CEA-TM: A Customer Experience Analysis Framework Based on Contextual-Aware Topic Modeling Approach

Ariona Shashaj, Davide Stirparo, Mohammad Kazemi

CEA-TM: A Customer Experience Analysis framework based on Contextual-aware Topic Modeling approach

Ariona Shashaj[1], Davide Stirparo[1], and Mohammad Kazemi[1]

No Institute Given

Abstract. Text mining comprises different techniques capable to perform text analysis, information retrieval and extraction, categorization and visualization, is experiencing an increase of interest. Among these techniques, topic modeling algorithms, capable of discovering topics from large documents corpora, has many applications. In particular, considering customer experience analysis, having access to topic coherent set of opinions expressed in terms of text reviews, has an important role in both customers side and business providers. Traditional topic modeling algorithms are probabilistic models words co-occurrences oriented which can mislead topics discovery in case of short-text and context-base reviews. In this paper, we propose a customer experience analysis framework which enrich a *state-of-art* topic modeling algorithm (LDA) with a semantic-base topic-tuning approach.

Keywords: NLP, topic modeling, text mining, word embedding

1 Introduction

The rapid growth of digital data over the internet, experienced during this last two decades, has drew attention on tools capable to organize, understand and search them. When it comes to consider textual data, text mining field, which comprises different aspect of text analysis, information retrieval and extraction, clustering, categorization and visualization [7], is becoming a key enabling technology. In this context, topic modeling algorithms, which are able to infer latent topics from large documents corpora, have many applications such as user interest profiling [20], content classification [18], topic-driven comments rating [12], analysing and understanding customer satisfaction [19]. Traditional algorithms of topic modeling are based on probabilistic generative models, such as *probabilistic Latent Semantic Analysis* (pLSA) [8] and *Latent Dirichlet Allocation* (LDA) [3], where each topic is represented through a probability distribution over words and documents are represented through a distribution over topics. Even though pLSA and LDA are the *state-of-art* topic modeling algorithms and find applications in many fields, their intrinsic unpredictive nature can leads to results which are topics difficult to understand and they don't provide any tool in order to perform application-oriented topic tuning. Further, when considering short text documents like customer's reviews where the actual meaning of reviews is mostly context-base, an approach which relay on just words co-occurrence might fail discovering contextual topics. Many proposals in literature have addressed short text documents topic modeling challenges. In [1] and [11]

the authors argue that techniques like word embedding [17] should be consider in order to exploit semantic relations among words. Similar to this works, in this paper we propose a customer experience analysis framework which combine LDA topic modeling approach with word embedding technique. Our contribution comprises: a *semantic topic coherence score* build on top of an word embedding model; *LDA parameter tuning*, where LDA parameters such as number of topics and the prior Dirichlet parameters, are tuned in order to maximize the overall topics semantic coherence score; a *topic tuning approach* based on clustering which split and merge topics with the final goal to maximize topics semantic coherence score. This results in a semi-automatized topic modeling framework where human evaluations is limited only to the final step of topic description. The remainder of this paper is organized as follows. In Section 2 we motivate our work and review related works. The proposed topic modeling it is presented in Section 3, while Section 4 shows evaluation of the approach. Finally, conclusion remarks and future works are given in Section 5.

2 Related works

Topic Modeling is a promising text mining technique in the context of social science, capable to explore and gain meaning from a large set of textual corpora [15]. In literature, there are many methodologies proposed in the context of topic modeling [16]. *Non-negative Matrix Factorization* NMF [2] is a deterministic approach based on a non-negative matrix decomposition problem given the number of topics. It imposes non-negative constraints on every element of the matrix. The *probabilistic Latent Semantic Analysis* pLSA [8] derives from *Latent Semantic Analysis* [6] and it represents topics through multinomial random variables. Each word is assigned to a single topic, whereas different words in the same document can be assigned to different topics. Relaxing the assumption of assigning a document to a single topic makes this approach the first attempt towards probabilistic generative models.

Latent Dirichlet Allocation LDA [3] projects document into a vector space by considering the number of occurrences and represents topics through a probability distribution over words, whereas documents are represented through a probability distribution over topics. In the last years, in the context of social science, several variants of LDA have been proposed, such as [22], [4].

On the other hand, in order to cope with sparse distribution of topics among short text corpora, such as Twitter feeds and product / offered service reviews, adaptions of LDA have been explored too. *Biterm Topic Modeling* (BTM) [5] exploits bi-terms co-occurrences, whereas in *Dirichlet Multinomial Mixture* (DMM) [21] the authors assume that there is a single latent topic per document and they introduced a collapsed Gibbs algorithm in order to sample topic for a document considering a conditional probability. GPU-DMM [11] enrich DMM with word embedding technique in order to exploit semantic relation among words. Even though the assumption of single topic per document seems to be reasonable for short text documents, in some case like caring services offered in the tourism domain, reviewers don't coherently discuss just a single topic over a comment. In this paper, similar as in [11] we enrich LDA algorithm with word embedding

technique, and further, we developed a parameters and topic tuning approach based on word embedding score.

3 Topic Modeling: Contextual-aware approach

Latent Dirichlet Allocation is one of the most popular and most used topic modeling techniques. Despite the popularity, there are some uncertainty about the validity and reliability of the LDA results.
This study overcomes aforementioned uncertainties by defining an approach that addresses three LDA's challenges: 1-Hyper-parameters tuning. 2-Evaluation of the model's reliability. 3-Control the validly interpreting the resulting topics. We propose a methodology named Contextual-aware Topic Modeling approach, that answers these challenges and also improves the overall result.

In our approach, *text pre-processing* techniques and *vectorization* are used in order to clean, normalize and vectorized text data. Since this very step has been explained in many other papers and case studies, here we would not cover it. Then we perform *word-embedding* which is the base of semantic topic coherence score. *Semantic topic coherence score* plays an important role in the field of determination of reliability and validity interpretation of the result. The process of topic modeling is performed by, first, executing the *LDA parameter tuning* step. After finding the best parameters settings and train the LDA model, (We've considered the Sklearn [14] implementation of LDA in our work) the next step is topic tuning phase which is about cleaning topics. First, it will try to improve the topic quality by applying clustering and then find similar topics and merge them together. In both steps, the *semantic topic coherence score* will intervene in order to evaluate the result. From now on we will explain each step in detail.

3.1 LDA Tuning

Concerning the first challenges, we need to define a proper tuning process in order to find the best values for *hyper_parameters* to get the optimum result. First LDA requires an estimation of the number of topics $n\_components$, for training. Second, needs to tune the LDA prior parameters α which is the distribution of topics per document and β which is the probability distribution of words per topic and finally the maximum iteration over each document max_iter is the last parameter related to the implementation of LDA which should be tune.
As we explained in the previous section, we use the semantic topic coherence score to find the best value for each hyper parameters which give us the optimum LDA result.

3.2 Coherence Score

In order to address the second LDA challenges and define a topic evaluation method, we define a coherence topic score which exploit semantic relations among words associated to the same topic. The cosine similarity it is calculated considering two scalar vectors A and B, and it returns a value from 0 to 1. Closer you get to the maximum, more similar A and B are. In our analysis, Word2Vec model, which projects words semantic meaning into a vector space embedding, was trained on custom dataset and the very first use of this score is evaluate the LDA result. Once the topics are obtained from the model, and

expressed through a limited set of words, the method, first calculates the cosine similarity between all the possible pairs of Word2Vec vector projections of the words and then makes the average of all these values for a generic topic.

Algorithm 1: Coherence Score

$Top\_Topic(K, W)$: top words array according to the LDA distribution per topic
$word2vec$: The word2vec model trained on custom dataset
K: Number of Topics
W: number of words per topics semantic coherence score associated to the topics

```
1  begin
2      for t ∈ [0, K] do
3          for i ∈ [0, W] do
4              vec[t, i] = word2vec(Top_Topic[t, i])
5          end
6      end
7      Topic_Score ⟵ []for t ∈ [0, K] do
8          Pairs ⟵ combination(vec[t], 2)
           Topic_Score[t] = ∑_{pair∈Pairs}(cos_sim(pair[1], pair[2]))/‖Pairs‖
9      end
10 end
11 Return Avg_Score = (∑_{S∈Topic_Scores} S)/K
```

In algorithm 1 we show how the semantic coherence score used to evaluate LDA topic. First, $vec[t, i]$, the Word2Vec projection for each word which represents a topic it is retrieved (lines 2-5). Then, the coherence score associated to single topic it is calculated as the average of the overall cosine similarity between pairs of words (lines 8-11).
Finally, the overall score is the average achieved considering all topics. Hence by using the score method we can find the optimum value of each *hyper_parameters* which will be used to train LDA model and obtain topics which achieve the maximum coherence score.

3.3 Topic Cleaning

In the previous step, we obtained optimal LDA parameters settings that maximize the semantic coherence score, however, high coherence score will not guarantee to have clean topics, because the score is a mathematical calculation. Unclean topics can be classified as:

- *Dirty Topics*: A topic is mixed with two or more semantic groups of topic words with different meanings. This dirty topic should be split to two or more correct topics
- *Redundant Topic*: These duplicate topics should be merged to a single topic.

To split and merge raw topics, we applied a new post–processing method based on the word embedding and unsupervised clustering techniques in which we consider Word2Vec as the word embedding model and as the unsupervised clustering method we used Density–based Spatial Clustering of Applications with Noise (DBSCAN). By using the output of tuned topic modeling, we created a list of top 25 most frequent words of each topic and we will use it in post-processing method. Below we are going to explain in details the preparation steps involved in topic cleaning.

Word2Vec Topic projection: In our point of view, the proposed method is based on the hypothesis that one topic will form one semantic cluster in the

word embedding space and a dirty topic is a mixture of multiple topics, so it contains multiple semantic clusters. Suppose that the *i-th* topic result denoted as T_i, where $T_i = \{w_j^i : j \in [0, \|T_i\|]\}$ is the array of the top words with the highest probabilities. Then, we define as f_{we} the projection of a word into Word2Vec vector space, where $\overline{w} = f_{we}(w)$ and $\overline{w}$ is the p-dimensional word embedding vector of w. In this way, the vector space corresponding to the top words of the *i-th* topic T is defined as $\overline{T_i} = \{\overline{w}_i^j = f_{we}(w_j^i) : j \in [0, \|T_i\|]\}$. In order to perform the Word2Vec projection we consider a pre-trained model on our custom target dataset.

Dimensional Reduction: When we have too many features(p-dimensional), observations become harder to cluster. In an attempt to reducing dimensional we create a mixed approach by using the *Principal Components Analysis* (PCA) [10] and *t-distributed Stochastic Neighbor Embedding* (t-SNE) [13]. PCA is a linear feature extraction technique which is focused on placing dissimilar data points far apart in a lower dimension representation, on the other hand t-SNE is a non-linear manifold and represent similar data points close together which is essential for our type analysis. We reduce initial number of dimensions linearly with PCA down to 10% latent variables, then we will apply t-SNE on the PCA result.

3.4 Topic cleaning: DBSCAN Clustering

DBSCAN is a clustering method that is used in machine learning to separate clusters of high density from clusters of low density region.One important feature of DBSCAN is that we do not need to fix the number of clusters before executing it. The DBSCAN algorithm automatically will estimate the clusters considering two input parameters:

- *eps*: The maximum distance between two samples for one to be considered as in the neighborhood of the other.
- *min_samples*: The number of samples (or total weight) in a neighborhood for a point to be considered as a core point.

The major challenge of using DBSCAN algorithm is to find a right setting of hyper-parameters (*eps* and *min_samples* values) to fit in to the algorithm for getting accurate results. Since the DBSCAN needs to have a distance between two samples, we calculated the Euclidean distance between all the top words in each topic and then sorted them out. Then we set the calculated euclidean distance as our *eps* range and now a fix range for *min_samples* is needed.
The algorithm we've implemented, loop through these two parameter's range and returning the possible clusters scenarios. For each clusters it calculates the silhouette score, and chose the first parameters that have the top score, and consequently, the cluster labels respect that best parameters will be the final result of our analysis in this step. Nevertheless the topic cleaning phase is not completed because we have to select the best clusters for each topic.

3.5 Topic Cleaning: Clusters evaluation

As a first step, the size of each cluster (C_i) for a generic topic T_j, will be checked. If size of $\|C_i\|$ is lower than a threshold named *min_cluster_size*, we will not consider that cluster further in the analysis. This *weak clusters*, named outliers,

contribute as noise in a topic definition.
Clusters, C_j, where $\|C_j\| > min\_cluster\_size$, named as super cluster are candidates for topics definition. The remain part of the approach split topics which contains more than one cluster with dimension greater than $min\_cluster\_size$ into two topics. We've also define a threshold max_topic_size for the maximum number of words, to use in topic definition. In case, cluster dimensions are greater than max_topic_size, a subset of max_topic_size words it is selected, which achieve the highest semantic coherence score.

Algorithm 2: Evaluate Cluster Result

```
1  C: Collection of clusters calculated for each topic where C[i,j] contains words array which
   characterized the j-th cluster of the i-th topic; min_size: minimum threshold for cluster
   dimensions; max_size: maximum threshold for final topic dimensions T: Resulting topics list
   after cluster analysis
2  begin
3      T ⟵ {}
4      Candidate_C ⟵ {C[i,t] : C[i,t] ∈ C ∧ ‖C(i,t)‖ > min_size};
5      forall C[i,t] ∈ Candidate_C do
6          if ‖C[i,t]‖ < max_size then
7              T ⟵ T ∪ w : w ∈ C[t,i]
8          end
9          else
10             C[i,t]^{max_size} : subset which maximize semantic coherence score
11             T ⟵ T ∪ w : w ∈ C[i,t]'
12         end
13     end
14     return T
15 end
```

In algorithm 2, C refers to the calculated clusters, whereas min_size and $max\ \_size$ are respectively the minimum cluster dimension threshold to use for discovering *weak clusters* and the maximum topic dimension threshold. Candidate clusters are selected among those which dimension is greater than min_size (line 3), whereas output topics are selected among clusters which size is lower than max_size (line 6) or subsets of max_size elements of clusters with dimension greater than that and which achieve the maximal semantic coherence score (lines 9-10).

3.6 Topic cleaning: Merge

Merging similar topics is the last step to be performed. The final goal of merge phase is to discover pairs of topics with at least 40% equal words, to join them as a unique topic. We use the clean topics list, obtained in the previous phases (algorithm 2) and identifies which topic pairs have the condition to be merged. To carry out this join, the words of the two topics will be grouped into a single list then we eliminate duplicate words and subsequently we calculate the coherence maximum semantic coherence scores. In algorithm 3 we show the implemented approach in order to automatically perform the join of similar topics. The algorithm takes as input, the list of topics T, which results after the clustering and topic splitting phase and a threshold Th_sim to be used in order to check the join condition. Topics list it is scrolled consecutively, and each topic it is compared with the next in row (lines 4-5).
If a pair of topics it is found such that they have a subset of Th_sim words the

same, then considering that the join condition it is satisfied, the two topics are merged (lines 7-16). A final check it is done, in order to consider topics which didn't participate in any merge (lines 19-22). Finally, the resulting list of merged topics it is returned, as well as, the list of deleted ones (topics which participate into a merge) and the list of topics which did not take part at any merge.

We observe, that even though the previous cluster analysis clean topic from eventually noise, expressed as *dirty cluster* which have a lower size, and split into more than one represented through semantic related words, eventually in some cases it can happen than one topic participates in more than one merge (multiple merges per single topic). In that case, it is necessary to break ties between the involved merges.
By setting the threshold Th_sim greater than half of the dimension of the words collection which represent a topic, it can prevent the situation of multiple merges per single topic, however this will result in a lower probability to discover semantic related topics. We show in algorithm 4 the proposed approach capable to clean a collection of merged topics affected by multiple merges per topic.

Algorithm 3: Creation of a list that contains merge topics and all topics without a merge

```
1  T[K,W]: array topics results from algorithm 2 which contains K topics each represented through W words. Th_sim : a threshold which hold the merge condition. Two topics are merged if they both contains at least Th_sim words T' : the resulting list of topics after merging T^D : list of removed topics T^NM: list of not merged topics
2  begin
3      T' <-- {}
4      T^D <-- {}
5      T^NM <-- {}
6      for i ∈ [1, K − 1] do
7          for j ∈ [i + 1, K] do
8              counter = 0
9              for w ∈ T_i do
10                 if w ∈ T_j then
11                     counter = counter + 1
12                 end
13             end
14             if counter >= Th_sim then
15                 T' <-- T' ∪ (T_i ∪ T_j)
16                 T^D <-- T^D ∪ T_i
17                 T^D <-- T^D ∪ T_j
18             end
19         end
20     end
21     for i ∈ [0, K] do
22         if T_i ∉ T^D then
23             T^NM <-- T^NM ∪ T_i
24         end
25     end
26     Return T', T^D, T^NM
27 end
```

Algorithm 4 consider as input respectively, $T^{'}$ the collection of topics resulted after merge, T^D a collection of the original topics which participate into a merge (part of the topics list resulted from the clustering and splitting approach - Section 3.5) and collection T^{NM} which are topics not merged. As a first step, we define as *size_merged*, the number of the original topics merged (line 2). Then we define a matrix $Co(size\_merged, size\_merged))$ in order to keep track of the semantic coherence score if the corresponding original are merged. So, $Co[i, j]$ will be equal to 0 if topics $T^D[i], T^D[j] \in T^D$ are not merged during the execution of algorithm 4, otherwise it will be equal to the the coherence score calculated considering the merged topic $T^D[i] \cup T^D[j]$ (lines

4-9). Observe that Co is symmetric, and that $Co[i,j] = Co[j,i]$. To identify the merged topics which are going to be the output of the final output of the merged approach we consider to merge each topic $T^D[i]$ with the one in order to achieve the maximal semantic coherence score(lines 11-14). Finally, single original topics are added to the output (lines 13-17).

Algorithm 4: Check for the presence of topics participating in multiple merges

```
1  T': list of merged topics obtained in algorithm 3. T^D: list of deleted topic which were in
   algorithm 3. T^NM: list of topics which didn't participate in any merge. word2vec: word2vec
   model pre-trained on custom dataset. T*: list of merged topics without multiple merges per
   topic
2  begin
3      size_merged ⟵ ||T^D||
4      Co(size_merged, size_merged) ⟷ init(0)
5      for i ∈ [0, size_merged] do
6          for j ∈ [0, size_merged] do
7              if T_i^D ∪ T_j^D ∈ T' then
8                  Co[i,j] = Co[j,i] = CoherenceScore((T_i^D ∪ T_j^D), word2vec)
9              end
10         end
11     end
12     for i ∈ [0, size_merged] do
13         j* = max_{j in [0, size_merged]} Co[i,j]
14         T* ⟵ T* ∪ (T_i^D ∪ T_j^D)
15     end
16     for t ∈ T^NM do
17         T* ⟵ T* ∪ t
18     end
19     return T*
20 end
```

4 Evaluations

For the evaluation of the proposed solution, we consider a real application scenario, in particular we refer to the scenario detailed as part of the POR PUGLIA FESR C-BAS (Customer Behavior Analysis System) [1]. The domain specific dataset was created considering three main review's sources, such as "`Booking`", "`TripAdvisor`", and "`Google map`"'s reviews related to tourism activities in Puglia[2]. For implementation we used python libraries such as scikit-learn for the LDA model, spacy and nltk for the preprocessing and gensim for the word2vec model.

First step is applying the pre-processing technique, which is going to consider as input the text comments and will return an output a collection of word-tokens. Table 1 shows an example of the pre-processing output pipeline. The second step consist in tuning LDA model considering the output of the pre-processing pipeline (Section 3.1). By using this optimal parameters setting we trained the model in order to obtain the initial topics collection, which are going be tuned in the next steps. Each topic is represented through the set of *25*-top words according to the LDA weights.

[1] https://www.c-bas.eu/

[2] Southern region in Italy - https://en.wikipedia.org/wiki/Apulia

Original text comment	**Text pre-processing** output
'lovely hotel, terraces with views over the old town, tastefully furnished, clean and stylish rooms. Would definitely stay here again. It is in a great location. There is parking available near to the hotel. Reception can advise you where to park.'	['hotel', 'terrace', 'view', 'town', 'room', 'stay', 'location', 'park', 'hotel', 'reception', 'advise', 'park']

Table 1: Pre-processing pipeline output example

In table 2 we show an example of the output topic obtained from LDA training.

Topic 0				
food	show	place	service	restaurant
wine	staff	make	recommend	have
order	time	come	eat	get
dinner	go	price	pasta	menu
cook	take	masseria	seafood	lunch

Table 2: Sample topic with 25 keywords

Cluster results		Top 10 Words	
Cluster 1	**Cluster 2**	**Cluster 1**	**Cluster 2**
show	food	show	food
place	service	place	wine
make	restaurant	make	order
recommend	wine	recommend	eat
have	staff	have	dinner
time	order	time	pasta
come	eat	come	menu
get	dinner	get	cook
go	pasta	go	seafood
price	menu	take	lunch
take	cook		
masseria	seafood		
	lunch		

Table 3: Result of the cluster analysis on topic 0

By observing the words part of *Topic 0*, it seems clear enough that this topic is about food and restaurant, however there are some words that have nothing to do with food and restaurant, such as show, place, make and time. The third step of our approach is the cluster analysis, which the final goal is to clean topics and eventually split them in two or more clusters of semantically related words (Section 3.5). An example of the output of clustering analysis considering *Topic 0* it is shown in table 3. By observing these clusters, we can say that all the words in cluster 1 have no connection with the topic (which is about food and restaurant), however we got an excellent result in cluster 2 where all words are completely related to the topic. The next step is to reduce the number of words in each cluster to top 10 words, which provides the highest semantic coherence (Table 3) and remove also impractical words. As last step of the cluster analysis is the selection of the best clusters in terms of have a clear meaning between all the others. In order to perform this, the best clusters which have the highest semantic coherence score are selected. In case of *Topic 0*, the algorithm chooses cluster 2 (discarding cluster 1), exactly as we might have expected.
As results of the cluster analysis steps, we might expect to have redundant topics with similar meaning. The final goal of the merge phase of Topic Cleaning 3.6 is to reduce this redundancy. In figure 1, we show the execution of the merge phase. There are two topics, *Topic 0* and *Topic 1* that have been identified for merging. This topics have exactly 4 equal words such as "hotel", "room", "staff"

and "service". We've set the merging threshold equal to 40%. In table 4 we show the final results of our work where all topics are completely clear. We show keywords for each topic which helped us to find the main aspect of every topic.

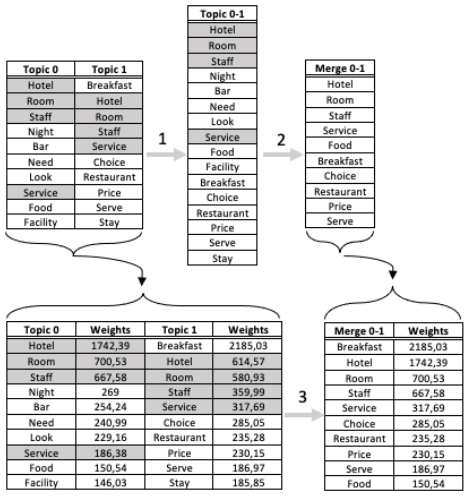

Fig. 1: Merge Process

Table 4, shows us how our model has obtained two topics for the low score reviews. The first topic is about reservation, while the second one is about the location. The neutral score reviews have six topics. We have some general topics about breakfast, location, hotel services, mobility, room but, one specific topic (Topic 5) is about noise pollution, especially in the night due to a dog. It should be noted that topics 6 and 7 in neutral reviews are the result of the merge phase. The high score reviews have 4 topics which are mobility, hotel services, location, and accommodation which is the result of the merge phase. These results approve the validity of our approach otherwise we would face difficulty in interpreting topics. Since we categorise reviews score at the beginning of our process, now we can have a better picture of what are the people opinions about each places. For example we have three same Topics which is about location and we can say that the Location Topic with low review score contains negative feedback and people were not satisfied with the place location and we can go on and elaborate all topics respect to their review score.

Topic ID	**Review Score**	**Main Aspect**	**Keywords**
Topic 0	Low	Reservation	Book, Pay, Say, Check, Tell
Topic 1	Low	Location	Restaurant, Place, Location, Walk, Town, Area, pool
Topic 2	Neutral	Mobility	Walk, Town, City, Center, Minute
Topic 3	Neutral	Location	Location, Stay, Place, Area, Recommend
Topic 4	Neutral	Breakfast	Breakfast, Food, Dinner, Well, Menu
Topic 5	Neutral	Noise Pollution	Street, Night, Noisy, Noise, Dog, Neighbour
Topic 6	Neutral	Room	Room, Breakfast, Restaurant, Get, Hotel, Look, Find
Topic 7	Neutral	Hotel Services	Hotel, Breakfast, Room, Staff, Restaurant
Topic 8	High	Hotel Services	Hotel, Staff, Room, Service, Restaurant
Topic 9	High	Mobility	Walk, City, Town, Host, Restaurant, Owner, Minute
Topic 10	High	Location	Pool, Town, Walk, Area, Restaurant
Topic 11	High	Accommodation	Location, Room, Breakfast, Staff, Place, Stay, Hotel, Recommend, Town, Clean

Table 4: Final Topic Model Result

5 Conclusion and future works

Our analysis focused on the tourism sector and in particular on tourist facilities such as hotels, B&Bs, restaurants, etc., which allows us to see the main topics of interest to customers. The division of reviews by low, neutral, and high score gave us the opportunity to have an even more clearer picture of people's opinions, for example, in low score reviews, people talked more about room problems while in the high score they talked more about the beauty of the place. Our primary goal by defining this method was to implement an automatic approach capable of carrying out these operations every time that new data is provided. The starting point of this method is based on the LDA topic model which can provides unsatisfactory topics. Subsequent operations of cleaning and merging the topics were fundamental and allowed us to obtain very clear topics, with higher coherence than the initial topics. The strengths of this method concern the ability to obtain excellent clean topics automatically and yet it can be be applied to different domains of customer reviews.

Since our approach is totally automatic, in some cases the choice of clusters based on coherence can lead to the selection of not very clear topics and perhaps discarding other which are more interpretable. Still the best evaluator of topic is obviously human, however the propose parameter and topic tuning approach is important in the development of a semi-automated approach of customer experience analysis based on topic modeling. LDA is a very powerful technique for the qualitative analysis of large corpora because of its highly interpretable topics. However, LDA ignores the temporal aspect present in many document collections. The next step in our work will work on DTM(Dynamic Topic modeling) instead of LDA and try implement the DTM method inside our approach. Dynamic Topic Models (DTMs) [9] address the LDA problem which is ignorance of the temporal aspect present in many document by extending the idea of LDA to allow topic representations to evolve over fixed time intervals such as years.

Acknowledgments

Funding/Support: This work was supported by the POR PUGLIA FESR 2014-2020 project C-BAS "Customer Behaviour Analysis System".

References

1. Adji B.Dieng, F.J., M.Blei, D.: Topic modeling in embedding spaces . Transactions of the Association for Computational Linguistics **8** (2020)
2. Arora, S., Ge, R., Moitra, A.: Learning topic models–going beyond svd. In: 2012 IEEE 53rd annual symposium on foundations of computer science. pp. 1–10. IEEE (2012)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. the Journal of machine Learning research **3**, 993–1022 (2003)
4. Chang, J., Boyd-Graber, J., Blei, D.M.: Connections between the lines: augmenting social networks with text. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 169–178 (2009)

5. Cheng, X., Yan, X., Lan, Y., Guo, J.: Btm: Topic modeling over short texts. IEEE Transactions on Knowledge and Data Engineering **26**(12), 2928–2941 (2014)
6. Dumais, S.T., et al.: Latent semantic indexing (lsi) and trec-2. Nist Special Publication Sp pp. 105–105 (1994)
7. Gupta, V., Lehal, G.S., et al.: A survey of text mining techniques and applications. Journal of emerging technologies in web intelligence **1**(1), 60–76 (2009)
8. Hofmann, T.: Probabilistic latent semantic indexing. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. pp. 50–57 (1999)
9. Iwata, T., Watanabe, S., Yamada, T., Ueda, N.: Topic tracking model for analyzing consumer purchase behavior. In: Twenty-First International Joint Conference on Artificial Intelligence. Citeseer (2009)
10. Kramer, M.A.: Nonlinear principal component analysis using autoassociative neural networks. AIChE journal **37**(2), 233–243 (1991)
11. Li, C., Wang, H., Zhang, Z., Sun, A., Ma, Z.: Topic modeling for short texts with auxiliary word embeddings. In: Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval. pp. 165–174 (2016)
12. Ma, Z., Sun, A., Yuan, Q., Cong, G.: Topic-driven reader comments summarization. In: Proceedings of the 21st ACM international conference on Information and knowledge management. pp. 265–274 (2012)
13. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(11) (2008)
14. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in python. the Journal of machine Learning research **12**, 2825–2830 (2011)
15. Ramage, D., Rosen, E., Chuang, J., Manning, C.D., McFarland, D.A.: Topic modeling for the social sciences. In: NIPS 2009 workshop on applications for topic models: text and beyond. vol. 5, p. 27 (2009)
16. Rania Albalawi, T.H.Y., Benyoucef, M.: Using topic modeling methods for short-text data: A comparative analysis. Artificial Intelligence and Deep Learning for Network Management and Communication (2020)
17. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. nature **323**(6088), 533–536 (1986)
18. Sriram, B., Fuhry, D., Demir, E., Ferhatosmanoglu, H., Demirbas, M.: Short text classification in twitter to improve information filtering. In: Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval. pp. 841–842 (2010)
19. Sutherland, I., Kiatkawsin, K.: Determinants of guest experience in airbnb: a topic modeling approach using lda. Sustainability **12**(8), 3402 (2020)
20. Weng, J., Lim, E.P., Jiang, J., He, Q.: Twitterrank: finding topic-sensitive influential twitterers. In: Proceedings of the third ACM international conference on Web search and data mining. pp. 261–270 (2010)
21. Yin, J., Wang, J.: A dirichlet multinomial mixture model-based approach for short text clustering. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 233–242 (2014)
22. Zhang, H., Giles, C.L., Foley, H.C., Yen, J.: Probabilistic community discovery using hierarchical latent gaussian mixture model. In: AAAI. vol. 7, pp. 663–668 (2007)