RESEARCH METHODS

2.1. **System Overview**

System created in this research is a system for multi-aspect sentiment analysis that implements Random Forest and Word Embeddings methods. Random Forest algorithm is a popular algorithm because Random Forest can be widely applied to prediction problems and does not require a lot of parameters tuning [10]. In addition, the method was chosen because it sees the potential of the Random Forest method in research [5] which produces a fairly high accuracy value (90.48%) and the Word Embeddings method in research [6] which produces an average accuracy of 64%.

In addition to creating a system for sentiment analysis, this research also tested the Word Embeddings method with the best results based on accuracy and running time efficiency. The Word Embeddings methods tested were Word2Vec and GloVe. The Word2Vec and GloVe methods were chosen because they represent two common categories of word embeddings. Word2Vec is one of the prediction-based Word Embeddings methods, and GloVe is count-based [11][12][13].

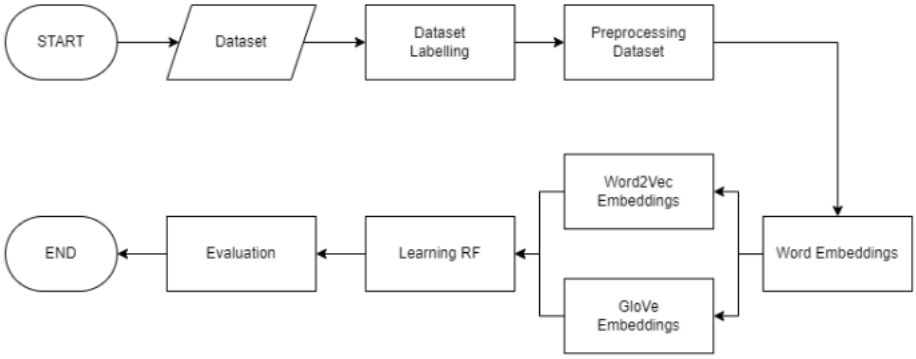

Figure 1. System overview

2.2. **Dataset**

Dataset used is the TripAdvisor Hotel Reviews obtained from the Kaggle website. The dataset contains 20491 rows of data about tourist reviews of the hotels visited. The components in the dataset consist of a review column which contains tourist reviews of the hotels visited in English, and a rating column which contains the rating given by tourists to the hotels visited.

Table 1. Example of review dataset content in English

Review	Rating
poor value stayed monaco seattle july, nice hotel priced 100- 150 night not, hotel takes beating quotient, experience simply average, nothing exceptional paying 300+ n't can't terribly disappointed, wife stayed nicest suites 200/night felt like overpaying, best advice shop, quality-wise league average marriott nice best western,	2
good choice hotel recommended sister, great location room nice, comfortable bed- quiet- staff helpful recommendations restaurants, pike market 4 block walk, stay,	5
posh twist moment walked knew unique hotel, ambiance set mood intriguing satisfying, bedroom small intimate girl want pillows finest sheets comforter wanted wrap bed mate, extremely pleased decor comfort extras room,	5

2.3. **Dataset Labelling**

In this multi-aspect sentiment analysis, five aspects are used: Room, Location, Cleanliness, Price, and Service. The aspects are based on the reviews given by hotel visitors in the TripAdvisor Hotel Review dataset. Aspect labeling is done manually on the reviewed aspects. The dataset is labeled with 1 representing positive and -1 representing negative. In this study, there are 3000 rows of data that are manually labeled according to the aspects reviewed. Table 2 is an example of dataset labeling.

Table 2 . Example of labelling on the dataset

Review	Room	Location	Cleanliness	Price	Service
good choice hotel recommended sister, great location room nice, comfortable bed- quiet- staff helpful recommendations restaurants, pike market 4 block walk, stay,	1	1			1
ritz canal street terrible location equally horrible hotel rooms dirty service terrible best trip left flew home,	-1	-1	-1		-1
pink black funky modern building great location 3 minutes walk ramblas gothic centre, stylish modern rooms watch curved bedside table, friendly staff good food,	1	1			1

The labeling process resulted in the data distribution shown in Table 3 below:

Table 3. Data distribution of dataset labelling results

Aspect	Room	Location	Cleanliness	Price	Service
Total data (pos and neg)	2043	1921	1848	1839	1944
Total positive aspects	1127	1042	919	919	1027
Total negative aspects	916	879	929	920	917

2.4. **Dataset Preprocessing**

In this process, several things are done to prepare the data before entering the feature extraction and modeling stages. Preprocessing is necessary because preprocessed data can improve model performance [14]. Preprocessing is done by labeling the dataset (dataset labelling). Then the dataset is cleaned by removing capital letters (case folding) and punctuation marks (punctual removal). After that the dataset is separated into individual words (tokenization). Furthermore, the dataset is tidied up by removing stop-words (stop-word removal) and word affixes (stemming) [1]. The use of all these preprocessing techniques together will certainly get the most optimal results [15].

2.4.1. **Case Folding**

After labeling, the dataset is cleaned by removing capital letters so that all words in the dataset are in lowercase.

2.4.2. **Punctual Removal**

The next step is to remove the punctuation/symbols present in the dataset. This process is done so that the dataset only contains words without any punctuation/symbols that can interfere with the analysis process.

Table 4. List of symbols removed in the punctual removal process

.	,	/	@	!	#	_	-
+	?	‘	“	$	&	*	‡
(	)	=	%	©	Ã	:	;

2.4.3. **Tokenization**

The next step is to tokenize the dataset. This token aims to separate sentences into words.

2.4.4. **Stop-word Removal**

After the sentence is separated into words, stop-word removal is performed. Stop-words are words that have no meaning or important information such as personal pronouns, conjunctions, possessive pronouns etc. The dictionary used for this process is English

without modifying the words that must be removed. The stop-word dictionary used is the NLTK Corpus for English. Table 5 is an example of stop-words that were removed:

Table 5. Example of removed stop-word

i	me	my	we	our	while	of	that
what	which	up	down	each	few	than	too
do	don't	should	but	if	why	when	here

2.4.5. **Stemming**

The last preprocessing stage is to remove the affixes contained in the word. It is intended that the dataset only consists of basic words. Stemmer used in this research is PorterStemmer.

Table 6. Example of stemming

Before	After
'recommended'	'recommend'
'recommendations'	'recommendation'
'stayed'	'stay'
'flew'	'fly'

2.5. **Word Embeddings**

In this process, the Word Embeddings method is applied to convert words into a digital representation for modeling. Word Embeddings are fixed-length vectors that represent a word [11]. The Word Embeddings methods used are Word2Vec and GloVe. The application of the two methods is done separately (parallel) because one of the objectives of this research is to see the best Word Embeddings method. The results of this process will be analyzed in the next process.

2.5.1. **Word2Vec**

Word2Vec is one of the simple and accessible Word Embeddings methods [16]. The Word2Vec training process helps the system to recognize the vector representation of the word through the framework of neural network [3]. There are two main architectures that make up Word2Vec: Continuous Bag-of-Words (CBOW) and Continuous Skip-gram Model (skip-gram) [12]. The CBOW architecture basically tries to predict the target word from a list of surrounding context words. The model takes a distributed representation of the context

words to try to predict the target word. CBOW treats the context as a “bag” of words, where the order in which they occur does not matter. The goal of the CBOW model is to predict a word if the surrounding words are known [7].

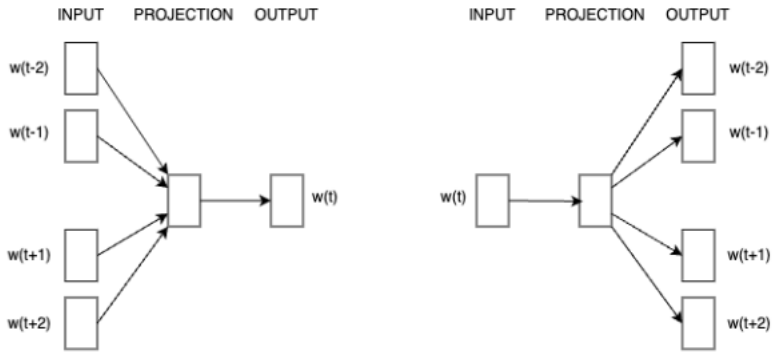

Figure 2. Architecture model of CBOW and Skip-gram

In comparison, a skip-gram takes a word as input and tries to accurately predict the words before and after the input. Skip-Gram takes one word at a time and learns to predict adjacent words within a specified window. The model basically tries to learn and predict the context words around the specified input word [12]. According to [7], skip-gram can operate more accurately for less training data. Whereas CBOW can be trained faster than skip-gram and is slightly more accurate for frequently occurring words.

Word2Vec works on a set of sentences, first building a vocabulary based on words that appear in the set of sentences more than a user-defined threshold (to remove noise), and then applying the CBOW algorithm or Skip-gram algorithm to the input document to learn a vector representation of words in D-dimensional space. Large text collections are often used to train Word2Vec. Alternatively, or in addition, thematic textual collections can be used to train Word2Vec, so as to better record the usage of words in a particular domain [7].

In this study, the vector-size parameter of 100, window (the maximum distance between the current word and the surrounding words (context) used to predict the target word) of 5 words. min_count (the minimum number of times a word appears in the dataset) of 3 times.

2.5.2. **GloVe**

Aside from Word2Vec, a frequently used Word Embeddings method is Global Vectors (GloVe). According to [13] the GloVe method uses a global matrix factorization method. The matrix represents the occurrence or absence of words in a document. The GloVe method is

a log-bilinear method, or a count-based method. GloVe learns the relationship of words by calculating how often words co-occur with each other in a given corpus. The probability ratio of occurrence of words has the potential to encode some form of meaning as well as help improve performance on word analogy problems [13]. GloVe works according to the following steps [17]:

1. Collect word co-occurrence statistics in the form of a word co-occurrence matrix X. Each element of X_{ij} represents the number of times word *i* appears in the context of word *j*.
2. Define soft constraints for each word pair:

   $$w_i^T w_j + b_i + b_j = \log(X_{ij})$$

   Where *w* – main word vector, *w* - context word vector, b_i, b_j is the scalar bias for main words and context words.
3. Define a cost function.

   $$J = \sum_{i=1}^{V} \sum_{j=1}^{V} f(X_{ij}) (w_i^T w_j + b_i + b_j - \log X_{ij})^2$$

   Where f is a weighting function that helps us prevent learning only from very common word pairs. The function is defined as follows:

   $$f(X_{ij}) = \begin{cases} (\frac{X_{ij}}{x_{max}})^{\alpha} & if\ X_{ij} < x_{max} \\ 1 & otherwise \end{cases}$$

2.6. **Learning RF**

In the RF Learning process, an analysis model is built using the Random Forest method. The Random Forest method works by creating a decision tree where as the tree grows a random factor is given so that the trees become diverse [5]. When the forest becomes diverse, the model works by dividing the features into random subsets and finding the best feature from the subset. The following algorithm explains the RF method in more detailed manner [10]:

The parameters used according to research [5] are the number of trees of 150, the criterion is defined using 'gini', the maximum depth is 150, and the maximum features are defined using 'sqrt'.

Algorithm 1 : Random Forest prediction for **x** value

Input : Training set D_n, tree count M, total sample a_n, *mtry, nodesize,* and $\boldsymbol{x}$
Output: Random Forest prediction for **x** value

Begin.

1. **for** j = 1…M **do:**
2. Select point a_n, with substitution, uniformly in the Dn
3. Set P = (X) contains a list of cells associated with the root of the tree.
4. Set $P_{final} = 0$ (empty list)
5. **while** P ≠ 0 **do:**
6. Suppose A is the first element of P
7. **if** A contain less than *nodesize* or if all Xi ∈ A is equal **then**
8. Delete cell A from list P
9. $P_{final} \leftarrow$ **Concat**(P_{final} , A)
10. **else**
11. Select uniformly, without substitution, a subset of *mtry*
12. Choose the best fraction in *a* by optimizing the criterion
13. Cut cell a based on the best fraction
14. Delete cell A from list P
15. $P_{final} \leftarrow$ **Concat**(P_{final} , A)
16. **end**
17. **end**
18. Calculate the prediction value on **x** which is equal to the average Y_i that goes to cell **x** in partition P_{final}.

End.

2.7. **Evaluation**

In this system evaluation, the data that has been analyzed is calculated for performance. System performance is measured using a predetermined accuracy metric. The accuracy metric is used to measure the performance of the model that has been built. For this test, 10-fold cross validation will be carried out by dividing the train data by 90% and the test data by 10% randomly to ensure the performance testing of the system. Accuracy is measured by comparing the number of correct predictions with the number of all predictions.

$$accuracy = \frac{t_p + t_n}{t_p + t_n + f_p + f_n}$$

with t_p are true positive, t_n are true negative, f_p and f_n are false positive and false negative.

Additionally, running time efficiency is also a factor that is taken into account to find out which algorithm is more efficient. After measuring the accuracy and running time, the two combinations of methods are compared to find the best Word Embeddings method to produce the most optimal results.