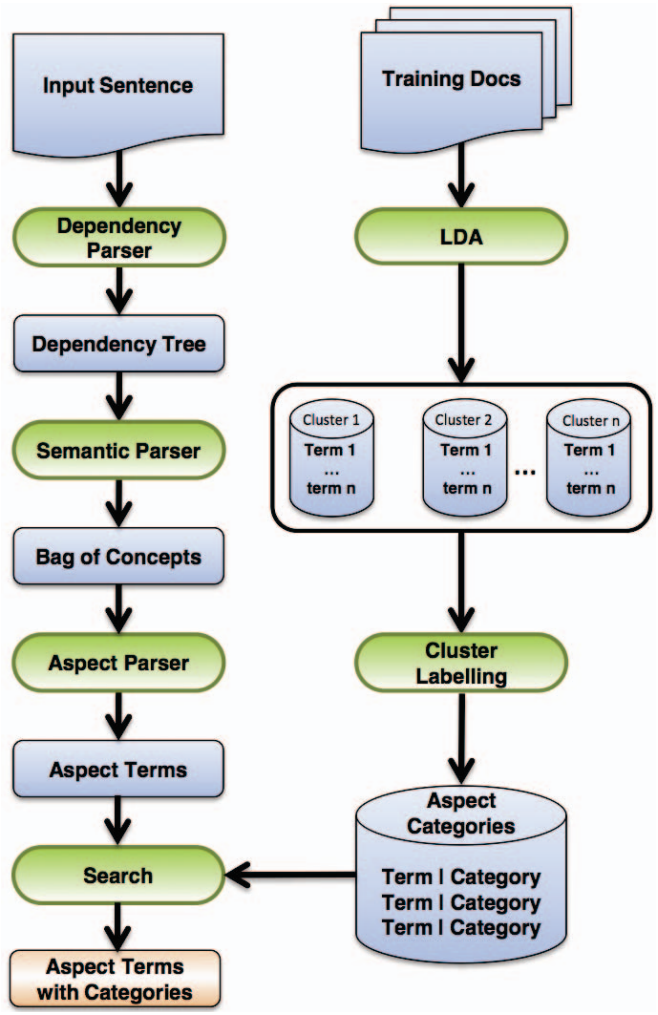

Figure 2. Aspect extraction flowchart

IV. Aspect Term Extraction

A. Pre-Processing

Pre-processing is a key step of the proposed aspect parser. Our pre-processing method consists of two steps:

- First, the sentence is sent to *Stanford-dependency-parser*[1] to obtain the dependency-parsed output.
- A lemmatizer (*Stanford Lemmatizer*) is then used to lemmatize each word in the dependency-parsed output.

If we switch the steps the parser may produce wrong results, since after lemmatization a sentence can be grammatically incorrect.

B. Aspect Parser

The aspect parser is based on two general types of rules:

- Rules for the sentences having the subject of verb.
- Rules for the sentences which do not have the subject of verb.

[1]http://nlp.stanford.edu:8080/parser

A dependency relation is a binary relation characterized by the following features:

- The type of the relation that specifies the nature of the (syntactic) link between the two elements in the relation.
- The head of the relation, which is the pivot of the relation. Core syntactic and semantics properties (e.g., agreement) are inherited from the head.
- The dependent is the element that depends on the head and which usually inherits some of its characteristics (e.g., number, gender in the case of agreement).

Most of the times, the head of the relation is considered as the active token, although some rules represent an exception. Once the active token has been identified as the trigger of a rule, there are several ways to compute its contribution, depending on how the dependency relation and the properties of the tokens match with the rules. The preferred way is to consider the contribution not of the token alone, but in combination with the other elements in the dependency relation.

First of all, Stanford parser is used to obtain the dependency parse structure of the sentences. Then, handcrafted dependency rules [30] are employed on the parse trees to extract aspects.

1) Subject Noun Rule: Trigger: when the active token is found to be the syntactic subject of a token. *Behavior:* if an active token h is in a subject noun relationship with a word t then -

1) if t has any adverbial or adjective modifier and the modifier exists in SenticNet then we extract t as an aspect.
2) if the sentence does not have auxiliary verb, i.e., *is, was, would, should, could etc.*, then -
   - if the verb t is modified by an adjective or adverb or is in *adverbial clause modifier* relation with another token then both h and t are extracted as an aspect. In (1), *battery* is in a subject relation with *lasts* and *lasts* is modified by an adjective modifier *little*, then the aspects *last* and *battery* are extracted.

     (1) The battery lasts little

   - if t has any direct object relation with a token n (where n is a *Noun*) and n is not in SenticNet, then n is extracted as an aspect. In (2), *like* is in direct object relation with *lens*, then the aspect *lens* is extracted.

     (2) I like the lens of this camera

   - if t has any direct object relation with a token n (where n is a *Noun*) and n exists in SenticNet, then the token n is extracted as an aspect term. In the dependency parse tree of the sentence, if another token n_1 is connected to n using any dependency relation and n_1 is a *noun*, then we extract n_1 as an aspect. In (3), *like* is in direct object relation with *beauty*, which is connected to *screen* via a preposition relation. Both the aspects *screen* and *beauty* are extracted.

     (3) I like the beauty of the screen

- if t is in open clausal complement relation with a token t_1, then the aspect t-t_1 is extracted, if t-t_1 exists in the opinion lexicon. If, using some other relations, t_1 is connected with a token t_2 whose part of speech is noun, then t_2 is extracted as an aspect.
  In (4) *like* and *comment* are in clausal complement relation and *comment* is connected to *camera* using a preposition relation. Here, *camera* is a *Noun*, so *camera* is extracted as an aspect.

  (4) I would like to comment on the camera of this phone

3) A copula is the relation between *the complement of a copular verb* and *the copular verb*. If the token t is in copula relation and the copular verb exists in the implicit aspects lexicon, then we extract t as an aspect term. t represents an *Implicit Aspect Clue* (IAC). In (5), *expensive* is extracted as an aspect.

   (5) The car is expensive

4) If the token t is in copula relation with a copular verb and the part-of-speech of h is *Noun*, then h is extracted as an explicit aspect. In (6), *camera* is extracted as an aspect.

   (6) The camera is nice

5) If the token t is in copula relation with a copular verb and the copular verb is connected to a token t_1 using any dependency relation, where t_1 is a verb, then we extract t_1 as an implicit aspect term, if it exists in the implicit aspect lexicon. In addition, if t exists in the implicit aspect lexicon, then it is also extracted as an implicit aspect term. In (7) *lightweight* is in copula relation with *is* and *lightweight* is connected to the word *carry* by open clausal complement relation. Here, both *lightweight* and *carry* are extracted as aspects.

   (7) The phone is very lightweight to carry

2) Sentences which do not have subject noun relation in their parse tree: For sentences which do not have noun subject relation in their parse trees, we use the following rules to extract aspects -

1) if an adjective or adverb h is in infinitival or open clausal complement relation with a token t and h exists in the implicit aspects lexicon, then we extract h as an aspect. In (8), we extract *big* as an aspect as it is connected to *hold* using a clausal complement relation.

   (8) Very big to hold

2) if a token h is connected to a noun t using a prepositional relation, then we extract both h and t as aspects. In (9) *sleekness* is extracted as an aspect.

   (9) Love the sleekness of the player

3) if a token h is in a direct object relation with a token t, then we extract t as the aspect. In (10), *mention* is in a direct object relation with *price*, so we extract *price* as an aspect.

   (10) Not to mention the price of the phone

3) Additional Rules:

- For each aspect term extracted above - if an aspect term h is in co-ordination or conjunction relation with another token t, then we also extract t as an aspect. In (11), first *amazing* is extracted as an aspect term. As *amazing* is in conjunction relation with *easy*, then we also extract *use* as an aspect.

  (11) The camera is amazing and easy to use

- A noun compound modifier is any noun that serves to modify the head noun. If t is extracted as an aspect and t has noun compound modifier h, then we extract the aspect h-t and remove t from the aspect list. In (12), *chicken* is extracted as an aspect using the rules described above. As *chicken* and *casserole* are in *noun compound modifier* relation, then we extract *chicken casserole* as an aspect and remove *chicken* from the aspect list.

  (12) We ordered the chicken casserole, but what we got were a few small pieces of chicken, all dark meat and on the bone

C. Noise Filtering

We have developed a list of some non-aspect terms (*I, You, They, People, It, This etc.*) The list contains 57 common non-aspect terms which are often extracted by the proposed aspect parser. This list is used to clean the noisy non-aspect terms from the extracted aspect list. Each aspect extracted by the aspect parser is first searched in the list and if it is found it is not considered as aspect term. This step actually enhances the precision of the aspect term extractor.

D. Aspect Category Inference

The process of obtaining aspect categories of terms can be seen as a 2-stage workflow -

- The clusters generated by LDA are manually labeled by aspect category. The main hypothesis of this procedure is that humans can easily assign aspect terms to aspect categories based on common-sense knowledge and domain knowledge. If the majority of aspect terms of a cluster belong to an aspect category A_i then the cluster is labeled by the A_i. Results indeed confirm the effectiveness of the hypothesis, since in each cluster only few aspect terms from other aspect categories can be found. The generated clusters are stored in a database, where each aspect term or word is labeled by its aspect category. If an aspect term t_i occurs within the top 50 words in more than one cluster, then we choose only the cluster for which t_i has the highest probability and t_i is labeled by the corresponding aspect category.

- After obtaining the aspect terms from a sentence with the aspect term extraction procedure, we search each aspect term in the database and, if it is found, then it is labeled by its aspect category. Otherwise, nothing is done.

V. Evaluation

In this section, we first discuss the performance of Sentic LDA and then we show the performance of the overall aspect extraction framework.

A. Aspect Extraction Corpus

For the evaluation of the proposed approach we used Semeval 2014 dataset[2]. Since our method focuses on both aspect term and aspect category detection, we only used the Restaurants corpus (as the aspect category labeling is available only for this corpora). In this dataset, aspect categories existing in a sentence are present but the aspect terms in the sentence are not assigned to any aspect category. Therefore, we manually labeled each aspect term by its corresponding aspect category. We obtained a total of 3713 sentences for training and 1025 sentences for testing in Semeval 2014 restaurants dataset. We compared our method with the best state of the art on the test sentences. It has to be noted that our method does not need any prior training.

B. Sentic LDA Performance

1) Dataset and Parameter Settings: To evaluate the proposed methodology, a collection of 235,793 hotel reviews was obtained. This benchmark data set was collected by Wang, Lu, and Zhai (2010) [15], from the hotels review site TripAdvisor.com. As proposed by [23] we removed the punctuations, stop words and words which occur less than 5 times in the corpus. There are 8 aspect categories labeled in the dataset - Overall, Value, Rooms, Location, Cleanliness, Check in/Front desk, Service and Business Service. We excluded *Overall* from the aspect category list because it does not present any definite aspect. In fact, experiments confirmed that considering *Overall* as an aspect category increases the amount of noise in the LDA result. Many non aspects like *night*, *walk*, *actual* are included as the top probable aspect terms in a cluster.

We have selected SAS [23] and DF-LDA [20] for a comparison with our approach. These two methods are indeed well known in this area and are recognized as best state of the art. For all models, we took a single sample as posterior inference after 3000 iterations of Gibbs sampling with 200 burn-in iterations. We used default parameters for the DF-LDA as mentioned in the code package[3]. As DF-LDA needs some supervision provided by the must-link and cannot-link constraints, we used the words in the same seeding aspect to generate must-link constraints and the words from different seeding aspects to generate cannot-link constraints. Let K be the number of topics, which is 7 in our case. For SAS model we used α as 50/K, β as 0.1 and γ as 250.

[2]http://alt.qcri.org/semeval2014/task4/index.php?id=data-and-tools

[3]http://pages.cs.wisc.edu/~andrzeje/research/df_lda.html

For Sentic LDA, we initialized α as 50/K, while different β are present for each word. For each general word w_g and a topic t_k, β_{gk} was initialized to 0.1 as suggested by [23]. If a seed word w_s belongs to an aspect category t_k then β_{sk} was initialized to 0.7 otherwise β_{sk} was set to 0.1. The higher β value of a seed word for its aspect category is used to supervise the LDA process so that a word should be in the most similar cluster. Following the suggestion of [23] in our case we used γ as 250.

2) Comparison with the State of the Art: Table I shows some qualitative results and a comparison between the proposed Sentic LDA and state-of-the-art methods. It can be seen that Sentic LDA has produced most relevant aspect terms compared to other algorithms.

Red colored words in Table I indicate clustering errors, i.e., the words are placed in the wrong cluster. We also used *Rand-index* and *Entropy* as metrics to evaluate the proposed algorithm. As explained before, LDA generates data D_{LDA}= $d_1, d_2, d_3, ..., d_n$. Here, n is the number of clusters, while d_i represents a single cluster, containing top k probable words. We developed a gold standard dataset D_{gold}=$g_1, g_2, ..., g_n$. The gold standard data were developed based on the generated output of LDA. Each of the 6 clusters was manually labeled; considering the top 50 words in each cluster, we assigned the aspect category which seems to be most prominent according to the human judgment.

a) Rand-index: is a measure of similarity between the LDA output and the gold standard dataset. The larger the value of Rand-index the better result we obtain. For n data points and for each pair of this data, let x represent the number of times two data points belong to the same cluster in both D_{LDA} and D_{gold}; y represents the number of times they do not occur in the same cluster in generated output and gold standard. Then, the rand index is calculated as follows:

$$RandIndex = 2 * \frac{(x+y)}{n*(n-1)} \tag{6}$$

b) Entropy: measures the proportion of the gold standard cluster g_i presented in the generated cluster d_i. The smaller values of entropy denote better accuracy. Equation 7 measures the entropy of a cluster.

$$Entropy(d_i) = -\sum_{j=1}^{n} Pr_i(g_j) \log_2 Pr_i(g_j) \tag{7}$$

$$Entropy(d_{LDA}) = \sum_{i=1}^{n} \frac{|d_i|}{|d_{LDA}|} Entropy(d_i) \tag{8}$$

Table II shows the evaluation results and the comparison between Sentic LDA and the other state-of-the-art LDA algorithms on the same TripAdvisor dataset. It can be noticed that both DF-LDA and SAS performed quite similar based on the Rand-index, even if SAS performance is slightly better. On the other hand, Sentic LDA outperformed both DF-LDA and SAS in terms of Rand-index. Taking entropy as an evaluation metric, we found SAS outperformed DF-LDA, while Sentic LDA again achieved the best performance.

Table I
EXAMPLE OF THE TOP WORDS PER ASPECT GENERATED BY LDA

Aspect	DF-LDA	SAS	Sentic LDA
Location (Seeds: location, area, street, airport, site)	location, car, price, view, taxi, staff, area, parking, place, hotel	location, site, parking, restaurant, hotel, view, service, area, street, place	location, area, place, hotel, airport, distance, car, parking, view, city
Service (Seeds: service, staff, waiter, manager, bar)	service, food, wine, bar, towel, breakfast, bed, large, security, manager	service, waiter, staff, security, hospitality, food, clean, manager, maintenance, lounge	staff, hospitality, service, manager, waiter, waitress, helpful, maintenance, reception, pool
Value (Seeds: value, cheap, price, quality, expensive)	value, worth, pay, kitchen, cheap, discount, nonsmoking, cost, dollar, tidy	price, worth, expensive, cost, room, cheap, value, quality, location, money	value, cost, price, cheap, worth, money, pay, dollar, expensive, bargain

Moreover, it is clear that using both metrics the performance of all LDA algorithms deteriorate as ranking size increases. The reason why the performance of SAS is better than DF-LDA lies in its ability to generate aspect specific words (words which are very related to the seeds). In this case, the seeds provide sufficient supervision in the clustering process. Sentic LDA uses common-sense knowledge based similarity between words so that two related and similar words are forced to be in the same cluster. Here, common-sense knowledge acts as a supervision to the LDA model. In Table I, we show 10 top aspect terms generated by all LDA algorithms under Location and Service aspect category. The aspect terms marked in *red* are the ones which are incorrectly generated by the LDA algorithm.

C. Overall Performance of the Aspect Parser

In this section, we discuss the overall performance obtained by the algorithm described in Section IV. First, Sentic LDA was employed on the corpus described in Section V-A. Then, each sentence of the corpus was parsed through the unsupervised aspect term parser; those terms were later searched in the clusters generated by Sentic LDA in order to obtain aspect categories. It should be noted that our prior work [31] only described the aspect term extraction method and did not include the extraction of aspect category.

We experimented on Semeval 2014 aspect-based sentiment analysis dataset. Table III shows the results of the experiment carried out on Semeval 2014 dataset. We compare the proposed method with the best system in Semeval 2014 contest, showing that our approach improves over this system in both precision and recall. Overall, the precision shows high accuracy is obtained. However, the recall suffers from the noise existing in the extracted aspects.

The aspect categories are FOOD, SERVICE, PRICE, AMBIENCE and ANECDOTES. Therefore, for LDA we used 6 as a number of clusters. [32] used a supervised approach to obtain aspect category extraction. We found a fundamental issue in their approach. It can predict how many aspect categories exist in a sentence but it is unable to detect the definite aspect category of an aspect term.

Table II
PERFORMANCE COMPARISON OF LDA ALGORITHMS

	Training Dataset	DF-LDA	SAS	Sentic LDA
Rand Index	top 5	0.78	0.79	0.91
	top 15	0.77	0.79	0.89
	top 30	0.75	0.77	0.86
Entropy	top 5	1.21	1.15	0.85
	top 15	1.43	1.20	0.91
	top 30	1.75	1.45	0.95

In real applications, it is important to know the aspect category of an aspect term so that aspect-based sentiment analysis can be done more efficiently. For example, the sentence *The restaurant was expensive, but the menu was great* has *expensive* and *menu* as aspect terms. Both algorithms can detect the aspect terms accurately but [32] cannot obtain the category of *menu* and *expensive*, i.e., *FOOD* and *PRICE*, respectively.

Therefore, [32] cannot detect the sentiment expressed on these two aspect categories as well, though their approach detects these categories in the sentence. However, the sentiment expressed on *menu* and *expensive* can be detected. This underlines the importance to know the aspect category of each aspect term in a sentence, in order to perform aspect-based sentiment analysis. [32] trained their system on the training data available in Semeval 2014 data.

In our case, since we do not need any training data, we evaluated our system on the available training data, which gave us 91.15% precision and 87.21% recall. However, as a state-of-the-art system evaluated on the training data does not exist, this comparison is not reported.

Table III
RESULTS ON SEMEVAL 2014 DATASET

Algorithm	Precision	Recall
[32]	91.04%	86.24%
Proposed Approach	92.12%	88.25%

Now, we want to consider the output of the aspect parser to be true only when it correctly detects the aspect term and its corresponding aspect category. This experiment shows a decrease both in precision and recall. We obtained 78.21% recall and 82.42% precision in this experiment.

VI. Conclusion

The shift from a read-only to a read-write Web made users more enthusiastic about interacting, sharing, and collaborating through social networks, online communities, blogs, wikis, and other online collaborative media. In recent years, this collective intelligence has spread to many different areas of the Web, with particular focus on fields related to our everyday life such as commerce, tourism, education, and health.

Most sentiment analysis tools are limited to the extraction of a polarity value associated to the whole document, rather than distinguishing into single aspects. In addition, such methods mainly rely on parts of text in which emotional states are explicitly expressed and, hence, they are unable to capture opinions and sentiments that are expressed implicitly. Because they are mainly based on statistical properties associated with words, in fact, such tools are easily tricked by linguistic operators such as negation and disjunction.

In this work, we presented a novel framework, termed Sentic LDA, which integrates common-sense reasoning in the calculation of word distributions in the LDA algorithm, thus enabling the shift from syntax to semantics in aspect-based sentiment analysis. Sentic LDA goes beyond a mere statistical approach to aspect extraction by leveraging on the semantics associated with words and, hence, highly improves clustering.

References

[1] E. Cambria, "Affective computing and sentiment analysis," *IEEE Intelligent Systems*, vol. 31, no. 2, pp. 102–107, 2016.

[2] E. Cambria, B. Schuller, B. Liu, H. Wang, and C. Havasi, "Statistical approaches to concept-level sentiment analysis," *IEEE Intelligent Systems*, vol. 28, no. 3, pp. 6–9, 2013.

[3] E. Cambria, A. Hussain, C. Havasi, and C. Eckl, "Common sense computing: From the society of mind to digital intuition and beyond," in *Biometric ID Management and Multimodal Communication*, ser. Lecture Notes in Computer Science, J. Fierrez, J. Ortega, A. Esposito, A. Drygajlo, and M. Faundez-Zanuy, Eds. Berlin Heidelberg: Springer, 2009, vol. 5707, pp. 252–259.

[4] J. Petterson, W. Buntine, S. M. Narayanamurthy, T. S. Caetano, and A. J. Smola, "Word features for latent dirichlet allocation," in *Advances in Neural Information Processing Systems*, 2010, pp. 1921–1929.

[5] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2004, pp. 168–177.

[6] A.-M. Popescu and O. Etzioni, "Extracting product features and opinions from reviews," in *Proceedings of EMNLP*, 2005, pp. 3–28.

[7] S. Blair-Goldensohn, K. Hannan, R. McDonald, T. Neylon, G. A. Reis, and J. Reynar, "Building a sentiment summarizer for local service reviews," in *Proceedings of WWW-2008 workshop on NLP in the Information Explosion Era*, 2008, p. 14.

[8] C. Scaffidi, K. Bierhoff, E. Chang, M. Felker, H. Ng, and C. Jin, "Red opal: product-feature scoring from reviews," in *Proceedings of the 8th ACM conference on Electronic commerce*. ACM, 2007, pp. 182–191.

[9] Y. Hu, J. Boyd-Graber, B. Satinoff, and A. Smith, "Interactive topic modeling," *Machine learning*, vol. 95, no. 3, pp. 423–469, 2014.

[10] Z. Chen and B. Liu, "Mining topics in documents: standing on the shoulders of big data," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 1116–1125.

[11] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999, pp. 50–57.

[12] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[13] I. Titov and R. McDonald, "Modeling online reviews with multi-grain topic models," in *Proceedings of the 17th international conference on World Wide Web*. ACM, 2008, pp. 111–120.

[14] S. Branavan, H. Chen, J. Eisenstein, and R. Barzilay, "Learning document-level semantic properties from free-text annotations," *Journal of Artificial Intelligence Research*, vol. 34, no. 2, p. 569, 2009.

[15] H. Wang, Y. Lu, and C. Zhai, "Latent aspect rating analysis on review text data: a rating regression approach," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 783–792.

[16] Y. Lu, C. Zhai, and N. Sundaresan, "Rated aspect summarization of short comments," in *Proceedings of the 18th international conference on World wide web*. ACM, 2009, pp. 131–140.

[17] W. X. Zhao, J. Jiang, H. Yan, and X. Li, "Jointly modeling aspects and opinions with a maxent-lda hybrid," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2010, pp. 56–65.

[18] J. D. Mcauliffe and D. M. Blei, "Supervised topic models," in *Advances in neural information processing systems*, 2008, pp. 121–128.

[19] Y. Lu and C. Zhai, "Opinion integration through semi-supervised topic modeling," in *Proceedings of the 17th international conference on World Wide Web*. ACM, 2008, pp. 121–130.

[20] D. Andrzejewski, X. Zhu, and M. Craven, "Incorporating domain knowledge into topic modeling via dirichlet forest priors," in *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009, pp. 25–32.

[21] T. Wang, Y. Cai, H.-f. Leung, R. Y. Lau, Q. Li, and H. Min, "Product aspect extraction supervised with online domain knowledge," *Knowledge-Based Systems*, vol. 71, pp. 86–100, 2014.

[22] J. Jagarlamudi, H. Daumé III, and R. Udupa, "Incorporating lexical priors into topic models," in *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2012, pp. 204–213.

[23] A. Mukherjee and B. Liu, "Aspect extraction through semi-supervised modeling," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, 2012, pp. 339–348.

[24] E. Cambria, A. Hussain, C. Havasi, and C. Eckl, "Sentic computing: Exploitation of common sense for the development of emotion-sensitive systems," in *Development of Multimodal Interfaces: Active Listening and Synchrony*, ser. Lecture Notes in Computer Science, A. Esposito, N. Campbell, C. Vogel, A. Hussain, and A. Nijholt, Eds. Berlin: Springer, 2010, pp. 148–156.

[25] E. Cambria, D. Olsher, and D. Rajagopal, "SenticNet 3: A common and common-sense knowledge base for cognition-driven sentiment analysis," in *AAAI*, Quebec City, 2014, pp. 1515–1521.

[26] S. Poria, A. Gelbukh, E. Cambria, P. Yang, A. Hussain, and T. Durrani, "Merging senticnet and wordnet-affect emotion lists for sentiment analysis," in *Signal Processing (ICSP), 2012 IEEE 11th International Conference on*, vol. 2. IEEE, 2012, pp. 1251–1255.

[27] S. Poria, A. Gelbukh, E. Cambria, D. Das, and S. Bandyopadhyay, "Enriching SenticNet polarity scores through semi-supervised fuzzy clustering," in *IEEE ICDM*, Brussels, 2012, pp. 709–716.

[28] E. Cambria and A. Hussain, *Sentic Computing: A Common-Sense-Based Framework for Concept-Level Sentiment Analysis*. Cham, Switzerland: Springer, 2015.

[29] E. Cambria, J. Fu, F. Bisio, and S. Poria, "AffectiveSpace 2: Enabling affective intuition for concept-level sentiment analysis," in *AAAI*, Austin, 2015, pp. 508–514.

[30] S. Poria, E. Cambria, A. Gelbukh, F. Bisio, and A. Hussain, "Sentiment data flow analysis by means of dynamic linguistic patterns," *IEEE Computational Intelligence Magazine*, vol. 10, no. 4, pp. 26–36, 2015.

[31] S. Poria, E. Cambria, L.-W. Ku, C. Gui, and A. Gelbukh, "A rule-based approach to aspect extraction from product reviews," *SocialNLP 2014*, p. 28, 2014.

[32] S. Kiritchenko, X. Zhu, C. Cherry, and S. M. Mohammad, "Nrc-canada-2014: Detecting aspects and sentiment in customer reviews," *SemEval 2014*, p. 437, 2014.