Implicit aspect-based opinion mining and analysis of airline industry based on user generated reviews

Kanishk Verma
School of Computing
Dublin City University
Dublin, Ireland
kanishk.verma2@mail.dcu.ie
0000-0001-7172-4098

Abstract

Mining opinions from reviews has been a field of ever-growing research. These include mining opinions on document level, sentence-level and even aspect level. While explicitly mentioned aspects in a user-generated review have been widely researched, very little work has been done in gathering opinions on aspects that are implied and not explicitly mentioned. In this paper, the present is a novel study for extracting and analyzing implicit aspects and opinions from airline reviews. Through this study, an airline domain-specific aspect-based corpus and a technique that augments pre-trained word embeddings for sequential with stochastic gradient descent optimized *conditional random fields* was devised and developed. The results of this method were used to extract and classify implied aspects from opinionated texts using *machine and ensemble learning.*

Keywords – *Conditional random field, stochastic gradient descent, machine learning, ensemble learning, implicit aspects, augmenting word embeddings, classification, corpus, sequential labelling*

1 Declaration

Not applicable.

2 Introduction

Travel and tourism are well-liked terms amongst all generations of people. The airline industry is a key facilitator in this domain. For this industry, serving its customers with not only cost-effective but also satisfactory service options is paramount. [1] Opinions are very important to businesses and organizations because they always want to find consumer or public opinions about their products and features. [2] In this 21st information age, with constant development in social and web media, a multitude of platforms are available like *Trip Advisor*, *Airline Ratings* etc. for consumers to express their views on air travel. This serves in favor of the airline companies, as it becomes their one-stop to access rich customer feedback information. However, many times, due to a variety of reasons like paid promotions, fraudulent and unstructured nature of these reviews, insightful information nature of these reviews, insightful information cannot be extracted. So, a need is felt to have a mechanism that gathers cognizance in terms of the perception of customers on airline-specific aspects. [3]

Liu and Zhang et. al. defined the term opinion as *"a concept covering sentiment, evaluation, appraisal, or attitude held by a person"* [2] Aspects and entities are more like topics in a text document. Hu and Liu et. al. coined this type of analysis as feature-based sentiment analysis. [4] Aspect or entity-based analysis identifies the target of the opinion. It is a fine-grained approach to text analysis.

2.1 Paper nomenclature

In this paper, an *entity* is the *feature of the airline* and an *implicit aspect* or sub-aspect is its *attribute*. Examples for entities are food, cabin, seat, staff etc. Since these entities in themselves can have various attributes associated with them. It becomes important to divide them further into sub-aspects or implicit aspects.

For example, a sentence in a review could read *"the cabin was cold, smelly and a bit weary".* Here, the entity *cabin* is accompanied its attributes like *temperature*, *fragrance* and *condition*. The phrases or terms like *"cold", "smelly",* and *"a bit weary"* are terms that imply an opinion to each individual attribute of the entity *cabin.* This paper devises a technique to identify airline-specific entities from such implicit phrases or terms. This approach helps in making a fine-grained analysis of opinions and maps them accurately to respective entity-implicit aspect pair.

2.2 Research motivation

Understand which passenger airline industry-specific aspects can be leveraged for implicit aspect-based opinion mining is one of the key focus of research. Also, how will these implicit aspects be engineered to be annotated[1] in order to build a one of a kind domain specific sentiment corpus. Furthermore, what specific lexicon[2] generation techniques can influence this type of opinion mining.

2.3 Data

Trip advisor and Airline ratings are online microblogging platforms primarily used for viewing reviews and experiences of travelers either travelling to the same destination or other, all over the globe. Usually, people before making airline ticket purchases do read reviews. [4]

In this study, 3000 reviews were collected within a time period of 1 month with an aim to study public's opinion with respect to 16 Airlines (see appendix A). From these 3000 reviews, after curating, only 1803 reviews were determined to be relevant for this study. Detailed statistical analysis was carried on the dataset to understand the quality of it. This statistical analysis information is available in table 1.

Number of reviews	1803
Number of sentences	9591
Average number of sentences in a review	5
Type Token Ratio [5]	0.27
Most common word	Flight
Average word length	7
Labelled unique word corpus size	3280

Table 1. Dataset Statistics

In summary, the goal of this study is to extract implied aspects and opinions from airline reviews. To achieve this goal, a new dataset was created, which to my knowledge, is the first time a dataset specifically for implicit aspects of airline reviews is created. Using a *supervised lexicon-based technique*, few experiments were run to gather insightful information about airline-based implied aspects and opinions. The results of which were favorable for the study. Further in this paper, discussions are on methodology, issues and challenges, experimental setup and evaluations/results of this approach.

3 Methodology

The methodology of this study consists of multiple modules. Each module was developed keeping in mind that the dataset is fresh, new and one of kind. So, the methodology pipeline includes *data collection*, *corpus statistics*, *annotation*, *feature engineering*, *sequence labelling*, and *classification tasks*.

3.1 Entity and Aspect selection

Post dataset statistical analysis, the two annotators carefully read about 500 reviews. Features of the passenger aircraft, services offered by the airlines both in and off the flight were formulated in a list. After curating the list, a data-driven decision led to enlist entities into 8 categories. The representation of these 8 implicit entity-aspect pairs can be found in table 2.

Entity	Implicit Aspect(s)
Food	*Service* *Temperature* *Taste*
Entertainment	*Visual* *Audio* *General*
Cabin	*Condition* *Fragrance* *Size*
In-flight service	*Temperature* *Operations* *Facility*
Off-flight service	*Ticketing* *General* *Facility*
Staff	*Behaviour* *General*
Seat	*Operations* *Comfort*
Possession	*Handling* *General*

[1] Annotation: It is like a metadata tag to markup specific elements in a dataset.

[2] Lexicon: It is a component of natural language processing that contains grammatical information about individual words or strings.

Table 2. Entity-wise implicit aspect list

3.2 Data Annotation

Manual annotation and labelling of all the reviews using Doccano [6] annotation tool was conducted. An inter-annotator agreement guideline [7] was also set up. (See appendix A). Annotation was done on two levels i.e. ***entity level*** and ***implicit aspect level***. So, using Cohen's Kappa coefficient [8] the annotators' agreement level was determined. The results of the Kappa coefficient are available in the Evaluation section.

3.3 Feature Engineering

The feature engineering task was divided in two methods, one to capture word features and the other to gather numeric representations of the word features. (See Appendix B)

Augmenting Word Embeddings

The numeric representations like count vectorizer and TF-IDF are more frequency based and lack contextual information [9] The dataset for this study being small and limited, a need was felt to augment[3]. So, a pre-trained word embedding model with word vector representation of the dataset. So, pre-trained Glove [10] vectors trained on user-generated were used. These pre-trained vectors were augmented with Word2Vec [11], [12] for corpus embeddings. Also, the parameters augmented are one's that considered maximum distance between focus word and its contextual neighbor. (See appendix D)

Word Features	Parts-of-speech tags; Dependency parsing
Numeric word representation	Count Vectorizer; Term Frequency – Inverse Document Frequency; Augmenting Word Embeddings

Table.3 Feature Engineering Tasks

Sequence Labelling with Conditional Random Fields

Sequence Labelling is a supervised learning[4] task where a label is assigned to each element of a sequence. For our study, to extract words and classify them into respective entities, a conditional random fields algorithm was selected. Conditional random fields [13] adjust to a variety of statistically correlated features as input just like a sequential classifier. Also, like a generative probabilistic model it trades-off decisions at different sequence to obtain a global optimal labelling. (See appendix E).

The CRF model was optimized using stochastic gradient descent[5] with L2 regularization[6]. This is done to maximize the likelihood of the CRF and be represented as follows,

$$\log P(Y|X) = w \,.\, \varphi(Y,X) - \left(log \sum\nolimits_{Y^T} e^{w\varphi(Y,X).} \right) [eq.1]$$

After taking derivates on the above equation, we get below,

$$\frac{d}{dw} \log P(Y|X) = \varphi(Y,X) - L^2 \sum\nolimits_{Y^T} P(Y^T|X)\varphi(Y^T,X) \; [eq.2]$$

Where it means $\varphi(Y,X)$ to add correct feature and subtract $P(Y^T|X)$ which is expectation of features and L2 is a regularization penalty term.

Classification for implicit aspect extraction

The aspect extraction task needed classifier models that could accurately predict the aspect. Different algorithms were used to classify and compare how accurate each model was to classify these sub-aspects. Algorithms like Support Vector Machine, Decision Trees, Random Forest, a bagging ensemble learning algorithm Voting Classifier and a boosting ensemble learning algorithm XGBOOST were used. (See appendix F)

[3] Made the word embeddings larger and stronger by adding Glove embeddings

[4] It means learning a mapping between a set of input variables X and output variables Y and applying these mappings, predictions can be made for unseen data

[5] Gradient Descent: An optimization algorithm used to minimize some function iteratively.

[6] L2 regularization: It is a penalty regularization technique which does not let the algorithm over-fit.

4 Data setup

4.1 Data Preprocessing

Using standard pre-processing techniques like removing domain-specific stop words, removal of unnecessary punctuations, spell correction, converting numbers to words, and word standardization. The motivation for doing so was to avoid misleading the training model. Also, since the data was user-generated, there were many contractions of words, for example, "*couldn't"*, "*can't"*, "*aren't"*, "*I'm"* etc., were seen quite often in the texts. So, fixing these contraction words was also a part of the study. Words like "*couldn't"* were replaced by "*could not"*. (See Appendix G)

4.2 Corpus Statistics

The data being user-generated was raw and unstructured. It is the first time this group of reviews was considered text mining and analysis. So, two statistical strategies, viz, type-token ratio [5] and Zipf's distribution [14] were used to determine variability in the dataset.

Type Token ratio (TTR) is represented as follows, (See Appendix H)

$$TTR = \frac{(number\ of\ types)}{(number\ of\ tokens)}\ [eq.3]$$

Source	TTR Score
Trip Advisor	0.35
Airline Ratings	0.37

Table 4 Type Token Ratio Scores

TTR Scores are low for both data sources, this means that there are many repeated terms in the corpus. (See appendix H) Zipf's law states that a relationship between frequency of word (f) and its position in the list i.e. its rank (r) is inversely proportional to one another.

$$f\ \propto \frac{1}{\mathrm{r}}\ [eq.4]$$

4.3 Manual Annotation

As explained in the methodology, the annotation was done on two levels using Doccano software[6] There are detailed examples and explanation of this manual annotation strategy.

INPUT: "Overall the experience was comfortable and spacious with delicious meals" **Output:** [("experience was comfortable", "Inflight"), ("spacious", cabin), ("delicious meals", "food)]

Table 5. Detailed example of Level 1 annotation

Once, entity-level tuples[7] were tagged containing a word or word phrases with entity-name, as seen in Table 5. After completing entity level annotation, another fine-grained approach to classify entity-wise word or word phrases to their respective implied aspects was conducted, details of which are available in table 6.

INPUT: [("experience was comfortable", Inflight), ("spacious", cabin), ("delicious meals", "food)] **OUTPUT:** [[("experience", inflight-operations), ("comfortable", inflight-operations)], [("spacious", cabin-size)], [("delicious", food-taste), ("meals", food-service)]]

Table 6. Detailed example of Level 2 annotation

4.4 Inter-Annotator Agreement

As explained in the methodology of this experimental study, after adhering with the guidelines in the inter-annotator agreement, and using SK-Learn Kappa score library, the Cohen's Kappa [8] score for level of agreement was calculated.

5 Experimental Setup

5.1 Training Data Preparation

The experiment study used techniques described in the methodology section for preparing the training data. Taking an example sentence, this process will be explained in detail Example sentence: "*Overall, the experience was comfortable and spacious with delicious meals."*

Entity Level	
Entity	Word/Phrases

[7] Tuples are a data type that is similar but also distinct from the list data type. The instances are characterized by having fixed attributes and the elements of a tuple instance can differ in data type amongst one another.

In-flight service	Experience was comfortable
Cabin	Spacious
Food	Delicious meals
Implicit Level	
Aspect	Word
Inflight Operations	Experience
Inflight Operations	Comfortable
Cabin Size	Spacious
Food Taste	Delicious
Food Service	Meals

Table 7. Annotated and labelled list of example sentence

From this review, words like experience, comfortable, spacious, delicious, and meals were identified as aspect terms and their semantic and syntactic information was extracted by parsing them through off-the-shelf state of the art models like Stanford Core NLP API [15] to extract part-of-speech (POS) tags and dependency tags and Vader for sentiment score. (See appendix B)

Using these techniques, a list of features was generated which consisted of main-word, main-word POS tag, dependent word, dependent word POS tag, main-word sentiment score, dependent word sentiment score, dependency tag, previous, and next word

For the task of sequence labelling to identify the entity a word or word phrase belongs to, the tuples were added with their respective labels i.e. the label added to a tuple was the label that word belonged to.

For example, Tuple, "*("delicious", "JJ", "meals", "NNS", 0.6, 0.0, "advmod", "spacious", "meals)*" has main word *food*, so a new entry to this was made as *"f"*, which became the *Y* or dependent variable. After getting results from the CRF model, the entity-id i.e. if it was classified as "*food*" so its id was "*f*".

Entity	Entity-ID
Food	f
Cabin	c
Entertainment	e
Staff	st
Seat	s
Off-flight	o
In-flight	i
Possession	p

Table 8. Entity ID List

Once the correct entity is identified, the next step is to classify which aspect is mentioned in the sentence. Then to the word feature tuple, ENTITY-ID is added to the training data and it is then vectorized.

5.2 Count Vectorization

For this experiment study, since the methodology does try to keep certain punctuations and special characters, a need is felt to create its own tokenizer. The results for an example sentence, as follows

Sentence: *"so overall I highly recommend this airline"*

Vectors: {"*so"*:5, *"overall"*:3, *"I":*2, *"highly"*:1, *"recommend":*4, *"this" :*6, *"airline" :*0}

5.3 TF-IDF Vectorization

For this experiment study, the TF-IDF score for the words in the feature sets was calculated using sci-kit TF-IDF vectorizer. Table 9 has the result of TF-IDF scores for all corpus words.

Word	TF-IDF score
Basic	0.965545
Redemption	0.965545
Rescue	0.958253

Table 9. TF-IDF Vectorization

5.4 Augmented Word Embeddings

As mentioned in the methodology, the corpus of this experimental study was small. So, a word embedding model using Word2Vec for the corpus was trained. And a pre-trained Twitter Glove Embeddings consisting of a vocabulary size of 1.2 million words and 27 billion tokenized twitter words with a 100-dimensional vector was selected.

Using the algorithm 1, a set of new vector embeddings were merged using pre-trained Glove and corpus Word2Vec embeddings.

Algorithm 1 *Word embedding vector generation using pre-trained glove vectors*

Inputs

$S = [W_1, W_2, W_3, ..., W_n]$, *Input sentence S contains n words*

path = path where downloaded embeddings are stored

GloveVec = Pretrained Glove Vectors

Output

Word2Vec Embedding Model

```
// Load the Glove Pre-trained Vectors
with open(path):
  gloveVec = embedding vectors // Create Word2Vec Embedding for Airline Corpus
word2vec = Word2Vec Create Model

for word, vector in zip(word2vec.index2word, word2vec.vectors) do
 | w2v = dict(word: vector)
end
// Vectors for airline Corpus are loaded
for each W_i in S do
  if W_i exists in gloveVec then
    extract vecW_i  MV_i = vecW_i end                                  end
  else if W_i exists in w2v then
    extract vecW_i  MV_i = vecW_i end
  else
    extract vecW_i  MV_i = generateNewvecW_i end
```

With this algorithm 1, a new set of word embeddings were generated to vectorize textual information in the feature tuple.

5.5 Cosine Similarity Index

Along with the word embeddings, cosine similarity between main and dependent word was added as a new feature. (See Appendix D).

These new features were then used to classify opinionated texts into their respective implicit-aspect classes.

5.6 Handling Class Imbalance

After annotation, there was a high imbalance amongst implicit aspect classes of almost all entities. The imbalance for the entity *cabin* was handled using an oversampling technique called Synthetic Minority Oversampling Technique. [16] (See appendix F). SMOTE was performed for all 8 entities.

Results of SMOTE imbalance handling for the entity: *Cabin* is as follows

Class: {"*Condition"*:182, *"Size"*:182, *"Temperature"*: 117, *"Fragrance"*: 102}

This could be visualized as a scatter distribution shown in figure 3

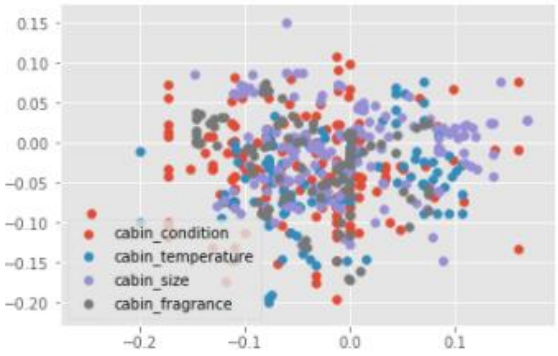


Figure 1. Cabin class imbalance rectified with SMOTE

5.7 Implicit Aspect Classification

A total of 8 models were created for each entity i.e. there are independent classification models for training each entity. The reason for creating 8 models is to devise a perfect model for recognizing and classifying each *Entity* with its own *Implicit Aspect.*

This experiment study makes use of state-of-the-art classification algorithms. Three of which were ensemble learning techniques. These include Gradient boosting algorithm – XGBOOST, a Voting Bagging algorithm using three tree-based classification techniques like Decision Trees, Random Forest, and Extra Trees Classifier. And other machine learning techniques like SVM, Decision Tree.

The reason for using these different algorithms was to gather insightful information on the performance of classification which was evaluated based on ROC-AUC [17] score and F1 [18] scores. (See Appendix I)

6 Evaluation and Results

This experimental study using state-of-the-art techniques and algorithms is a new approach to mine and extract implicit aspects from opinionated texts.

The first evaluation was for the annotation of the dataset using Cohen's Kappa Co-efficient. The two annotators' agreement scores ranged from 80.48% to 82.13% for entity level and implicit aspect level annotation. (See Appendix A.

The second evaluation was for the sequence labelling task using stochastic gradient descent with L2 regularization Conditional Random Field. This was done to classify texts in 8 different entities.

The ROC-AUC score achieved for this task is 96.5% and an F1 score of 94.56%. (Appendix I)

The third evaluation was for the classification task using five different classification algorithms. (See Appendix I)

A detailed ROC-AUC score evaluation metric is available in Table 10 (Highlighted in green provides best score)

Entity	Algorithms				
	S	D	R	V	X
Food	84%	92%	94%	94.8 %	94.7 %
Cabin	75 %	75 %	85%	85.6 %	77 %
Entertainment	73.6 %	79.9 %	83.1 %	84.3 %	85.9 %
In-flight	60.3 %	70.3 %	72.2 %	74.9 %	71.2 %
Off-flight	66.4 %	86.2 %	84.9 %	84.8 %	89.8 %
Possession	66.9 %	66.9 %	70.5 %	73.3 %	73.4 %
Seat	66 %	73.7 %	75 %	75.7 %	78%
Staff	75.6 %	76.9 %	80.9 %	82.1 %	81.4 %

Table 10 ROC-AUC Scores for classification of entities

In table 10, S stands for Support Vector Machines, D for Decision Trees, R for Random Forest, V for Voting Classifier, and X for XGBOOST algorithms. In all these machine learning and ensemble learning classification algorithms, the bagging technique of ensemble using tree-based classifiers has out-performed all other classification algorithms. (Appendix I)

7 Issues and Challenges

Manual annotation was a big challenge. Being humans, everyone has a different outlook on implied meanings. One can think of words like "*boarding, de-boarding, take-off", and "landing"* as in-flight operations. But, if a person takes a minute to read the review and understand the concept, "*boarding", "de-boarding", "take-off"*, and *"landing"* are off-flight facilities provided by the airlines. So, using corpus statistic techniques and adhering to the inter-annotator guidelines the annotators made mutually agreeable decisions. (See Appendix A)

The word *spacious* in this study made it a bit challenging for the labeling sequence task. It is a word that occurred quite frequently in the reviews. Also, if used within the same sentence or context of "*cabin"* it means that the *"cabin"* was big implying to its attribute *"size"* and in context of *"seat"*, it means that the *"seat"* had enough leg room implying to its attribute of "*comfort"*. This word has two implicit meanings. This is a double implicit problem. Such a problem was tackled by making use of T-distributed stochastic nearest neighbors for word embeddings clustering techniques [19] This clustering technique allows word distances of such words to be mapped with each implicit aspect-entity pair. Wherever the words were close, it was mapped to the respective implicit aspect-entity pair. (See appendix D).

For example, *"spacious"* occurs in the same vector space as of "*size"* for cabin and "*comfort"* for the seat. So, the word cosine distance between spacious, size and comfort was added as a feature to the occurrence of the word spacious.

8 Related Work and Improvements

Our research concentrates on implicit aspect extraction, opinion lexicon generation, and engineering an annotated implicit aspect-based sentiment corpus that can influence implicit opinion mining from consumer reviews in the airline industry. Few studies that are done in this realm of implicit aspect-based opinion mining and extraction but very few on implicit aspect-based opinion mining

In a research study proposed by *Chinsha T C et al.*[20] the methodology proposes a syntactic based approach using dependency parsing[8]. In other research for comparing word representations for implicit classification [21] Both these studies use *SentiWord Net* and have dataset restrictions. The present study intends to extend the results of these two papers. By using a syntactic approach to group implicit aspect synonyms for a larger dataset. As the two studies were restricted to 170 and SemEval dataset respectively.

Research dealing with the double-implicit problem[9] in opinion mining and sentiment analysis proposes a protocol to derive a labelled

[8] Dependency parsing: A methodology that is used to extract grammatical structure from sentences.

[9] Double-implicit: Word or word-phrases that not only describe an entity but also the opinion of the entity.

corpus for implicit polarity and aspect analysis. [22] The work in this paper is limited to only Chinese restaurant reviews. The present study addresses not only the dataset limitation but also the labelling of the corpus technique by using Type/token Ratio and other corpus statistic techniques which are explained in the experimental setup section 4.

Another study using two corpora proposed a hybrid model to support Naïve Bayes training to identify implicit aspects[23] This corpus and dictionary-based approach is limited to only adjective type words of a sentence. The present study extends this work by taking considering a combination of adjectives, adverbs, nouns, and other part-of-speech indicators and uses ensemble learning for classification

A study conducted on implicit aspect indicator extraction, models relations between the polarity of a document and its opinion target using Conditional Random Field (CRF)[24] This method is limited however to only cellular device data and the entities are picked from a pre-trained Stanford CRF model. Our work extends Conditional Random Field and extends it to the airline domain

9 Conclusion and future work

The present research study using a supervised machine learning approach provides a novel technique to overcome the implicit opinion and aspect mining problem. It does so by, identifying eight different airline industry-specific aspects that can be leveraged for the task of opinion mining. They include fine-grained entities like the cabin, entertainment, food, in-flight service, off-flight service, seat, staff, and possessions. The annotation is done on two levels, one on the entity level and the other is on the sub-aspect level, which allows for a more detailed label construction. The two annotators in this experiment study have a very good agreement on annotated terms. This can be reflected by Cohen's Kappa score ranging from 0.77 to 0.80. So, it can be said that the corpus derived from this study, can be used as a gold standard for implicit aspect-based mining tasks for airline reviews.

This experimental study presents a novel approach of dividing the implicit aspect-based opinion mining task into two levels, one using stochastic gradient descent with L2 regularization for improving conditional random fields to identify entities. This is done with a ROC-AUC Score of **96.58%**, F statistic score of **94.56%**, and with **0.01** degrees of a mean absolute error on testing data. The second level is to classify each entity into an implicit aspect sub-group. For this state-of-the-art machine and ensemble learning algorithms are used. From the experiments, it is found that ensemble learning outperformed the machine learning approaches. The ROC-AUC scores for ensemble learning algorithms like Voting Classifier range from **73% to 94.8%** and the boosting algorithm like XGBOOST range from **71% to 94.7%** for all eight entities. Synthetic Minority Oversampling technique proved to be an effective performance improver for the classification and extraction of implicit aspects tasks.

The scope of this experimental study is limited to a few reviews, as possible future work, another study can carry forward the methods proposed in this paper to a larger dataset. Also, another possible future work can be implementing a neural architecture of these proposed methods.

10 Conflict of Interest

The author Kanishk Verma declares that there is no conflict of interest for the presented research study.

11 Ethics approval

The present study has successfully followed all research ethics and post receiving of approval from the university, the study continued. Prior to gathering data from the websites, written confirmation and approval was collected from TripAdvisor and Airline Ratings.com

12 Availability of data

The manually annotated corpus data is publicly available under Creative Commons Attribution 4.0 International through Zenodo: https://zenodo.org/record/4126975#.X5RR4IhKjIU

13 Funding

This proposed study was conducted under the guidance and supervision of Dr. Brian Davis at Dublin City University for thesis purposes.

14 Bibliography

[1] F. Misopoulos, M. Mitic, A. Kapoulas, and C. Karapiperis, "Uncovering customer service experiences with Twitter: the case of airline industry," *Management Decision*, vol. 52, no. 4, pp.

705–723, Jan. 2014, doi: 10.1108/MD-03-2012-0235.

[2] B. Liu and L. Zhang, "A Survey of Opinion Mining and Sentiment Analysis," in *Mining Text Data*, C. C. Aggarwal and C. Zhai, Eds. Boston, MA: Springer US, 2012, pp. 415–463.

[3] K. Wongleedee, "Customer satisfaction in the airlines industry: comparison between low-cost and full service airlines," *Актуальні проблеми економіки*, no. 1, pp. 218–222, 2017.

[4] M. Hu and B. Liu, "Mining and Summarizing Customer Reviews," p. 10.

[5] M. C. TEMPLIN, *Certain Language Skills in Children: Their Development and Interrelationships*, NED-New edition., vol. 26. University of Minnesota Press, 1957.

[6] *doccano/doccano*. doccano, 2020.

[7] "Natural Language Annotation for Machine Learning [Book]." https://www.oreilly.com/library/view/natural-language-annotation/9781449332693/ (accessed Aug. 16, 2020).

[8] A. Rosenberg and E. Binkowski, "Augmenting the kappa statistic to determine interannotator reliability for multiply labeled data points," in *Proceedings of HLT-NAACL 2004: Short Papers*, Boston, Massachusetts, USA, May 2004, pp. 77–80, Accessed: Aug. 15, 2020. [Online]. Available: https://www.aclweb.org/anthology/N04-4020.

[9] O. Levy, Y. Goldberg, and I. Dagan, "Improving Distributional Similarity with Lessons Learned from Word Embeddings," *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 211–225, 2015, doi: 10.1162/tacl_a_00134.

[10] J. Pennington, R. Socher, and C. Manning, "GloVe: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, Oct. 2014, pp. 1532–1543, doi: 10.3115/v1/D14-1162.

[11] Y. Goldberg and O. Levy, "word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method," *arXiv:1402.3722 [cs, stat]*, Feb. 2014, Accessed: Apr. 22, 2020. [Online]. Available: http://arxiv.org/abs/1402.3722.

[12] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin, "Advances in Pre-Training Distributed Word Representations," *arXiv:1712.09405 [cs]*, Dec. 2017, Accessed: Apr. 21, 2020. [Online]. Available: http://arxiv.org/abs/1712.09405.

[13] J. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," p. 10.

[14] D. M. W. Powers, "Applications and Explanations of Zipf's Law," 1998, Accessed: Aug. 16, 2020. [Online]. Available: https://www.aclweb.org/anthology/W98-1218.

[15] "Document (Stanford CoreNLP API)." https://nlp.stanford.edu/nlp/javadoc/javanlp-3.5.0/edu/stanford/nlp/dcoref/Document.html (accessed Aug. 17, 2020).

[16] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *jair*, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/jair.953.

[17] S. Wu and P. Flach, *A scored AUC Metric for Classifier Evaluation and Selection*. 2005.

[18] Z. C. Lipton, C. Elkan, and B. Narayanaswamy, "Thresholding Classifiers to Maximize F1 Score," *arXiv:1402.1892 [cs, stat]*, May 2014, Accessed: Aug. 17, 2020. [Online]. Available: http://arxiv.org/abs/1402.1892.

[19] S. Smetanin, "Google News and Leo Tolstoy: Visualizing Word2Vec Word Embeddings with t-SNE," *Medium*, Nov. 16, 2018. https://towardsdatascience.com/google-news-and-leo-tolstoy-visualizing-word2vec-word-embeddings-with-t-sne-11558d8bd4d (accessed Apr. 23, 2020).

[20] C. T. C and S. Joseph, "A syntactic approach for aspect based opinion mining," in *Proceedings of the 2015 IEEE 9th International Conference on Semantic Computing (IEEE ICSC 2015)*, Feb. 2015, pp. 24–31, doi: 10.1109/ICOSC.2015.7050774.

[21] C. Braud and P. Denis, “Comparing Word Representations for Implicit Discourse Relation Classification,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, 2015, pp. 2201–2211, doi: 10.18653/v1/D15-1262.

[22] H.-Y. Chen and H.-H. Chen, “Implicit Polarity and Implicit Aspect Recognition in Opinion Mining,” in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Berlin, Germany, Aug. 2016, pp. 20–25, doi: 10.18653/v1/P16-2004.

[23] E. H. Hajar and B. Mohammed, “Hybrid approach to extract adjectives for implicit aspect identification in opinion mining,” in *2016 11th International Conference on Intelligent Systems: Theories and Applications (SITA)*, Oct. 2016, pp. 1–5, doi: 10.1109/SITA.2016.7772284.

[24] I. Cruz, A. Gelbukh, and G. Sidorov, “Implicit Aspect Indicator Extraction for Aspect-based Opinion Mining,” p. 18.

Appendix

Appendix A.

Refer to the Inter-annotator guideline agreement[1] for annotation rules as per the project scope.

Doccano

Doccano is an open source text annotation tool which is used to label, or annotate text data for:

- Text Classification
- Entity Extraction
- Sequence to Sequence translation

It has an easy to use GUI for annotating text data. First, the text data is loaded onto the software by creating a new project. Then, the entities are defined by the user. Doccano gives us the liberty to add colors and custom keyboard shortcuts to the entities, so as to assist in the manual task. Then, the text is manually labelled against the set entities. Once completed, the data can be saved in JSON format, that is JavaScript Object Notation. For each annotated project, two files are saved. One contains the original text data along with the entities and tagged words. The second file gives an insight of total counts for tags of each entity in the whole project.[1]

Entity-Aspect Selection:

500 reviews were read carefully, enlisting broad categories that were reviewed. Based on the classification of the reviews, 8 entities were observed to be significant. As majority of the reviews talked about food, staff, seat, cabin, entertainment, in-flight service, off-flight service, the same naming convention was followed to name the entity groups. For example, all the words and phrases that indicate opinions about the food served in the flight are annotated under the entity "food". Following is a wordcloud for "Food" entity. A wordcloud is a graphical representation of word frequency, such that the most common, or the most repeated word has the greatest prominence by size in the figure.

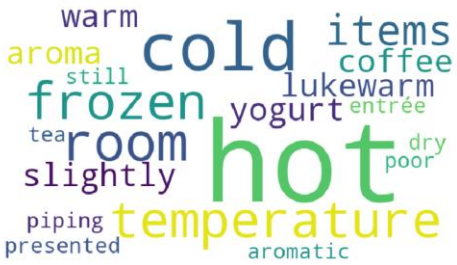
Food:

Similarly, at a step further in the hierarchy, these entities were classified based on the aspect the opinion focused on. For example, most of the reviews that talked about food, notified the reader towards the food temperature, taste etc. As a result, after comparing the number and weightage of each opinion, the Food entity was classified into 3 aspects- temperature, taste and service. This categorization was chosen as it classified all the opinions about food. Following are the word clouds for all the aspects of Food entity:

Food Service:

Food taste:

Food temperature:



The main check is to note that at all times, the words in either of the categories should not match with the words in any other category. This way, a clear and distinct boundary is maintained between the categories.

Following a similar process, wordcloud for all entities are prepared:

Seat

Seat Aspects:

Comfort

Operations

Possession

Possession aspects:

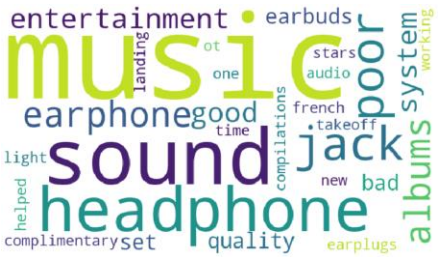
Handling

Operations

Entertainment

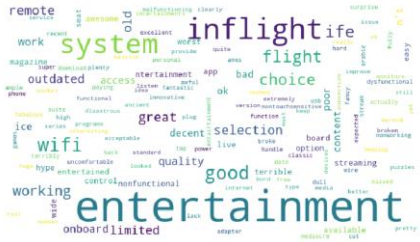
Entertainment Aspects:

Audio



Visual

General



Staff

Staff Aspects:

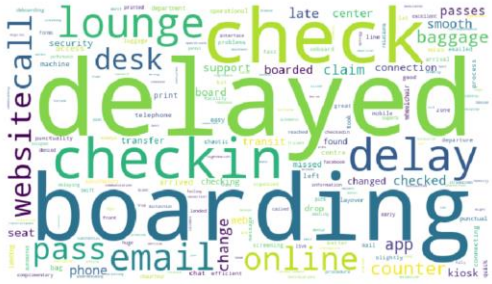
Behaviour

General

Off-flight Service

Off-flight Service Aspects:

Facility



General

Ticketing

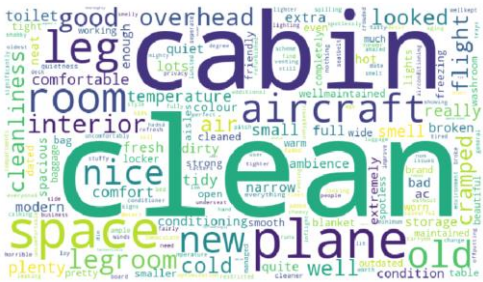
In-flight Service

Inflight Service Aspects:

Facility

Operations

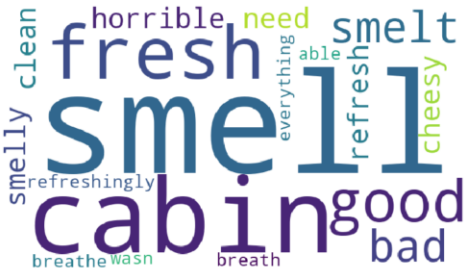
Cabin



Cabin Aspects:

Condition

Fragrance



Size

Temperature

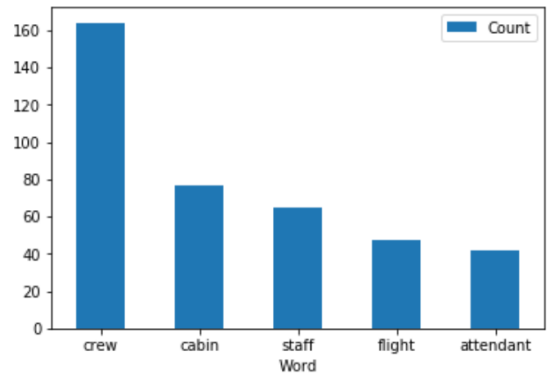


Figure: 5 most used words for "Staff" entity

Appendix B.

Cohen's Kappa Coefficient

Since the scope of the project followed supervised learning, which requires labelled data to train the

models, the process of annotating/labelling the text data with the respective entities was needed. For any supervised machine learning model to have high accuracy, it is important that the labels consistency and data integrity is maintained throughout the training data. An inter-annotator agreement was formed to make sure that both the annotators label the data in a similar manner. To keep a track of how similar the annotations are, Cohen's Kappa coefficient is used.[2]

The Kappa coefficient value for annotating entities came out to be 0.8048, whereas the Kappa coefficient value for annotating respective aspects came out to be 0.8213.

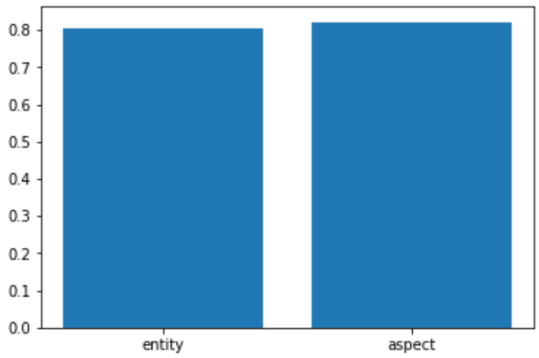


Figure: Kappa Co-efficient for entity and aspect

Furthermore, Kappa coefficient is also calculated as per each entity, to take into account any particular entity being labelled differently by the two annotators. These are plotted in a bar plot below:

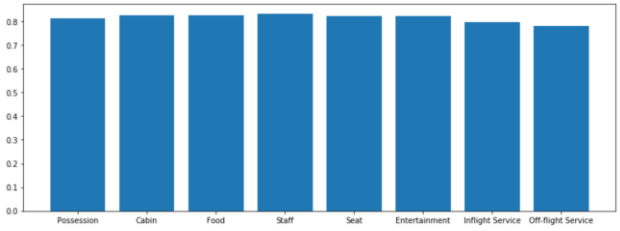


Figure: Kappa Co-efficient for all entities

Highest Kappa coefficient value is observed for 'Staff' entity, which is approximately 0.8317, and the lowest is observed for 'Off-flight Service', approximately 0.7814.

Since, all the coefficient value have a score greater than 0.75, it is safe to conclude that the annotations are quite similar in nature.

Pos Tag

The POS Tagger reads the text and assigns parts of speech to all the words appearing in the vocabulary. These parts of speech could be noun, verb, adjective etc. The Penn Treebank tag set by Stanford specifies certain naming conventions for all the parts of speech.[3]

Tanle I

Abbreviations of part-of speech tags- Penn Treebank Tag Set

Serial Number	POS Tag	Description
1	CC	Coordinating conjunction
2	CD	Cardinal number
3	DT	Determiner
4	EX	Existential *there*
5	FW	Foreign word
6	IN	Preposition or subordinating conjunction
7	JJ	Adjective
8	JJR	Adjective, comparative
9	JJS	Adjective, superlative
10	LS	List item marker
11	MD	Modal
12	NN	Noun, singular or mass
13	NNS	Noun, plural
14	NNP	Proper noun, singular
15	NNPS	Proper noun, plural
16	PDT	Predeterminer
17	POS	Possessive ending
18	PRP	Personal pronoun
19	PRP$	Possessive pronoun
20	RB	Adverb
21	RBR	Adverb, comparative
22	RBS	Adverb, superlative
23	RP	Particle
24	SYM	Symbol
25	TO	*to*
26	UH	Interjection
27	VB	Verb, base form

28	VBD	Verb, past tense
29	VBG	Verb, gerund or present participle
30	VBN	Verb, past participle
31	VBP	Verb, non-3rd person singular present
32	VBZ	Verb, 3rd person singular present
33	WDT	Wh-determiner
34	WP	Wh-pronoun
35	WP$	Possessive wh-pronoun

Dependency Parsing

Dependency Parsing helps to describe the grammatical relationships between the words of a sentence, by specifying what are called "*dependencies*". These dependencies are binary relations- a grammatical relation that holds among a governor/head and a dependent. The following is the list to map the abbreviations of the Dep tags to their respective description.[4]

Table II

Universal Dependency Relations

Serial Number	Dependency Tag	Description
1	acl	clausal modifier of noun
2	advcl	adverbial clause modifier
3	advmod	adverbial modifier
4	amod	adjectival modifier
5	appos	appositional modifier
6	aux	auxiliary
7	case	case marking
8	cc	coordinating conjunction
9	ccomp	clausal complement
10	clf	classifier
11	compound	compound
12	conj	conjunct
13	cop	copula
14	csubj	clausal subject
15	dep	unspecified dependency
16	det	determiner
17	discourse	discourse element
18	dislocated	dislocated elements
19	expl	expletive
20	fixed	fixed multiword expression
21	flat	flat multiword expression
22	goeswith	goes with
23	iobj	indirect object
24	list	list
25	mark	marker
26	nmod	nominal modifier
27	nsubj	nominal subject
28	nummod	numeric modifier
29	obj	object
30	obl	oblique nominal
31	orphan	orphan
32	parataxis	parataxis
33	punct	punctuation
34	reparandum	overridden disfluency
35	root	root
36	vocative	vocative
37	xcomp	open clausal complement

Appendix C.

Count Vectorizer:

Here, the collection of text reviews is converted into a matrix of token counts. The basic operation of this technique is to check each word in the document and count the number of their representations and create a matrix of these counts.

For this experiment study, since the methodology does try to keep certain punctuations and special characters, a need is felt to create own tokenizer. The results for an example sentence:

Table

Count Vectorizer using Sci-kit Learn

Sentence: *'so overall I highly recommend this airline'*						
So	Over-all	I	High-ly	Recomm-end	Th-is	airline
5	3	2	1	4	6	0

TF-IDF

It is commonly referred as TF-IDF. It can be divided as two terms namely, Term Frequency and Inverse Document Frequency.

Term Frequency (TF) can be defined as a ratio of count of the word present in a sentence to the length of the sentence.

Inverse Document Frequency (IDF) can be termed as measure of rareness of a term in the corpus. Article words like "a", "an" or "the" appear in almost every corpus, but rare words might not be present in all documents.

Appendix D.

Word Embedding

Word embedding as the name suggests is a collective name for language modelling and feature engineering techniques of Natural Language Processing. In this technique, the word phrases are mapped to vectors of real numbers.[5]

Before going in the details of our methodology for implementation of word embeddings, there are certain terminology that needs to be understood in context of word embeddings.

Language Model: The concept of a language model has a probabilistic character. It is essentially described as a function that provides a probability distribution of strings drawn from a vocabulary[1].

Vector Space Models: An algebraic model to represent text documents as vector of identifiers. Documents can be represented as

$d_j = (w_{1,j}, w_{2,j}, w_{3,j}, .., w_{t,j})$, wherein each dimension is a separate term in the document.

Distributional Semantics: In 1954, Harris stated that the basis of distributional semantics is distributional hypothesis i.e. similarity of distribution in linguistics is resulted by similarity in meaning.

n-gram: They are essentially sequencing of characters or words extracted from a text. It can be deduced as a set of n consecutive characters from a word.

Since this experiment study has limited and a small size of corpus, a decision was made for using pre-trained Twitter Glove vectors. The approach for this experiment study includes training a Word2Vec model for the experiment corpus on-top of the pre-trained Twitter Glove vectors.[6]

CBOW or Continuous Bag of Words: It is a methodology that tends to predict the probability of a word given a context. A context can either be a single or a group of words. The objective function of CBOW language model is as follows

$$J_\Theta = \frac{1}{T}\sum_{t=1}^{T} \log\ p\ (w_t |\ w_{t-n}, .., w_{t-1}, w_{t+1}, .., w_{t+n})$$

Where, a training corpus containing a sequence of T training words $w_1, w_2, w_3, ..., w_T$ that belongs to vocabulary V of size |V| and Θ is the parameters of the model.

Advantages of using CBOW:

1. Generally, it performs superior to deterministic methods because of its probabilistic nature.

2. Unlike a co-occurrence matrix, it does not have huge RAM requirements.

Limitations of using CBOW:

1. For example, the word Apple can mean both fruit and company. CBOW will take an average of both contexts and place it in the middle of a cluster of both these entities.

2. Optimization is highly important, else the training using a CBOW model will take forever.

Skip Gram: The aim of a skip gram language model is to predict the context given a word. It follows the inverse of CBOW's architecture. In simpler terms, skip-gram model will use the centre word

[1] Vocabulary: Set of unique words in a text corpus is referred to as a vocabulary.

to predict the surrounding words, unlike a CBOW model which uses surrounding words to predict centre word.

The skip-gram objective function sums up the log probabilities of the surrounding n words to the right and left of the target word w_t and can be represented as below

$$J_\theta = \frac{1}{T}\sum_{t=1}^{T}\sum_{-n \le j \le n} \log\, p\left(w_{t+j}|w_t\right)$$

So, instead of computing P (w_t) target word given w_{t+j} surrounding words, skip-gram computes surrounding word given target word.

Negative Sampling:

Let's consider, a pair (w, c) where w and c determine word and context respectively.

If the pair of word and context derive from the training data then it can be notated as

P (D=1 | w,c) - (a)

and if the word pair does not come from training data then it can be simply represented as

P (D = 0 | w,c) – (b)

So, from equations a & b, one can rewrite b as

P (D=0 | w,c) = [1 – P(D=1 | w,c)]

Assuming, there are Θ parameters controlling this distribution and can be represented as follows,

P (D = 1 | w,c, Θ)

The goal is to make all observations come from training data. And in order to do so, we have to maximise this probability and it can be denoted as below

$$\arg max_\theta \; \Pi_{(w,c)\epsilon D}\, P\,(D = 1\,|w,c;\Theta)$$

$$= \arg max_\theta \log \Pi_{(w,c)\epsilon D}\, P\,(D = 1\,|w,c;\Theta)$$

$$= \arg max_\theta \sum_{(w,c)\epsilon D} P(D = 1\,|w,c;\Theta)$$

Using soft-max[2] distribution, above equation can be rewritten as follows,

$$\mathrm{P}\,(\mathrm{D}\; =\; 1\,|\mathrm{w},\mathrm{c};\;\Theta) = \frac{1}{1+\; e^{-Vc*Vw}}$$

This can be represented as objective function as follows,

$$\arg max_\theta = \sum_{(w,c)\epsilon D} \log \frac{1}{1+\; e^{-Vc*Vw}}$$

The only limitation of the above is that it allows same (w,c) pair combinations to occur.

So, ahead a mechanism will be developed that prevents vectors with same value. This can be achieved by introducing (w,c) pairs that are not in the data. So generate new pairs which are not in training data and are represented as below

$$D^1 = random(w,c)pairs$$

Since, these pairs are assumed to be incorrect, this approach is named as negative sampling and the objective function can now be optimized as below,

$$\arg max_\theta\, \Pi_{(w,c)\epsilon D}\; p(=1\,|c,w;\;\theta)\,.\,\Pi_{(w,c)\epsilon D}\; P(D = 0\,|c,w;\;\theta)$$

$$= \arg max_\theta\, \Pi_{(w,c)\epsilon D}\; p(D = 1\,|c,w;\;\theta)\,.\,\Pi_{(w,c)\epsilon D}\; [1 - P(D = 1 \mid \mathrm{c,w};\Theta)$$

$$= \arg max_\theta \sum_{(c,w)\epsilon\, D^1} \log P\,(D = 1|c,w;\Theta) + \sum_{(c,w)\epsilon\, D^1} \log P\,(D = 0|c,w;\Theta)$$

$$= \arg max_\theta \sum_{(c,w)\epsilon\, D^1} \log P\,(D = 1|c,w;\Theta) + \sum_{(c,w)\epsilon\, D^1} \log[1 -\; P\,(D = 0|c,w;\Theta)\,]$$

[2] Soft-max: It is a normalized exponential function that takes vector a of R real numbers as input and normalizes it into a probability distribution consisting of R probabilities proportional to the exponential of input numbers

$$= \arg max_{\Theta} \sum_{(c,w)\in D^1} \log \frac{1}{1 + e^{-Vc*Vw}} + \sum_{(c,w)\in D^1} \log[1 - \frac{1}{1 + e^{-Vc*Vw}}]$$

$$= \arg max_{\Theta} \sum_{(c,w)\in D^1} \log \frac{1}{1 + e^{-Vc*Vw}} + \sum_{(c,w)\in D^1} \log \frac{1}{1 + e^{Vc*Vw}}$$

Replacing, $\frac{1}{1+e^{-x}}$ $by\ \sigma(x), we\ get$

$$\arg max_{\Theta} \sum_{(c,w)\in D^1} \log \frac{1}{1 + e^{-Vc*Vw}} + \sum_{(c,w)\in D^1} \log \frac{1}{1 + e^{Vc*Vw}}$$

$$= \arg max_{\Theta} \sum_{(c,w)\in D^1} \log\ \sigma(V_c . V_w) + \sum_{(c,w)\in D^1} \log\ \sigma(-V_c . V_w)$$

The aim is to represent that $D\ \cup D^1$ depicts the entire corpus.

Context Window

The context window determines which contextual neighbors are taken into account when estimating the vector representations

context window is the maximum window size (i.e. the maximum distance between the focus word and its contextual neighbors). This parameter is the easiest one to adjust using existing software, which is why it is comparatively well studied. Larger windows are known to induce embeddings that are more 'topical' or 'associative', improving their performance on analogy test sets, while smaller windows induce more 'functional' and 'synonymic' models, leading to better performance on similarity test sets.

Visualizing elements of the Word2Vec Model

Cosine Similarity- computes similarity between a simple mean of the projection weight vectors of the given words and the vectors for each word in the model. The method corresponds to the word-analogy and distance scripts in the original word2vec implementation. It is a metric used to measure how similar the documents are irrespective of their size

$$Cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\|\ \|\vec{b}\|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2}\ \sqrt{\sum_1^n b_i^2}}$$

where, $\vec{a} \cdot \vec{b} = \sum_1^n a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n$ is the dot product of the two vectors.

The higher the value of COS Theta, the higher the similarity.

```
def cosine_distance_wordembedding_method(s1, s2):
    import scipy
    vector_1 = np.mean([w2v[word] for word in s1],axis=0)
    vector_2 = np.mean([w2v[word] for word in s2],axis=0)
    cosine = scipy.spatial.distance.cosine(vector_1, vector_2)
    print('Word Embedding method with a cosine distance asses that our two sentences are similar to',round((1-cosine)*100,2),'%')
```

Figure: Code to calculate Cosine Similarity

The result for the Adjective-Noun pairs comes out to be 73.21%.

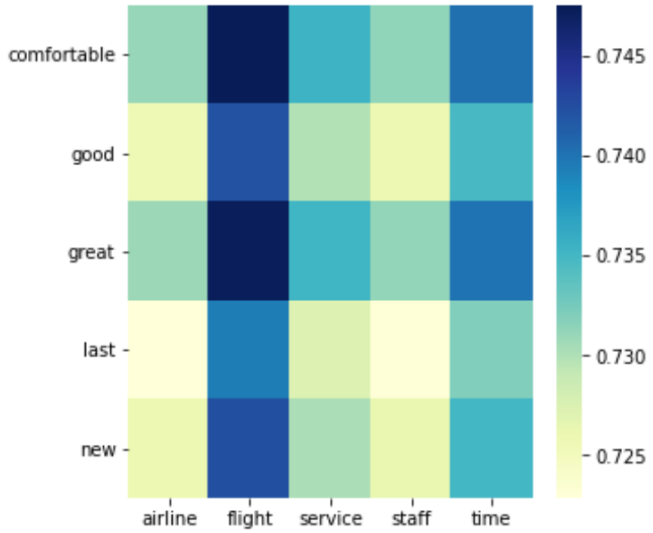


Figure: Heatmap between Adjectives and Nouns

T-SNE (t-distributed stochastic neighboring embedding)

T-SNE is quite useful in case it is necessary to visualize similarity between objects which are located into multidimensional space. With a large dataset, it is becoming more and more difficult to make an easy-to-read t-SNE plot, so it is common

practice to visualize groups of the most similar words.

Hyperparameters of T-SNE:

- **perplexity**: It is a value which in context of T-SNE, may be viewed as a smooth measure of the effective number of neighbours. It is related to the number of nearest neighbours that are employed in many other manifold learners
- **n_components**: dimension of the output space
- **n_iter**: Maximum number of iterations for optimization
- **init**: Initialization of embedding matrix

The following visualization can be useful to understand how Word2Vec works and how to interpret relations between vectors captured from your texts before using them in neural networks or other machine learning algorithms:

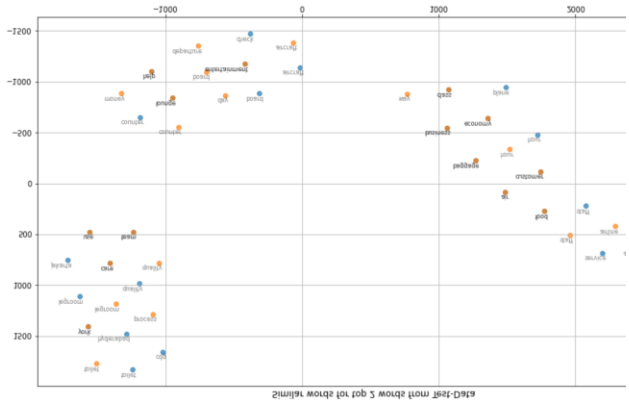


Figure: Words in vector space

Interpretation:

From the Test Dataset, using TF-IDF we found that the words "Food" and "Hour" are most common. So, to find the words in the embedding that are most associated with these two words, we plotted a TSNE-plot. As, described before, TSNE finds the nearest neighbor embedding for the words and thus, the TSNE plotted shows clusters of words that are closely embedded together. Orange highlights the words that are associated for the word-HOUR, Blue highlights the words that are associated for the word-FOOD. and the Brown highlighted words are associated with both the words Hour and Food

Appendix E.

Conditional Random Fields:

In sequential labelling or learning, previously most of the work was done using two machine learning approaches. One of which was a generative probabilistic method and the other was a sequential classification method.

The generative probabilistic method depends on k-order generative probabilistic models of paired input and label sequences using either Hidden Markov Models or Multi-level Markov Models. This approach though provides a good training and decoding algorithms of Markov Models it requires more strict conditional independence assumptions. Thus, making it impractical to use a windowed sequence of input as well as surrounding labels to make a label dependent on such a sequence.

As demonstrated in work of maximum-entropy by McCallum and Ratnaparkhi, many correlated features can be handled by a sequential classifiers like linear-classifiers, AdaBoost and support vector machines. Generative models can trade off decisions at different positions against one another, this cannot be done by Sequence Classifiers. This compelled even the best sequential learning classifiers to use heuristic combinations of forward-moving and backward-moving sequential classifiers.

Conditional Random fields brings the best out of both worlds of generative probabilistic modelling and sequential label classification.[7]

It can adjust to a variety of statistically correlated features as input just like a sequential label classifier. And just like a generative probabilistic model it can trade off decisions at different sequence to obtain a global optimal labelling.

Lafferty et al. defined conditional random field on a set of X observations with a set of Y labels, for example X might range over sentences and Y might range over part-of-speech tags. These random variables X and Y are jointly distributed, but in a discriminative framework, a conditional model is constructed p(Y | X) from paired observations and label sequences.

The principle is based on the fact that the conditional probability of a label Y_v depends on a label Y_w if and only if there is affinity with Y_v

The joint distribution over the label sequences Y given X has the form:

$P\ \Theta\ (y \mid x) \propto \exp\ (\sum_{e\ \epsilon\ E,K} \lambda_k\ f_k\ (y \mid e,x) + \sum_{v\ \epsilon\ V,K} \mu_k\ g_k(v, y|v, x) - (2)$

where x is data sequence,

y is label sequence,

y |s is the set of components of y associated with vertices in subgraph S, f_k and g_k are feature functions and Θ is the set of weight parameters.

$\Theta = (\lambda_1, \lambda_2, \lambda_3,\ldots; \mu_1, \mu_2, \mu_3,..)$

Typically to the subset of {0,1}, the feature functions f_k and g_k maps a set of observations X to a real number. The feature functions are built in such a way that the observations X_i are modelled as a vector. These are usually hand-crafted Boolean values.

Appendix F.

Classification algorithms

SVM

For defining the hyperplane, the following equation is used,

$$w^T.x + b = 0$$

where, w denotes weight vector, x is the input vector and b is bias.

This helps in creating a hyperplane with as big a margin as possible.[8]

Decision Tree

In the beginning of this algorithm, the whole training dataset is the root of the tree, where root node represents the entire population. Each box represented in the above figure is a node at which tests (T) are applied to recursively split the dataset in smaller groups. The letters (A, B, C) at each leaf node represent the labels assigned to every observation.

The test (T) is basically making the best choice to reduce the entropy to minimum and thereby improving information gain to maximum. This process is carried recursively till entropy is minimized among all branches of the tree.[9]

Entropy and information gain are calculated as follows,

$$Entropy = \sum_{i=1}^{c} -p_i.log_2 p_i$$

$$Information\ Gain = Entropy_{before-split} - Entropy_{after-split}$$

Boosting

It is an implementation of gradient boosted decision trees.[10]

For a given dataset, with n examples and m features $D = \{(x_i, y_i)\}, (|D| = n, xi \in R\ m, yi \in R)$, the output predicted by such a tree ensemble technique can be depicted as below,

$$y^T{}_i = \varphi(x_i) = \sum_{k=1}^{K} f_k(x_i),\ \ f_k \epsilon F$$

where $F = \{f(x) = w_{q(x)}\}\{q: R^m \to T, w \epsilon R^T\}$ describes the space of the trees.

Random Forest

Random Forest is essentially an ensemble classifier that uses several decision trees and then outputs the class that is predicted by the maximum number of trees. It is a robust method andproves to output high accuracy, because of it not being dependent on any particular decision tree, but abunch, or forest of them. The idea implements Breiman's "bagging" technique , which is a way to decrease the variance of the prediction by generating supplementary data o train from dataset using several combinations with repetition, therefore producing multi-sets of the original data.[11]

Voting Classifier

Voting Classifier is an ensemble technique which is based on a simple working mechanism, that is 'voting'. Several different algorithms are trained on the dataset, and the output of each is combined

to predict the final class. It works on a 'majority' principle, and the class being predicted by the greatest number of classifiers, is chosen as the ensemble result for the data. The models used were decision trees, random forest and extra trees classifier. Extra trees classifier, or extremely randomized trees uses all the data available in the training set to build each decision tree with depth set to one, also called as stump. Furthermore, the best split to form the root node or any other node is determined by searching in a subset of randomly selected features having size equal to square root of the number of features. For each selected feature, the split is chosen randomly. Therefore, the degree of randomness is more extreme than that of random forest. Thus, although decision tree, random forest and extra trees, all

implement decision trees, they have different understanding of the data. Hence, the output of each of these classifiers is taken into consideration and the class predicted the maximum number of times is voted as the final predicted class.[12]

ADASYN

It was observed that some aspects inspite of being important, were not talked about much. For example, food temperature is an important aspect of food, but the reviews containing food temperature aspect were quite less in number than that of the reviews talking about food taste. Similarly, reviews containing cabin fragrance aspect were less in number than the reviews containing cabin condition aspect. Such a difference in numbers would create an unwanted bias in the model, increasing the chances of overfitting. To overcome this problem, a technique known as ADASYN was used. Adaptive Synthetic Sampling Method for Imbalanced Data is an oversampling technique for minority classes which tackles the imbalanced classification problems. In ADASYN, the degree of imbalance, d is calculated by dividing the number of minority class samples by the number of majority class samples.

Consider Inflight-Service, which has two aspects- Operations and Facility. The initial count of data for

these two aspects was 327 and 263 respectively, which clearly indicates imbalance. The degree of

imbalance d would be 263/327 which is 0.8 . For each datapoint in minority dataset, synthetic data is

generated so as to bring the count closer to the majority class data count, while avoiding high

redundancy. The algorithm increased the minority class datapoint count to 278, thereby increasing the

degree of imbalance to 278/327 which is 0.85, bring the number of datapoints for 'facility' closer to

'operations'.

Appexdix G.

The steps taken for pre-processing the text are described in section.

Given a sequence of words, we construct vector of features for each of the labelled words. These vectors describing the features contain the following encoded features

1. Lower – case of word

2. previous word

3. next word

4. Word Length

5. Part-of-speech tag

6. Dependent Word

7. Dependent Word Part-of-speech tag

8. Dependency Tag

It is crucial to understand the fact that the stopwords removal step is both, a boon and a bane, as removal of these words leads to breakage of the sentence structure, making it difficult to analyze the text semantically. Therefore, in dependency parsing step, the text was used without removing the stopwords. Another part of preprocessing text is dealing with contraction, which means shortening of words or syllables. It was noticed that several words were present in the data in many different forms, for instance, the

term “could not” was present in terms of “couldn’t” as well. These contractions occur depending upon the tone of the reviewer or the context of the review. It is often seen that the implied meaning of the phrase does not differ, but the model considers them as different words, leading to poor training. Therefore, the need arises to alter the text in such a way that the model links up the different variations that have the same implied meaning. In this example, we change the term “couldn’t” to “could not”. Such expansion of contracted terms helps with text standardization. Apart from this, all the text is changed to lowercase, to create a uniform text dataset, which initially contained a mixture of uppercase and lowercase texts. Additionally, numerals are converted to words, for example- ‘$3000’ is changed to ‘three thousand dollars’.

Corpus can be defined as a collection of textual data, or a body of writing, that is based around a particular subject. The reviews after the above steps are added collectively to a list of reviews, henceforth referred to as “Corpus”. This corpus could be thought of a collection of all the scraped data, for all the airlines, referring to many different entities and opinions- after cleaning and preprocessing. This corpus serves as a basis of document for further steps.

Appendix H.

TTR

There are some rules for calculating TTR, which are adapted in this study. These rules include following,

a) Compound nouns and hyphen words are considered as one word

b) Parts of verbal phrases are considered as separate words, example, phrase like “*meals were served”* counts as three tokens, *meals, were and served*

c) Contractions are considered as two words, example *couldn’t,* is counted as *could not*

Results of TTR

Since, the present study is for user generated data for airlines, it is expected that there will be words that might be repeated quite often. Data is gathered for 16 airlines from two different websites and the type token ratio is observed to be between 0.2 to 0.6 for almost all airlines.

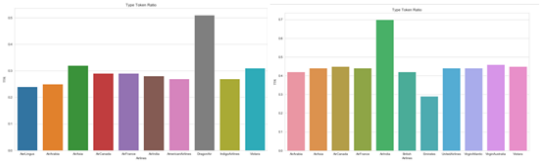


Figure: 5 most u

Type token ratio between both data sources is observed to be 0.27, which means that there are many words that are repeated between them.

Zipf’s other law states that the number of meanings (m) of a word is the square root of its frequency.

$$Given\ first\ law, m\ \propto \frac{1}{\sqrt{\mathrm{m}}},$$

$$m\ \propto \sqrt{\mathrm{f}}$$

This means that the second most repeated word will have a frequency that is half of the first word and the third most repeated word will have a frequency that is half of the second most repeated word.

As seen below, our corpus does follow Zipf’s distribution.[13]

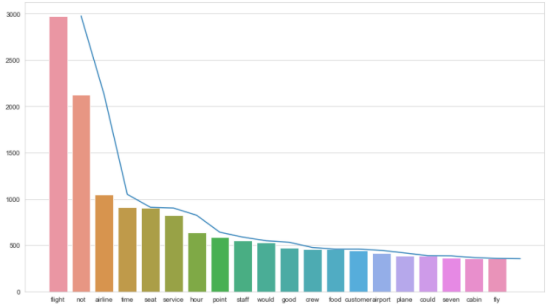


Figure: 5 most u

Appendix I.

The project can majorly be divided into these parts- Entity extraction, Aspect

identification/extraction, sentiment analysis. Several parameters are used to check the level of

righteousness of the project.

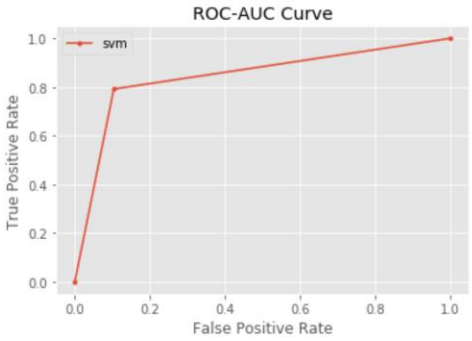
A point to be pondered about is as to which of the performance metrics should be taken into account, to better judge the model. The most common idea, “accuracy” works best when the false positives and false negatives have similar cost. However, the airline reviews contained an unequal number of positive and negative opinions for different aspects- because opinions are a subjective matter and could differ for any two people. Therefore, the performance metrics used were F1, precision and recall. These are defined below:

Precision: The measure of the correctly identified positive cases from collectively all the predicted positive cases. It is beneficial when the costs of False Positives is high.

Recall: The measure of the correctly identified positive cases from collectively all the actual positive cases. It is significant when the cost of False Negatives is high. Mutually, F1 score is the weighted average of Precision and Recall, and takes both false positives and false negatives into account. Therefore, it proved to be the best choice.

Performance metrics for “Food” entity based on different approaches, simultaneously applying SMOTE:

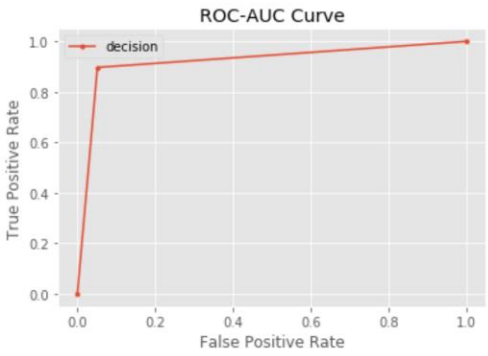
1. SVM



Classification Report is as follows

	precision	recall	f1-score	support
food_service	0.86	0.65	0.74	310
food_taste	0.75	0.88	0.81	317
food_temperature	0.78	0.89	0.83	152
accuracy			0.79	779
macro avg	0.80	0.81	0.80	779
weighted avg	0.80	0.79	0.79	779

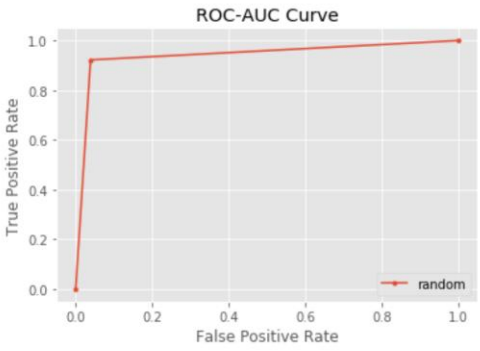
2. Decision Tree



Classification Report is as follows

	precision	recall	f1-score	support
food_service	0.89	0.87	0.88	318
food_taste	0.87	0.90	0.88	300
food_temperature	0.98	0.95	0.96	139
accuracy			0.90	757
macro avg	0.91	0.91	0.91	757
weighted avg	0.90	0.90	0.90	757

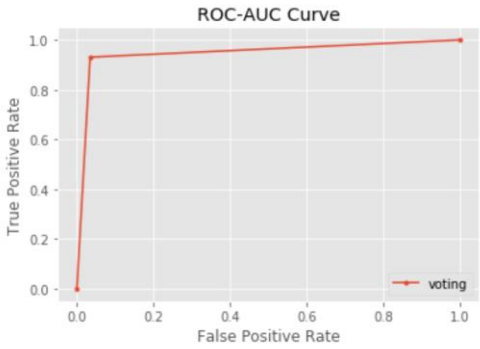
3. Random Forest



Classification Report is as follows

	precision	recall	f1-score	supp
food_service	0.91	0.92	0.91	
food_taste	0.91	0.92	0.91	
food_temperature	0.99	0.94	0.96	
accuracy			0.92	
macro avg	0.93	0.92	0.93	
weighted avg	0.92	0.92	0.92	

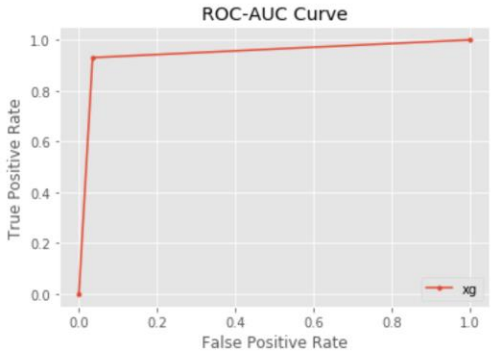
4. Voting Classifier



Classification Report is as follows

	precision	recall	f1-score	supp
food_service	0.91	0.91	0.91	
food_taste	0.91	0.93	0.92	
food_temperature	0.99	0.97	0.98	
accuracy			0.93	
macro avg	0.94	0.93	0.94	
weighted avg	0.93	0.93	0.93	

5. XG-Boost



Classification Report is as follows

	precision	recall	f1-score	support
food_service	0.94	0.90	0.92	318
food_taste	0.91	0.95	0.93	305
food_temperature	0.97	0.94	0.96	108
accuracy			0.93	731
macro avg	0.94	0.93	0.94	731
weighted avg	0.93	0.93	0.93	731

References

[1] *doccano/doccano*. doccano, 2020.

[2] A. Rosenberg and E. Binkowski, "Augmenting the kappa statistic to determine interannotator reliability for multiply labeled data points," in *Proceedings of HLT-NAACL 2004: Short Papers*, Boston, Massachusetts, USA, May 2004, pp. 77–80, Accessed: Aug. 15, 2020. [Online]. Available: https://www.aclweb.org/anthology/N04-4020.

[3] Kristina Toutanova and Christopher D. Manning. 2000. Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. In Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000), pp. 63-70.

[4] Moore R, Caines A, Graham C and Buttery P Incremental Dependency Parsing and Disfluency Detection in Spoken Learner English Proceedings of the 18th International Conference on Text,

Speech, and Dialogue - Volume 9302, (470-479)

[5] Y. Goldberg and O. Levy, "word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method," *arXiv:1402.3722 [cs, stat]*, Feb. 2014, Accessed: Apr. 22, 2020. [Online]. Available: http://arxiv.org/abs/1402.3722.

[6] J. Pennington, R. Socher, and C. Manning, "GloVe: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, Oct. 2014, pp. 1532–1543, doi: 10.3115/v1/D14-1162.

[7] J. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," p. 10.

[8] J. A. K. Suykens and J. Vandewalle, "Least Squares Support Vector Machine Classifiers," *Neural Processing Letters*, vol. 9, no. 3, pp. 293–300, Jun. 1999, doi: 10.1023/A:1018628609742.

[9] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, no. 3, pp. 660–674, May 1991, doi: 10.1109/21.97458.

[10] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco California USA, Aug. 2016, pp. 785–794, doi: 10.1145/2939672.2939785.

[11] A. Cutler, D. R. Cutler, and J. R. Stevens, "Random Forests," in *Ensemble Machine Learning: Methods and Applications*, C. Zhang and Y. Ma, Eds. Boston, MA: Springer US, 2012, pp. 157–175.

[12] S. Saha and A. Ekbal, "Combining multiple classifiers using vote based classifier ensemble technique for named entity recognition," *Data & Knowledge Engineering*, vol. 85, pp. 15–39, May 2013, doi: 10.1016/j.datak.2012.06.003

[13] D. M. W. Powers, "Applications and Explanations of Zipf's Law," 1998, Accessed: Aug. 16, 2020. [Online]. Available: https://www.aclweb.org/anthology/W98-1218.

INTER- ANNOTATOR AGREEMENT

Inter-annotator agreement is the degree of how similar two or more annotators can label the same annotation tag for a certain category.

Annotation is a manual task which acts as the backbone of this project. If not done carefully, it may lead to inaccurate training and predictions. Therefore, to make sure that annotations are correct and uniform throughout, several guidelines concerning the entities are listed and should form as a base for the task.

A. All the annotation related tasks are to be done via open-source Doccano tool.

B. Concepts are distinctly defined in 8 entities that are all annotated separately.

1. Food: Phrases that consist of observations made by the reviews about the food served in the flight. This includes all edible items, including beverages.
   Categories:
   a. Temperature: Phrase focusing on the temperature of the food being served. For example, “The burger was cold”.
   b. Service: Phrase indicating the act of food being served. For example, “Food service was slow”.
   c. Taste: Phrase focusing on the taste of the food. For example, “The snacks were delicious”.
2. Cabin: Expressions that give an insight into the cabin, the body or the interiors of the plane.
   Categories:
   a. Condition: How well was the plane/cabin maintained. For example, “The plane was clean”.
   b. Fragrance: If the plane was smelling of something, or there was a pleasant fragrance. For example, “the cabin was smelling bad”.
   c. Size: Specifies the size of the plane/ cabin/ alley. For example, “The alley was too narrow”.
   d. Temperature: Whether the air conditioning was hot/warm/cold etc. For example, “It was freezing cold”.
3. Staff: Opinions about the staff members of the airlines. This includes air hostess, pilot, ground staff.
   Categories:
   a. Behaviour: Phrase highlighting the attitude and conduct of the staff members. For example, “The air hostess was very helpful”.
   b. General: Phrase highlighting the operations and duties of the staff. For example, “The air hostess served food on time”.
4. Seat: Phrases about the in-flight seats.
   Categories:
   a. Comfort: Opinion about anything impacting the comfort of the seat. Includes backrest, cushioning and legroom. For example, “The seats were hard”.
   b. Space: Opinion about the space or size of the chair. For example, “The seats were congested”.
   c. Operation: Phrases indicating to seat operations or seat assigning processes. For example, “Seats got exchanged”.
5. Possession: Opinions relating to the possessions and luggage of the passenger.
   Categories:

a. Handling: Opinions about the handling and conduct towards the luggage. For example, “My bag was lost”.
b. General: Opinions linked to general aspects of the luggage. For example, “I was allowed to carry two bags”.

6. Entertainment: Phrases relating to the in-flight entertainment system.
   Categories:
   a. Visual: Opinions about the visual aspects of the system. For example, “The TV screen was large”.
   b. Audio: Opinions about the sound and audio quality of the system. For example, “The headphone sound was unclear”.
   c. General: Opinions about the general aspect of the system. For example, “Good collection of movies”.
7. Inflight Service: Opinions about the service towards the customer inside the plane.
   Categories:
   a. Operations: Phrases related to in-flight operations and processes. For example, “Seat belt light was not working”.
   b. Facility: Phrases concerning the in-flight facilities provided to the passenger. For example, “Extra blankets were provided”.
8. Off-flight Service: Opinions about the services outside the plane.
   a. Facility: Phrases related to the off-flight facilities availed by the customers. For example, “Pick-up car was arranged”.
   b. Ticketing: Phrases related to ticketing and booking tasks. For example, “Refund was provided”.
   c. General: Phrases indicating general and operational aspects of the off-flight services. For example, “The flight was delayed”.

C. The minimum words or phrases are to be annotated, such that it clearly helps to identify:

1. The entity/aspect being reviewed.

2. The quality of the entity/aspect.

3. The opinion about the entity/aspect.

D. Annotated data is to be saved in the form of JSON [JavaScript Object Notation] files for the project pipeline.

E. The first 200 reviews have to be common between both the annotators. This will be used to calculate the Kappa Coefficient, which indicates the similarity of the label annotation by the annotators. The Kappa Coefficient is checked at three levels- entity level, aspect level and entity-aspect pair level. Speculation is needed whenever a Kappa coefficient value falls below 0.75 and this step is to be repeated until the Kappa coefficient crosses the threshold value of 0.75

F. Regular re-checking of the annotated labels is mandatory, after annotating every 100 reviews.

G. In case of any new finding, make sure to update the Guidelines document and inform about the same to the fellow annotator.

H. In case of any conflict, discuss the same with the fellow annotator for a common worked-out resolution. Special effort should be taken to follow the above guidelines strictly.