applied sciences

MDPI

Article

Topic Word Embedding-Based Methods for Automatically Extracting Main Aspects from Product Reviews

Sang-Min Park [1,†], Sung Joon Lee [2,‡] and Byung-Won On [2,*,‡]

1 AI Labs, Saltlux Inc., Gangnam-gu, Seoul 06147, Korea; smpark@saltlux.com
2 Department of Software Convergence Engineering, Kunsan National University, Gunsan, Jeollabuk-do 54150, Korea; 1501386@kunsan.ac.kr
* Correspondence: bwon@kunsan.ac.kr; Tel.: +82-63-469-8913
† Current address: Daewoong Building, 538 Eonju-ro, Gangnam-gu, Seoul 06147, Korea.
‡ Current address: 558 Daehak-ro, Gunsan-si, Jeollabuk–do 54150, Korea.

Received: 16 March 2020; Accepted: 26 May 2020; Published: 31 May 2020

Abstract: Detecting the main aspects of a particular product from a collection of review documents is so challenging in real applications. To address this problem, we focus on utilizing existing topic models that can briefly summarize large text documents. Unlike existing approaches that are limited because of modifying any topic model or using seed opinion words as prior knowledge, we propose a novel approach of (1) identifying starting points for learning, (2) cleaning dirty topic results through word embedding and unsupervised clustering, and (3) automatically generating right aspects using topic and head word embedding. Experimental results show that the proposed methods create more clean topics, improving about 25% of Rouge–1, compared to the baseline method. In addition, through the proposed three methods, the main aspects suitable for given data are detected automatically.

Keywords: word embedding; aspect detection; opinion summarization

1. Introduction

Opinion Summarization is one of the most important problems in sentiment analysis and opinion mining areas currently. Given a large collection of review text documents about a particular product in automobiles or smartphones as input, state–of–the–art opinion summarization methods usually provide (1) a few main aspects, (2) pros and cons per aspect, and (3) extractive or abstractive summary in favor of/against each aspect.

In this article, an aspect is formally defined as *a focused topic* about which a majority of online consumers mainly talk in the particular product. For instance, according to domain experts, *design*, *function*, *performance*, *price*, *quality*, and *service* are the main aspects in automobiles [1]. Regardless of manual or automatic approaches, if we can precisely mine key aspects from a large corpus, we can easily measure pros and cons ratios per aspect. Nowadays, since data–based market research has become popular, the aspect detection problem has been studied actively.

To address this challenging problem of *automatically* detecting a few main aspects from a large corpus, association rule mining–based, lexicon–based, syntactic–based, and machine learning–based approaches are presented as solutions [2]. However, despite active research on the aspect detection problem, it is still not trivial to find the right aspects for each domain. To make matters worse, major aspects are different for each domain and differ slightly by different consumer groups even on the same domain. In addition, today demand for handling large–scale text data and the existence of lexical semantics (ambiguous words) make this problem even harder. In particular, to surmount the

challenges for which conventional solutions against this problem no longer work, we propose a novel automatic method of mining main aspects through topic analysis.

Given a collection of text documents, a probabilistic topic model extracts major latent topics that are mainly spoken in the collection. A topic θ_i consists of relevant words, where each relevant word w has its probability value $P(w)$ that indicates its importance (weight) of w within θ_i. For example, θ_i is represented as a set of relevant words like {bank:0.5, saving: 0.35, interest: 0.11, loan: 0.04}. Probabilistic Latent Semantic Analysis (pLSA) or Latent Dirichlet Allocation (LDA), well–known as topic models, extracts topics from large text documents by the number of topics (k) entered by a domain expert. Then, he/she manually determines the label (title) of each topic $L(\theta_1), \ldots,$ and $L(\theta_k)$ by looking at the relevant words and taking advantage of domain knowledge. For example, $L(\theta_i)$ is very likely to be the word *finance* as the label if he/she guesses roughly with the topic words like 'bank', 'saving', 'interest', and 'loan'. In other words, *finance* includes the comprehensive meaning of all words in θ_i.

To mine aspects from large text documents, Figure 1 presents the overall system of the proposed approach, where we first extract accurate topics that are mostly covered in the collection and then generate important aspects through clean topics. To acquire high–quality topics, we propose novel two approaches that improve the shortcomings of the existing topic models.

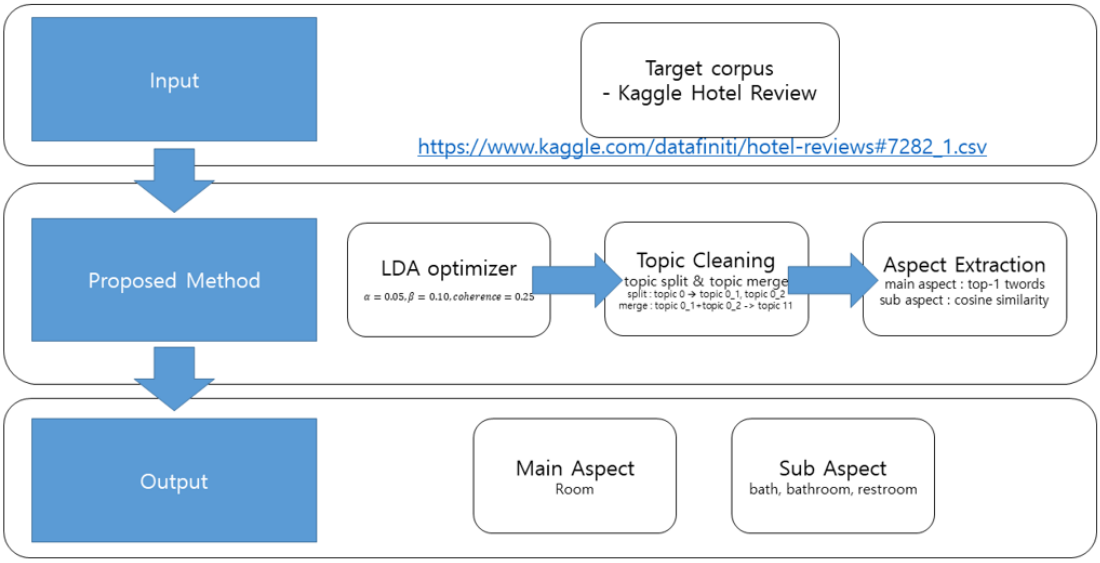

Figure 1. The overall system of the proposed approach.

- Firstly, we propose a new algorithm of estimating proper priors for a given topic model. The Dirichlet priors that yield the strong association of all words in a topic (topic coherence) are chosen while the topic model with different prior values is executing iteratively. In particular, we propose the word embedding–based measure to compute the average topic coherence value. We also propose a new method of automatically identifying the number of true topics in advance, where the number of true topics means the number of topics originally in a given data that is unknown. Empirically, the number of topics that yields the highest topic coherence value is chosen while the topic model with different numbers of topics is executing iteratively.
- Secondly, we propose a novel Topic Cleaning (TC) method that correct raw topics output by the existing topic model through topic splitting and merging processes. To split and merge incorrect topics, we propose a new post–processing method based on the word embedding and unsupervised clustering techniques in which we consider word2vec [3], GloVe [4], fastText [5], ELMo [6], and BERT [7] as the word embedding model and Density–based Spatial Clustering of Applications with Noise (DBSCAN) [8] as the unsupervised clustering method.

Next, we propose three automatic methods of automatically labelling clean topics after topic splitting and merging. Each topic is given a label (title) of the topic that represents the overall meaning of the topic as a single keyword. It is also assigned sub–labels that assist the semantics of the topic. Our proposed methods are based on (1) topic words, (2) head words obtained by dependency parsing of the sentences related to a given topic, or (3) a hybrid of topic and head words.

To the best of our knowledge, the proposed approach is the first study to (1) develop word embedding–based topic coherence measure, (2) to correct dirty topics generated by an existing topic model using the word embedding and unsupervised clustering techniques, and (3) to generate main and sub aspects based on clean topics and dependency parsing of topic sentences. In addition, to find top–k sentences relevant with a given topic, we propose a new probabilistic model based on Maximum Likelihood Estimation (MLE). Our experimental results show that the proposed approach outperforms LDA as the baseline method.

The remainder of this article is organized as follows—in Section 2, we introduce both background knowledge and existing methods related to this work. In particular, we discuss the novelty of our method, in addition to main differences between previous studies and our work. In Section 3, we describe the details of the proposed method and explain/analyze experimental set–up and results in Section 4. We also discuss how the results can be interpreted in perspective of previous studies and limitations of the work in Section 5. Finally, we summarize our work and future direction in Section 6.

2. Background

In this section, we discuss the problem motivation and recent studies related to the proposed method.

2.1. Motivation

The proposed method deals with three specific sub problems—(1) Automatic and empirical prior estimation problem, (2) topic cleaning problem, and (3) automatic topic labelling problem. The final aim is to automatically find main aspects based on the solutions that handle such three problems. In next subsection, we will describe the detailed things of each problem.

2.1.1. Automatic Prior Knowledge Estimation

Since topic models use the Bayesian probability model, the prior probabilities α and β should be given as the input parameter. Here, α is the distribution of topics in a document and β is the distribution of words in a topic.

Previously, the default values have been widely used as the priors or arbitrarily determined by domain experts. However, such approaches have the following disadvantages.

- First, the more accurate the priors for a given data is, the faster the convergence is. However, we cannot expect this benefit if we use the existing methods.
- Second, it is hard to manually find the exact priors from a large–sized data set.
- Third, the topic model with inaccurate priors is likely to extract dirty topics, which prevent us from understanding the nature of the given data set.

As such, it is significant to grasp proper priors in advance given a data set, in order to obtain good topic results. According to a recent study in Reference [9], there exists prior knowledge suitable for a domain and the accuracy of topic models will be greatly improved if the domain–optimized prior knowledge is used in the initial step.

To optimize α and β, some traditional topic models use non–parametric hierarchical prior distribution. The non–parametric hLDA is based on the Chinese Process to detect the number of clusters. However, as the main disadvantage, it is considerably affected by larger clusters. In addition, note that text data are usually ambiguous and noise big data so that the mathematically–based model cannot be applied simply. In practice, to handle such noise data, the most realistic approach is to use

an empirical method that automatically determine the priors by considering all combinations of α and β values, which is not covered in the existing studies.

2.1.2. Topic Cleaning

Figure 2 shows an example of topics θ_1, θ_2, ..., and θ_k. The labels of θ_1, θ_2, ..., and θ_k are words 'service', 'clean', ..., and 'price'. Each topic is represented as the probability distribution of words relevant with the topic label. The topic words such as 'room' and 'bellboy' are relevant with the topic label 'service'. The probability value of the word 'room' is $P(w = \text{'room'}|\theta_1) = 0.02$ and that of the word 'bellboy' is $P(w = \text{'bellboy'}|\theta_1) = 0.01$. The sum of the probability values of all words in θ_1 is 1. This is $\Sigma_{w \in V} P(w|\theta_{i=1,\ldots, \text{or } k}) = 1$, where V is a set of vocabularies (unique words) in the collection of text documents. In addition, as the weight of θ_i is π_i, the sum of the weights of all topics should be 1, i.e., $\Sigma_{i=1}^{k} \pi_i = 1$.

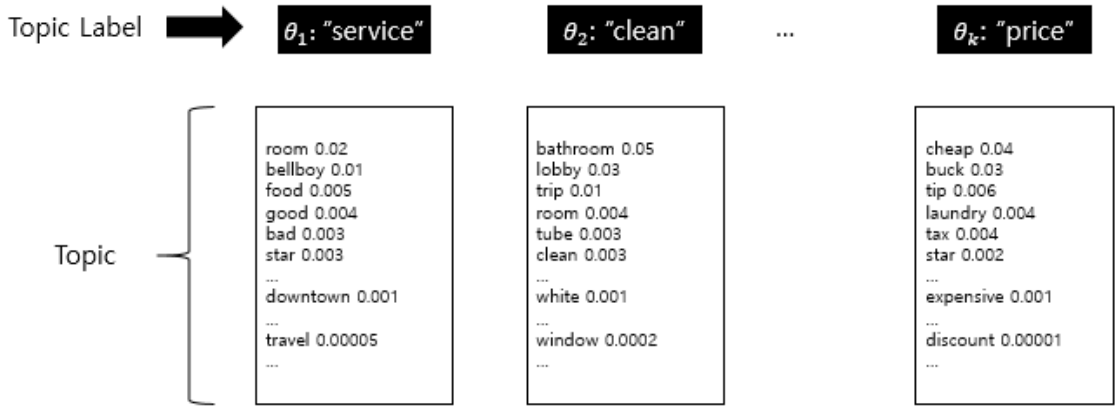

Figure 2. An example of topics through Latent Dirichlet Allocation (LDA).

If we grasp the correct topics using any topic model, we can easily infer main aspects through the topic labels. Unfortunately, compared to topic results summarized by human evaluators who go over all text documents in a collection, pLSA or LDA has mostly failed to show satisfactory results in real applications. The main reasons are as follows:

1. Often, the results of topic models are dirty. A topic is mixed with two or more semantic groups of topic words with different meanings. To better understand topic results, this dirty topic should be split to two or more correct topics, each of which consists of only words representing the unified meaning, as shown in Table 1.
2. There exist redundant topics with similar meaning in topic result. These duplicate topics should be merged to a single topic, as shown in Table 2. After topic splitting and merging, the topic results will be more accurate and better understood. Besides, meaningless topics in which most words have stop words (e.g., 'a', 'the', 'in', etc.) should be removed from the topic result.

2.1.3. Automatic Topic Labelling

In topic result, a domain expert manually labels each topic by looking at the topic words and taking advantage of domain knowledge. In general, this task of manually labelling topics is both labor–intensive and time–consuming. It is also hard to know the exact meaning of a topic by observing only words in the topic. If topic words are ambiguous or the semantics among topic words are significantly variant, there is a high likelihood of incorrect interpretation by reflecting the expert's subjective opinion. Through the existing topic model, we can simply understand the superficial tendency hidden in review documents. For these reasons, an automatic yet accurate topic labelling method is required for practical use.

Table 1. An example of the topic split problem—$L(\theta_{i,j})$ = {performance, design}.

Topic Label	Topic Word w	Probability $P(w)$	Split Topic Labels
$L(\theta_{i,j})$ = {performance, design}	break	0.0026	$L(\theta_i)$ = {performance}
$L(\theta_{i,j})$ = {performance, design}	horsepower	0.0008	$L(\theta_i)$ = {performance}
$L(\theta_{i,j})$ = {performance, design}	noise	0.0007	$L(\theta_i)$ = {performance}
$L(\theta_{i,j})$ = {performance, design}	inside	0.0006	$L(\theta_j)$ = {design}
$L(\theta_{i,j})$ = {performance, design}	cornering	0.0003	$L(\theta_i)$ = {performance}
$L(\theta_{i,j})$ = {performance, design}	fuel efficiency	0.0003	$L(\theta_i)$ = {performance}
$L(\theta_{i,j})$ = {performance, design}	exterior	0.0003	$L(\theta_j)$ = {design}
$L(\theta_{i,j})$ = {performance, design}	refined	0.0003	$L(\theta_j)$ = {design}
$L(\theta_{i,j})$ = {performance, design}	clean	0.0003	$L(\theta_j)$ = {design}
$L(\theta_{i,j})$ = {performance, design}	nimble	0.0003	$L(\theta_j)$ = {design}

Table 2. An example of the topic merge problem—$L(\theta_{\{i\}})$ = {performance} and $L(\theta_{\{j\}})$ = {performance}.

Topic Label	Topic Word w	Probability $P(w)$	Merged Topic Label
$L(\theta_i)$ = {performance}	engine	0.0091	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_i)$ = {performance}	motor	0.0041	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_i)$ = {performance}	brake	0.0026	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_i)$ = {performance}	base	0.0009	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_i)$ = {performance}	interlocking	0.0006	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_i)$ = {performance}	smart	0.0006	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_i)$ = {performance}	speaker	0.0006	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_i)$ = {performance}	aerodynamic	0.0006	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_i)$ = {performance}	exchange	0.0003	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_i)$ = {performance}	switch	0.0003	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_j)$ = {performance}	turbo	0.0045	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_j)$ = {performance}	engine	0.0029	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_j)$ = {performance}	pure	0.0011	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_j)$ = {performance}	state	0.0008	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_j)$ = {performance}	exchange	0.0006	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_j)$ = {performance}	occur	0.0006	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_j)$ = {performance}	unrest	0.0006	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_j)$ = {performance}	engine sound	0.0003	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_j)$ = {performance}	humidity	0.0003	$L(\theta_{\{i,j\}})$ = {performance}
$L(\theta_j)$ = {performance}	disappointed	0.0003	$L(\theta_{\{i,j\}})$ = {performance}

2.2. Related Work

The most similar existing problem is the aspect–based opinion summarization that has been actively studied in the field of sentiment analysis since 2016. However, the existing aspect extraction problems are radically different from our problem in this article. The existing problems are represented as multi–classification of a candidate word to one of aspects, where an aspect is not **a focused topic mainly talked in a large collection** but some feature that describes any product in part (e.g., in the sentence "the battery of Samsung Galaxy 9 is long", the word 'battery' is one of features).

The existing aspect extraction problem is largely catogorized to two different problems of extracting *explicit* and *implicit* aspects from review documents. A recent study [10] introduces the explicit aspect extraction problem, where aspect words are extracted and then clustered [11–14], and the implicit aspect extraction problem, where aspects that are not explicitly mentioned in text strings are assigned to one of the pre-defined aspects [15,16].

Extracting explicit aspects is relatively easier than implicit aspects. To address the explicit aspect problem, the most popular approach is based on the association rule mining technique that finds the frequent noun and noun phrase set in text documents [17]. Furthermore, based on the frequent-based method, Popescu et al. used Point-wise Mutual Information (PMI) [18], while Ku et al. used the

TF/IDF method [19], to find main aspects from candidate noun phrases. Besides, Moghaddam and Ester removed noise terms by using additional pattern rules in addition to the frequency-based aspect extraction method [20]. Even though these approaches are simple and effective, they are limited because it is difficult to find low-frequency aspects and implicit aspects. Ding et al. presented a holistic lexicon-based approach by which low-frequency aspects can be detected. In their method, the authors first exploited opinion bearing words from other sentences and reviews and then found a noun closest to the opinion bearing words [21]. Rather than using the opinion lexicon like Ding et al.'s method, Turney first proposed the syntactic–based approach based on POS tags [22] and Zhuang et al. used the dependency parsing to find all possible aspect candidates from review documents [23]. In recent years, learning–based approaches have been widely used. Zhai et al. proposed a semi–supervised learning method based on Expectation–Maximization (EM) with sharing words (e.g., battery, battery life, and battery power) and lexical similarity based on WordNet to label the aspect of the extracted word [24]. In addition, Lu et al. proposed a multi–aspect sentence labelling method in which each sentence in a review document is labelled to an aspect. The authors modified the existing topic model with minimal seed words to encourage a correspondence between topics and aspects, and then predict overall multi–aspect rating scores [25]. To analyze people's opinions and sentiments toward certain aspects, References [26–28] attempted to learn emotion and knowledge information based on a sentic Long Short-Term Memory (LSTM)–based model that extends LSTM with a hierarchical attention mechanism. Unlike their approaches, the proposed method focuses on finding main aspects automatically using clean topics. To improve the performance of text classification, Reference [29] proposed a capsule network using dynamic routing that can learn hierarchically. Because the capsule network integrates multiple sub layers layer, text classification is improved from single label text classification to multi–label one.

By and large, most recent aspect extraction methods are grouped to three categories: (1) Unsupervised methods, (2) Supervised methods, and (3) Semi-supervised methods. The unsupervised methods are based on Dependency Parsing [30], Association Rule Mining [31], Mutual/Association [32], Hierarchy [33], Ontology [34], Generative Model [35,36], Co-occurrence [37], Rule-based [38], and Clustering [39]. The supervised methods are based on Association Rule Mining [40], Hierarchy with Co-occurrence [41], Co-occurrence [42], Classification [43], Fuzzy Rule [44], Rule-based [45], and Conditional Random Fields [46]. The semi-supervised methods are based on Association Rule Mining [47], Clustering [48], and Topic Model [49].

Rather than traditional topic models, Reference [50] used only word2vec to find topics. All vocabularies are mapped to word embedding spaces and clustered to topics, each of which is a group of relevant words in the corpus. In contrast, the proposed method basically uses some traditional topic model because the traditional topic models are likely to work well to obtain brief summary from large documents. In the approach, a topic model is first applied to find topics and then main aspects are exploited from the clean topics. To clean the topics, various word embedding methods such as word2vec, fastText, GloVe, ELMo, and BERT are used. Reference [51] extracted some topics from the embedding space using only word–level embedding, while the proposed method takes advantage of both word–level and sentence–level embedding to detect main topics from a given large corpus.

All existing approaches based on any topic model [25,35,36,49] are quite different from ours in that the Bayesian network of the original topic model is modified and prior knowledge such as pre–defined aspects [25], explicit aspects [35], opinion bearing words [36], aspect dependent sentiment lexicon [52], and cannot/must–links [49] are used. Alternatively, we do not change any existing topic model directly and do not use any prior knowledge like opinion words. In addition, to enforce a direct correspondence between aspects and topics, we propose optimal starting points for learning, topic cleaning, and aspect labelling based on topic structures.

3. The Proposed Approaches

Our proposed method consists of two steps. The first step is to extract raw topics from a large collection of text documents through a given topic model like pLSA or LDA and the second step is to automatically generate each topic label that is one of main aspects in our context. If we use one of the existing topic models, the topic result is poor. As we already pointed out in the introduction section, one of the reasons is that the existing models are likely to fall into local optima because (unknown domain–optimized) prior knowledge such as Dirichlet priors and the number of topics are not used in the initial step. Another reason is that raw topics are incomplete and even dirty because of the topic split and merge problems.

The goal of the proposed method in the first step is to (1) identify the best priors for a given topic model; (2) execute the topic model with the priors; and (3) clean raw topics by removing meaningless topics, splitting ambiguous topics, and merging redundant topics with similar meaning. Then, the second step of the proposed method aims at automatically generating topic labels, each of which is an aspect in the collection. In next sections, we will describe the detailed algorithms for each proposed method.

3.1. Automatic Identification of Domain-Optimized Priors and Number of True Topics

To address the problem of automatically identifying initial priors suitable to a given domain, we focus on an empirical approach instead of mathematical optimization solutions. Dirichlet priors (α and β) and the number of topics are the priors for a given topic model. α and β adjust the distribution of topics per document and the probability distribution of words per topic, respectively. If α increases, the number of topics per document increases and if β increases, the probability values of a few words within the topic increase. In existing topic models, the prior probabilities such as α and β are determined either by default ($\alpha = \frac{50}{\text{\# of topics}}$ and $\beta = 0.1$) or arbitrarily decided by domain experts. However, we propose Algorithm 1, where the average topic coherence is measured while increasing the values of α and β by 0.05, and the values of α and β which maximize the average topic coherence value are returned. To measure the topic coherence value (corresponding to Function *MeasureTopicCoherence*() in Line 11), to the best of our knowledge, we propose a new topic coherence measure based on word embedding for the first time. Given a topic $\theta_i = \{w_1, w_2, w_3\}$, where w_i is a word, *MeasureTopicCoherence*() (1) converts each word $w_i \in \theta_i$ to p–dimensional word embedding vector v_{w_i} through a domain–customized word embedding model (f_{we}) which is one of word2vec, GloVe, fastText, ELMo, and BERT; (2) creates all possible combinations of vector pairs—e.g., $S = \{(v_{w_1}, v_{w_2}), (v_{w_1}, v_{w_3}), (v_{w_2}, v_{w_3})\}$; (3) computes $sim(v_{w_i}, v_{w_j})$, the similarity of each pair through $\frac{v_{w_i} \cdot v_{w_j}}{||v_{w_i}||||v_{w_j}||}$; and (4) returns the topic coherence value $\frac{\sum_{(v_{w_i}, v_{w_j}) \in S} sim(v_{w_i}, v_{w_j})}{|S|}$, where $|S|$ is the number of pairs in S.

Similarly, the optimal number of topics is computed using the above *MeasureTopicCoherence*() function. A topic model is individually executed with each different number of topics. Next, the proposed method computes the average topic coherence value from the topic results and then finds the number of topics with the maximum topic coherence value.

3.2. Cleaning Dirty Topics

In our point of view, the proposed method is based on the hypothesis that one topic forms one semantic cluster in the word embedding space and a dirty topic is a mixture of multiple topics, so it contains multiple semantic clusters. In the pre-processing step, we use f_{we} by learning simple neural networks like Continuous Bag of Words (CBOW) and Skip–gram from a domain given as input. Suppose that the topic result is θ_0 and other topics after a topic model is executed. In addition, $\theta_0 = \{w_1, w_2, w_3, w_4, w_5\}$, where w_i is a word. For $w_i \in \theta_0$, $f_{we}(w_i) = v_{w_i}$, where v_{w_i} is the p–dimensional word embedding vector of w_i. In this way, the vectors corresponding to all words in θ_0 like v_{w_1}, v_{w_2}, v_{w_3}, v_{w_4}, and v_{w_5} are created and then projected to the p–dimensional word embedding vector space.

Similar vectors will be located close to the vector space, while disjoint vectors will be far away from the vector space.

Algorithm 1: The empirical approach for identifying best priors

Input: A collection of text documents C and Number of topics m;
Output: Optimal Dirichlet priors α and β;
for $(\alpha = 0; \alpha \leq \frac{50}{m}; \alpha += 0.05)$ **do**
  $T = \phi$;
  for $(\beta = 0; \beta \leq 1; \beta += 0.05)$ **do**
    // TopicModel: LDA;
    $T = \text{TopicModel}(C, m, \alpha, \beta)$;
    $TC[m] = 0$;
    $TotalTopicCoherence = 0$;
    for $(i = 0; i < m; i++)$ **do**
      $TC[i] = MeasureTopicCoherence(T)$;
      $TotalTopicCoherence += TC[i]$;
    end
    $AvgTopicCoherence = \frac{TotalTopicCoherence}{m}$;
  end
end
return $argmax_{AvgTopicCoherence}(\alpha, \beta)$;

Next, the proposed method clusters the word embedding vectors using the unsupervised clustering method. In this problem, the unsupervised clustering method should be used because the number of true clusters is not known in advance. In the experiment, we used DBSCAN as the unsupervised clustering method. If a remaining cluster mostly has stop words or does not represent the united meaning (Line 14 in Algorithm 2), it is removed in the topic result. On the one hand, if there is one giant cluster and several small clusters in the clustering result (Line 16 in Algorithm 2), the main stream of θ_0 is a giant cluster and the remaining clusters are noise. Therefore, the topic words in the remaining small clusters except the giant cluster are removed in θ_0. For example, let us assume that the clustering result is $\{w_1, w_2, w_3, w_5\}$ and $\{w_4\}$. In this case, the original θ_0 is changed to the clean $\theta_0 = \{w_1, w_2, w_3, w_5\}$. On the other hand, if the clustering result is divided into approximately half (Line 18 in Algorithm 2), it is a topic split problem. That is, this means that there are two clusters with different meanings within a single topic. After the split process is completed, the merge process is performed for each pair of split topics (Line 22 to 27 in Algorithm 2).

3.3. Automatic Generation of Main Aspects from Clean Topics

To mine main aspects in the collection, we view each of clean topics to an aspect. The next problem is how automatically but yet accurately we can generate an aspect (topic label/title) corresponding to each topic. Please note that labelling each topic is not difficult because the topic is clean (or has a unified meaning) through Algorithm 2. Based on topic information that we can only use, we propose three new methods for detecting main aspects.

Algorithm 2: The proposed topic cleaning method

Input: A collection of text documents C, priors α^*, β^*, and m^*;
Output: CleanTopics;
$T = \phi$; //T is a list of topics generated by a topic model;
T = TopicModel(C, α, β, m^*); //$\mathbb{V}$ is the space of the word embedding vectors of all topic words;
$\mathbb{V} = \phi$;
for ($\theta_j \in T$) **do**
  for ($w_i \in \theta_j$) **do**
    $\mathbb{V} = f_{we}(w_i)$;
    $\mathbb{V} = \mathbb{V} \cup \{f_{we}(w_i)\}$;
  end
end
Clusters = ϕ; //Clusters is a list of clusters clustered by DBSCAN;
Clusters = DBSCAN($\mathbb{V}$, ϵ, $minPts$);
CleanTopics = ϕ; //CleanTopics is a list of clean topics;
for ($\theta_{\{i,j\}}$ ∈Clusters) **do**
  if Ratio of stop words among topic words in $\theta_{\{i,j\}} \geq 0.9$ **then**
    Remove $\theta_{\{i,j\}}$;
  end
  else if $|\theta_i| >> |\theta_j|$ in $\theta_{\{i,j\}}$ **then**
    CleanTopics = CleanTopics $\cup$ $\{\theta_i\}$;
  end
  else if $|\theta_i| \sim |\theta_j|$ in $\theta_{\{i,j\}}$ **then**
    // The topic $\theta_{\{i,j\}}$ should be split to θ_i and θ_j;
    $\theta_i \longleftarrow \theta_{\{i,j\}}$; $\theta_j \longleftarrow \theta_{\{i,j\}}$;
    CleanTopics = CleanTopics $\cup$ $\{\theta_i, \theta_j\}$;
  end
end
// Topics θ_i and θ_j should be merged to $\theta_{\{i,j\}}$;
for ($\theta_i \in$ CleanTopics) **do**
  for ($\theta_j \in$ CleanTopics) **do**
    if $\theta_i \sim \theta_j$ **then**
      $\theta_{\{i,j\}} \longleftarrow \theta_i, \theta_j$;
      CleanTopics = CleanTopics $\cup$ $\{\theta_{\{i,j\}}\}$;
    end
  end
end

The first method is based on topic words. As an example, see a topic $\theta_i = \{w_1 : 0.5, w_2 : 0.3, w_3 : 0.2\}$. For each word $w_i \in \theta_i$, $f_{we}(w_i) = v_{w_i}$, where f_{we} is the domain–customized word embedding model. In this way, the vectors of all topic words in θ_i (i.e., v_{w_1}, v_{w_2}, and v_{w_3}) are created and then projected to the word embedding vector space. Then, a new vector v_* that are the most relevant with v_{w_1}, v_{w_2}, and v_{w_3} is computed by $\frac{v_{w_1}+v_{w_2}+v_{w_3}}{|\theta_i|}$, where $|\theta_i|$ is the number of vectors in θ_i and $+$ is the element–wise addition between vectors. Finally, a vocabulary w_* close to v_* is chosen in the domain–customized word embedding model, where w_* is considered as an aspect explaining θ_i to the best. In Line 6 of Algorithm 3, $f_{we}^{-1}(v_*)$ is the inverse of $f_{we}(w_*)$. The reason why we use the inverse function is that w_* is the closest to v_* when the value of $f_{we}(w_*)$ is the smallest.

Algorithm 3: The first aspect auto–generation method

Input: A topic $\theta_i = \{w_1, \ldots, w_l\}$;
Output: A topic label w_*;
for ($w_i \in \theta_i$) **do**
  $f_{we}(w_i) = v_{w_i}$;
end
$v_* = \frac{v_{w_1}+\ldots+v_{w_l}}{|\theta_i|}$;
$w_* = f_{we}^{-1}(v_*)$;

The second method is based on head words extracted by natural language processing techniques. In the pre-processing step, all text documents in a collection are segmented to a set of sentences. Now, given a topic $\theta_i = \{w_1 : 0.5, w_2 : 0.3, w_3 : 0.2\}$, the goal of the second method is to find the most relevant top–k sentences. Given θ_i and a sentence s_i, the probability value that s_i is relevant with θ_i is first computed by Equation (7) and then only top–k sentences with the highest s^* value are selected from a total of sentences in the collection.

$$s^* = argmax_{s_i} P(s_i|\theta_i) \tag{1}$$

$$= argmax_{s_i} \frac{P(\theta_i|s_i)P(s_i)}{P(\theta_i)} \tag{2}$$

$$= argmax_{s_i} \frac{P(s_i)}{P(\theta_i)} P(\theta_i|s_i) \tag{3}$$

$$= argmax_{s_i} Z(\theta_i) \prod_{w_i \in \theta_i} P(w_i|s_i) \tag{4}$$

$$= argmax_{s_i} Z(\theta_i) \prod_{w_i \in \theta_i} P_{s_i}(w_i|\theta_i) \tag{5}$$

$$= argmax_{s_i} \prod_{w_i \in \theta_i} logP_{s_i}(w_i|\theta_i) \tag{6}$$

$$= argmax_{s_i} \sum_{w_i \in \theta_i} logP_{s_i}(w_i|\theta_i), \tag{7}$$

where $P_{s_i}(w_i|\theta_i) = \frac{P(w_i)}{\sum_{w \in \theta_i \cap tokens(s_i)} P(w)}$, where $P(w_i)$ is a probability of the i–th topic word computed by a topic model and $P(w)$ is one probability of topic words appearing in both the topic θ_i and the i–th sentence s_i.

By Bayes' theorem, Equation (1) is converted to Equation (2). The $\frac{P(s_i)}{P(\theta_i)}$ in Equation (3) is the normalized term so it can replaced by $Z(\theta_i)$. Since $Z(\theta_i)$ is treated as a constant, it can be omitted because it does not significantly affect the results. If we replace $P(\theta_i|s_i)$ in Equation (3) with the term for each word, it is represented as $\prod_{w_i \in \theta_i} P_{s_i}(w_i)$ in Equation (5). Finally, we obtain Equation (7) by taking a log on Equation (5).

Then, for $s_i \in$ top–k sentences $\{s_1, \ldots, s_k\}$, the dependency parsing is carried out to find head words from s_i. In the dependency parsing, there exist governor and dependent words in a given sentence. Let us assume one sentence like "I saw a beautiful flower on the road". In this case, the word 'beautiful' is the dependent and the word 'flower' is the governor. Throughout this article, we call governors head words for convenience. If some head words appear frequently cross top–k sentences, such words can be considered to be key words representing the topic θ_i. Then, such head words w_1^H, $w_2^H, \ldots$ are converted to the vectors $v_{w_1^H}, v_{w_2^H}, \ldots$ through the domain–customized word embedding model. The next step is similar to the first method. v_* is computed by averaging element-wise sum of the vectors. Finally, a vocabulary w_* close to v_* is chosen in the domain-customized word embedding

model, where w_* is considered as an aspect describing θ_i to the best. Algorithm 4 is the pseudo code for the second aspect auto–generation method.

Algorithm 4: The second aspect auto–generation method

Input: A topic $\theta_i = \{w_1 : P(w_1), \ldots, w_l : P(w_l)\}$ and a set of sentences in a collection;
Output: A topic label w_*;
for ($s_i \in$ A set of sentences in the collection) **do**
  $s_i^* = \sum_{w_i \in \theta_i} log(\frac{P(w_i)}{\sum_{w \in \theta_i \cap tokens(s_i)} P(w)})$;
end
HeadWords = ϕ;
for ($s_i \in$ Top–k sentences with the highest s_i^*) **do**
  HeadWords = HeadWords $\cup$ DependencyParser(s_i);
end
for ($w_i \in$ Frequently–used head words) **do**
  $f_{we}(w_i) = v_{w_i}$;
end
$v_* = \frac{v_{w_1} + \ldots + v_{w_l}}{|\theta_i|}$;
$w_* = f_{we}^{-1}(v_*)$;

The third method is a hybrid of the first and second methods. In other words, the third method is working based on the consideration of both word-level approach (from the first method) and sentence-level approach (from the second method). Technically, it selects both top–k topic words from the first method and top–k head words from the second method. Then, the words are converted to the vectors and v_* is computed by averaging element-wise sum of the vectors. Finally, a vocabulary w_* close to v_* is chosen in the domain-customized word embedding model, where w_* is considered as an aspect explaining the topic θ_i to the best. Algorithm 5 is the pseudo code for the third aspect auto–generation method.

Algorithm 5: The third aspect auto-generation method

Input: A topic $\theta_i = \{w_1 : P(w_1), \ldots, w_l : P(w_l)\}$ and a set of sentences in a collection;
Output: A topic label w_*;
WordSet = Top–k topic words from Algorithm 3;
WordSet = WordSet $\cup$ Top–k frequently–used head words from Algorithm 4;
for ($w_i \in$ WordSet) **do**
  $f_{we}(w_i) = v_{w_i}$;
end
$v_* = \frac{v_{w_1} + \ldots + v_{w_{2k}}}{|\theta_i|}$;
$w_* = f_{we}^{-1}(v_*)$;

3.4. Automatic Generation of Sub-Aspects

Given a clean topic θ_i, using one of Algorithms 3–5, the main aspect of θ_i is extracted. Specifically, the top–k topic words within θ_i are translated to the word embedding vectors and then projected to the word embedding space. Since the top–k topic words are relevant one another, the center of all the vectors is a vector (v_*) that represents θ_i semantically well. Then, the main aspect (w_*) of θ_i is the word corresponding to the center vector from the vocabulary dictionary. To find top–k sub–aspects most relevant to w_*, the cosine similarity between w_* and each topic word within θ_i is computed and finally the top–k topic words with the highest cosine similarity values are selected as the sub–aspects.

4. Experimental Validation

In this section, we discuss the results of the proposed methods for the defined three problems.

4.1. Evaluation of Best Priors and Number of True Topics

In the experiment, we used about 10,000 review documents about hotels in US, where the total numbers of words, vocabularies, and sentences are 282,934, 12,777, and 40,819, respectively [53]. As a topic model, we executed Gibbs sampling–based LDA [54] and the numbers of topics and of words per topic are 10 and 50, respectively. Table 3 shows the reason why we selected ten topics as the output of LDA. As the number of topics increases to 3, 5, 7, and 10, we measured the values of Rouge–1 and the numbers of split and merge topics, to see how clean the topic result is. The Rouge–1 is the most well–known metric used to quantify how similar two sets are. In this set–up, for a clean topic θ_i, we first selected the top 100 review documents that are the most relevant with θ_i retrieved by the LDA, and then several human evaluators made a list of important and meaningful keywords after they went over the documents. We denote such a list by X as the solution set to θ_i. We also had another set Y that is a list of topic words in θ_i. Now the Rouge–1 metric is calculated by $\frac{|X \cap Y|}{|X|}$. Y set cleaned by the proposed method is very similar to the solution set if the value of the Rouge–1 is close to 1. In addition, if the numbers of split and merge topics are low in the topic result post-processed by the proposed method, the topic result is more clean than the raw topics generated by the existing LDA. While all Rouge–1 results are little different, the total number of the split and merge topics is small when the number of topics is ten. As a result, we selected 10 as the number of topics in all experiments. Unexpectedly, it turns out that only 1∼2 topics have three or more groups with different meanings when the number of topics is 3 but each topic has two groups with different meanings when the number of topics is 7.

Table 3. The optimal number of topics.

Number of Topics	Rouge–1	Number of Split Topics	Number of Merge Topics	Total
3	0.546	9	0	9
5	0.511	8	1	9
7	0.453	10	2	12
10	0.512	4	1	5

For lexical semantics, we focus on word embedding including word2vec, fastText, GloVe, ELMo, and BERT to surmount the shortcomings of both one-hot encoding and WordNet–based methods. Table 4 supports our claim that using best priors makes topic result to be more clean. In most empirical experiments using LDA, the default $\alpha = \frac{50}{\text{Number of topics} = 10} = 5$ and $\beta = 0.1$. According to our manual investigation, the Rouge–1 results are the worst when we used the default values. In contrast, when ‘$\alpha = 1.5$ and $\beta = 0.08$, the Rouge–1 result is the best, indicating that the topic result is close to the solution set and the fastText is the best among different word embedding models. Please note that Table 4 summarizes the results of the fastText that we selected as the best word embedding model. We can also observe that the word2vec, GloVe, and fastText are the models re–trained with our own dataset show better Rouge–1 results than ELMo and BERT that are not re–trained with our own dataset.

To evaluate Algorithm 1, we measured topic coherence values according to different α and β values. Because of the result in Table 4, we selected fastText as the best word embedding method. Figure 3 clearly shows that the average topic coherence value is about 0.2 when $\alpha = 1.5$ and $\beta = 0.08$. The topic coherence value is the highest in all priors. These results indicate that topic words are closely relevant one another in the particular prior values.

Table 4. The best Dirichlet priors.

	Word Embedding Model	α	β	Rouge–1
Best	word2vec	2.0	0.08	0.553
	fastText	**1.5**	**0.08**	**0.582**
	GloVe	1.0	0.04	0.569
	ELMo	2.0	0.08	0.553
	BERT	1.5	0.16	0.536
Worst	word2vec	5.0	0.2	0.508
	fastText	5.0	0.2	0.508
	GloVe	5.0	0.2	0.508
	ELMo	5.0	0.18	0.494
	BERT	2.0	0.1	0.520

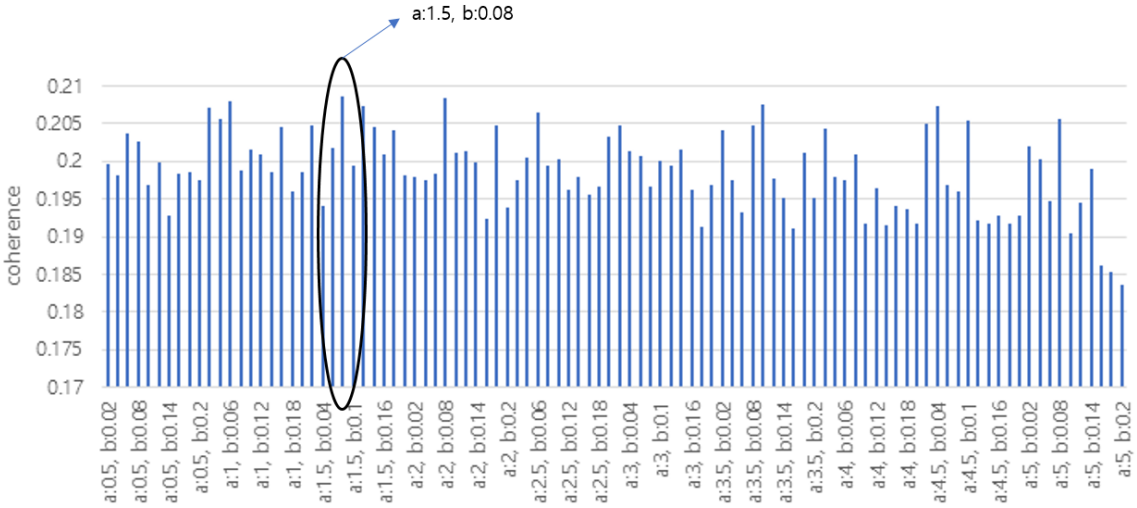

Figure 3. Average topic coherence values of Algorithm 1 according to different α and β values (In the figure, α and β are represented as *a* and *b*).

4.2. Evaluation of Clean Topics

Figure 4 shows the Rouge–1 results of the baseline and proposed methods. Each Rouge–1 result is the average of all topic Rouge–1 values. We can indirectly know that the words in each topic are closely relevant each other and the topic words represent the same meaning. The baseline method generates topics by LDA, while the proposed method (Algorithm 2) cleans incomplete topics generated by LDA. The best case is to use fastText in the proposed method. The Rouge–1 values of the baseline and proposed methods are 0.725 and 0.582. Our proposed method improves up to 25% compared to the baseline method. Furthermore, the proposed method outperforms the baseline method in all cases of using different word embedding models.

Table 5 compares the baseline and proposed methods by counting the number of topics which must be split and merged, and meaningless topics output by each method. For this evaluation, we hired three human evaluators who had nothing to do with our research. After going over the topic results and the review documents, they manually checked to see how many dirty topics are in the output. If the number of the split, merge, and meaningless topics are small, the output by either the baseline method or the proposed method will be more clean. Table 5 clearly shows that the topic results output by the proposed method are more clean than by the baseline method.

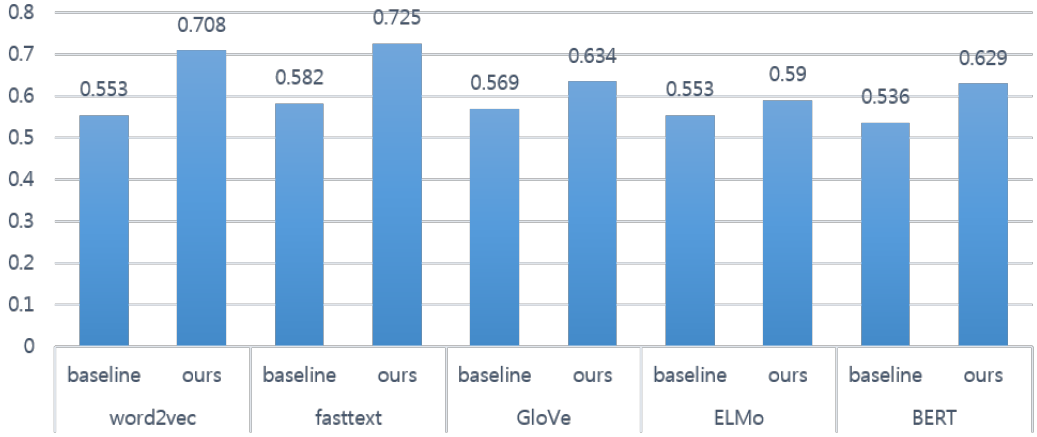

Figure 4. The Rouge–1 results of the baseline and proposed methods.

Table 5. Results of clean topics.

Evaluation	Method	# of Split Topics	# of Merge Topics	# of Meaningless Topics	Total
Human Evaluator 1	Baseline	4	1	0	5
	Ours	1	0	0	1
Human Evaluator 2	Baseline	1	0	0	1
	Ours	0	0	0	0
Human Evaluator 3	Baseline	2	1	0	3
	Ours	1	0	0	1

Figure 5 shows a clustering result of a topic after DBSCAN is performed. This shows a gigantic cluster (in black) and two small clusters (in blue and red). In this figure, we view the words belonging to the small clusters to the noises in the topic. Thus, we remove the small clusters except for the giant cluster. Because of space limitation, we leave out another cases in which there are 2∼3 clusters with different meanings but the numbers of words in those clusters are equal. In this case, the topic is split into 2 or 3 topics.

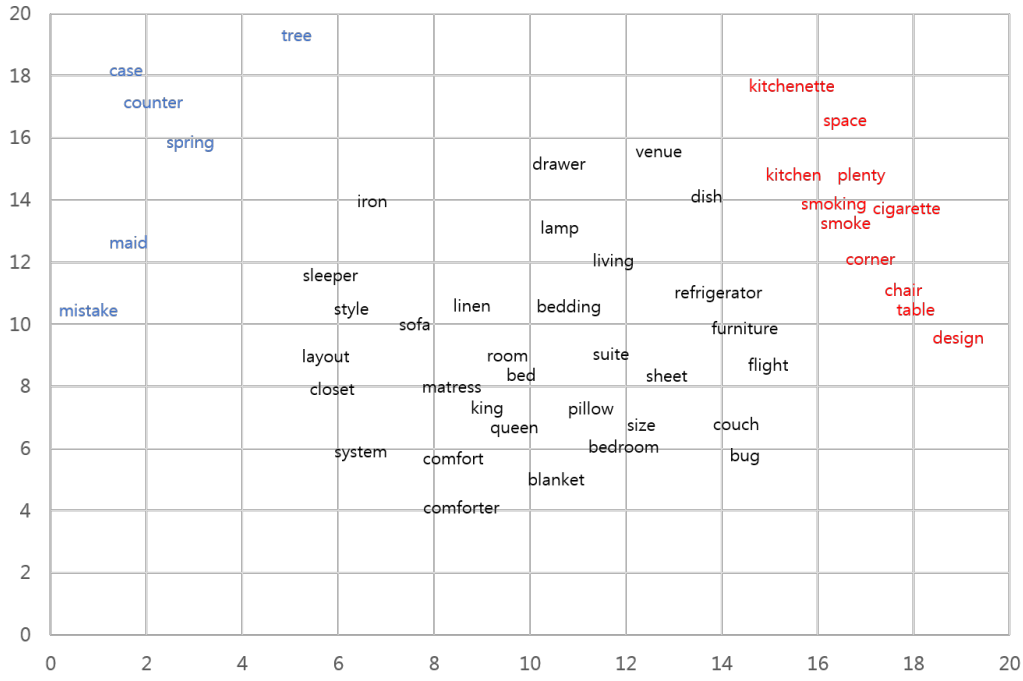

Figure 5. A clustering result of DBSCAN.

Figure 6 depicts a real example of the topic split and merge problems. Through word embedding and DBSCAN, a topic including clusters with different meanings is divided to two or more topics. On the other hand, the small clusters will be combined if it turns out that they have the same meaning at the topic merge stage. If each of the sub topics is meaningless, then it is removed. In a nutshell, we can split and merge raw topics by LDA through word embedding and unsupervised clustering like DBSCAN in order to make us clearly understand the semantics of the topics generated by LDA.

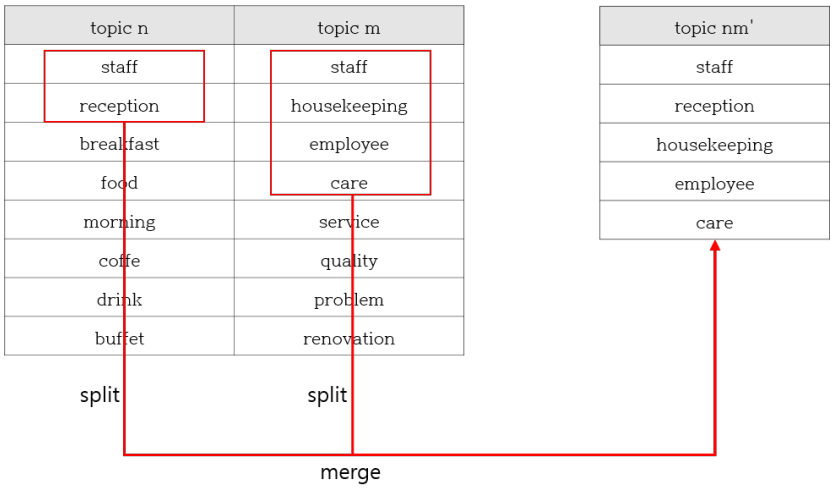

Figure 6. A real example of split and merge topics.

4.3. Evaluation of Main Aspects and Sub-Aspects

After topics are clean, we view each clean topic to an aspect. The goal of the proposed method is to automatically generate one word as an aspect which corresponds to a clean topic. If the number of the clean topics post–processed by the proposed method is five, we can figure out almost five stuffs about which most users mainly talked in the collection. Each aspect is a feature that describe a product or not. In some time, an aspect is a story not related to the product features but interested by most users. Table 6 shows the main and sub aspects per topic. The main aspect is more general than sub aspects that describe/support the details of the main aspect. In Algorithms 3–5, given a topic θ_i, a word with the lowest w_* is chosen as the main aspect. To automatically generate the sub aspects, the top–5 words with the highest probability value are first selected and then each word is converted to a vector from a word embedding model. Next, the average embedding is calculated from the five word embedding vectors. Finally, find the top–3 words that are close to the average embedding vector in the collection. Table 6 shows each topic identifier, main aspect, three sub aspects, a sentence, and a score. To evaluate the auto–generated aspects, we hired 30 volunteers who were college students. For a topic, each human evaluator carefully and manually went over top–100 review documents that are arranged with the topic by LDA. After he/she understood the content of the topic, he/she scored between 1 and 5 on the Ricardo scale in which 1 means that these aspects have nothing to do with the topic and 5 means that these aspects are very relevant with the topic. The score of the table is the average of the scores determined by the 30 human evaluators. In addition, the column 'Top sentence' of the table is one sentence that summarizes the top–100 review documents relevant with the topic. They extracted the sentence after they completely understood the top–100 documents. By and large, it appears that the main and sub aspects are very relevant with the semantics of all topics. Interestingly, some aspects are the same. For example, the main aspect of topics 1 and 8 is 'hotel' but the meanings of the aspects are slightly different because topic 1 says the hotel rating, while topic 8 is relevant with the location. As shown in the table, because main topics of our dataset are drink, hotel, staff, room, night, and breakfast, we conjecture that many users are likely to be interested in them as the main aspects.

Table 6. Main and sub aspects per topic.

Topic ID	Main Aspect	Sub Aspect	Description	Score
0	drink	brunch voucher salad	. . . drinks and a snack meal available	4.0
1	hotel	decoration accommodation opinion	The decoration of the room has a luxury feel. . .	4.3
2	hotel	praise period loyalty	Rancho Valencia was perfect from beginning to end	3.6
3	staff	bellman understaffed acknowledge	The staff is very friendly and helpful	4.3
4	room	restroom roomy layout	. . . room was clean with nice views	3.7
5	room	bathtub bath tube	The bathroom needs to be upgraded	3.8
6	night	overnight weekend getaway	Stayed here for two nights and had a great time	3.5
7	location	resturant supermarket attraction	It was a great location for our purpose	3.7
8	hotel	landscaping tennis court	Pretty tropical landscaping, really nice pool fitnessa are	3.8
9	room	loudly construction sound	Room spacious and clean, heater efficient but pretty noisy	3.5
10	breakfast	cocktail menu tasting	They had a very nice breakfast too	4.3

To automatically generate the aspect of each topic, we proposed three different auto-labelling methods. One is based on topic words; another is on head words; and the other is on a hybrid of topic and head words. Table 7 illustrates the results of the three methods. The second column means the average of the scores evaluated by human evaluators. The experimental result shows that the second proposed method of auto-generating the aspects of the topics is better than the other individual methods.

Table 7. Comparison of three proposed methods for automatically generating aspects.

Aspect Detection Method	Score
Algorithm 3 using topic words	2.2
Algorithm 4 using head words	4.4
Algorithm 5 using both topic and head words	3.6

5. Discussion

In this section, we interpret the results of the proposed methods in perspective of previous studies and of the working hypotheses. We also discuss limitations of the work and future research directions.

In this work, the final goal of our research is to automatically label topics, each of which is considered to be an aspect about which a lot of review writers talk in a particular product. To achieve this goal, we should tackle three sub problems that are real challenges for data–driven market research.

The first sub problem is to find the proper priors of a given data set. The previous studies showed that there exist priors suitable for each particular data set so that the topic model used by inaccurate priors can provide dirty topics as the result. In turn, the topic labelling method is considerably affected by such inaccurate topic results. Table 4 shows the topic coherence scores by means of fastText, the best of variant word embedding models, according to different values of three priors—α, β, and # of topics. As we already discussed in Section 2.1.1, because real text data are not clean, the existing statistically-based prior estimation approach does not work well. Therefore, the empirical approach like Algorithm 1 actually helps in order to find the priors suitable for a specific data.

The second sub problem is to clean dirty topics usually resulting from the traditional topic models. If such mixed and redundant topics are corrected through the proposed topic merge and split algorithms, labelling each topic is easy and we can clearly understand the nature of the given data. To validate our claim, we did additional experiments in which we downloaded 186 review posts about K5 cars in one Korean web site—Bobaedream and then compare split topics to the original topics through topic coherence measure. Table 8 shows the ratio of original topics to split topics, indicating that the average score of topic coherence is improved by up to 25% by the split topics more than by the original topics.

To see the effectiveness of the merged topics, we first collected all pairs of two split topics. For example, for each pair of two split topics—θ_i and θ_j, Algorithm 2 merges θ_i to θ_j if $\theta_i \sim \theta_j$, where the similarity score between θ_i and θ_j is computed by word embedding. In this way, the merged topics are the *predicted result* performed by Algorithm 2. When θ_i and θ_j are combined, human evaluators find both top–k sentences relevant with θ_i and both top–k sentences relevant with θ_j. Then, human evaluators manually judged whether the meaning of the most sentences relevant with θ_i is similar (different) to (from) that of θ_j. Through this manual investigation, the *actual result* is obtained. Using both actual and predicted results, we consider the confusion matrix to compute average recall, precision, and F_1–score. Figure 7 shows the results of the merged topics. Specifically, the F_1–score is 0.91 and the accuracy value is 0.93. Thus, this implies that the merged topics obtained by Algorithm 2 are more consistent than the original topics obtained by the traditional topic model.

The third sub problem is to automatically find main and sub aspects that are the label of each topic. As shown in Table 6, the proposed three methods work well to detect main aspects from review data about hotels. For example, most review documents mainly talk about hotel breakfast, hotel interior, service, room, location, and landscape. Table 7 shows the average of the scores evaluated by human evaluators. The score of Algorithm 4 is 4.4 that is close to 5 as the perfect score. This survey result means that the proposed main aspect detection method based on clean topics work well. Until now, all previous methods have not considered clean topics to finally detect main aspects from large review text documents.

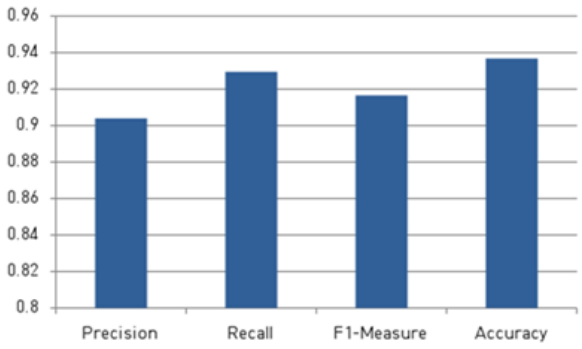

Figure 7. Results of merge topics.

The approaches proposed in this article might be affected by the quality of the raw review data. As the input data, review documents, including advertisements or spam, make it difficult to extract the main aspects. In addition, it will be hard to find important topics from review documents that

contain little text or non-text such as emoticons and abbreviations. Therefore, it is expected that the proposed approaches will achieve a higher performance if the best technique capable of both collecting high-quality review data and filtering unnecessary data is employed. In this case, there is still room to study in our future work.

Table 8. Results of original and split topics.

Topic ID	Original Topic Coherence	Split Topic Coherence	Improvement Rate
0	0.49	0.61	0.25
1	0.5	0.61	0.23
2	0.49	0.6	0.23
3	0.51	0.66	0.3
4	0.51	0.64	0.25
5	0.5	0.63	0.27
6	0.49	0.6	0.21
7	0.5	0.61	0.24
8	0.51	0.64	0.26
9	0.48	0.61	0.26
10	0.5	0.63	0.25
11	0.51	0.64	0.26
12	0.5	0.64	0.27
13	0.5	0.62	0.24
14	0.51	0.64	0.26
15	0.49	0.61	0.24
16	0.48	0.61	0.27
17	0.5	0.63	0.26
18	0.5	0.6	0.21
19	0.49	0.6	0.22
Average	0.5	0.62	0.25

6. Conclusions

In data-driven market research, given a collection of large review posts about a particular product, it is important to automatically find main aspects that are mainly talked by most customers. For each aspect, the automatic method that quickly and easily grasps the public's preferences for the product and by finding specific reasons they like or dislike will provide useful information to customers who want to purchase the product or company marketers who want to sell the product. In particular, finding main aspects automatically is the essential task in market research.

To address this problem, we focus first on utilizing the traditional topic model that extracts topics from large review text documents. In our standpoint, a topic is represented as an aspect. However, one obstacle of the existing topic models is that the topic results are often dirty. In other words, there exist ambiguous topics, redundant topics, and meaningless topics. To improve the dirty topics, we propose an automatic method (Algorithm 1) that estimates priors that fit the input data and a novel topic cleaning method (Algorithm 2) that both split ambiguous topics and merge duplicate topics. Moreover, based on the cleaned topics, we propose novel word–level (Algorithm 3), sentence–level (Algorithm 4), and hybrid (Algorithm 5) methods that automatically label each topic as an aspect. All algorithms are based on word embedding model. In Algorithm 1, various word embedding models are used to measure topic coherence. In Algorithm 2, word embedding models are used to merge redundant topics. In Algorithms 3–5, to find the center word among relevant words in each topic, word embedding models are used.

In a collection of review text documents about hotels, the experimental results show that Algorithm 1 using fastText is the best word embedding model with priors $\alpha = 1.5$ and $\beta = 0.08$. The rouge–1 score of Algorithm 1 is 0.725, while that of the baseline method with default priors is 0.582, improving up to 25%. In addition, comparing original topics to split topics, Algorithm 2 improves to about 25% on average. In case of merged topics rather than original topics, the average F_1–score is about 0.91. These results clearly show that the topics cleaned by Algorithm 2 is more useful than the

original topics in order to label topics. Finally, human evaluators judges how useful the proposed three methods of automatically labelling topics are. In particular, the survey scores of the sentence-level method are about 4.4 on average that is close to 5 as the perfect score. As a case study, main aspects about the hotel data set are breakfast, interior, service, location, and landscaping. All of these results indicate the proposed methods provide a simple, fast, and automatic way to mine main aspects from a large review data.

To the best of our knowledge, this work is the first study to consider (1) an empirical method that is effective in estimating the priors from ambiguous and noisy text data, (2) a topic clean method that splits and merges ambiguous and redundant topics, and (3) word-level and sentence-level topic labelling methods that are based on various word embedding models.

In our future work, we plan to apply the proposed method to various domains including hotels, restaurants, rent-a-car, and so forth, and improve the accuracy of the proposed method. We will also propose a new abstractive summarization method based on clean topic information. Furthermore, we will re-implement the proposed method to MapReduce-based method for processing Big Data.

Author Contributions: All authors of this article discussed the contents of the manuscript and actively contributed in the processing of implementation. Conceptualization, B.-W.O.; methodology, B.-W.O.; software, S.-M.P. and S.J.L.; validation, B.-W.O. and S.-M.P.; formal analysis, B.-W.O.; investigation, B.-W.O.; resources, B.-W.O.; data curation, S.-M.P. and S.J.L.; writing—Original draft preparation, B.-W.O.; writing—Review and editing, B.-W.O.; visualization, S.-M.P.; supervision, B.-W.O.; project administration, B.-W.O.; funding acquisition, B.-W.O. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by National Research of Korea (NRF) grant funded by Korea government (MSIT) (No. NRF–2019R1F1A1060752).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Park, S.-M.; On, B.-W. Latent topics based product reputation mining. *J. Intell. Inf. Syst.* **2017**, *23*, 39–70.
2. Maharani, W.; Widyantoro, D.H.; Khodra, M.L. Aspect-based opinion summarization: A survey. *J. Theor. Appl. Inf. Technol.* **2017**, *95*, 448–456.
3. Gensim. Word2vec Embeddings. 2019. Available online: https://radimrehurek.com/gensim/models/word2vec.html (accessed on 2 January 2019).
4. Jeffery, P.; Richard, S.; Christopher, D.M. GloVe: Global Vectors for Word Representation. 2019. Available online: https://nlp.stanford.edu/projects/glove/ (accessed on 2 January 2019).
5. Facebookresearch. Fasttext. 2019. Available online: https://github.com/facebookresearch/fastText (accessed on 2 January 2019).
6. TensorFlow Hub. ELMo: Embeddings from Language Model. 2019. Available online: https://tfhub.dev/google/elmo/1 (accessed on 2 January 2019).
7. Google Research. BERT: Bidirectional Encoder Representations from Transformers. 2019. Available online: https://github.com/google-research/bert (accessed on 2 January 2019).
8. Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, OR, USA, 2–4 August 1996; pp. 226–231.
9. Oh, S.; On, B.-V. A MapReduce-based prior probability optimization algorithm for topic extraction. *J. KIISE* **2018**, *45*, 478–488. [CrossRef]
10. Gaillat, T.; Stearns, B.; Sridhar, G.; McDermott, R.; Zarrouk, M.; Davis, B. Implicit and explicit aspect extraction in financial microblogs. In Proceedings of the 1st Workshop on Economics and Natural Language Processing, Melbourne, Australia, Brussels, Belgium, 31 October–4 November 2018; pp. 55–61.
11. Chen, Z.; Liu, B. Topic modelling using topics from many domains, lifelong learning and big data. In Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 21–26 June 2014; pp. 703–711.
12. Poria, S.; Cambria, E.; Gelbukh, A. Aspect extraction for opinion mining with a deep convolutional neural network. *Knowl. Based Syst.* **2016**, *108*, 42–49. [CrossRef]

13. Qiu, G.; Liu, B.; Bu, J.; Chen, C. Opinion word expansion and target extraction through double propagation. *Comput. Linguist.* **2011**, *37*, 9–27. [CrossRef]
14. Shu, L.; Liu, B.; Xu, H.; Kim, A. Supervised opinion aspect extraction by exploiting past extraction results. *arXiv* **2016**, arXiv:1612.07940.
15. Liu, B. *Sentiment Analysis and Opinion Mining*; Morgan & Claypool Publishers: San Rafael, CA, USA, 2012.
16. Pontiki, M.; Galanis, D.; Papageorgiou, H.; Manandhar, S.; Androutsopoulos, I. SemEval-2015 Task 12: Aspect Based Sentiment Analysis. In Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), Denver, CO, USA, 4–5 June 2015; pp. 486–495.
17. Hu, M.; Liu, B. Mining and summarizing customer reviews. In Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, WA, USA, 22–25 August 2004; pp. 168–177.
18. Popescu, A.-M.; Nguyen, B.; Etzioni, O. OPINE: Extracting product features and opinions from reviews. In Proceedings of HLT/EMNLP on Interactive Demonstrations, Vancouver, BC, Canada, 6–8 October 2005; pp. 32–33.
19. Ku, L.-W.; Liang, Y.-T.; Chen, H.-H. Opinion extraction, summarization and tracking in news and blog corpora. In Proceedings of the AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs, Palo Alto, CA, USA, 27–29 March 2006; pp. 100–107.
20. Moghaddam, S.; Ester, M. Opinion digger: An unsupervised opinion miner from unstructured product reviews. In Proceedings of the 19th ACM International Conference on Information and Knowledge Management, Toronto, ON, Canada, 25–29 October 2010; pp. 1825–1828.
21. Ding, X.; Liu, B.; Yu, P.S. A holistic lexicon-based approach to opinion mining. In Proceedings of the 1st International Conference on Web Search and Data Mining, New York, NY, USA, 11–12 February 2008; pp. 231–240.
22. Turney, P.D. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Stroudsburg, PA, USA, 6–12 July 2002; pp. 417–424.
23. Zhuang, L.; Jing, F.; Zhu, X.Y. Movie review mining and summarization. In Proceedings of the 15th ACM International Conference on Information and Knowledge Management, Arlington, VI, USA, 6–11 November 2006; pp. 43–50.
24. Zhai, Z.; Liu, B.; Xu, H.; Jia, P. Clustering product features for opinion mining. In Proceedings of the 4st International Conference on Web Search and Data Mining, Hong Kong, China, 9–12 February 2011; pp. 347–354.
25. Lu, B.; Ott, M.; Cardie, C.; Tsou, B. Multi-aspect sentiment analysis with topic models. In Proceedings of the 11th IEEE International Conference on Data Mining Workshops, Vancouver, BC, Canada, 11 December 2011; pp. 81–88.
26. Cambria, E. Affective computing and sentiment analysis. *IEEE Intell. Syst.* **2016**, *31*, 102–107. [CrossRef]
27. Hussain, A.; Cambria, E. Semi-supervised learning for big social data analysis. *Neurocomputing* **2018**, *275*, 1662–1673. [CrossRef]
28. Ma, Y.; Peng, H.; Cambria, E. Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive LSTM. In Proceedings of the 32nd AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; pp. 5876–5883.
29. Yang, M.; Zhao, W.; Ye, J.; Lei, Z.; Zhao, Z.; Zhang, S. Investigating capsule networks with dynamic routing for text classification. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 3110–3119.
30. Zainuddin, N.; Selamat, A.; Ibrahim, R. Improving twitter aspect-based sentiment analysis using hybrid approach. In Proceedings of the 2016 Asian Conference on Intelligent Information and Database Systems, Da Nang, Vietnam, 14–16 March 2016; pp. 151–160.
31. Mankar, S.A.; Ingle, M.D. Implicit sentiment identification using aspect based opinion mining. *Int. J. Recent Innov. Trends Comput. Commun.* **2015**, *3*, 2184–2188.
32. Yan, Z.; Xing, M.; Zhang, D.; Ma, B. EXPRS: An extended pagerank method for product feature extraction from online consumer reviews. *Inf. Manag.* **2015**, *52*, 850–858. [CrossRef]

33. Yu, J.; Zha, Z.-J.; Wang, M.; Wang, K.; Chua, T.-S. Domain-assisted product aspect hierarchy generation: Towards hierarchical organization of unstructured consumer reviews. In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, Edinburgh, UK, 27–31 July 2011; pp. 140–150.
34. Lazhar,F.; Yamina, T.-G. Mining explicit and implicit opinions from reviews. *Int. J. Data Min. Model. Manag.* **2016**, *8*, 75–92. [CrossRef]
35. Karmaker, S.S.K.; Sondhi, P.; Zhai, C. Generative feature language models for mining implicit features from customer reviews. In Proceedings of the 2016 ACM Conference on Information Knowledge Management, Indianapolis, IN, USA, 24–28 October 2016; pp. 929–938.
36. Sun, L.; Chen, J.; Li, J.; Peng, Y. Joint topic-opinion model for implicit feature extracting. In Proceedings of the 2015 IEEE International Conference on Intelligent Systems and Knowledge Engineering, Taipei, Taiwan, 24–27 November 2015; pp. 208–213.
37. Makadia, N.; Chaudhuri, A.; Vohra, S. Aspect-based opinion summarization for disparate features. *Int. J. Adv. Res. Innov. Ideas Educ.* **2016**, *2*, 3732.
38. Wan, Y.; Nie, H.; Lan, T.; Wang, Z. Fine-grained sentiment analysis of online review. In Proceedings of the 12th International Conference on Fuzzy Systems and Knowledge Discovery, Zhangjiajie, China, 15–17 August 2015; pp. 1406–1411.
39. Chen, L.; Martineau, J.; Cheng, D.; Sheth, A. Clustering for simultaneous extraction of aspects and features from reviews. In Proceedings of the NAACL–HLT, San Diego, CA, USA, 12–17 June 2016; pp. 789–799.
40. Liu, B.; Hu, M.; Cheng, J. Opinion observer: Analyzing and comparing opinions on the web. In Proceedings of the 14th International Conference on World Wide Web, Chiba, Japan, 10–14 May 2005; pp. 342–351.
41. Panchendrarajan, R.; Ahamed, N.; Murugaiah, B.; Sivakumar, P.; Ranathunga, S.; Pemasiri, A. Implicit aspect detection in restaurant reviews using co-occurence of words. In Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, San Diego, CA, USA, 12–17 June 2016; pp. 128–136.
42. Dosoula, N.; Griep, R.; den Ridder, R.; Slangen, R.; Schouten, K.; Frasincar, F. Detection of multiple implicit features per sentence in consumer review data. In Proceedings of the 2016 International Baltic Conference on Databases and Information Systems, Riga, Latvia, 4–6 July 2016; pp. 289–303.
43. El Hannach, H.; Benkhalifa, M. Hybrid approach to extract adjectives for implicit aspect identification in opinion mining. In Proceedings of the 11th IEEE International Conference on Intelligent Systems: Theories and Applications, Mohammedia, Morocco, 19–20 October 2016.
44. Afzaal, M.; Usman, M.; Fong, A.C.M.; Fong, S.; Zhuang, Y. Fuzzy aspect based opinion classification system for mining tourist reviews. *Adv. Fuzzy Syst.* **2016**, *2016*, 14. [CrossRef]
45. Poria, S.; Cambria, E.; Ku, L.-W.; Gui, C.; Gelbukh, A. A rule-based approach to aspect extraction from product reviews. In Proceedings of the 2nd Workshop on Natural Language Processing for Social Media, Dublin, Ireland, 24 August 2014; pp. 28–37.
46. Bhatnagar, V.; Goyal, M.; Hussain, M.A. A proposed framework for improved identification of implicit aspects in tourism domain using supervised learning technique. In Proceedings of the 2016 International Conference on Advances in Information Communication Technology & Computing, pages Advances in Fuzzy Systems, Bikaner, India, 12–13 August 2016.
47. Jiang, W.; Pan, H.; Ye, Q. An improved association rule mining approach to identification of implicit product aspects. *Open Cybern. Syst. J.* **2014**, *8*, 924–930. [CrossRef]
48. Hai, Z.; Chang, K.; Cong, G.; Yang, C.C. An association–based unified framework for mining features and opinion words. *ACM Trans. Intell. Syst. Technol.* **2015**, *6*, 26. [CrossRef]
49. Xu, H.; Zhang, F.; Wang. W. Implicit feature identification in Chinese reviews using explicit topic mining model. *Knowl. Based Syst.* **2015**, *76*, 166–175. [CrossRef]
50. Uehara, H.; Ito, A.; Saito, Y.; Yoshida, K. Prior-knowledge-embedded LDA with word2vec—For detecting specific topics in documents. In *PKAW 2019: Knowledge Management and Acquisition for Intelligent Systems, Proceedings of the Pacific Rim Knowledge Acquisition Workshop, Cuvu, Fiji, 26–27 August 2019*; Springer: Cham, Switzerland, 2019; pp. 115–126.
51. Dieng, A.B.; Ruiz, F.J.R.; Blei, D.M. Topic modelling in embedding spaces. *arXiv* **2019**, arXiv:1907.04907.
52. Xu, X.; Cheng, X.; Tan, S.; Liu, Y.; Shen, H. Aspect-level opinion mining of online customer reviews. *China Commun.* **2013**, *10*, 25–41. [CrossRef]

53. Scavuzzo, N. Datafiniti/Hotel-Reviews. 2017. Available online: https://data.world/datafiniti/hotel-reviews (accessed on 15 February 2018).
54. Phan, H.-P.; Nguyen, C.-T. A Java Implementation of Latent Dirichlet Allocation (LDA) using Gibbs Sampling for Parameter Estimation and Inference. 2008. Available online: http://jgibblda.sourceforge.net/ (accessed on 15 February 2018).

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).