Measuring Transcript Relevance and Certainty through Sentence Classification and Semantic Similarity Analysis

Cyrez B. Ronquillo
Institute of Computer Science
University of the Philippines Los Baños
Los Baños, Laguna
cbronquillo@up.edu.ph

Reginald Neil C. Recario
Institute of Computer Science
University of the Philippines Los Baños
Los Baños, Laguna
rcrecario@up.edu.ph

ABSTRACT

This paper presents a study that attempts to measure how factual a given statement is and how much it claims is relevant. The study is divided into two major parts: Sentence Classification and Semantic Similarity Analysis. For sentence classification, supervised classifiers were trained: Support Vector Machine (SVM) with Radial Basis Function (RBF) Kernel, Logistic Regression (LR), and Artificial Neural Network (ANN). Tuning was done to the models to create different setups and find out which setup produces the best results. The dataset used for the models were 33 US Debate Transcripts. After dataset preprocessing, 22,611 sentences remained, and 63 features were extracted from the sentences. The Logistic Regression model was determined to be the most reliable model. The Support Vector Machine scored the highest Training Accuracy. However, the Logistic Regression model is the most balanced model, based on the five different evaluation metrics. For Semantic Similarity Analysis, news articles were extracted from 16 different satiric and reliable websites to assess an input statement's certainty. Triplet Extraction Approach was used to compare the input sentence to different articles to be able to provide a corresponding similarity score and to classify it as reliable, satiric, or unverified. Using a set data for testing, the Semantic Similarity Analysis scored 80% accuracy.

CCS Concepts

• Theory of Computation→ Theory and algorithms for application domains→ Machine Learning Theory
• Computing methodologies→ Artificial Intelligence→ Natural Language Processing

Keywords

Fact checking, machine learning, supervised learning, natural language processing, semantic similarity analysis, sentence classification, support vector machine, logistic regression, artificial neural network.

1. INTRODUCTION

Transcripts and other forms of documentation provide an avenue to record information. In a textual form, transcripts give a detail of what was spoken by people in a setting and may even describe events happening in that setting. Transcripts of every institution, especially of the government, are regarded to be true, correct and concise as they provide a detail of what was said by a figure, say for example, a president or a government spokesperson. With that in mind, transcripts can be used in fact-checking. They can also be used to determine whether a new claim is related and is based on facts given such transcripts. They can also be used to detect a fake news.

Fake news, as defined by Lazer et al [1], is "fabricated information that mimics news media content in form but not in organizational process or intent". According to Elemia [2], the spreading of fake news has been a controversy for the past few months in the Philippines. Advancements in computing technologies can be used to determine verifiable sources of information.

"Fake news detection" is a task for categorizing unverified news along with a series of facts or verified news, associated with a measure of certainty [3].

According to Hassan et al. [4], it is nearly impossible to reach the Holy Grail – a fully automated fact-checker. However, it is still possible to implement a program whose accuracy is as close to a fully automated one. They implemented ClaimBuster, the first ever sentence ranking application, wherein its sentence classification is through analysis of sentence structure. Modeled as a classifier and ranker, the application makes use of a 3-class classification (Check-Worthy Factual (CFS), Unimportant Factual (UFS), or Non-Factual (NFS)) using Multinomial Naive Bayes Classifier, Support Vector Machine, and Random Forest Classifier evaluated by a 4-fold cross validation. The authors pointed out that the classification models used performed better accuracy on NFS and CFS than UFS.

Hassan et al [5] provided fine tuning on a recent work on Claimbuster which included details of the ground truth collection, use of participants to label sentences, the current components of the Claimbuster system which collects claims from social media like Twitter, debate transcripts, and Hansard Australian parliament transcript proceedings, together with fact-check sources like CNN, a new agency, and Politifact, a US fact-checking website on claims made by elected officials. The authors used the same 3-class classification, the same set of classification models and k-fold cross validation from their previous study.

Another related work by Baly et al [6] focused on predicting the news factuality of a news medium and studied bias alongside data used from various sources such as identified target websites, Wikipedia pages, news media Twitter accounts, news web URL structure, and information related to traffic generated. They utilized a set of features from data sources defined from previous works they have reviewed and used Support Vector Machine as the classification model on a 5-fold cross validation setup. Using accuracy, macro-averaged F1 score, and Mean Average Error as

Philippine Computing Journal Dedicated Issue on Natural Language Processing, pages 34–43
Vol. XIV No. 1 August 2019

metrics, they concluded that Wikipedia features are less useful for factuality but performs reasonably well for bias [6].

Constantino Jr., et al [7] on their work on news verification used Sentence Similarity Analysis and Hidden Markov Model. They used several reliable and satiric websites and made use of the Triplet Extraction Approach as one of their methods for Sentence Similarity Analysis. They used 30 sentences to evaluate their system and conducted a survey with 30 respondents to assess the results of their system. Triplet Extraction Approach provided the highest accuracy results as compared to other methods they tried in their study.

In this paper, we present an approach to determine how factual a statement is and provide a measure of how factual a statement claimed. We created a system that combines relevance checking and certainty checking which were not present in the previous works we reviewed at the time this work was conceptualized.

We used the terms transcript relevance and transcript certainty. We define transcript relevance as a rating of how interesting the transcript's content is and how much it claims is relevant. Transcript certainty, on the other hand, is defined as a rating of how factual a transcript is; a degree of certainty of a given transcript.

The study aimed to create an application that measures how factual a given statement is and measure how much it claims is factual. Specifically we aimed to: (1) collect local news from reliable and satirical websites using a web scraper; (2) collect all 33 US Presidential Debate transcripts for training and test data for relevance verification; (3) determine the transcript relevance and classify each of the transcript's sentences as Check-Worthy Factual (CFS), Unimportant Factual (UFS), or Non-Factual (NFS) using Support Vector Machine (SVM), Logistic Regression (LR), and Artificial Neural Network (ANN); (4) evaluate the performance of the models used for sentence classification by computing the accuracy, precision, recall, and F1 score; (5) verify the certainty of a sentence's content using the collected news through Semantic Similarity Analysis using Triple Extraction Approach; and (6) compute a score for each sentence in the transcript based on the weights and model used.

In the study, the transcript relevance is determined using Sentence classification while the transcript certainty is determined using Semantic Similarity Analysis. The kind of ANN used in this study is feedforward ANN.

With the growing research interests focused on fake news detection, this work and the previous works can be used as a tool to determine certainty and relevance of information provided by political figures and of media outlets as well. In the Philippines, media outlets like CNN Philippines, Rappler, and Vera Files may integrate such fact-checking tool in their system.

2. OUR APPROACH

This section summarizes how we were able to implement a design to meet our objectives. The two separate tasks were (1) Sentence Classification and (2) Semantic Similarity Analysis. Figure 1 shows the overall view of how the major tasks come together to realize the objectives of this study.

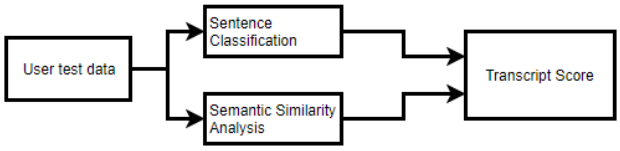

Figure 1. A visual flow of the steps done in the study.

The relationship between the two tasks and how they produce results are explained in the following subsections.

For simplicity, we divided this section into four subsections: (1) technologies and libraries used, (2) Sentence Classification, (3) Semantic Similarity Analysis, and (4) Generation of Transcript Score.

2.1 Technologies and libraries used

The main programming language used for the study is Python 3.6.4. Computations performed were executed in a 64-bit Ubuntu Linux 16.04 LTS OS. Python libraries that aided in numerical computation were NumPy 1.14.1 and SciPy 1.0.0. Scikit-learn 0.19.1 was used in building and training the Support Vector Machine. Tensorflow CPU 1.6.0 library was used in building and training the Artificial Neural Network and Logistic Regression. The Watson-Developer-Cloud 1.0.0 Python library helped in extracting some features of the sentences that were used for training and testing. Python requests 2.18.4 library was used for retrieving the ground truth values for the sentence classification via the ClaimBuster API. The Natural Language Toolkit (NLTK) 3.2.5 library in Python was also used for Natural Language Processing in Sentence Classification and Semantic Similarity Analysis. Newspaper (newspaper3k 0.2.5) library by Python was used for web crawling and scraping. SQLite3 2.5 was used for news storage. The Python library that aided in triplet extraction is the BLLIPParser 2016.9.11 by generating the semantic tree structure of a given sentence.

2.2 Sentence Classification

One of the two major tasks done is Sentence Classification. In Sentence Classification, sentences are classified and scored according to classes. Details were further explained within this subsection.

A visual summary of the major steps for Sentence Classification is shown on Figure 2. The subsections reflect the major steps done to carry out this study.

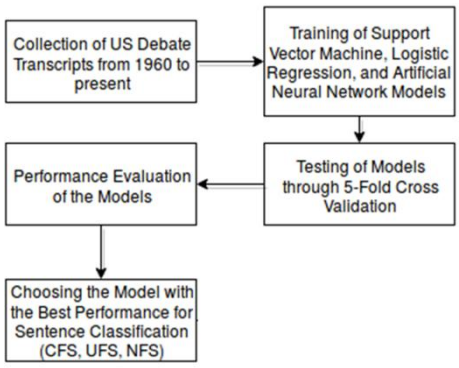

Figure 2. A visual flow of the steps done in the study.

2.2.1 Data collection, preprocessing and storage

Transcripts collected for training the models were Presidential US Debate transcripts from 1960 to 2012 obtained from the dataset used in a previous study [4]. Thirty presidential US debate transcripts were the contents of the dataset obtained, with 23,075 sentences extracted. However, only 20,788 sentences were kept because sentences with less than 5 words were removed. In addition, the three US Presidential Debate transcripts of 2016 were added as part of the dataset. A total of 22,611 sentences after the additional transcripts were added. This dataset was used for the Sentence Classification. The debate transcripts data were used as a means of comparison since it was used in the original study with 63 extracted features which was also used in the study [4]. This will be further explained in section 2.5.

2.2.2 Test and Training Data Split for Sentence Classification

Sentences obtained from the 33 US presidential debates were split for training and test data with around 16,000 for training and 4,000 for testing. Each model underwent a five-fold cross-validation. There were five iterations of training for every model per parameter set-up. Parameters that varied for each training is the training epoch size, the learning rate for the Logistic Regression and Artificial Neural Network, and the kernel coefficient (gamma) for the Support Vector Machine.

The training and testing accuracy, and other metrics of the models are computed as the average of the metrics retrieved for each iteration of the five-fold cross-validation. A previous study [8] stated that shuffling the dataset will assure that the values obtained from the models will be consistent, will make the experiment reproducible, and will avoid overfitting.

2.2.3 Sentence Classification and Scoring

Sentences in the dataset were classified according to their structure and other features to be extracted based on [4]. They were classified as Non-Factual Sentences (NFS), Check-Worthy Factual Sentences (CFS), and Unimportant Factual Sentences (UFS).

2.2.3.1 Non-Factual Sentences (NFS)

These are sentences that do not have any factual claim in their content. This classification of sentence also shows subjectivity and should produce the lowest score in the models.

2.2.3.2 Check-worthy Factual Sentences (CFS)

These sentences are known to have factual claims. Its content is of interest by the public in determining whether it is factual or faulty. "Check-worthy" was defined in a paper [3] as a sentence whose content is highly disputable and interesting to research more on. This classification of sentence should produce the highest score in the models.

2.2.3.3 Unimportant Factual Sentences (UFS)

These are sentences that have factual claims but are not check-worthy. Unimportant Factual Sentences are not check-worthy because its content is not of interest by the public to whether its content is true or not. This classification should produce a score greater than NFS but less than CFS.

This study classified and scored sentences based on the above-mentioned classifications using the following models: Support Vector Machine (SVM), Logistic Regression (LR), and Artificial Neural Network (ANN). Because of time constraints, considering the data collection, data processing, training and testing, we limited the use of models to only three based on their popularity in use from other related research works.

The models classified CFS as positives while NFS and UFS were classified as negatives. The scores provided were used for evaluating the performance of the models. Different thresholds, specifically 0.3, 0.4, and 0.5, were used to avoid dataset bias.

2.2.4 Feature Extraction

In this study, features extracted in each sentence were similar to what were extracted in Hassan and other researchers' study [4] except for the TF-IDF. A total of 63 features were extracted for each sentence. The following sample sentence is used to explain the different features: *I have fought against – well, one of them would be the marketing assistance program.* Given the sample sentence, we can get the following features:

1. **Sentiment:** The Watson Developer Cloud API was used to calculate the sentiment score of a sentence. The sample sentence above has a sentiment score of 0.675968.
2. **Word Count:** RegexpTokenizer of NLTK was used to slice the sentence into words. The sample sentence above has 14 words.
3. **Part-of-Speech (POS) Tags:** NLTK's *pos_tag* function accepts a word or an array of words as parameter and returns the corresponding POS tag of each word. There are a total of 35 POS tags in the NLTK POS classifier since the classifier used the Penn Treebank Tagger. In the sample sentence above, there are 3 words with POS tag NN (noun, singular or mass) and 2 words with POS tag IN (preposition or subordinating conjunction).
4. **Entity Type:** The Watson Developer Cloud API was used to extract different entities from sentences. There are a total of 26 types of unique entities. The sample sentence above has labeled "marketing" as an entity "JobTitle".

Do note that what is collected are not the words or sentences themselves but the features of these sentences from the transcripts.

2.2.5 Model Parameters

Different parameters for each model were chosen to create various model combinations. The threshold for the ground truth was also manipulated to avoid data bias. Threshold values 0.3 and 0.4 were used for testing because the 0.5 threshold was found to produce only 2,513 UFS out of the 22,611 sentences in the dataset. The training epoch sizes were manipulated to determine the maximum number of passes over the training dataset are needed to obtain higher scores [9]. Training epoch sizes 1,000, 2,000, and 5,000 were used. For the Artificial Neural Network and Logistic Regression model, the learning rate was manipulated to see how it would take effect in minimizing the cost function of gradient descent [9]. Learning rate values used where 0.01 and 0.001. For the Support Vector Machine, the gamma parameter was manipulated and given values 0.01, 0.001, and 'auto' where auto is the reciprocal of the feature size ($\frac{1}{63}$). The gamma affects the kernel output of the SVM model which is the Radial Basis Function (RBF) kernel.

2.2.6 Performance Evaluation

The following performance measures were used to compare the results for each model: true relative error, precision, recall, and F1 score [10].

2.2.6.1 True Relative Error

True Relative Error (TRE) was used to measure how close the computed values (Actual Value or AV) in the model are to the ground truth values (True Value or TV).

$$TRE = \frac{TV-AV}{TV} \times 100 \quad (1)$$

where TV refers to the true value and AV refers to the actual value.

2.2.6.2 Precision

Precision is also known as the positive predicted value. This was used to compute for the relationship between the total number of sentences that were classified in the test data as CFS (True Positive and False Positive) and the number of sentences that were actually CFS (True Positive).

$$Precision = \frac{TP}{TP+FP} \times 100 \quad (2)$$

where TP refers to the true positive and FP refers to the false positive.

2.2.6.3 Recall

Recall is also known as the *sensitivity*. This was used to compute for the relationship between the correctly classified CFS (True Positive) and the sentences that were supposedly CFS but were classified as UFS and NFS (False Negative).

$$Recall = \frac{TP}{TP+FN} \times 100 \quad (3)$$

where TP refers to the true positive and FN refers to the false negative.

2.2.6.4 F1 Score

F1 Score is known to be the harmonic mean of recall and precision. It was used to test the accuracy of the results generated by the models.

$$F1Score = 2 \times \frac{Precision \times Recall}{Precision+Recall} \times 100 \quad (4)$$

2.3 Semantic Similarity Analysis

As mentioned in the technologies and libraries subsection, the Natural Language Toolkit (NLTK) and BLLIP Parser library in Python were used for Semantic Similarity Analysis (SSA). Since this paper [7] regarding Fact Checking found that the *Triplet Extraction Approach*, also known as the Subject-Verb-Object (SVO) Approach, is the most effective in computing for the semantic similarity, this was used for this study as well. Other approaches will also be considered in the future since this approach is only accurate for simple sentence inputs.

The flow summary of the whole semantic similarity analysis can be seen in Figure 3.

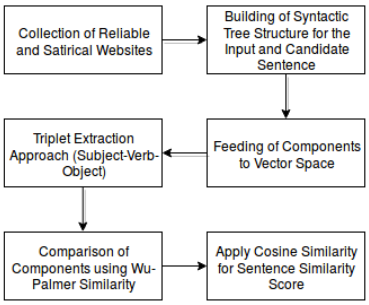

Figure 3. A visual flow for the Semantic Similarity Analysis component.

This subsection is divided into five parts: (1) Collection of reliable and satirical websites, (2) Building of Syntactic Tree Structure, (3) Triplet Extraction Approach and Feeding of Components to Vector Space, (4) Comparison of components using Wu-Palmer Similarity and (5) Applying Cosine Similarity for Sentence Similarity Score. These subsections correspond to the visual flow shown on Figure 3.

2.3.1 Collection of reliable and satirical websites

News for certainty verification were collected via web scraping. Collected news were categorized into two: news obtained from reliable websites and news obtained from satirical websites. The news were collected over a span of five months from January 2018 to May 2018. Data collection was updated until July 2019 totaling to 3,028 reliable news articles and 1,076 satirical news articles as shown in Figure 4.

```
$ python3 print_db_size.py
STATS
3028 news article are in reliable_news db
1076 news article are in satirical_news db
```

Figure 4. A screenshot of the Python program run on the collected news articles both reliable and satirical.

The study is limited to accept input transcripts whose content is verifiable via online news in the Philippines. The application can only process transcripts that are written in English since the model, based on 63 features, (see sections 2.2.1 and 2.2.4) are written in English.

Verification of each sentence's certainty is based on information gathered in Philippine news articles only. These data (collected news) were used for the Semantic Similarity Analysis. A crawler was deployed to different satirical and reliable websites. It was configured to visit the websites recursively to be able to reach all the news articles that can be found inside. Important information such as URL, title, author, and news body were scraped in every web page and were stored in the database. Tables 1 and 2 show the list of websites where the spiders were deployed.

Table 1. List of reliable websites used for the study

http://www.cnnphilippines.com/
http://newsinfo.inquirer.net/
http://www.philstar.com/
https://mb.com.ph/
http://news.abs-cbn.com/

Table 2. List of satirical websites used for the study

https://adobochronicles.com/
http://filipinofreethinkers.org/
http://dutertenews.com/
https://agilanews.wordpress.com/
https://pinoytrending.altervista.org/
https://dutertetrendingnews.blogspot.com/
https://dutertedefender.com/
https://www.maharlikanews.com/
https://mindanation.com/
https://globalnews.favradio.fm/
https://pinoynewsblogger.blogspot.com/

Table 1 contains the list of some popular media outfits in the Philippines. Table 2 contains a list of satirical or fake news websites as listed in a Wikipedia page.

2.3.2 Building of Syntactic Tree Structure

The Charniak-Johnson (BLLIP) reranking parser was used to generate the syntactic tree structure of the sentence. The Noun Phrase (NP) encountered in the first subtree was considered as the Subject of the Sentence, the deepest verb found in the Verb Phrase (VP) subtree was used as the Verb, while the nouns Noun Phrases (NP) or Adjective Phrases (ADJP) found in the last subtree were the Objects of the sentence. A swap between the Subject and Object occurs when the sentence is found to be in Passive Voice form.

2.3.3 Triplet Extraction Approach and Feeding of Components to Vector Space

In the Triplet Extraction Approach, we track the subject, verb, and object (SVO) of a sentence [10]. The SVO of the sentences derived from the sources are compared to the segments from other sentences. After getting the object components, they were converted to vector space in preparation for *Cosine Similarity* computation.

2.3.4 Comparison of components using Wu-Palmer Similarity

Sentence components were compared against other sentence components using Wu-Palmer Similarity. Wu and Palmer [11] defined a similarity measure that can be calculated using words' positions or depths in their respective structures relative to the depth of the most Least Common Subsumer (LCS).

Given two concepts X and Y, their similarity is expressed as:

$$Sim_{WP}(\mathrm{X},\mathrm{Y}) = \frac{2 \times \mathrm{N}}{N1+N2} \quad (5)$$

Where $\mathrm{Sim_{WP}}$ is the Wu-Palmer Similarity, N1 and N2 are the number of arcs between the two concepts X, Y and the ontology root R, and N is the number of arcs between the LCS and the ontology root R[12].

2.3.5 Applying Cosine Similarity for Sentence Similarity Score

Cosine similarity is "a measure of similarity between two vectors by measuring the cosine of the angle between them" [13].

$$cos(\theta) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (6)$$

Equation 6 is the formula for the Cosine Similarity computation. A_i represents the vectorized form of the input sentence while B_i represents the vectorized form of the candidate sentence in the database. The similarity score of the two sentences were computed by multiplying the similarity score of each component to its corresponding coefficient and getting the sum of the products. The coefficients are adjusted depending on the presence and absence of different components after extraction. Part of this study determined whether a certain sentence lies heavier on the satiric kind of news rather than the reliable kind or vice versa. The study computed for the similarity score of each sentence in the database relative to the input sentence.

2.4 Generation of Transcript Score

Once we reach this point, after all the steps provided, it is expected that every sentence should be assigned with two different scores: the sentence relevance (R) weight and the sentence certainty (C) weight. The number of sentences will be denoted as *n*. The formula that was used for computing the rate of credibility of the transcript as a whole is shown in Equation 7. The value of $cred_t$ may range from -1 to 1. A credibility rate between -1 and 0 means that most of the sentences in the given transcript are satiric, fake, or faulty. A credibility rate between 0 and 1 means that most of the sentences in the given transcript are reliable, truthful, or factual.

$$cred_t = \frac{\sum_{S=1}^{n} (R_s \times C_s)}{n} \quad (7)$$

3. RESULTS AND DISCUSSIONS

An interface for the application was created in order to easily test input data. Figure 5 shows the interface with an input transcript data and transcript score while Figure 6 shows a zoomed in version of the upper part of the application focused on the input data and the toggle buttons for certainty, relevance, and transcript score.

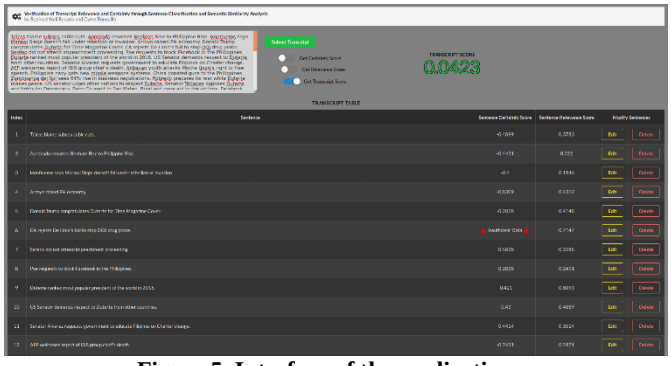

Figure 5. Interface of the application.

The certainty score ranges from -1.00 to 1.00 with negative values as “satirical” and positive values indicating “reliable”. A certainty score of zero means there is insufficient data to determine the certainty of the given input. The relevance scores, on the other hand ranges from 0 to 1.00.

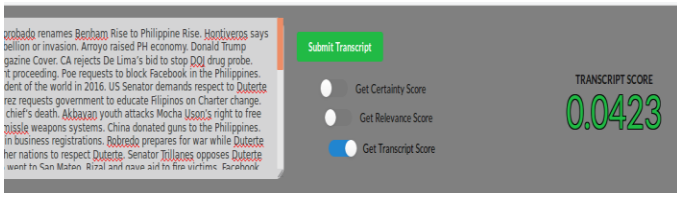

Figure 6. A screenshot portion of the application focused on the input field part; the toggle buttons for certainty, relevance, and transcript scores; and the score of the transcript.

The transcript score provided has three possible colors: red, black, and green indicating a negative, neutral and positive score. Figure 7 shows these scores respectively.

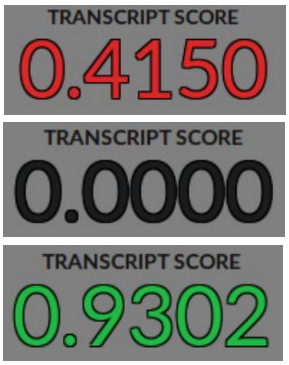

Figure 7. Sample scores indicating a negative (top), neutral (middle), and positive (bottom) scores. The scores are differentiated by their colors.

For the Semantic Similarity Analysis module, the testing performed was similar to a previous study [7]. Fifty (50) sentences were fed to the model for testing. The sentences were generated based on the articles collected from the websites to determine whether they would capture the correct classification. The sentences did not necessarily come from the title or body of any article. The module was tasked to classify whether a given sentence is Reliable or Satirical. Insufficient Evidence is also a possible output if the module cannot find a news article in the database that has a similar SVO of the input sentence. The model scored an accuracy of 80.00%. Table 3 shows 10 from the 50 test sentences, used for the Semantic Similarity Analysis, with known result as either "Reliable" or "Satirical", and Table 4 shows the summary of the testing performed.

Table 3. List of 10 sample test data sentences for the Semantic Similarity Analysis

Sentence Number	Test Sentence used for the Semantic Similarity Analysis
4	Arroyo raised PH economy.
12	AFP welcomes report of ISIS group chief's death.
14	Philippine navy gets new missile weapons systems.
18	US senator urges other nations to respect Duterte.
22	A holdover from PNoy admin tries to sabotage Meralco's 7-Billion peso refund.
26	Marcos calls for help among agencies to help ensure road safety.
27	Cayetano: OFW rescue video not a political move.
30	Philippines now in China's mercy over maritime dispute.
35	US warns China over missiles deployed on 3 PH reefs.
43	Grab starts operation in Naga.

Table 4: Comparison of Computed Scores and Expected Output of the Semantic Similarity Analysis module on 10 sample data sentences.

Sentence Number	Computed Score and Classification	Expected Result
4	83.09 Satirical	Satirical
12	74.31 Satirical	Satirical
14	81.00 Satirical	Satirical
18	72.17 Satirical	Satirical
22	77.66 Satirical	Satirical
26	100.00 Reliable	Reliable
27	100.00 Reliable	Reliable
30	100.00 Satirical	Reliable
35	100.00 Reliable	Reliable
43	100.00 Reliable	Reliable

Using the same 10 test sentences, we tested the system for obtaining relevance scores. We attained the following as shown in Table 5.

Table 5: Computed Relevance Scores for the 10 sample data sentences.

Sentence Number	Computed Relevance Scores
4	43.12
12	53.64
14	57.77
18	25.70
22	77.59
26	23.67
27	30.64
30	50.33
35	91.08
43	27.24

Table 6 summarizes the results for Semantic Similarity Analysis using 50 test sentences. An extra column called "Insufficient data" is included to compensate for two sentences that were not classified as either "Reliable" or "Satirical".

Table 6: Confusion Matrix for testing sentences as either "Reliable" or "Satirical"

		Predicted		
		Reliable	**Satirical**	**Insufficient Data**
Actual	**Reliable**	19	5	1
	Satirical	3	21	1

For the Sentence Classification module, there were 18 possible setup combinations for the Logistic Regression (LR) and Artificial Neural Network (ANN) Models while there were 27 possible setup combinations for the Support Vector Machine (SVM). Three ground truth thresholds, **thr**, were considered: 0.3, 0.4, and 0.5. Three training epoch sizes, **TE**, were considered: 1,000, 2,000, and 5,000. Two learning rate values, **LR**, for LR and ANN were considered: 0.01 and 0.001. And three kernel coefficient values, **gamma**, for SVM were considered: 0.01, 0.001, *auto* (1/63).

The raw dataset consists of 32,487 sentences. However, preprocessing was done to provide more accurate results [4]. The preprocessing done in the previous study was also followed where: (1) the moderator speakers were removed; (2) crowd reactions and other extra words enclosed in [], (), and {} were removed; (3) interrogative sentences were removed; and (4) only sentences with greater than five words are maintained. After the preprocessing, a total of 22,611 sentences remained.

Five-fold cross-validation was done to ensure that the values obtained from the models are consistent, makes the experiment reproducible, and avoids overfitting [8]. Cross validation "allows models to be tested using the full training set by means of repeated resampling; thus, maximizing the total number of points used for testing" [14]. For the given dataset, 18,088 sentences were used for training the model while the remaining 4,523 sentences were used for testing the model accuracy and other evaluation metrics. The ground truth values were extracted from the ClaimBuster API by feeding the dataset to it. The API generates a probabilistic value of how much the sentence claims. Classifying the scores generated depends on the threshold (thr) parameter given to the model. With the corresponding value of thr, a value of 0 is given for NFS and UFS ($\leq$thr) while a value of 1 is given for CFS ($\geq$thr). The reason for manipulating the threshold for ground truth is because of data bias. A threshold value of 0.5 would produce only 2,513 UFS out of the 22,611 sentences in the dataset. For a threshold value of 0.4, it would produce 4,765 UFS. While for a threshold value of 0.3 would produce 8,835 UFS.

3.1 Accuracy

The formula for the True Relative Error (TRE) was used to compute for the accuracy in training and testing the different setups.

3.1.1 Training Accuracy

In all the three threshold values plugged for training, the **Support Vector Machine** setup where **training epoch size is 5,000** and **gamma is 'auto'** scored the highest training accuracy of 83.19% for thr=0.3, 90.68% for thr=0.4, and 95.06% for thr=0.5. However, it is not always the case that having a high training accuracy would produce the best performing model. The higher the training accuracy of a model, the higher the probability of a model to overfit the training data that could lead to faulty predictions [9]. Table 7 shows the top three training accuracies achieved at different thresholds, epoch sizes, and learning models.

Table 7: Top Three Training Accuracies for Various Thresholds, Training Epoch Sizes, and Learning Rates/Gamma

Type (thr=0.3)	**SVM**	**ANN**	**LR**
TE=1000 LR/gamma = 0.01	37.89	**76.73**	80.66
LR/gamma = 0.001	36.01	**76.73**	75.84
TE=5000 LR/gamma = 0.01	82.46	75.21	**81.57**
LR/gamma = auto	**83.19**	–	–
Type (thr=0.4)	**SVM**	**ANN**	**LR**
TE=2000 LR/gamma = 0.01	87.17	**87.24**	88.56
TE=5000 LR/gamma = 0.01	90.06	76.24	**89.07**
LR/gamma = auto	**90.68**	–	–
Type (thr=0.5)	**SVM**	**ANN**	**LR**
LR/gamma = 0.001	35.17	**92.88**	89.18
TE=5000 LR/gamma = 0.01	94.55	91.62	**93.56**
LR/gamma = auto	**95.06**	–	–

3.1.2 Testing Accuracy

For thr=0.3 and 0.4, the **Logistic Regression** with **training epoch size of 5,000** and **learning rate of 0.01** scored the highest testing accuracy of 81.43%. This means that even though a model can score a very high training accuracy, it does not always guarantee a high testing accuracy. For thr=0.5, the **Support Vector Machine** with **training epoch size of 5,000** and **gamma is 0.001**.

Table 8 shows the top three testing accuracies achieved at different thresholds, epoch sizes, and learning models.

Table 8: Top Three Testing Accuracies for Various Thresholds, Training Epoch Sizes, and Learning Rates/Gamma

Type (thr=0.3)	**SVM**	**ANN**	**LR**
TE=1000 LR/gamma = 0.01	38.65	**76.04**	80.41
TE=5000 **LR/gamma = 0.01**	**79.04**	74.76	**81.43**
Type (thr=0.4)	**SVM**	**ANN**	**LR**
TE=2000 LR/gamma = 0.01	60.04	**87.07**	88.50
TE=5000 **LR/gamma = 0.01**	88.55	76.07	**88.98**
LR/gamma = 0.001	**88.78**	80.40	86.67
Type (thr=0.5)	**SVM**	**ANN**	**LR**
LR/gamma = 0.001	11.46	**92.92**	89.13
LR/gamma = 0.001	**93.51**	91.96	90.21
TE=5000 LR/gamma = 0.01	93.37	91.80	**93.45**

3.2 Precision

The positive predicted value or precision will help determine the score of how good a model can classify Check-Worthy Factual Sentences. For thr = 0.3, the **Artificial Neural Network** with **training epoch size of 5,000** and **learning rate of 0.001** scored the highest precision of 78.60%. For thr = 0.4, an **Artificial Neural Network** again scored the highest precision score of 84.78% but having parameters **training epoch size of 2,000** and **learning rate of 0.01**. For thr = 0.5, a **Logistic Regression** with **training epoch size of 1,000** and **learning rate of 0.001** scored the highest precision with a value of 91.84%.

Table 9 shows the top three precisions achieved at different thresholds, epoch sizes, and learning models.

Table 9: Top Three Precisions for Various Thresholds, Training Epoch Sizes, and Learning Rates/Gamma

Type (thr=0.3)	**SVM**	**ANN**	**LR**
TE=5000 LR/gamma = 0.01	**70.17**	68.17	**68.81**
LR/gamma = 0.001	69.96	**78.60**	55.84
Type (thr=0.4)	**SVM**	**ANN**	**LR**
TE=2000 **LR/gamma = 0.01**	35.57	**84.78**	73.22
TE=5000 LR/gamma = 0.01	75.70	68.32	**75.19**
LR/gamma = 0.001	**75.74**	61.47	63.71
Type (thr=0.5)	**SVM**	**ANN**	**LR**
LR/gamma = 0.001	10.99	72.66	**91.84**
LR/gamma = 0.001	75.86	**80.42**	78.69
LR/gamma = auto	**76.02**	–	–

3.3 Recall

The *sensitivity* or recall will compute for the relationship between the correctly classified CFS, and the classified UFS or NFS but are actually CFS. For thr = 0.3 and 0.4, the **Support Vector Machine** with **training epoch size of 2,000** and **gamma of 0.001** scored the highest recall of 99.93% and 100% ,respectively. For thr = 0.5, the **Support Vector Machine** also has the highest score of 97.90% but with a different **training epoch size of 1,000**. This means that the SVM model can easily classify CFS with the given parameters.

Table 10 shows the top three recalls achieved at different thresholds, epoch sizes, and learning models.

Table 10: Top Three Recalls for Various Thresholds, Training Epoch Sizes, and Learning Rates/Gamma

Type (thr=0.3)	**SVM**	**ANN**	**LR**
TE=1000 LR/gamma = 0.01	90.31	**79.45**	88.73
LR/gamma = 0.001	97.59	71.66	**97.26**
LR/gamma = 0.001	**99.93**	64.08	94.45
Type (thr=0.4)	**SVM**	**ANN**	**LR**
LR/gamma = 0.001	91.52	**80.87**	54.41
LR/gamma = 0.001	**100.00**	61.86	60.60
TE=5000 LR/gamma = 0.01	66.89	65.72	**69.18**
Type (thr=0.5)	**SVM**	**ANN**	**LR**
TE=1000 LR/gamma = 0.01	72.93	**71.28**	47.57
LR/gamma = 0.001	**97.90**	57.46	2.65
TE=5000 LR/gamma = 0.01	59.08	68.58	**56.12**

3.4 F1 Score

The *harmonic mean* of the Precision and Recall or the F1 Score takes into account both the false positives and the false negatives into one score. It is somehow similar to what accuracy computes but is more useful especially on uneven class distributions. For thr = 0.3 and 0.4, the **Logistic Regression** with **training epoch size of 5,000** and **learning rate of 0.01** has the highest value of F1 score with 75.56% and 72.04%, respectively. For thr = 0.5, the **Support Vector Machine** with **training epoch size of 2,000** and **gamma of 0.001** scored the highest with a value of 67.06%.

Table 11 shows the top three F1 scores achieved at different thresholds, epoch sizes, and learning models.

Table 11: Top Three F1 Scores for Various Thresholds, Training Epoch Sizes, and Learning Rates/Gamma

Type (thr=0.3)	**SVM**	**ANN**	**LR**
TE=1000 LR/gamma = 0.01	53.33	**71.99**	72.57
TE=5000 **LR/gamma = 0.01**	**75.03**	69.47	**75.56**
Type (thr=0.4)	**SVM**	**ANN**	**LR**
LR/gamma = 0.001	32.77	**63.92**	50.79
TE=5000 **LR/gamma = 0.01**	70.98	59.51	**72.04**
LR/gamma = 0.001	**71.92**	61.44	64.26
Type (thr=0.5)	**SVM**	**ANN**	**LR**
LR/gamma = 0.001	**67.06**	48.64	27.25
TE=5000 LR/gamma = 0.01	66.21	**64.30**	**65.28**

3.5 Summary of best performing models from various configurations

We present in Table 12 a summary of best performing models after performing the various configurations described in this paper.

Table 12: Best Performing Models on Different Evaluation Metrics for varying thresholds

Threshold	Train Accuracy	Test Accuracy	Precision	Recall	F1 Score
0.3	SVM (83.19)	LR (81.43)	ANN (78.60)	SVM (99.93)	LR (75.56)
0.4	SVM (90.68)	LR (88.98)	ANN (84.78)	SVM (100.00)	LR (72.04)
0.5	SVM (95.06)	SVM (93.51)	LR (91.84)	SVM (97.90)	SVM (67.06)

4. CONCLUSION AND RECOMMENDATIONS

The study was able to show an approach to determine how factual a statement is and provide a measure of how factual a statement claimed. Divided into two major parts, Sentence Classification and Semantic Similarity Analysis, we were able to create a system that combined relevance checking and certainty checking. We trained Support Vector Machine (SVM) with Radial Basis Function (RBF) Kernel, Logistic Regression (LR), and Artificial Neural Network (ANN) for sentence classification with different setups and determined that the Logistic Regression model is the most reliable model. We also determined that Support Vector Machine scored the highest Training Accuracy. However, the Logistic Regression model is the most balanced model, based on the five different evaluation metrics used in the study. In the Semantic Similarity Analysis, we extracted news articles from 16 satiric and reliable websites and utilized the Triplet Extraction Approach for us to determine a similarity score and classify input sentences as either reliable or satiric. We were able to achieve an 80% accuracy in terms of classifying the test data as either reliable or satiric using a set of 50 test sentences. An extra classification called "Insufficient data" catches test sentences that the system cannot determine as either reliable or satiric.

With the growing research interests focused on fake news detection and works focused on determining how factual statements are, this work can be used as a tool to determine certainty and relevance of information provided in our society. The framework presented in this study can used and improved by agencies who use fact-checking tools to improve their service.

For future research on sentence classification, one could explore on extracting more features in the dataset such as term frequency-inverse document frequency but for a local transcript dataset. The reason for using TF-IDF in a previous study [4] is because their target sentences for testing are from US transcripts, like their dataset. Other features could also be extracted and explored further. Another thing that can be considered for future research is to explore other parameters of the models used such as the kernel function for the Support Vector Machine model.

5. REFERENCES

[1] D. M. Lazer, M. A. Baum, Y. Benkler, A. J. Berinsky, K. M. Greenhill, F. Menczer, M. J. Metzger, B. Nyhan, G. Pennycook, D. Rothschild, M. Schudson, S. A. Sloman, C. R. Sunstein, E. A. Thorson, D. J. Watts, J. L. Zittrain. (2018). *The science of fake news: Addressing fake news requires a multidisciplinary effort*. Science Vol. 359, Issue 6380, pp. 1094-1096. [Online]. Available: https://science.sciencemag.org/content/359/6380/1094

[2] Elemia, C. (2017). *Mocha uson: 'ako po ay biktima ng fake news'*. [Online]. Available: https://www.rappler.com/nation/184243-mocha-uson-claims-victim-fake-news

[3] Conroy, N. J., V. L. Rubin, and Y. Chen. (2015). *Automatic deception detection: Methods for finding fake news*. Proceedings of the Association for Information Science and Technology, vol. 52, October 2015, pp. 1–4.

[4] Hassan, N., B. Adair, J. Hamilton, C. Li, M. Tremayne, J. Yang, and C. Yu. (2015). *The quest to automate fact-checking*. October 2015. [Online]. Available: https://idir.uta.edu/~naeemul/file/factchecking-cj2015-hassan-submission.pdf

[5] Hassan, N., F. Arslan, C. Li, and M. Tremayne. (2017). *Toward Automated Fact-Checking: Detecting Check-worthy Factual Claims by ClaimBuste*r. KDD 2017. August 2017. [Online]. Available: https://doi.org/10.1145/3097983.3098131

[6] Baly, R., G. Karadzhov, D. Alexandrov, J. Glass, P. Nakov. (2018). *Predicting Factuality of Reporting and Bias of News Media Sources*. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3528–3539. [Online]. Available: http://aclweb.org/anthology/D18-1389

[7] Constantino Jr., E. B., H. C. Malijan, and C. N. Peralta. (2017). *Factu: An android application for news verification using sentence similarity analysis and hidden markov model*. Undergraduate Special Problem. Institute of Computer Science, University of the Philippines Los Baños. Unpublished work.

[8] Adona, J. C. M. and R. N. C. Recario. (2017). *Coursera's moocs sentiment analysis: Dimensionality reduction visualization and model prediction explanation*. Undergraduate Special Problem. Institute of Computer Science, University of the Philippines Los Baños. Unpublished work.

[9] Raschka, S. (2015). *Python Machine Learning*. Packt Publishing Ltd., 2015.

[10] Muller, A.C. and S. Guido. 2016. *Introduction to Machine Learning with Python*. OReilly Media, Inc.

[11] Z. Wu and M. Palmer. (1994). "*Verb semantics and lexical selection*". Proceedings of 32nd annual Meeting of the Association for Computational Linguistics. June 27-30; Las Cruces, New Mexico.

[12] Geussoum, D., M. Miaoui, C. Tadj. (2016). "A modification of Wu and Palmer Semantic Similarity Measure". UBICOMM 2016: The

Tenth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies.

[13] Yang D., Xu B., Yang B., Wang J. (2013). *Secure Cosine Similarity Computation with Malicious Adversaries*. In: Chaki N., Meghanathan N., Nagamalai D. (eds) Computer Networks & Communications (NetCom). Lecture Notes in Electrical Engineering, vol 131. Springer, New York, NY. [Online] Available: https://link.springer.com/chapter/10.1007%2F978-1-4614-6154-8_52

[14] Rao, R.B., G. Fung, and R. Rosales. (2008). *On the Dangers of Cross-Validation*. An Experimental Evaluation. SDM.