# A Heuristic-driven Uncertainty based Ensemble Framework for Fake News Detection in Tweets and News Articles

Sourya Dipta Das[a], Ayan Basak[a], Saikat Dutta[b]

[a]Razorthink Inc, USA
[b]IIT Madras, India

## Abstract

The significance of social media has increased manifold in the past few decades as it helps people from even the most remote corners of the world to stay connected. With the advent of technology, digital media has become more relevant and widely used than ever before and along with this, there has been a resurgence in the circulation of fake news and tweets that demand immediate attention. In this paper, we describe a novel Fake News Detection system that automatically identifies whether a news item is “real” or “fake”, as an extension of our work in the CONSTRAINT COVID-19 Fake News Detection in English challenge. We have used an ensemble model consisting of pre-trained models followed by a statistical feature fusion network , along with a novel heuristic algorithm by incorporating various attributes present in news items or tweets like source, username handles, URL domains and authors as statistical feature. Our proposed framework have also quantified reliable predictive uncertainty along with proper class output confidence level for the classification task. We have evaluated our results on the COVID-19 Fake News dataset and FakeNewsNet dataset to show the effectiveness of the proposed algorithm on detecting fake news in short news content as well as in news articles. We obtained a best F1-score of **0.9892** on the COVID-19 dataset, and an F1-score of **0.9073** on the FakeNewsNet dataset.

*Keywords:* COVID-19, Language Model, Fake News, Ensemble, Heuristic, Uncertainty, Dropout, Bayesian Approximation

*URL:* dipta.juetce@gmail.com (Sourya Dipta Das), ayan.basak@razorthink.com (Ayan Basak), saikat.dutta779@gmail.com (Saikat Dutta)

## 1. Introduction

Fake news represents the press that is used to spread false information and hoaxes through conventional platforms as well as online ones, mainly social media. There has been an increasing interest in fake news on social media due to the political climate prevailing in the modern world [1, 2, 3], as well as several other factors. Detecting misinformation on social media is as important as it is technically challenging. The difficulty is partly due to the fact that even humans cannot accurately distinguish false from true news, mainly because it involves tedious evidence collection as well as careful fact checking. With the advent of technology and ever-increasing propagation of fake articles in social media, it has become really important to come up with automated frameworks for fake news identification.

In this paper, we describe our system which performs a binary classification on news items from social media and classifies it into "real" or "fake". We have used transfer learning in our approach as it has proven to be extremely effective in text classification tasks, with a reduced training time as we do not need to train each model from scratch. The primary steps for our approach initially include text preprocessing, tokenization, model prediction, and ensemble creation using a soft voting schema. Post evaluation, we have drastically improved our fake news detection framework with a Statistical feature fusion network (SFFN) with uncertainty estimation, and followed by a heuristic post-processing technique where both network takes into account the effect of important aspects of news items like username handles, URL domains, news source, news author, etc as statistical features. This approach has allowed us to produce much superior results when compared to other models in their respective datasets. We have also provided performance analysis of predictive uncertainty quality with proper metrics and showed improvement in overall performance, robustness of the SFFN with ablation study. We have also additionally performed an ablation study of the various attributes used in our post-processing approach.

Our algorithm is also applicable to detection of fake news items in long news articles. In this context, we have evaluated the performance of our approach on FakeNewsNet dataset [4]. Along with the news titles, we have also utilized the actual news body (document) in this case. We have tried out the modified BERT [5] based model as NewsBERT on these documents to obtain the prediction vectors, which can be used as additional features for our model. Using these features, we have shown that there has been a drastic improvement in the overall accuracy and F1-score in the case of FakeNewsNet. We have also quantified the model uncertainty in the fake news classification task.

## 2. Related Work

### *2.1. Fake News Detection*

Traditional machine learning approaches have been quite successful in fake news identification problem. Reis et al. [6] has used feature engineering to generate hand-crafted features like syntactic features, semantic features etc. The problem was then approached as a binary classification problem where these features were fed into conventional Machine Learning classifiers like K-Nearest Neighbor (KNN) [7], Random Forest (RF) [8], Naive Bayes [9], Support Vector Machine (SVM) [10] and XGBOOST (XGB) [11], where RF and XGB yielded results that were quite favourable. Shu et al. [12] have proposed a novel framework TriFN, which provides a principled way to model tri-relationship among publishers, news pieces, and users simultaneously. This framework significantly outperformed the baseline Machine Learning models as well as erstwhile state-of-the-art frameworks. With the advent of deep learning, there has been a significant revolution in the field of text classification, and thereby in fake news detection. Karimi et al. [13] has proposed a Multi-Source Multi-class Fake News Detection framework that can do automatic feature extraction using Convolution Neural Network (CNN) based models and combine these features coming from multiple sources using an attention mechanism, which has produced much better results than previous approaches that involved hand-crafted features. Zhang et al. [14] introduced a new diffusive unit model, namely Gated Diffusive Unit (GDU), that has been used to build a deep diffusive network model to learn the representations of news articles, creators and subjects simultaneously. Ruchansky et al. [15] has proposed a novel Capture-Score-Integrate (CSI) framework that uses an Long Short-term Memory (LSTM) network to capture the temporal spacing of user activity and a doc2vec [16] representation of a tweet, along with a neural network based user scoring module to classify the tweet as real or fake. It emphasizes the value of incorporating all three powerful characteristics in the detection of fake news: the tweet content, user source, and article response. Monti et al. [3] has shown that social network structure and propagation are important features for fake news detection by implementing a geometric deep learning framework using Graph Convolutional Networks. Julio et al. [17] have used a supervised approach for fake news classification using hand-crafted features like linguistic, lexical, psycholinguistic and semantic features, as well as news source and environmental features. They have applied traditional machine learning models on this data like KNN, Naive Bayes, Random Forest, SVM and XGBOOST, out of which Random Forest and XGBOOST have achieved the best results. Zellers et al. [18] have introduced a novel fake news generation model, GROVER, that possesses a GPT-like architecture. It has the capability to generate very realistic fake news items in a controlled manner, including various associated meta information like title, news source, publication date, author list, etc. GROVER also outperforms other deep-pretrained models while discriminating between real and fake news articles, hence, it is a powerful model for fake news generation and detection. Bang et al. [19] have tried to develop a robust model for fake news detection that can generalize across different test sets. They have shown their results by performing experiments on two different test sets - FakeNews-19 and Tweets-19. In one approach, they have fine-tuned transformer based language models using robust loss functions, that did not help to improve the F1-score on the FakeNews-19 dataset by much as compared to the traditional cross-entropy loss; however, it showed better generalization on the Tweets-19 dataset. They have also performed an influence-based data cleansing which has improved model robustness and adaptability. Shu et al. [20] has proposed an automated Fake News Detection framework, dEFEND, that uses a deep hierarchical co-attention network which takes into account the news items and user comments, and provides a classification output along with viable explanations.

### *2.2. Fake News Detection in news articles*

Felber [21] has analyzed the performance of some classical Machine Learning models using several linguistic features such as n-gram, readability, emotional tone and punctuation along with various preprocessing techniques like stop word removal, stemming/lemmatization, link removal. Shushkevich et al. [22] has used an ensemble technique consisting of Bidirectional LSTM (Bi-LSTM), SVM, Logistic Regression, Naive Bayes. Their combination of Logistic Regression and Naive Bayes models has produced results that are within 5% of state-of-the art results on the given dataset. Sharif et al. [23] have tried out various techniques like SVM, CNN, Bi-LSTM, and CNN+BiLSTM with tf-idf and Word2Vec embedding techniques, where SVM with tf-idf features has produced the best results. Gautam et al [24]. has proposed a solution where they have combined topical distributions obtained using Latent Dirichlet Allocation (LDA) and contextualized representations obtained using XLNet. These features are then passed through a 2-layer Feed Forward Neural Network in order to obtain the final classification output. Li et al. [25] has proposed an ensemble model consisting of various pre-trained models like BERT, RoBERTa, ERNIE, etc. using five-fold five-model cross validation. Their pseudo label algorithm has also been able to improve overall model performance.

### *2.3. Language models*

Most of the current state-of-the-art language models are based on Transformer [26] and they have proven to be highly effective in text classification problems. They provide superior results when compared to previous state-of-the-art approaches using techniques like Bi-directional LSTM, Gated Recurrent Unit (GRU) based models etc. The models are trained on a huge corpus of data. The introduction of the BERT [5] architecture has transformed the capability of transfer learning in Natural Language Processing. It has been able to achieve state-of-the art results on downstream tasks like text classification. RoBERTa [27] is an improved version of the BERT model. It is derived from BERT's language-masking strategy, modifying its key hyperparameters, including removing BERT's next-sentence pre-training objective, and training with much larger mini-batches and learning rates, leading to improved performance on downstream tasks. XLNet [28] is a generalized auto-regressive language method. It calculates the joint probability of a sequence of tokens based on the transformer architecture having recurrence. Its training objective is to calculate the probability of a word token conditioned on all permutations of word tokens in a sentence, hence capturing a bidirectional context. XLM-RoBERTa [29] is a transformer [26] based language model relying on Masked Language Model Objective. DeBERTa [30] provides an improvement over the BERT and RoBERTa models using two novel techniques; first, the disentangled attention mechanism, where each word is represented using two vectors that encode its content and position, respectively, and the attention weights among words are computed using disentangled matrices on their contents and relative positions, and second, the output softmax layer is replaced by an enhanced mask decoder to predict the masked tokens pre-training the model. ELECTRA [31] is used for self-supervised language representation learning. It can be used to pre-train transformer networks using very low compute, and is trained to distinguish "real" input tokens vs "fake" input tokens, such as tokens produced by artificial neural networks. ERNIE 2.0 [32] is a continual pre-training framework to continuously gain improvement on knowledge integration through multi-task learning, enabling it to learn various lexical, syntactic and semantic information through massive data much better.

### *2.4. Uncertainty*

Model uncertainty is a very important concept that is related to the model parameters. In order to capture model uncertainty, a prior distribution needs to be assigned over each weight in a neural network. Gal et al. [33] has developed a new theoretical framework casting dropout training in deep neural networks (NNs) as approximate Bayesian inference in deep Gaussian processes. They have shown that a neural network with arbitrary depth or non-linearities can be analogous to a probabilistic deep Gaussian process when dropout is applied before every weight layer. This theory presents tools to model uncertainty with dropout NNs, and shows a considerable improvement in predictive log-likelihood and Root Mean Squared Error (RMSE) compared to existing state-of-the-art methods. Lakshminarayanan et al. [34] has proposed a novel approach to estimate the predictive uncertainty using ensembles of deep neural networks.

This approach produced superior results when compared to traditional Bayesion Neural Networks, with an added advantage of being readily parallelizable and requiring less hyperparameter tuning. It also takes into account the data uncertainty as it produces higher uncertainty values for out-of-distribution examples.

## 3. Dataset Description

We have chosen 2 datasets to test our approach that have the necessary attributes that we require to extract statistical features:

**COVID-19 Fake News:** The dataset [35] for CONSTRAINT COVID-19 Fake News Detection in English challenge was provided by the organizers on the competition website[1]. It consists of data that have been collected from various social media and fact checking websites, and the veracity of each post has been verified manually. The "real" news items were collected from verified sources which give useful information about COVID-19, while the "fake" ones were collected from tweets, posts and articles which make speculations about COVID-19 that are verified to be false. The original dataset contains 10,700 social media news items, the vocabulary size (i.e., unique words) of which is 37,505 with 5141 words in common to both fake and real news. It is class-wise balanced with 52.34% of the samples consisting of real news, and 47.66% of fake samples. These are 880 unique username handle and 210 unique URL domains in the data.

**FakeNewsNet:** We have also evaluated the performance of our fake news detection system on the FakeNewsNet dataset [4], which consists of two datasets with news content, social context, and spatiotemporal information: PolitiFact and GossipCop. In PolitiFact, the political news items are reviewed by journalists and domain experts, who review and provide fact-checking evaluation results to claim news articles as fake or real. GossipCop is a website for fact-checking entertainment stories aggregated from various media outlets. GossipCop provides rating scores on the scale of 0 to 10 to classify a news story as the degree from fake to real. Most news items on

[1] https://competitions.codalab.org/competitions/26655

GossipCop have a rating less than 5, which aligns with its purpose to showcase more fake stories. In order to collect real entertainment news items, the E! Online website [2] is crawled. It is a well known trusted media website for publishing entertainment news items. The articles from E! Online are considered as real news articles, while the ones from GossipCop are considered fake. The original dataset consists of 16817 real news items and 5323 fake news items from GossipCop, and 624 real news items and 432 fake news items from PolitiFact. However, we believe that Twitter's policy to remove certain fake news items from time to time, has prevented us from obtaining the entire dataset. We have been able to procure 15151 real news items and 5323 fake news items from GossipCop, and 610 real news items and 401 fake news items from PolitiFact. We have done a 80-10-10 split of the data into training, validation and test sets.

## 4. Methodology

Our goal in this paper is to design a common fake news classification pipeline framework for both tweets and news items. For this method, we have used some easily available meta-data of tweets or news to boost the performance of the framework. We are also providing the uncertainty value along with predictions to make this framework suitable for active learning, as well as solving domain adaptation related problems. We have used an ensemble of Pre-trained deep learning based language model for text classification and have fed the prediction vector from that ensemble model to another Approximate Bayesian Neural Network based feature fusion network along with some statistical features computed from meta-data of those news or tweets. Initial prediction vector from that fusion model is further tuned with a heuristic-based post processing approach to boost the qualitative performance of the model. Our proposed method consists of six main parts: (a) Text Preprocessing, (b) Tokenization, (c) Backbone Model Architectures, (d) Ensemble, (e) Statistical Feature Fusion Network, (f) Predictive Uncertainty Estimation Model, and (g) Heuristic Post Processing. The overall architecture of our system is shown in Figure-1. A more detailed description is provided in the fol-

[2] https://www.eonline.com/ap

lowing subsections:

### *4.1. Text Preprocessing*

Some social media items, like tweets, are mostly written in colloquial language. Also, they contain various other information like usernames, URLs, emojis, etc. We have filtered out such attributes from the given data as a basic preprocessing step, before feeding it into the ensemble model. For tweets, We have used the tweet-preprocessor[3] library from Python to filter out such noisy information from tweets. For News articles, we have removed any username, URLs from Instagram, Facebook, Twitter etc.

### *4.2. Tokenization*

During tokenization, each sentence is broken down into tokens before being fed into a model. We have used a variety of tokenization approaches[4] depending upon the pre-trained model that we have used, as each model expects tokens to be structured in a particular manner, including the presence of model-specific special tokens. Each model also has its corresponding vocabulary associated with its tokenizer, trained on a large corpus data like GLUE, wikitext-103, CommonCrawl data etc. During training, each model applies the tokenization technique with its corresponding vocabulary on our news data. We have used a combination of BERT [5], XLNet [28], RoBERTa [27], XLM-RoBERTa [29], DeBERTa [30], ERNIE 2.0 [32] and ELECTRA [31] models and have accordingly used the corresponding tokenizers from the base version of their pre-trained models.

### *4.3. Backbone Model Architectures*

We have used a variety of pre-trained language models[5] as backbone models for text classification. For each model, an additional fully connected layer is added to its respective encoder sub-network to obtain prediction probabilities for each class- "real" and "fake" as a prediction vector. We have used transfer learning in our approach in

[3] pypi.org/project/tweet-preprocessor/

[4] huggingface.co/docs/tokenizers/python/latest/

[5] huggingface.co/models

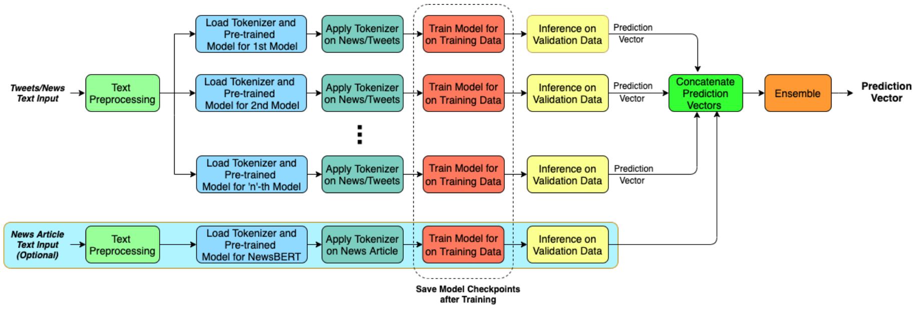
Tweets/News Text Input
Text Preprocessing
Load Tokenizer and Pre-trained Model for 1st Model
Apply Tokenizer on News/Tweets
Train Model for on Training Data
Inference on Validation Data
Prediction Vector
Load Tokenizer and Pre-trained Model for 2nd Model
Apply Tokenizer on News/Tweets
Train Model for on Training Data
Inference on Validation Data
Prediction Vector
Concatenate Prediction Vectors
Ensemble
Prediction Vector
Load Tokenizer and Pre-trained Model for 'n'-th Model
Apply Tokenizer on News/Tweets
Train Model for on Training Data
Inference on Validation Data
Prediction Vector
News Article Text Input (Optional)
Text Preprocessing
Load Tokenizer and Pre-trained Model for NewsBERT
Apply Tokenizer on News Article
Train Model for on Training Data
Inference on Validation Data
Save Model Checkpoints after Training

Figure 1: Fake News Identification Initial Process Block Diagram

this problem. Each model has used some pre-trained model weights as initial weights. Thereafter, it fine-tunes the model weights using the tokenized training data. The same tokenizer is used to tokenize the test data and the fine-tuned model checkpoint is used to obtain predictions during inference.

### *4.4. Ensemble*

In this method, we use the model prediction vectors obtained from inference on the news titles for the different models to obtain our final classification result, i.e. "real" or "fake". Our main motivation behind using an ensemble of various fine-tuned pre-trained language models are to utilize knowledge extracted by the respective models from the corresponding dataset in which it is trained on. However, in the case of FakeNewsNet dataset, we obtain an additional prediction vector using NewsBERT on the news body, that is also appended to the existing feature set. All the features used here are obtained from the raw text data only. To balance an individual model's limitations, an ensemble method can be useful for a collection of similarly well-performing models. We have experimented with two approaches: soft voting and hard voting, that are described in the following figure:

#### *4.4.1. Soft Voting*

In this approach, we calculate a "soft probability score" for each class by averaging out the prediction probabilities of various models for that class. The class that has a higher average probability value is selected as the final prediction class. Probability for "real" class, $P^r(x)$ and probability for "fake" class , $P^f(x)$ for a tweet $x$ is given by,

$$P^r(x) = \sum_{i=1}^{n} \frac{P_i^r(x)}{n} \tag{1}$$

$$P^f(x) = \sum_{i=1}^{n} \frac{P_i^f(x)}{n} \tag{2}$$

where $P_i^r(x)$ and $P_i^f(x)$ are "real" and "fake" probabilities by the $i$-th model and $n$ is the total number of models.

### *4.4.2. Hard Voting :*

In this approach, the predicted class label for a news item is the class label that represents the majority of the class labels predicted by each individual model. In other words, the class with the most number of votes is selected as the final prediction class. Votes for "real" class, $V^r(x)$ and Votes for "fake" class , $V^f(x)$ for a tweet $x$ is given by,

$$V^r(x) = \sum_{i=1}^{n} I(P_i^r(x) \geq P_i^f(x)) \tag{3}$$

$$V^f(x) = \sum_{i=1}^{n} I(P_i^r(x) < P_i^f(x)) \tag{4}$$

where the value of $I(a)$ is 1 if condition $a$ is satisfied and 0 otherwise.

### *4.5. Statistical Feature Fusion Network*

Our basic intuition behind using statistical features is that meta-attributes like username handles, URL domains, news source, news author, etc. are very important aspects of a news item and they can convey reliable information regarding the genuineness of such items. We have tried to incorporate the effect of these attributes along with our original ensemble model predictions by calculating probability vectors corresponding to each of them. We have used information about the frequency of each class for each of these attributes in the training set to compute these vectors. In our experiments, we observed that Soft-voting works better than Hard-voting. Hence our post-processing step takes Soft-voting prediction vectors into account. The steps taken in this approach are described as follows:

- First, we obtain the class-wise probability from the best performing ensemble model. These probability values form two features of our new feature-set.

- We collect all distinct values of a particular attribute from all the news items in our training data, and calculate how many times the ground truth is “real” or “fake” for this attribute.

- We calculate the conditional probability of this particular attribute indicating a real news item, which is represented as follows:

  $$P^{r}(x|attribute_{k}) = \frac{n(A)}{n(A)+n(B)} \tag{5}$$

  where n(A) = number of “real” news items containing the $attribute_k$, n(B) = number of “fake” news items containing the $attribute_k$, and k = 1,2,...,n. In our case, $attribute_1$ = ”URL domain” in case of COVID-19 Fake News dataset and ”news author” in case of FakeNewsNet, and $attribute_2$ = ”username handle” in case of COVID-19 Fake News dataset and ”news source” in case of FakeNewsNet. Similarly, the conditional probability of the particular attribute indicating a fake news item is given by,

  $$P^{f}(x|attribute_{k}) = \frac{n(B)}{n(A)+n(B)} \tag{6}$$

  We obtain a probability vector that forms two additional features of our new dataset.

- Similarly, we collect all other relevant attributes from all the news items in our training data, and calculate how many times the ground truth is “real” or “fake” for each one. This enables us to compute a two-dimensional prediction vector for each new attribute which can be appended to our current feature set. This approach enables us to create two types of feature-sets: one using the DL model predictions and the statistical features obtained from various attributes, and the other using ensembled DL model predictions and the statistical features. When we use the ensembled prediction features, we obtain superior results.

- In case there are multiple attributes of the same type in a sentence, the final probability vectors are obtained by averaging out the vectors of the individual attribute instances.

### *4.6. Predictive Uncertainty Estimation Model*

We have designed an approximate Bayesian neural network as a Statistical Feature Fusion Network (SFFN) for uncertainty estimation of fake news classification. We have applied Monte Carlo Dropout (MCDropout) [33] layer between hidden layers of the feature fusion network for Bayesian interpretation. In the case of Monte Carlo (MC) dropout, the dropout is applied both during training and inference. Hence, the model does not produce the same output each time inference is done on the same data point. Hence, MC dropout enables us to make random predictions that can be interpreted as samples from a probability distribution. As we are using fine tuned Pre-Trained language models for text classification, employing Monte Carlo Dropout (MCDropout) in between the model architecture was not feasible. From this model, we get the prediction vector along with its uncertainty value.

During inference, we ran multiple forward passes through the trained SFFN model with MCDropout, $f_{SFFN}$ for sample $x$ with different dropout masks. For predictive uncertainty estimation, the prediction vector of $N$ inferences with different dropout masks, $d_0, ..., d_N$ are accumulated. Here, $f_{SFFN}^{d_i}$ represents the model with dropout mask, $d_i$. Hence, for dropout masks ($d_i$), we obtain a sample of the possible model outputs, $f_{SFFN}^{d_0}(x), ...., f_{SFFN}^{d_N}(x)$ for that particular sample, $x$. We get an ensemble prediction by calculating the mean ($\mu_x$) and variance ($\sigma_x^2$) of this sample, which would be the mean of the model's posterior distribution for this sample and an approximation of the model's uncertainty.

$$v_p = \mu_x = \frac{1}{N} \sum_{i=0}^{N} f_{SFFN}^{d_i}(x) \tag{7}$$

$$c_u = \sigma_x^2 = \frac{1}{N} \sum_{i=0}^{N} \left[ f_{SFFN}^{d_i}(x) - v_p \right]^2 \tag{8}$$

Here, $v_p$ is the predictive posterior mean and $c_u$ is the model uncertainty.

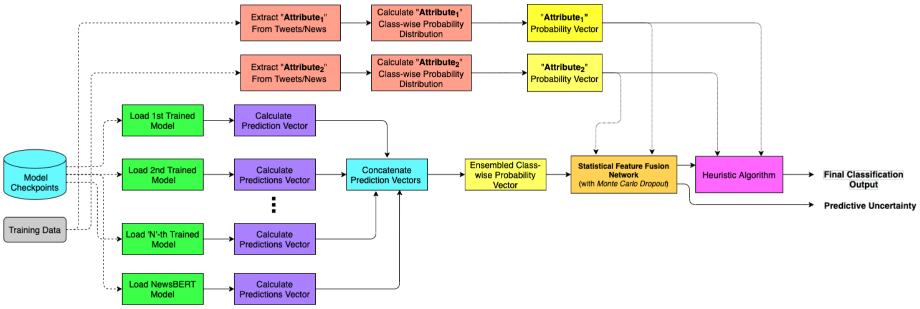
Extract "Attribute$_1$" From Tweets/News
Calculate "Attribute$_1$" Class-wise Probability Distribution
"Attribute$_1$" Probability Vector
Extract "Attribute$_2$" From Tweets/News
Calculate "Attribute$_2$" Class-wise Probability Distribution
"Attribute$_2$" Probability Vector
Model Checkpoints
Training Data
Load 1st Trained Model
Calculate Prediction Vector
Load 2nd Trained Model
Calculate Predictions Vector
Load 'N'-th Trained Model
Calculate Predictions Vector
Load NewsBERT Model
Calculate Predictions Vector
Concatenate Prediction Vectors
Ensembled Class-wise Probability Vector
Statistical Feature Fusion Network (with Monte Carlo Dropout)
Heuristic Algorithm
Final Classification Output
Predictive Uncertainty

Figure 2: Fake News Identification Post Process Block Diagram

### 4.7. Heuristic Post-Processing

In this approach, we have augmented our original framework with a heuristic approach that can take into account the effect of the statistical attributes mentioned in Section 4.5. This approach works well for data having attributes like URL domains, username handles, news source, and the like. Please note, for texts that lack these attributes, we rely only on ensemble model predictions. These attributes allow us to add meaningful features to our current feature set. We obtain new training, validation and test feature-sets obtained using class-wise probability vectors from ensemble model outputs as well as probability values obtained using statistical attributes from the training data. We use a novel heuristic algorithm on this resulting feature set to obtain our final class predictions. The intuition behind using a heuristic approach taking the statistical features into account is that if a particular feature can by itself be a strong predictor for a particular class, and that particular class is predicted whenever the value of a feature is greater than a particular threshold, a significant number of incorrect predictions obtained using the previous steps can be "corrected" back.

Table 1 shows some samples of the conditional probability values of each label class given the URL domain and username handle attributes in COVID-19 Fake News dataset, while Table 2 illustrates similar samples for news source and news author attributes in FakeNewsNet dataset. We have also shown the frequency of those attributes in the training data. The details of the heuristic algorithm is explained in the following pseudocode (Algorithm-1). In our experiment, the value of threshold used is 0.88. The post-processing architecture is shown in Figure-2.

**Algorithm 1** Heuristic Algorithm

```
Result: label ( “real” or “fake”)
if $P^r(x|attribute_1) > threshold$ AND $P^r(x|attribute_1) > P^f(x|attribute_1)$ then
    label = “real”
else if $P^f(x|attribute_1) > threshold$ AND $P^r(x|attribute_1) < P^f(x|attribute_1)$ then
    label = “fake”
else if $P^r(x|attribute_2) > threshold$ AND $P^r(x|attribute_2) > P^f(x|attribute_2)$ then
    label = “real”
else if $P^f(x|attribute_2) > threshold$ AND $P^r(x|attribute_2) < P^f(x|attribute_2)$ then
    label = “fake”
else if $P^r(x) > P^f(x)$ then
    label = “real”
else
    label = “fake”
end if
```

## 5. Experiments

### *5.1. Attribute Extraction*

In order to extract statistical features mentioned in Section 4.5, we have considered the username handles and URL domains from the COVID-19 Fake News dataset and news source and news domain from the FakeNewsNet dataset. Such attributes provide a significant lift in the final classification task since they contribute meaningful information regarding the origin of news items.

Table 1: Few Examples on URL Domain-name and Username attribute distribution data

| Example of URL Domain Name Prob. Dist. | | | |
|---|---|---|---|
| URL Domain Name | $P^r(x\|domain)$ | $P^f(x\|domain)$ | Frequency |
| news.sky | 1.0 | 0.0 | 274 |
| medscape.com | 1.0 | 0.0 | 258 |
| thespoof.com | 0.0 | 1.0 | 253 |
| newsthump.com | 0.0 | 1.0 | 68 |
| theguardian.com | 0.167 | 0.833 | 6 |

| Example of UserName Prob. Dist. | | | |
|---|---|---|---|
| UserName | $P^r(x\|username)$ | $P^f(x\|username)$ | Frequency |
| MoHFW_NDIA | 0.963 | 0.037 | 162 |
| DrTedros | 1.0 | 0.0 | 110 |
| ICMRDELHI | 0.9903 | 0.0097 | 103 |
| PIB_ndia | 1.0 | 0.0 | 83 |
| CDCMMWR | 1.0 | 0.0 | 34 |

Table 2: Few Examples on Author and Source attribute distribution data

| Example of Source Prob. Dist. | | | |
|---|---|---|---|
| Source | $P^r(x\|source)$ | $P^f(x\|source)$ | Frequency |
| people.com | 0.8869 | 0.1131 | 1769 |
| www.dailymail.co.uk | 0.8134 | 0.1866 | 943 |
| www.usmagazine.com | 0.8063 | 0.1937 | 697 |
| hollywoodlife.com | 0.8677 | 0.1323 | 446 |
| radaronline.com | 0.8805 | 0.1195 | 159 |

| Example of Author Prob. Dist. | | | |
|---|---|---|---|
| Author | $P^r(x\|author)$ | $P^f(x\|author)$ | Frequency |
| Amy Mistretta | 0.1175 | 0.8825 | 434 |
| Lindsay Valdez | 0.1175 | 0.8825 | 4340 |
| Daisy Maldonado | 0.0681 | 0.9319 | 411 |
| Dailymail.Com Reporter | 0.8889 | 0.1111 | 243 |
| Dave | 0.8869 | 0.1131 | 168 |

### *5.2. System Description*

We have fine-tuned our pre-trained models using AdamW[36] optimizer and cross-entropy loss after doing label encoding on the target values. We have applied softmax on the logits produced by each model in order to obtain the prediction probability vectors. The experiments were performed on a system with 16GB RAM and 2.2 GHz Quad-Core Intel Core i7 Processor, along with a Tesla T4 GPU, with batch size of 32. The maximum input sequence length was fixed at 128. Initial learning rate was set to 2e-5. The number of epochs varied from 6 to 15 depending on the model.

### *5.3. Evaluation Metrics*

For evaluation of fake news classification, we have used precision, recall, accuracy and f1-score to measure performance of models. We additionally have used two metric, negative log likelihood (NLL) loss and Brier score for evaluating predictive uncertainty of the model. More details on these metrics are following.

**Negative Log Likelihood:** The negative log-likelihood function produces a high value when all the values in a prediction vector are evenly distributed, i.e. when the classification is unclear. It also produces relatively high values in case of wrong classification. However, its value is very small when the output matches the expected value.

$$L_{\log}(y,p) = -(y\log(p) + (1-y)\log(1-p)) \tag{9}$$

where $p$ is prediction vector and $y$ is the true labels.

**Brier Score:** The brier score is a metric that is applied for prediction probabilities. It calculates the mean squared error between the predicted probabilities and actual values. It is quite similar in spirit to the log-loss metric, with a major difference being the fact that it is gentler in penalizing inaccurate predictions.

$$BS = \frac{1}{N}\sum_{i=1}^{N}(t_i - p_i)^2 \tag{10}$$

where, $t_i$ is the predicted probability and $p_i$ is the actual outcome.

### *5.4. Training Strategy*

We have used XLNet, RoBERTa, XLM-RoBERTa, DeBERTa, ELECTRA and ERNIE 2.0 as backbone models in the case of COVID-19 Fake News dataset, while XLNet, RoBERTa, XLM-RoBERTa, DeBERTa and NewsBERT served as backbone models for the FakeNewsNet dataset. The training procedure is carried out by adding a fully-connected dense layer at the end of each of these pre-trained models and fine-tuning it on the corresponding dataset. For NewsBERT model, we have used BERT with *uncased_small_base* pre-trained weights and modified it by stacking few consecutive fully-connected dense layers. We have used higher max sequence length for its embedding layer. We have trained this model with news article text, similarly to other backbone models for longer epochs. In order to train the Statistical Feature Fusion Network models, we used the feature samples created using prediction vector from ensemble of fine-tuned language models as features as well as the statistical features. The FakeNewsNet dataset is highly imbalanced, with 75% of the samples be-longing to the ”real” class and 25% belonging to the ”fake” class. In order to handle this problem of an imbalanced dataset, we have used the Synthetic Minority Oversampling Technique (SMOTE) [37]. We simply oversample the feature samples which belongs to the minority class in order to balance out the imbalance in class distribution, without providing any additional information to the model.

## 6. Results

### *6.1. Performance of Individual Models*

We have used each fine-tuned model individually to perform “real” vs “fake” classification. Quantitative results for COVID-19 Fake News dataset are tabulated in Table-4. We can see that XLM-RoBERTa, RoBERTa, XLNet and ERNIE 2.0 perform really well on the validation set. However, RoBERTa has been able to produce the best classification results when evaluated on the test set. We have also evaluated the performance of XLM-RoBERTa, RoBERTa, XLNet, DeBERTa, and NewsBERT on the FakeNewsNet dataset. NewsBERT has been able to achieve the best results on the validation set, while RoBERTa produces the best results on the test set.

Table 3: Individual model performance on validation and test set of COVID-19 Fake News Dataset

| **Model Name** | **Validation Set** | | | | **Test set** | | | |
|---|---|---|---|---|---|---|---|---|
| | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| XLM-RoBERTa (base) | 0.968 | 0.968 | 0.968 | 0.968 | 0.970 | 0.970 | 0.970 | 0.970 |
| RoBERTa (base) | 0.970 | 0.970 | 0.970 | 0.970 | **0.972** | **0.972** | **0.972** | **0.972** |
| XLNet (base, cased) | 0.975 | 0.975 | 0.975 | 0.975 | 0.966 | 0.966 | 0.966 | 0.966 |
| DeBERTa (base) | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 |
| ELECTRA (base) | 0.948 | 0.948 | 0.948 | 0.948 | 0.953 | 0.953 | 0.953 | 0.953 |
| ERNIE 2.0 | **0.976** | **0.976** | **0.976** | **0.976** | 0.969 | 0.969 | 0.969 | 0.969 |

Table 4: Individual model performance on validation and test set of FakeNewsNet Dataset

| **Model Name** | **Validation Set** | | | | **Test set** | | | |
|---|---|---|---|---|---|---|---|---|
| | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| XLM-RoBERTa (base) | 0.8548 | 0.8548 | 0.8548 | 0.8548 | 0.8631 | 0.8631 | 0.8631 | 0.8631 |
| RoBERTa (base) | 0.8636 | 0.8636 | 0.8636 | 0.8636 | **0.8652** | **0.8652** | **0.8652** | **0.8652** |
| XLNet (base, cased) | 0.8600 | 0.8600 | 0.8600 | 0.8600 | 0.8605 | 0.8605 | 0.8605 | 0.8605 |
| DeBERTa (base) | 0.8657 | 0.8657 | 0.8657 | 0.8657 | 0.8580 | 0.8580 | 0.8580 | 0.8580 |
| NewsBERT | **0.8694** | **0.8694** | **0.8694** | **0.8694** | 0.8626 | 0.8626 | 0.8626 | 0.8626 |

### *6.2. Performance of Ensemble Models*

We tried out different combinations of pre-trained models with both the ensemble techniques: Soft Voting and Hard Voting. Performance for different ensembles on the COVID-19 Fake News dataset are shown in Table-5 and 6. From the results, we can infer that the ensemble models significantly outperform the individual models, and Soft-voting ensemble method performed better overall than Hard-voting ensemble method. Hard-voting Ensemble model consisting of RoBERTa, XLM-RoBERTa, XLNet, ERNIE 2.0 and DeBERTa models performed the best among other hard voting ensembles on both validation and test set. Among the Soft Voting Ensembles, the ensemble consisting of RoBERTa, XLM-RoBERTa, XLNet, ERNIE 2.0 and Electra models achieved best accuracy overall on the validation set and a combination of XLNet, RoBERTa, XLM-RoBERTa and DeBERTa models produces the best classification result overall on the test set.

Table 5: Performance of Soft Voting for different ensemble models on validation and test set of COVID-19 Fake News Dataset

| **Ensemble Model Combination** | **Validation Set** | | | | **Test set** | | | |
|---|---|---|---|---|---|---|---|---|
| | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| RoBERTa+XLM-RoBERTa +XLNet | 0.9827 | 0.9827 | 0.9827 | 0.9827 | 0.9808 | 0.9808 | 0.9808 | 0.9808 |
| RoBERTa+XLM-RoBERTa +XLNet+DeBERT | 0.9832 | 0.9832 | 0.9832 | 0.9832 | **0.9831** | **0.9831** | **0.9831** | **0.9831** |
| RoBERTa+XLM-RoBERTa +XLNet+ERNIE 2.0 +DeBERTa | 0.9836 | 0.9836 | 0.9836 | 0.9836 | 0.9822 | 0.9822 | 0.9822 | 0.9822 |
| RoBERTa+XLM-RoBERTa +XLNet+ERNIE 2.0 +Electra | **0.9841** | **0.9841** | **0.9841** | **0.9841** | 0.9808 | 0.9808 | 0.9808 | 0.9808 |

Table 6: Performance of Hard Voting for different ensemble models on validation and test set of COVID-19 Fake News Dataset

| **Ensemble Model Combination** | **Validation Set** | | | | **Test set** | | | |
|---|---|---|---|---|---|---|---|---|
| | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| RoBERTa+XLM-RoBERTa +XLNet | 0.9818 | 0.9818 | 0.9818 | 0.9818 | 0.9804 | 0.9804 | 0.9804 | 0.9804 |
| RoBERTa+XLM-RoBERTa +XLNet+DeBERT | 0.9748 | 0.9748 | 0.9748 | 0.9748 | 0.9743 | 0.9743 | 0.9743 | 0.9743 |
| RoBERTa+XLM-RoBERTa +XLNet+ERNIE 2.0 +DeBERTa | **0.9832** | **0.9832** | **0.9832** | **0.9832** | **0.9813** | **0.9813** | **0.9813** | **0.9813** |
| RoBERTa+XLM-RoBERTa +XLNet+ERNIE 2.0 +Electra | 0.9822 | 0.9822 | 0.9822 | 0.9822 | 0.9766 | 0.9766 | 0.9766 | 0.9766 |

We have also evaluated our best ensemble model combination from the above approach, consisting of XLM-RoBERTa, RoBERTa, XLNet and DeBERTa, as well as a combination of the above models along with NewsBERT, on the FakeNewsNet dataset in Table 7 and 8. We have tried out both soft-voting and hard-voting ensembling techniques, and have observed that the addition of the features obtained from NewsBERT prediction vectors provides a boost to the final F1-score. Also, soft-voting performs slightly better than hard-voting on the test set.

Table 7: Performance of Soft Voting on validation and test set of FakeNewsNet dataset

| **Ensemble Model** | **Validation Set** | | | | **Test set** | | | |
|---|---|---|---|---|---|---|---|---|
| **Combination** | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| RoBERTa+XLM-RoBERTa +XLNet+DeBERTa | 0.8699 | 0.8699 | 0.8699 | 0.8699 | 0.8718 | 0.8718 | 0.8718 | 0.8718 |
| RoBERTa+XLM-RoBERTa +XLNet+DeBERTa +NewsBERT | **0.8783** | **0.8783** | **0.8783** | **0.8783** | **0.8765** | **0.8765** | **0.8765** | **0.8765** |

Table 8: Performance of Hard Voting on validation and test set of FakeNewsNet dataset

| **Ensemble Model** | **Validation Set** | | | | **Test set** | | | |
|---|---|---|---|---|---|---|---|---|
| **Combination** | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| RoBERTa+XLM-RoBERTa +XLNet+DeBERT | 0.8662 | 0.8662 | 0.8662 | 0.8662 | 0.8672 | 0.8672 | 0.8672 | 0.8672 |
| RoBERTa+XLM-RoBERTa +XLNet+DeBERT+NewsBERT | **0.8783** | **0.8783** | **0.8783** | **0.8783** | **0.8749** | **0.8749** | **0.8749** | **0.8749** |

### *6.3. Performance of Statistical Feature Fusion Network and Comparisons*

In this section, we have qualitatively measured the performance of our Statistical Feature Fusion Network (SFFN) with MCDropout with respect to SFFN. We have also compared the performance of various classical models like Logistic Regression, SVM, Decision Tree, Random Forest. As a feature input to SFFN, we have studied two different feature input types. The first type of feature set is created using the individual prediction vector from the various language models of the ensemble (soft-voting), with the conditional probability values of various attributes as statistical features and the second type of feature set is created using the prediction vector from the ensemble (soft-voting) of the language models with the same conditional probability features as the previous one.

In Table 9, we have experimented with some classical machine learning models on a new feature set created using the individual predictions from the language models of the best ensemble mentioned in Table 5, and the conditional probability values of URL domains and username handles for the COVID-19 Fake News dataset. In Table 10, we have tabulated the results of the same experiment, with the best ensemble from the Table 7, on the FakeNewsNet dataset using the conditional probability values of news author and news source.

Table 9: Individual DL model predictions + Statistical features on validation and test set of COVID-19 Fake News dataset

| **Model** | **Validation Set** | | | | **Test set** | | | |
|---|---|---|---|---|---|---|---|---|
| | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| Logistic Regression | 0.9841 | 0.9841 | 0.9841 | 0.9841 | 0.9832 | 0.9832 | 0.9832 | 0.9832 |
| SVM | 0.9825 | 0.9825 | 0.9825 | 0.9825 | 0.9827 | 0.9827 | 0.9827 | 0.9827 |
| Decision Tree | 0.9804 | 0.9804 | 0.9804 | 0.9804 | 0.9743 | 0.9743 | 0.9743 | 0.9743 |
| Random Forest | 0.9808 | 0.9808 | 0.9808 | 0.9808 | 0.9804 | 0.9804 | 0.9804 | 0.9804 |
| SFFN | 0.9841 | 0.9841 | 0.9841 | 0.9841 | 0.9822 | 0.9822 | 0.9822 | 0.9822 |
| SFFN with MCDropout | **0.9846** | **0.9846** | **0.9846** | **0.9846** | **0.9836** | **0.9836** | **0.9836** | **0.9836** |

Table 10: Individual DL model predictions + Statistical features on validation and test set of FakeNewsNet dataset using news title and article text

| **Model** | **Validation Set** | | | | **Test set** | | | |
|---|---|---|---|---|---|---|---|---|
| | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| Logistic Regression | 0.8714 | 0.8714 | 0.8714 | 0.8714 | 0.8644 | 0.8644 | 0.8644 | 0.8644 |
| SVM | 0.8772 | 0.8772 | 0.8772 | 0.8772 | 0.8678 | 0.8678 | 0.8678 | 0.8678 |
| Decision Tree | 0.8647 | 0.8647 | 0.8647 | 0.8647 | 0.8600 | 0.8600 | 0.8600 | 0.8600 |
| Random Forest | 0.8662 | 0.8662 | 0.8662 | 0.8662 | 0.8641 | 0.8641 | 0.8641 | 0.8641 |
| SFFN | 0.8788 | 0.8788 | 0.8788 | 0.8788 | 0.8672 | 0.8672 | 0.8672 | 0.8672 |
| SFFN with MCDropout | **0.8814** | **0.8814** | **0.8814** | **0.8814** | **0.8708** | **0.8708** | **0.8708** | **0.8708** |

Then, we have evaluated the performance of the same models on another type of feature set for COVID-19 Fake News and FakeNewsNet datasets respectively in Table 11 and 12. From these studies, we can conclude that SFFN with MCDropout got better accuracy than the other classical models and the feature set using average prediction

Table 11: DL soft-voting predictions + Statistical features on validation and test set of COVID-19 Fake News dataset

| Model | Validation Set | | | | Test set | | | |
|---|---|---|---|---|---|---|---|---|
| | Accuracy | Precision | Recall | F1 Score | Accuracy | Precision | Recall | F1 Score |
| Logistic Regression | 0.9836 | 0.9836 | 0.9836 | 0.9836 | 0.9831 | 0.9831 | 0.9831 | 0.9831 |
| SVM | 0.9831 | 0.9831 | 0.9831 | 0.9831 | 0.9827 | 0.9827 | 0.9827 | 0.9827 |
| Decision Tree | 0.9832 | 0.9832 | 0.9832 | 0.9832 | 0.9827 | 0.9827 | 0.9827 | 0.9827 |
| Random Forest | 0.9832 | 0.9832 | 0.9832 | 0.9832 | 0.9827 | 0.9827 | 0.9827 | 0.9827 |
| SFFN | **0.9841** | **0.9841** | **0.9841** | **0.9841** | 0.9832 | 0.9832 | 0.9832 | 0.9832 |
| SFFN with MCDropout | **0.9841** | **0.9841** | **0.9841** | **0.9841** | **0.9836** | **0.9836** | **0.9836** | **0.9836** |

Table 12: Performance of Soft Voting on validation and test set of FakeNewsNet dataset using title and document

| Model | Validation Set | | | | Test set | | | |
|---|---|---|---|---|---|---|---|---|
| | Accuracy | Precision | Recall | F1 Score | Accuracy | Precision | Recall | F1 Score |
| Logistic Regression | 0.8929 | 0.8929 | 0.8929 | 0.8929 | 0.8914 | 0.8914 | 0.8914 | 0.8914 |
| SVM | 0.8966 | 0.8966 | 0.8966 | 0.8966 | 0.8950 | 0.8950 | 0.8950 | 0.8950 |
| Decision Tree | 0.8939 | 0.8939 | 0.8939 | 0.8939 | 0.8976 | 0.8976 | 0.8976 | 0.8976 |
| Random Forest | 0.8939 | 0.8939 | 0.8939 | 0.8939 | 0.8998 | 0.8998 | 0.8998 | 0.8998 |
| SFFN | 0.9033 | 0.9033 | 0.9033 | 0.9033 | 0.9053 | 0.9053 | 0.9053 | 0.9053 |
| SFFN with MCDropout | **0.9039** | **0.9039** | **0.9039** | **0.9039** | **0.9053** | **0.9053** | **0.9053** | **0.9053** |

vector from best performing ensemble model with statistical feature is more significant than the other one for best performance.

### *6.4. Ablation Study on Heuristic Post-processing*

We augmented our Fake News Detection System with an additional heuristic algorithm to boost the accuracy of the model further. We have used the best performing ensemble model consisting of RoBERTa, XLM-RoBERTa, XLNet and DeBERTa for this approach. We have performed an ablation study by assigning various levels of priority to each of the features (for example, username>domain or author>source) and then checking which class's probability value for that feature is maximum for a particular news item, so that we can assign the corresponding "real" or "fake" class label to that particular item. For example, in one iteration, we have given URL domains a higher priority than username handles to select the label class. Results for different priority and feature set is shown in Table 13 and 14.

Another important parameter that we have introduced for our experiment is a threshold on the class-wise probability values for the features. For example, if the probability that a particular attribute that exists in a news item belongs to "real" class is greater than that of it belonging to "fake" class, and the probability of it belonging to the "real" class is greater than a specific threshold, we assign a "real" label to the item. The value of this threshold is a hyperparameter that has been tuned based on the classification accuracy on the validation set. We have summarized the results from our study with and without the threshold parameter in Tables 13 and 14.

As we can observe from the results, the URL domain plays a significant role for ensuring a better classification result when the threshold parameter is taken into account in case of COVID-19 Fake news dataset, while the news author plays a significant role in an analogous scenario in case of the FakeNewsNet dataset. The best results are obtained when we consider the threshold parameter and both the username and domain attributes in case of COVID-19 Fake News dataset, and the news author and news source along with the threshold in the case of FakeNewsNet dataset, with higher importances given to the username and news author. We have also performed a similar ablation study on the FakeNewsNet dataset using the author and source attributes.

Table 13: Ablation Study on Heuristic algorithm on COVID-19 Fake News Dataset

| **Combination of Attributes** | **with Threshold** | | **without Threshold** | |
|---|---|---|---|---|
| **(in descending order of Attribute Priority)** | **F1 Score on Validation Set** | **F1 Score on Test Set** | **F1 Score on Validation Set** | **F1 Score on Test Set** |
| {*username, ensemble model pred* } | 0.9831 | 0.9836 | **0.9822** | **0.9804** |
| {*domain, ensemble model pred* } | **0.9917** | 0.9878 | 0.9635 | 0.9523 |
| {*domain, username, ensemble model pred* } | 0.9911 | 0.9878 | 0.9635 | 0.9519 |
| {*username, domain, ensemble model pred* } | 0.9906 | **0.9883** | 0.9645 | 0.9528 |

### *6.5. Performance of Uncertainty Models*

We evaluate the performance of the predictive uncertainty of Statistical Feature Fusion Network (SFFN) on two mentioned datasets. We have used two proper scoring rules, Brier score and the negative log-likelihood loss where a lower score corresponds to a better performance in predicting uncertainty value. The scores and respective

Table 14: Ablation Study of Heuristic algorithm on FakeNewsNet Dataset

| **Combination of Attributes** | **with Threshold** | | **without Threshold** | |
|---|---|---|---|---|
| **(in descending order of Attribute Priority)** | **F1 Score on Validation Set** | **F1 Score on Test Set** | **F1 Score on Validation Set** | **F1 Score on Test Set** |
| {*news author, ensemble model pred* } | 0.8824 | 0.8883 | **0.8626** | **0.8677** |
| {*news source, ensemble model pred* } | 0.8887 | 0.8909 | 0.8510 | 0.8533 |
| {*news source, news author, ensemble model pred* } | **0.8939** | **0.9007** | 0.8516 | 0.8543 |
| {*news author, news source, ensemble model pred* } | **0.8939** | 0.9001 | 0.8510 | 0.8595 |

model accuracy are given in Table 15 below. The results clearly demonstrate that the SFFN model with Monte Carlo Dropout (MCDropout) leads to both improved predictive uncertainty and accuracy of the respective model for their corresponding datasets.

Table 15: Performance of Uncertainty Models (SFFN with MCDropout)

| **Dataset** | **Input Feature** | **Model** | **Validation Set** | | | **Test set** | | |
|---|---|---|---|---|---|---|---|---|
| | | | **F1 Score** | **NLL Loss** | **Brier Score** | **F1 Score** | **NLL Loss˜** | **Brier Score˜** |
| COVID-19 FakeNews | Individual DL models with Statistical feature | Vanila SFFN | 0.9841 | 0.1444 | 0.0158 | 0.9822 | 0.1544 | 0.0160 |
| | | SFFN with MCDropout | **0.9846** | **0.1792** | **0.0144** | **0.9836** | **0.2250** | **0.0156** |
| | DL Ensemble model with Statistical feature˜ | Vanila SFFN | **0.9841** | 0.0910 | 0.0150 | 0.9832 | 0.0949 | 0.0152 |
| | | SFFN˜with MCDropout | **0.9841** | **0.0660** | **0.0142** | **0.9836** | **0.0678** | **0.0144** |
| FakeNewsNet | Individual DL models with Statistical feature˜ | Vanila SFFN | 0.8788 | 0.5457 | 0.1108 | 0.8672 | 0.5649 | 0.1183 |
| | | SFFN˜with MCDropout | **0.8814** | **0.5320** | **0.1077** | **0.8708** | **0.5377** | **0.1139** |
| | DL Ensemble model with Statistical feature˜˜ | Vanila SFFN | 0.9033 | 0.3581 | 0.0793 | **0.9053** | 0.3350 | 0.0744 |
| | | SFFN with MCDropout | **0.9039** | **0.2631** | **0.0754** | **0.9053** | **0.2444** | **0.0710** |

### *6.6. Performance of Final Proposed Model and Comparisons*

We qualitatively evaluate the performance of the proposed method on the two mentioned datasets. In Table-16, we have shown that with the addition of feature fusion network, the performance of the framework has improved compared to other models and achieved state of the art results on both datasets. In our earlier work [38], we had shown that the heuristic post-processing approach improves the classification accuracy on the test set significantly. However, the incorporation of uncertainty estimation improves the model performance even more.

We have shown the comparison of the results on the test set obtained by our model before and after applying the post-processing technique against the top 3 teams in the leaderboard for the COVID-19 Fake News dataset, in Table 17.

Table 16: Performance of Ensemble SFFN (with MCDropout) Model with Heuristic Post-Processing on Different Datasets

| **Dataset** | **Model** | **Combination of Attributes˜**<br>**(in descending order of Attribute Priority)** | **F1 Score on**<br>**Validation Set** | **F1 Score on**<br>**Test Set** |
|---|---|---|---|---|
| COVID-19 FakeNews | Soft Voted Ensemble Model | { *username, domain, ensemble model pred* } | 0.9906 | 0.9883 |
| | Ensemble SFFN (with MCDropout) | { *username, domain, ensemble_SFFN_mcdropout pred* } | **0.9911** | **0.9892** |
| FakeNewsNet | Soft Voted Ensemble Model | { *news source, news author, ensemble model pred* } | 0.8939 | 0.9007 |
| | Ensemble SFFN (with MCDropout) | { *news source, news author, ensemble_SFFN_mcdropout pred* } | **0.8976** | **0.9073** |

Table 17: Performance comparison on test set on COVID-19 Fake News Dataset

| **Method** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
|---|---|---|---|---|
| Team g2tmn (*Rank 1* [39]) | 0.9869 | 0.9869 | 0.9869 | 0.9869 |
| Team saradhix (*Rank 2* [39]) | 0.9864 | 0.9865 | 0.9864 | 0.9864 |
| Team xiangyangli (*Rank 3* [39]) | 0.9860 | 0.9860 | 0.9860 | 0.9860 |
| Ensemble Model + Heuristic Post-Processing (Das et al. [38]) | 0.9883 | 0.9883 | 0.9883 | 0.9883 |
| SFFN (with MCDropout) + Heuristic Post-Processing | **0.9892** | **0.9892** | **0.9892** | **0.9892** |

## 7. Conclusion

In this paper, we have proposed a robust framework for identification of fake news items, which can go a long way in eliminating the spread of misinformation on sensitive topics. In our initial approach, we have tried out various pre-trained language models. Our results have significantly improved when we implemented an ensemble mechanism with Soft-voting by using the prediction vectors from various combinations of these models. Furthermore, we have been able to augment our system with a statistical feature fusion network and a novel heuristics-based post-processing algorithm by incorporation of statistical features, that has drastically improved the fake tweet detection accuracy. Our novel heuristic approach shows that meta-attributes like username handle, URL domain, news author, news source, etc. form very important features of news and analyzing them accurately can go a long way in creating a robust framework for fake news detection. We have also quantified the model uncertainty in the task of fake news detection by applying Monte Carlo dropout as a Bayesian approximation in the statistical feature fusion network. With empirical experiments, we have shown the overall performance increase after including uncertainty in the model.

Finally, we would like to pursue more research into how to extend our framework for an active learning based approach by utilizing uncertainty values, and incorporate other combinations of meta-attributes in our model perform on the given datasets. It would be really interesting to evaluate how our system performs on other generic Fake News datasets and also if different values of the threshold parameter for our post-processing system would impact its overall performance.

## References

[1] J. A. Tucker, A. Guess, P. Barberá, C. Vaccari, A. Siegel, S. Sanovich, D. Stukal, B. Nyhan, Social media, political polarization, and political disinformation: A review of the scientific literature, Political polarization, and political disinformation: a review of the scientific literature (March 19, 2018).

[2] D. P. Calvillo, B. J. Ross, R. J. Garcia, T. J. Smelter, A. M. Rutchick, Political ideology predicts perceptions of the threat of covid-19 (and susceptibility to fake news about it), Social Psychological and Personality Science 11 (8) (2020) 1119–1128.

[3] F. Monti, F. Frasca, D. Eynard, D. Mannion, M. M. Bronstein, Fake news detection on social media using geometric deep learning, arXiv preprint arXiv:1902.06673.

[4] K. Shu, D. Mahudeswaran, S. Wang, D. Lee, H. Liu, Fakenewsnet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media, Big Data 8 (3) (2020) 171–188.

[5] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805.

[6] J. C. Reis, A. Correia, F. Murai, A. Veloso, F. Benevenuto, Supervised learning for fake news detection, IEEE Intelligent Systems 34 (2) (2019) 76–81.

[7] L. Kozma, k nearest neighbors algorithm (knn), Helsinki University of Technology.

[8] L. Breiman, Random forests, Machine learning 45 (1) (2001) 5–32.

[9] I. Rish, et al., An empirical study of the naive bayes classifier, in: IJCAI 2001 workshop on empirical methods in artificial intelligence, Vol. 3, 2001, pp. 41–46.

[10] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, B. Scholkopf, Support vector machines, IEEE Intelligent Systems and their applications 13 (4) (1998) 18–28.

[11] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, 2016, pp. 785–794.

[12] K. Shu, S. Wang, H. Liu, Beyond news contents: The role of social context for fake news detection, in: Proceedings of the twelfth ACM international conference on web search and data mining, 2019, pp. 312–320.

[13] H. Karimi, P. Roy, S. Saba-Sadiya, J. Tang, Multi-source multi-class fake news detection, in: Proceedings of the 27th international conference on computational linguistics, 2018, pp. 1546–1557.

[14] J. Zhang, B. Dong, S. Y. Philip, Fakedetector: Effective fake news detection with deep diffusive neural network, in: 2020 IEEE 36th International Conference on Data Engineering (ICDE), IEEE, 2020, pp. 1826–1829.

[15] N. Ruchansky, S. Seo, Y. Liu, Csi: A hybrid deep model for fake news detection, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, pp. 797–806.

[16] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: International conference on machine learning, PMLR, 2014, pp. 1188–1196.

[17] J. C. Reis, A. Correia, F. Murai, A. Veloso, F. Benevenuto, Supervised learning for fake news detection, IEEE Intelligent Systems 34 (2) (2019) 76–81.

[18] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, Y. Choi, Defending against neural fake news, arXiv preprint arXiv:1905.12616.

[19] Y. Bang, E. Ishii, S. Cahyawijaya, Z. Ji, P. Fung, Model generalization on covid-19 fake news detection, arXiv preprint arXiv:2101.03841.

[20] K. Shu, L. Cui, S. Wang, D. Lee, H. Liu, defend: Explainable fake news detection, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 395–405.

[21] T. Felber, Constraint 2021: Machine learning models for covid-19 fake news detection shared task, arXiv preprint arXiv:2101.03717.

[22] E. Shushkevich, J. Cardiff, Tudublin team at constraint@ aaai2021–covid19 fake news detection, arXiv preprint arXiv:2101.05701.

[23] O. Sharif, E. Hossain, M. M. Hoque, Combating hostility: Covid-19 fake news and hostile post detection in social media, arXiv preprint arXiv:2101.03291.

[24] A. Gautam, S. Masud, et al., Fake news detection system using xlnet model with topic distributions: Constraint@ aaai2021 shared task, arXiv preprint arXiv:2101.11425.

[25] X. Li, Y. Xia, X. Long, Z. Li, S. Li, Exploring text-transformers in aaai 2021 shared task: Covid-19 fake news detection in english, arXiv preprint arXiv:2101.02359.

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, arXiv preprint arXiv:1706.03762.

[27] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: A robustly optimized bert pretraining approach, arXiv preprint arXiv:1907.11692.

[28] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, Q. V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, arXiv preprint arXiv:1906.08237.

[29] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, V. Stoyanov, Unsupervised cross-lingual representation learning at scale, arXiv preprint arXiv:1911.02116.

[30] P. He, X. Liu, J. Gao, W. Chen, Deberta: Decoding-enhanced bert with disentangled attention, arXiv preprint arXiv:2006.03654.

[31] K. Clark, M.-T. Luong, Q. V. Le, C. D. Manning, Electra: Pre-training text encoders as discriminators rather than generators, arXiv preprint arXiv:2003.10555.

[32] Y. Sun, S. Wang, Y. Li, S. Feng, H. Tian, H. Wu, H. Wang, Ernie 2.0: A continual pre-training framework for language understanding, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 8968–8975.

[33] Y. Gal, Z. Ghahramani, Dropout as a bayesian approximation: Representing model uncertainty in deep learning, in: international conference on machine learning, PMLR, 2016, pp. 1050–1059.

[34] B. Lakshminarayanan, A. Pritzel, C. Blundell, Simple and scalable predictive uncertainty estimation using deep ensembles, arXiv preprint arXiv:1612.01474.

[35] P. Patwa, S. Sharma, S. PYKL, V. Guptha, G. Kumari, M. S. Akhtar, A. Ekbal, A. Das, T. Chakraborty, Fighting an infodemic: Covid-19 fake news dataset, arXiv preprint arXiv:2011.03327.

[36] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, arXiv preprint arXiv:1711.05101.

[37] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, Smote: Synthetic minority over-sampling technique, Journal of Artificial Intelligence Research 16 (2002) 321–357. `doi:10.1613/jair.953`.
URL `http://dx.doi.org/10.1613/jair.953`

[38] S. D. Das, A. Basak, S. Dutta, A heuristic-driven ensemble framework for covid-19 fake news detection, arXiv preprint arXiv:2101.03545.

[39] P. Patwa, M. Bhardwaj, V. Guptha, G. Kumari, S. Sharma, S. PYKL, A. Das, A. Ekbal, S. Akhtar, T. Chakraborty, Overview of constraint 2021 shared tasks: Detecting english covid-19 fake news and hindi hostile posts, in: Proceedings of the First Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situation (CONSTRAINT), Springer, 2021.