A2Text-Net: A Novel Deep Neural Network for Sarcasm Detection

Conference Paper · December 2019

DOI: 10.1109/CogMI48466.2019.00025

5 authors, including:

Liyuan Liu
Kennesaw State University
15 PUBLICATIONS **77** CITATIONS
SEE PROFILE

Jennifer Priestley
Kennesaw State University
59 PUBLICATIONS **378** CITATIONS
SEE PROFILE

Yiyun Zhou
Kennesaw State University
11 PUBLICATIONS **87** CITATIONS
SEE PROFILE

Herman Ray
Kennesaw State University
31 PUBLICATIONS **75** CITATIONS
SEE PROFILE

A2Text-Net: A Novel Deep Neural Network for Sarcasm Detection

Liyuan Liu*, Jennifer Lewis Priestley*, Yiyun Zhou*, Herman E. Ray*, Meng Han†
†Corresponding author
*Analytics and Data Science Institute, Kennesaw State University
†College of Computing and Software Engineering, Kennesaw State University

***Abstract*—Sarcasm is a common form of irony in which users usually express their negative attitudes using contrary words. Predicting sarcasm is an essential part of investigating human social interaction. Improvements in classifying sarcasm have the potential to improve other dimensions of human sentiment (e.g., brand preference, political views). In face-to-face communication, the changing of voice, eye contact, physical position, etc. provides the audience with cues to detect sarcasm. However, detecting sarcasm exclusively with text is particularly challenging, given the lack of these subtle human-centric cues. In this study, we employed a new deep neural network: A2Text-Net to mimic the face-to-face speech, which integrates auxiliary variables such as punctuations, part of speech (POS), numeral, emoji, etc. to increase classification performance. The experiment results provide evidence that our A2Text-Net approach improves classification performance over conventional machine learning and deep learning algorithms.**

***Index Terms*—sarcasm detection, deep learning, machine learning, sentiment analysis, social media**

I. Introduction

Sarcasm is a general form of irony that uses language, which is inherently contradictory to what is intended. In face-to-face communication, the speaker's physical cues, tone, facial expressions, etc. could help audiences to detect sarcasm. However, in social media and other text-based environments, sarcasm detection becomes a difficult task. It is difficult to classify sarcastic information exclusively from text; the cues that provide important context are missing.

Improved sarcasm detection supports a better interpretation of customer sentiment, political commentary, and communication of information through social media platforms. For example, there are many applications using tweets to detect threats or disasters [1], [2], [3], but a sarcastic tweet could generate incorrect classification results if the prediction is exclusively based on the semantic interpretation of each word. For example, A twitter account tweeted "*And 12 bombers depart now ... to Lithuania!*" Using conventional sentiment analysis models, this tweet has a high possibility to be classified as the event of a threat. Actually, it turned out that Lithuania gave Russia zero points for its performance in the Eurovision Song Contest 2015 [4].

There are mainly two approaches used to detect sarcastic records: rule-based and classical machine learning and deep learning approaches [5]. In the rule-based approach, researchers try to recognize sarcastic records using specific pieces of evidence, such as using a positive verb, negative nouns, etc. For example, Riloff *et al.* [6] employed positive sentiment phrases and negative situation phrases to classify sarcasm. In machine learning and deep learning-based approaches, most researchers have applied bag of words as their features to classify sarcasm. For example, Khodak *et al.* employed bag-of-words methods to convert unstructured data to structured data. More recently, word embedding has become a popular method to represent text to vectors. In Ghosh *et al.* [7], weighted textual matrix factorization, word2vec representation, and GloVe representation were used to represent each word in the context vector. Their results show a word embedding approach increased the baseline approach performance. In the current study, we are not only focused on the model phase, but we also developed a framework to improve model selection using auxiliary features. We developed our A2Text-Net deep neural network that combined multiple auxiliary data with the output of word embedding. The framework of A2Text-Net is provided in Figure 1.

There are three layers in A2Text-Net. The first layer, named "Hypothesis Test Layer," aims to decide if the auxiliary variable is suitable to add to the text. The appropriate statistical test could be performed in this layer. For example, conducting a chi-square test on the frequency of punctuations between two groups (sarcastic records and not sarcastic records) could help determine if there is a significant difference in punctuations between two groups. The second layer is the "Feature Processing Layer." For text data, a word embedding layer could train the parameters of each word and convert unstructured text data to structured data. The dummy variables could be created regarding the auxiliary variables selected from the first layer. The two-channel data, word embedding outputs, and dummy auxiliary variables will be connected as the inputs into the third layer, the "Neural Network Layer."

The main contributions of this study are:

- A proposed novel deep neural network A2Text-Net to help increase the sarcasm detection models' performance. With experimental results from four datasets, we conclude that the proposed method not only improves classification performance but also improves users' understanding of the insights in the data more comprehensively.
- The proposed auxiliary variables can be selected using a statistical hypothesis test, and can improve the detection of sarcasm.

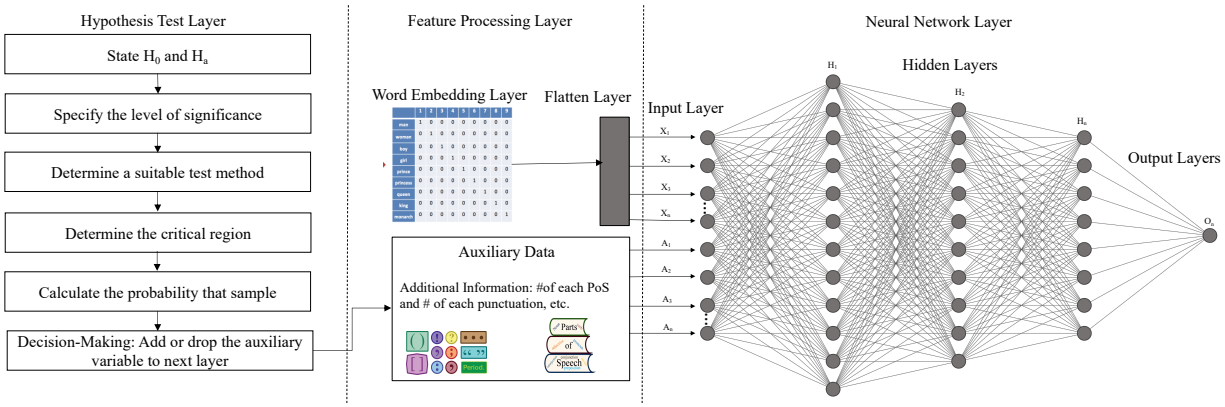


Figure 1: Framework of A2Text-Net Deep Neural Network

In this study, Section II discusses current research on the topic of sarcasm detection, Section III illustrates the newly developed methodology, Section IV discusses experimental results and Section V provides study conclusion and suggestions for future research.

II. Related Works

Researchers use three general approaches to classify sarcastic information: rule-based, machine learning-based, and deep learning-based approaches [5]. The details of some related research are shown in Table I. Abbreviations used in Table I: Naive Bayes (NB), Decision Tree (DT), Logistic Regression (LR), Support Vector Machines (SVM), Neural Networks (NN), and Convolutional Neural Network (CNN), F-1 Score (F).

In 2013, Riloff *et al.* first collected 3000 tweets and used a rule-based method to detect sarcasm. They presented a novel bootstrapping algorithm that could automatically detect the positive sentiment phrases and negative situation phrases from the collected tweets. Their experiment results show the features they selected generated a better F1 score compared with using standard features, such as N-grams [6]. Liebrecht *et al.* also performed their study using tweets data. They employed several features, such as N-grams, emotion marks, etc. to classify sarcastic records. Their study illustrated that sarcasm is often signaled by hyperbole, using intensifiers and exclamations. For the non-hyperbolic sarcastic records, they often received an explicit marker [8].

One year later, Maynard *et al.* proposed a sarcasm detection method using the polarity of tweets. They also extracted rules that enabled them to improve the detection accuracy. They involved hashtags in their analysis and found sarcastic records, which included the hashtags, can be detected more efficiently [16]. Buschmeier *et al.* also used hyperbole as one of their features and formulated the sarcasm detection problem as a supervised classification problem. They performed different classification algorithms on the imbalanced dataset. [9]. In 2015, Joshi *et al.* used sentiment incongruities, the largest positive/negative subsequence, positive and negative words, lexical polarity as features, and employed their designed model to classify sarcasm messages [11]. Rajadesingan *et al.* employed theories from behavioral and psychological studies to construct a behavioral modeling framework. They used several features, such as complexity-based, emotion expression-based, contrast-based etc. in their research. Their experiment results show the proposed framework has better performance compared with conventional methods. However, only using polarity to classify sarcasm is not sufficient; the polarity of sarcastic records is particularly difficult for both machine and humans in text-only environments [12].

Liu *et al.* employed punctuation symbols, recurring sequences, semantic imbalance rate, rhetorical feature, homophony, and construction as their input features. They first explored the characteristics of both English and Chinese sarcastic sentences in imbalanced social media data. They proposed a novel multi-strategy ensemble learning approach (MSELA) to handle the imbalanced problem [10]. Bouazizi *et al.* [17] proposed a pattern-based method to detect sarcasm in 2016. They separated sarcasm detection algorithms into four different types of analysis: sentiment-based, punctuation-based, syntactic-based, and pattern-based. They confirmed that the pattern-based analysis generated better performance compared with the other methods. However, in our proposed model, it is not necessary to separate sarcasm detection algorithms to different types. We could employ both sentiment, punctuation, syntactic and patterns simultaneously.

More recently, deep learning algorithms have increasingly received formalized research attention in the context of sarcasm detection - primarily because of their ability to handle non-linear combinations. The reason is deep learning algorithms could handle the non-linear combinations and uncertainly, especially for text data. More and more researchers applied deep learning algorithms to classify sarcasm instead of using conventional machine learning algorithms. Joshi *et al.* developed their detection models using word embedding similarity as features, and employed SVM to predict the sarcastic tweets on social media [13]. Felbo *et al.* used emoji occurrences to detect sarcasm on social media. The authors employed DeepMoji and LSTM as their models and confirmed that the diversity of their emotional labels yielded a performance improvement over previous methods [18]. In 2018, Hazarika *et al.* employed

	Salient Features	Models	Highest Reported Performance	Hypotheses Tested
Riloff et al. 2013 [6]	positive verbs, negative phrases	bootstrapped lexicons, SVM	F: 0.51	No
Liebrecht et al. 2013 [8]	N-grams, emotion marks, intensifiers	winnow classification	AUC: 0.67	No
Buschmeier et al. 2014 [9]	interjection, ellipsis, hyperbole, imbalance-based	SVM, LR, DT, RF, NB	F: 0.75	No
Liu et al. 2014 [10]	punctuation symbols, recurring sequences, semantic imbalance rate, rhetorical feature, homophony, construction	full-training, bagging, SMOTEBoost, AdaBoost, RF, undersampling multi-classifiers ensemble, LR	AUC: 0.90	No
Joshi et al. 2015 [11]	sentiment incongruities, largest positive or negative subsequence, positive and negative words, lexical polarity	LibSVM with RBF kernel	F: 0.80	No
Rajadesingan et al. 2015 [12]	complexity-based, emotion expression-based, contrast-based etc.	contrast approach, hybrid approach, SCUBA, random classifier, majority classifier, N-gram classifier	AUC: 0.83	No
Joshi et al. 2016 [13]	features based on word embedding similarity	SVM	F: 0.73	No
Hazarika et al. 2018 [14]	contextual features, user embeddings, etc	CNN, CNN–SVM, CUE–CNN, CNN based designed model	F: 0.86	No
Subramanian et al. 2019 [15]	emoji, text	designed model	F: 0.99	No

Table I: Meta Analysis of Related Studies

deep convolutional neural networks to detect sarcasm. They applied their designed models based on a pre-trained CNN model. They extracted the sentiment, emotion, and personality features and then performed their models on benchmark datasets. With the experiment results, their method could reach better performance compared with the state-of-the-art techniques [14]. In 2019, Subramanian *et al.* also performed a new sarcasm detection model named "ESD" that combined emoji and text information to classify sarcasm. Their model performed better compared with the state-of-the-art ones [15].

In this study, we found that sarcasm detection models do not have to be necessarily separated into the machine learning-based, deep learning-based, and rule-based approaches. Using our proposed A2Text-Net neural network, we can test any rule-based hypothesis in the first layer, then combine the word embedding results with auxiliary variables to training a deep neural network. The auxiliary variables are not limited to only one variable, such as punctuation, part of speech (POS), emoji, etc. We could use all of them if there exists a significant difference between the two groups. Compared with previous research, we can statistically analyze the significance of each rule, then employ either machine learning or deep learning algorithm to classify sarcasm. After testing our model on four public datasets, we found that our approach not only improved the detection of sarcasm but also supported improved overall insights.

III. Methodology

In this study, we employed logistic regression (LR), support vector machine (SVM), random forest (RF), deep neural networks (DNN), long-short-term memory recurrent neural networks (LSTM), gated recurrent units (GRU) as the baseline models and classify the sarcasm. We tested our proposed A2Text-Net neural network using the same parameter used in the DNN. This was necessary to ensure a fair, baseline comparison. The problem formulation and framework of A2Text-Net are discussed as follows.

A. A2Text-Net

There are three layers in the A2Text-Net neural network. The "Hypothesis Layer" selects the appropriate auxiliary variable that is added in the next "Feature Processing Layer." In this layer, a set of hypotheses could be conducted. For each pair of the ${H_0}^n$ and ${H_1}^n$, the appropriate statistical analysis was selected based upon the distribution of collected data. After computing the test, the statistic approach was employed to determine if there exists a significant difference between the sarcastic messages and not sarcastic messages. If yes, the auxiliary variable will be created based on the statistical results. For example, if we tested that there was a significant difference in punctuation between two groups, then each punctuation will be considered as an auxiliary variable. Multiple variables will be created as the auxiliary variables, including the frequency of punctuation in each record, such as the number of question marks, the number of

dashes, etc. The details of the first layer are shown in Algorithm 1. Regarding the distribution of the proposed auxiliary variable for two groups, the appropriate hypothesis test will be selected to test if there exists a significant difference between the two groups.

Algorithm 1 A2Text-Net: Algorithm for Hypothesis Layer

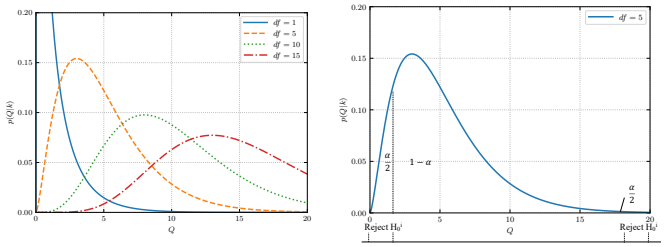
Input: a set of proposed null hypothesis and alternative hypothesis $\mathcal{H}_0 = ({H_0}^1, {H_0}^2, {H_0}^3, ..., {H_0}^n)$, $\mathcal{H}_1 = ({H_1}^1, {H_1}^2, {H_1}^3, ..., {H_1}^n)$ of each auxiliary variable c_n
Output: a set of auxiliary variable $\mathcal{A} = (A_1, A_2, A_3, ..., A_m)$ will be implemented to next layer

Initialization
for i in n **do**
  $\mathcal{S} \leftarrow$ test statistic
  $\rho \leftarrow$ distribution
  $\varsigma \leftarrow$ significant level
  $\mathcal{R} \leftarrow$ decision rule
  $\mathcal{S}_{new} \leftarrow$ calculate the statistic value of $\mathcal{S}$
  $p \leftarrow$ calculate p value
  if $p < \varsigma$ **then**
    ${H_0}^i$ was rejected, statistical significant
    $\mathcal{A}$.append (A_i)
  else
    ${H_0}^i$ was accepted, not significant
  end if
end for

In this study, we proposed two auxiliary variables: punctuation and POS. The chi-squared test is used to determine whether there is a significant difference between the expected frequencies and the observed frequencies in one or more categories [19] so that it is selected in this study. The calculation of chi-square is shown in Equation (1). O_i denotes the observed frequency, and E_i denotes the expected frequency.

$$\chi^2 = \sum_{k=1}^{n} \frac{(O_i - E_i)^2}{E_i} \tag{1}$$

Figure 2(a) shows the probability density function examples of



(a) Examples of Chi-square Distribution (b) Chi-square P-value Method

Figure 2: Chi-square Test

the chi-square distribution with different degrees of freedom. A chi-square random variable X with n degrees of freedom has probability density function shows below.

$$f(x) = \frac{x^{n/2-1} e^{-x/2}}{2^{n/2}\Gamma(n/2)} \qquad x > 0 \tag{2}$$

Figure 2(b) shows two-tail P-value method with chi-square distribution. When the test statistic (i.e. the chi square value) is significant, the auxiliary variable is passed to the next level. The second layer is the "Feature Processing Layer."

Algorithm 2 A2Text-Net: Algorithm for Feature Processing and Neural Network Layer

Input: trained word embedding with flatten output $\mathcal{X} = (X_1, X_2, X_3, ..., X_j)$, auxiliary variable output $\mathbb{A} = (A_1, A_2, A_3, ..., A_k)$, label for each record $\mathcal{Y} = (Y_1, Y_2, Y_3, ..., Y_j)$, number of layers in neural network $l \in \mathbb{N}^+$, learning rate η
Output: $\Delta_{backprop}$

Initialization weights ω
$X' = Concatenate(\mathcal{X}, \mathbb{A})$
${\Delta_{ij}}^{(l)} \leftarrow$the error for all l,i,j
$\delta_{ij}^{(l)} \leftarrow 0$ for all l,i,j
for i=1 to m **do**
  $Z^l \leftarrow feedforward({X'}^{(i)}, \omega)$
  $d^l \leftarrow Z(L) - Y(i)$
  $\delta_{ij}^{(l)} \leftarrow \delta_{ij}^{(l)} + Z_j^l * \delta_i^{(l+1)}$
  if $j \neq 0$ **then**
    ${\Delta_{ij}}^{(l)} \leftarrow \frac{1}{m} * \delta_{ij}^{(l)} + \eta * \omega_{ij}^{(l)}$
  else
    ${\Delta_{ij}}^{(l)} \leftarrow \frac{1}{m} * \delta_{ij}^{(l)}$
    where $\frac{\partial}{\partial \omega_{ij}^{(l)}} J(\omega) = {\Delta_{ij}}^{(l)}$
  end if
end for

The primary purpose of this layer is to convert the unstructured data to structured data, and connect text features with auxiliary features. The data preprocessing consists of several aspects: tokenization by words, lower case all text, removing stopwords and punctuations, snow stemming, and lemmatization.

Following, we train the word embedding model. Some pre-trained word embedding models also could be employed, such as word2vec [20], GloVe [21], etc. In this study, we trained the word embedding model instead of using the pre-trained models. The primary purpose of word embedding is to represent words and documents using a dense vector representation. The embedding layer is initialized with random weights and will learn an embedding for all of the words in the training dataset using a deep neural network [22]. A flattening layer is followed after the word embedding layer because it can reduce the word embedding layer dimensions to meet the input dimension requirement of the next layer. For the auxiliary features, the new features will be created regarding the results of the first layer since the context vectors and auxiliary variables have different dimensions that cannot act as the inputs to neural networks at the same time. For example, the auxiliary features could be the count of each POS tag in each record. The output of the flatten layer and auxiliary features layer will be concatenated together as the two-channel inputs to the next neural network layer.

In the "Neural Network Layer," a back-propagation deep neural network will be applied. The inputs of the neural network layer will be the word embedding outputs connected with a flattening layer of the text denotes to $\mathcal{X} = (X_1, X_2, X_3, ..., X_j)$, and the auxiliary variables which we decided by the first layer, indicates to $\mathbb{A} = (A_1, A_2, A_3, ..., A_k)$.

In this study, a ReLu function is employed in the hidden layer and a sigmoid function in the output layer. The equations of Relu (Equation (3)) and sigmiod (Equation (4)) functions are shown:

$$f(X) = max(X, 0) \tag{3}$$

$$sigmoid(X) = \frac{e^X}{e^X + 1} \tag{4}$$

The number of hidden layers is defined by evaluating the generalized error of each network.

B. Machine Learning and Deep Learning Algorithms

1) Logistic Regression: The goal of logistic regression is to model the event probability of a random variable. The generalized linear model function of logistic regression can be defined with the parameter θ. Compared with linear regression, logistic regression is appropriate when the dependent variables are dichotomous or multinomial. In this study, the binary dependent variable is sarcasm or not. Suppose the word embedding output is a set of $\mathcal{X} = (X_1, X_2, X_3, ..., X_j)$, the optimal solution of logistic regression is the maximum likelihood function as Equation (5) shows [23].

$$\begin{aligned} L(\theta|x) &= Pr(\mathcal{Y}|\mathcal{X}; \theta) \\ &= \prod_j Pr(Y_j|X_j; \theta) \\ &= \prod_j h_\theta(X_j)^{Y_j}(1 - h_\theta(X_j))^{1-Y_j} \end{aligned} \tag{5}$$

After finding the optimal solution of the likelihood, the logistic regression estimates a multiple linear regression function defined as:

$$log(\frac{P}{1-P}) = B_0 + B_1X_1 + ... + B_jX_j \tag{6}$$

2) Support Vector Machine (SVM): The central concept of SVM is to find an optimal hyperplane or set of hyperplanes, which can be used for classification, regression, or other tasks like outliers detection [24]. For example, suppose the training word embedding output is $\mathcal{X} = (X_1, X_2, ..., X_j)$ and given the labels $\mathcal{Y} = (Y_1, Y_2, ..., Y_j)$ where $\mathcal{Y} \in$ {-1,1}. An optimal hyperplane will be found to separate the training data points with maximal margin. The objective function has $\xi_i =$max$(0, 1 - Y_i(w * X_i - b))$.

$$\begin{aligned} \min \quad & \lambda\|w\|^2 + C\sum_{i=1}^{n} \xi_i \\ \text{s.t.} \quad & y_i(w * (x_i + b)) \geq 1 - \xi_i, \quad \xi_i \geq 0 \end{aligned} \tag{7}$$

Then we could write the dual functions regarding Equation (7) and use the Lagrange multipliers method to find the optimal solution for the SVM.

3) Random Forests: Random forests is a classification algorithm belonging to the ensemble learning methods. It combined several weak learners, such as constructing a multitude of decision trees and then outputting the class that is the model of the categories or obtaining the average of the prediction of each tree [25]. Bagging is the basic technique applied in random forests. In this study, let the outputs of word embedding as $\mathcal{X} = (X_1, X_2, ...X_j)$ and the responses as $\mathcal{Y} = (Y_1, Y_2, ..., Y_j)$. In the bagging method, the bagging will repeat α times to select a random sample with replacement from the training dataset and construct multiple trees with the random samples. Then the ensemble prediction will be made using the mean of each weak learner.

4) Deep Neural Networks: Deep Neural Networks, as a conventional deep learning algorithm, are widely used in regression and classification problems now. Since neural networks are robust and can handle non-linear combinations and unknown situations, they are commonly employed in many fields, such as cyber-attack [26], data privacy [27], [28], social media [29], healthcare [30] etc. In this study, the back-propagation neural network is used. The back-propagation methods enable the weights to be updated on the process and involve a multi-layer topology hat, which includes an input layer, a hidden layer, and an output layer [31]. More details of the neural network can be found in Algorithm 2.

5) Long-short-term Memory: LSTM is a recurrent neural network (RNN) widely used in sequence data. Compared with traditional vanilla RNNs, it can solve the vanishing gradient problem and address better performance regarding previous studies [32], [33]. There are three gates and memory cells. The gates aim to control the flow of information. Capturing the most important information and forget the less important information. Each memory block contains an input gate to control the flow of input activations into the memory cell, an output gate to control the output flow of cell activations into the rest of the network, and a forget gate [34]. For one LSTM unit, the equations of the three gates are shown below. i_t denotes to the input gate's activation vector, f_t denotes to the forget gate's activation vector, o_t denotes to the output gate's activation vector.

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \tag{8}$$

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \tag{9}$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \tag{10}$$

6) Gated Recurrent Unit: Compared with LSTM, a gated recurrent unit (GRU) has similar concepts but only with two gates in each memory block instead of three. The two gates in GRU are update gate and reset gate. Each gate includes a sigmoid function, as shown in Equation (11) and Equation (12). From the previous studies, GRU has better performance compared with LSTM on the smaller datasets [35].

$$u_t = \sigma_g(W_u x_t + U_u h_{t-1} + b_u) \tag{11}$$

$$r_t = \sigma_g(W_r x_t + U_r h_{t-1} + b_r) \tag{12}$$

IV. Experiments Results and Discussion

A. Dataset Description

In this study, we employed four public datasets and tested the performance of the A2Text-Net neural network. We use Python environment to develop the models, and the source code can be found from https://github.com/lliyuan1117/A2Text-Net.

- **News Headlines Dataset:** This dataset contains 26,709 news headlines collected from the "Onion" and "HuffPost" website. There are 14,985 text records labeled as "Not sarcastic (0)," and 11,724 text records are labeled as "Sarcastic (1)". The dataset could be download from the "Kaggle" website [36]. The data provider labeled the dataset.
- **Tweets Dataset A:** The original dataset is collected by Riloff *et al.* [6]. In their paper, they reported that a total of 3,000 tweets with 693 of them are annotated as sarcastic. However, the authors only publish the tweet's ID of their dataset. Many tweets have been removed from when the tweet IDs were initially collected. For this study, we collected 1,956 tweets, of which 308 of them are sarcastic records, and 1,648 of them are not sarcastic records.
- **Tweets Dataset B:** This is a larger sarcasm dataset collected by Ghosh *et al.* [37]. There are 54,931 records in the dataset, 25,872 of them are labeled as sarcastic tweets, and 29,059 are labeled as not sarcastic tweets. For Tweets dataset A and B, all of them were downloaded from a Github account [38]. The dataset creators labeled the data. The details could be found in their paper.
- **Reddit Dataset:** This dataset includes 4,692 Reddit comments.Of these comments, 2,346 are sarcastic records, and 2,346 are not sarcastic records.

B. Evaluation Metrics

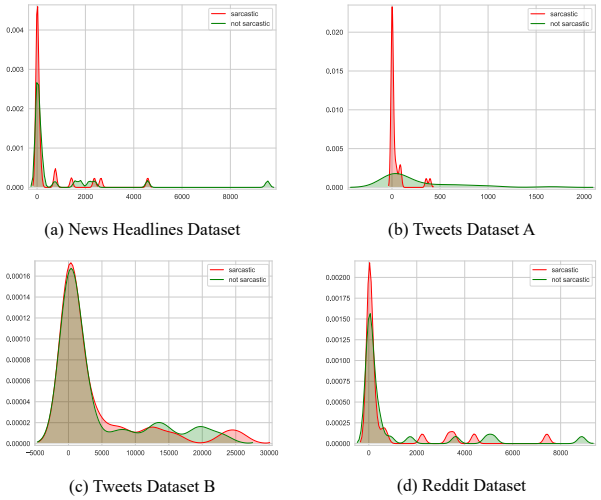
Computing just the accuracy score for a classification model gives a half-done view of the model's performance. For example, if we have an imbalanced dataset with 99% of instances in negative class and only 1% in the other. The accuracy could reach 99% if all the instances predicted as negative. However, this model is failed. There are more evaluation metrics, such as the confusion matrix, ROC curve, precision, recall, etc. In this study, we employ ROC AUC, recall, precision, and F1 score as the evaluation metrics. The F1 score is the harmonic average of precision and recall. It is popularly used to evaluate text classification.

We also apply 5-cross satisfied validation to test conventional machine learning and deep learning models and A2Text-Net neural network.

C. Hypothesis Test Layer

We have two pairs of hypotheses in this study. They are illustrated as follow:

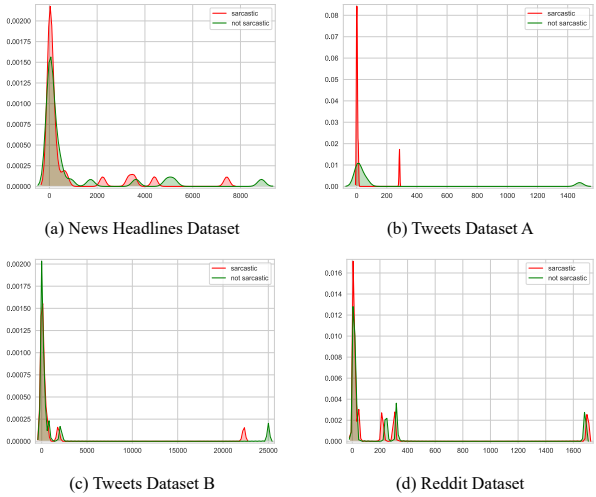
$H_0{}^1$*: the frequency distribution of punctuation is the same among the two groups.*



(a) News Headlines Dataset (b) Tweets Dataset A

(c) Tweets Dataset B (d) Reddit Dataset

Figure 3: Distribution of Punctuation Frequency between Two Groups



(a) News Headlines Dataset (b) Tweets Dataset A

(c) Tweets Dataset B (d) Reddit Dataset

Figure 4: Distribution of POS Frequency between Two Groups

$H_1{}^1$*: the frequency distribution of punctuation is different between the two groups.*
$H_0{}^2$*: the frequency distribution of POS is the same among the two groups.*
$H_1{}^2$*: the frequency distribution of POS is different between the two groups.*
We compared the distributions of punctuation and POS between the two groups. As Figure 3 and Figure 4 show, the distributions of two groups are different. We also employed a chi-square test to examine each pair of the hypotheses in this study using the four datasets. We found that both punctuation and POS significantly contribute to the classification of sarcasm. The chi-square test results are shown in Table III.

With the chi-square test results, punctuation and POS were added as the auxiliary variables to the next layer.

D. Feature Processing Layer and Neural Network Layer

In the "Feature Processing Layer," the two-channel data is processed. One channel is the text data, where we can convert

		DNN	LSTM	SVM	RF	LR	GRU	A2Text-Net
News Headlines Dataset	F-1	0.853	0.850	0.548	0.537	0.631	0.798	**0.862**
	Recall	0.853	0.850	0.553	0.462	0.685	0.800	**0.862**
	Precision	0.853	0.850	0.543	0.641	0.777	0.800	**0.863**
	ROC AUC	0.930	0.927	0.600	0.630	0.890	0.900	**0.937**
Tweets Dataset A	F-1	0.934	**0.993**	0.805	0.983	0.890	0.899	0.900
	Recall	0.940	**0.993**	0.944	0.968	0.943	0.975	0.910
	Precision	0.944	0.993	0.702	**0.999**	0.842	0.852	0.917
	ROC AUC	0.993	**0.997**	0.934	0.984	0.932	0.956	0.970
Tweets Dataset B	F-1	0.794	0.798	0.547	0.527	0.638	0.756	**0.801**
	Recall	0.794	0.798	0.580	0.464	0.675	0.756	**0.802**
	Precision	0.795	0.799	0.518	0.611	0.725	0.761	**0.803**
	ROC AUC	0.873	0.876	0.552	0.601	0.800	0.843	**0.884**
Reddit Dataset	F-1	0.699	0.690	0.601	0.563	0.581	0.654	**0.710**
	Recall	0.701	0.691	0.582	0.521	0.594	0.657	**0.711**
	Precision	0.704	0.695	0.622	0.612	0.607	0.664	**0.712**
	ROC AUC	0.774	0.763	0.614	0.595	0.653	0.730	**0.779**

Table II: Experiment Results for Four Datasets

		χ^2	df	sig.
News Headlines Dataset	Punctuation	2162.3	16	<**0.0001**
	POS	8869.5	29	<**0.0001**
Tweets Dataset A	Punctuation	2002.4	29	<**0.0001**
	POS	37.8	8	<**0.0001**
Tweets Dataset B	Punctuation	34583.7	31	<**0.0001**
	POS	1727.1	15	<**0.0001**
Reddit Dataset	Punctuation	7420.9	28	<**0.0001**
	POS	533.7	15	<**0.0001**

Table III: chi-square Test Results

the unstructured text data to a vector. The second channel is the auxiliary features. For each dataset, the punctuation and POS type will be analyzed as one feature. The value of each feature is the frequency of each punctuation or POS in the record. We first get all the type of punctuation that the dataset contained, then each punctuation will be one feature, and then the value will be the count of each punctuation. For example, in Tweets Dataset A, we created 30 punctuations features and 9 POS features.

We test logistic regression, support vector machine, random forest, deep neural network, LSTM, GRU only using word embedding data. In logistic regression, L2 regularization is performed to prevent the overfitting problem. We applied "rmsprop" as the optimization method to train the hyperparameters. Binary cross-entropy is operated as the loss function. In the support vector machine, a radial basis function kernel is utilized to map the non-linear separable data points into a higher dimensional space. In random forests, Gini is the criterion to measure the quality of a split.

In the neural network, a dropout rate sets to 0.2, and two hidden layers use the ReLu function, and the output layer uses a sigmoid function to classify the sarcasm status. The first hidden layer has 64 neurons, and the second hidden layer has 32 neurons. Similar to logistic regression, "rmsprop" optimizer is utilized to find the optimal solution of the binary cross-entropy object function. In LSTM and GRU, the units number set to 32 for both two hidden layers, "rmsprop" is set as the optimization method and binary cross-entropy is the object function, the dropout rate is 0.2. In A2Text-Net, all the parameters were arranged as same as the neural networks to ensure there was a fair comparison.

Table II shows the models' performance for four datasets. The machine learning and deep learning models only use text data; the A2Text-Net is the model combine text and auxiliary data together. We use F-1 score, recall, precision, and ROC AUC as our evaluation metrics. From the results of the experiments, we can see that our proposed A2Text-Net neural network has the best performance on three datasets. The Tweets Dataset A is a very small and imbalanced dataset, LSTM has the best performance to address the sarcasm classification problem on this dataset. It is evident that the A2Text-Net neural network could help the DNN models to achieve better classification results. Since the datasets we had are not as same as the authors used in their research, some tweets with the tweet's ID the authors published are deleted. So compared our proposed method with state-of-the-art methods is not a fair comparison. However, it is worth to mention that for "Tweets Dataset A", the authors reported performance is an F1 score equal to 0.51 [6], ours is 0.90.

V. Conclusion

In this study, we proposed a novel deep neural network to detect sarcasm, namely "A2Text-Net". The proposed method could be implemented in many areas, such as social media, product branding, customer service, etc. The three layers in the A2Text-Net neural network enables us to test each hypothesis we made, and statistically support selecting features before we train the deep neural networks. The experiment results show our proposed method could achieve better performance compared with other baseline models. In addition, the hypothesis test also could help us to interpret the data insights and found the patterns of sarcasm. A2Text-Net is a suitable model to detect sarcasm that allows us to add more relevant auxiliary features other than only

use text features. Our proposed method is also easy to adjust. The third layer in our A2Text-Net could adapt to any other deep learning models if they have better performance than the deep neural network. In future work, we will study how to use the output of the current one to improve other sentiment analysis projects.

REFERENCES

[1] M. Spitters, P. T. Eendebak, D. T. Worm, and H. Bouma, "Threat detection in tweets with trigger patterns and contextual cues," in *2014 IEEE Joint Intelligence and Security Informatics Conference*. IEEE, 2014, pp. 216–219.

[2] A. L. Wester, L. Øvrelid, E. Velldal, and H. L. Hammer, "Threat detection in online discussions," 2016.

[3] P. S. Earle, D. C. Bowden, and M. Guy, "Twitter earthquake detection: earthquake monitoring in a social world," *Annals of Geophysics*, vol. 54, no. 6, 2012.

[4] E. Forslid and N. Wikén, "Automatic irony-and sarcasm detection in social media," 2015.

[5] A. Joshi, P. Bhattacharyya, and M. J. Carman, "Automatic sarcasm detection: A survey," *ACM Computing Surveys (CSUR)*, vol. 50, no. 5, p. 73, 2017.

[6] E. Riloff, A. Qadir, P. Surve, L. De Silva, N. Gilbert, and R. Huang, "Sarcasm as contrast between a positive sentiment and negative situation," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 704–714.

[7] D. Ghosh, W. Guo, and S. Muresan, "Sarcastic or not: Word embeddings to predict the literal or sarcastic meaning of words," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1003–1012.

[8] C. Liebrecht, F. Kunneman, and A. van Den Bosch, "The perfect solution for detecting sarcasm in tweets# not," 2013.

[9] K. Buschmeier, P. Cimiano, and R. Klinger, "An impact analysis of features in a classification approach to irony detection in product reviews," in *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 2014, pp. 42–49.

[10] P. Liu, W. Chen, G. Ou, T. Wang, D. Yang, and K. Lei, "Sarcasm detection in social media based on imbalanced classification," in *International Conference on Web-Age Information Management*. Springer, 2014, pp. 459–471.

[11] A. Joshi, V. Sharma, and P. Bhattacharyya, "Harnessing context incongruity for sarcasm detection," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 2015, pp. 757–762.

[12] A. Rajadesingan, R. Zafarani, and H. Liu, "Sarcasm detection on twitter: A behavioral modeling approach," in *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. ACM, 2015, pp. 97–106.

[13] A. Joshi, V. Tripathi, K. Patel, P. Bhattacharyya, and M. Carman, "Are word embedding-based features useful for sarcasm detection?" *arXiv preprint arXiv:1610.00883*, 2016.

[14] D. Hazarika, S. Poria, S. Gorantla, E. Cambria, R. Zimmermann, and R. Mihalcea, "Cascade: Contextual sarcasm detection in online discussion forums," *arXiv preprint arXiv:1805.06413*, 2018.

[15] J. Subramanian, V. Sridharan, K. Shu, and H. Liu, "Exploiting emojis for sarcasm detection," in *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*. Springer, 2019, pp. 70–80.

[16] D. Maynard and M. A. Greenwood, "Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis," in *LREC 2014 Proceedings*. ELRA, 2014.

[17] M. Bouazizi and T. O. Ohtsuki, "A pattern-based approach for sarcasm detection on twitter," *IEEE Access*, vol. 4, pp. 5477–5488, 2016.

[18] B. Felbo, A. Mislove, A. Søgaard, I. Rahwan, and S. Lehmann, "Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm," *arXiv preprint arXiv:1708.00524*, 2017.

[19] A. Satorra and P. M. Bentler, "A scaled difference chi-square test statistic for moment structure analysis," *Psychometrika*, vol. 66, no. 4, pp. 507–514, 2001.

[20] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[21] 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)*, 2014, pp. 1532–1543.

[22] A. Gulli and S. Pal, *Deep Learning with Keras*. Packt Publishing Ltd, 2017.

[23] A. Ng, "Cs229 lecture notes," *CS229 Lecture notes*, vol. 1, no. 1, pp. 1–3, 2000.

[24] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[25] T. K. Ho, "Random decision forests," in *Proceedings of 3rd international conference on document analysis and recognition*, vol. 1. IEEE, 1995, pp. 278–282.

[26] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," *ACM SIGKDD Explorations Newsletter*, vol. 19, no. 1, pp. 22–36, 2017.

[27] P. Gundecha, G. Barbier, and H. Liu, "Exploiting vulnerability to secure user privacy on a social networking site," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 511–519.

[28] Y. Zhou, M. Han, L. Liu, J. S. He, and Y. Wang, "Deep learning approach for cyberattack detection," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2018, pp. 262–267.

[29] K. Cheng, J. Li, J. Tang, and H. Liu, "Unsupervised sentiment analysis with signed social networks," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[30] L. Liu, M. Han, Y. Zhou, and Y. Wang, "Lstm recurrent neural networks for influenza trends prediction," in *International Symposium on Bioinformatics Research and Applications*. Springer, 2018, pp. 259–264.

[31] C.-L. Chang and C.-H. Chen, "Applying decision tree and neural network to increase quality of dermatologic diagnosis," *Expert Systems with Applications*, vol. 36, no. 2, pp. 4035–4041, 2009.

[32] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[33] H. Zen, Y. Agiomyrgiannakis, N. Egberts, F. Henderson, and P. Szczepaniak, "Fast, compact, and high quality lstm-rnn based statistical parametric speech synthesizers for mobile devices," *arXiv preprint arXiv:1606.06061*, 2016.

[34] A. Azzouni and G. Pujolle, "A long short-term memory recurrent neural network framework for network traffic matrix prediction," *arXiv preprint arXiv:1705.05690*, 2017.

[35] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[36] R. Misra and P. Arora, "Sarcasm detection using hybrid neural network," *arXiv preprint arXiv:1908.07414*, 2019.

[37] A. Ghosh and T. Veale, "Magnets for sarcasm: Making sarcasm detection timely, contextual and very personal," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 482–491.

[38] MirunaPislar, "Sarcasm-detection," https://github.com/MirunaPislar/Sarcasm-Detection/blob/master/res/README.md, Feb 2018.