Performance Evaluation on Vector-Model Combinations

Increment 1

GitHub link: *https://github.com/trela47/NLP_Project*

Team Members:

Blessy Kuriakose - 11230255

Saisri Teja Pepeti - 11555656

Saikiran Yedulla - 11518301

Yamuna Bollepalli - 11552426

Goals and Objectives

Motivation:

We aim to explore the effectiveness of differing vectorization-and-model combinations in the text-based prediction task of spam detection. By this project we will be able to accomplish which vectorization will work with best model accuracy and least time complexity by comparing to each other.

Significance:

There are multiple vectorization techniques available but there is no proper model available to distinguish which technique and model suits the best for a specific dataset. In this project we will be able to provide the accurate model for a spam detection dataset along with the time complexity. So, this helps the organizations and individuals time as they are already aware of the model best suit for a particular dataset.

Moreover, we improved the accuracy of Support Vector Machine classification and Naive Bayes along with the least time complexity. We will also work with Random Forest classification and Decision Tree classification and assume that we brought good variation in the accuracy compared to the existing by well pretraining. We achieved 98.8 % accuracy from one of the models.

Objectives:

- Cleaned the raw text.
- Got the language-based vectors from the cleaned text.
- Prepared models for training and testing and tracked the metrics (confusion matrix)
- Evaluated the metrics to conclude the best combination.

Features:

We will be using the spam detection dataset from Kaggle; techniques will be applied on the raw text data to create the input features. After text cleaning, we apply TFIDF vectorization in which text is converted into a vector form. The ideas of Term Frequency (TF) and Inverse Document Frequency (IDF) are combined to form TFIDF. After completion, these vectors will be used to evaluate the model performance of multiple classification models like Random Forest, SVM, Decision tree, Naive Bayes based on model metrics.

Related Work:

Some papers use a straightforward absence/presence check on keywords to detect whether an email is spam or not, however we find that these do not offer the best in terms of accuracy for all emails. This is especially the case when some keywords, though often found in spam, could still appear in legit (non-spam) emails. There are also methods that are too direct with tokenization and which produce an accuracy that reaches only 89%. From the existing models we tried to improve the accuracy and also implement multiple models including Decision tree, SVM, Random Forest and will be displaying which model provided the best accuracy with minimum time complexity.

Dataset:

We are using a dataset of emails which contain a number of emails in text format and they are coupled in a single folder. Other than the normal alphabet found in words, the text also includes numbers, punctuations, and special characters. When we converted those dataset into an excel sheet we have absorbed nearly 940 emails. It comes pre-split into training and testing sets and contains one line containing the subject heading (not used in our implementation) and another with the body of the email.

Features:

Each technique is a feature of your project. What type of features you are using from your dataset.

The raw features of our dataset are the text itself of each individual email. Techniques such as:

A bit of preprocessing including lowering the case, processing with TextBlob, and lemmatization are completed, so that the more insignificant differences that might separate two like-words are bypassed.

Tf-idf and word2vec are applied to gather insight into the underlying patterns of these raw features.

Tf-idf (Term frequency- Inverse document frequency) vectorized form of the emails to build and test several models.

Tf-idf makes use of frequency as well as the inverse document frequency to create a function based probabilistic model that uses the intricacies of word patterns and usage context in a mathematical formulation to make predictions on category, in this case spam vs ham (not-spam).

By using transform function to train the data and sending this trained data to the classification models as the performance of machine learning algorithms is improved by scaling the numerical (vector) input variables to a standard range. The two most common methods are normalization and standardization.

Performance matrices of the model which is how model is trained and performing up to the bench mar k of expectation on accuracy:

Confusion matrix

Accuracy

Precision

Recall

f-Score

AUC

Naïve Bayes: Naive Bayes algorithm is for prediction and Tf-Idf Vectorizer for feature extraction. A straightforward and probabilistic conventional machine learning algorithm is Naive Bayes

Decision Tree: The dataset features will be divided using a decision tree technique and a cost function. It used to delete branches that might use unnecessary features.

SVM: This algorithm is extremely effective for recognizing patterns and categorizing it to a certain class or group. The SVM model achieved more accuracy than other Decision tree, Naïve Bayes and Random Forest classification techniques.

Random Forest: This is a meta estimator that employs averaging to increase predicted accuracy and reduce overfitting after fitting numerous decision tree classifiers to distinct dataset subsamples.

Where the LSTM and BiLSTM considered as a base model for the data preprocessing for Word2Vectorisation and FastText.

Analysis of implementation:

With the taken data set we implemented ML models and NLP techniques for measuring the accuracies by comparing to each model with respect to accuracies so, here the main purpose of implementing these models irrespective of the available models in the internet is that those internet models are well trained over millions of records so they could achieve good accuracy eventually, likewise that the data set plays the important role where, preprocessing the data for each model in specific format is challenging as per the data selection and then accuracy increases using the same base model itself.

Implementation:

- We are importing all the necessary libraries like Pandas, Matplotlib, Numpy, Sklearn, NLTK, TextBlob then from NLTK we are downloading PUNKT and wordnet.
- For validation we are importing classification report, F1 score, accuracy score, confusion matrix and roc_auc_score, precision score from Sklearn metrics, using all these metrics we will validate our classification models to check the better performing model
- Post loading data set we have performed to bring up the header information which has the labels of ham and spam with label numbers for the given text
- We performed data preprocessing immediately, under preprocessing data cleaning is one of the sections where we remove all the unnecessary data which affects our models performance.
- For better classification we have divided the spam and ham data and then by grouping them the cleaned data

- We are assigning train and test sets to our text and ‘label’(spam or ham) data with text size 0.3(30%) along with random state 42
- Then performing lemmatization for all the words
- By using Textblob here which has predefined rules which helped to calculate the polarity of sentences
- Tf-idf and word2vec are applied to gather insight into the underlying patterns of these raw features. Tf-idf (Term frequency- Inverse document frequency) vectorized form of the emails to build and test several models.
- Tf-idf makes use of frequency as well as the inverse document frequency to create a function based probabilistic model that uses the intricacies of word patterns and usage context in a mathematical formulation to make predictions on category, in this case spam vs ham (not-spam).
- Preparing the text data by locating the line containing the body of the email, splitting each line into a list of words, removing “stopwords”, and singling out only words containing alphabetic characters for further processing.
- Creating a word dictionary and making a Bag-of-words from the 2500 most common words (of those prepared in step 1) in the training set.
- After implementing BOW and TF-Idf vectorization transformer on our data we are training our models
- Feature extraction process with Word2vec
- Training the model using various supervised algorithms(Naive Bayes, Random Forest, Support Vector Machine, Decision tree.)
- Selecting the model that gives better accuracy and using it to predict the test data (predicting the emails as spam or not spam)
- For assessment we have created a function which includes all the validation metrics to evaluate our test.
- After validation on all our models we achieved also plotted confusion matrix to find the variation in comparison of expected and predicted

  Naive Bayes - 86%

  Decision Tree - 94%

Random Forest - 95%

SVM - 98.88%

- Then by performing the LSTM which helps a multiple word string in LSTM to determine the class to which it belongs. When working with natural language processing, this is quite beneficial. The model will be able to determine the true meaning in the input string and will provide the most correct output class if the right layers of embedding and encoding are used in the LSTM. The idea of utilizing LSTM for classifying ham,spam texts which is further explained in the code that follows.
- Also, the embedding layer maps the texts to vectors, and the Bi-LSTM layer extracts the vector's features to produce the final sequence. The final sequence will then be categorized using a softmax function in the fully linked layer.
- By taking training size as 20% of test size in LSTM and then we perform numbers, white spaces, punctuations, special characters, hyperlinks and replacing newlines for cleaning the text in sentences.
- After cleaning we assign the data to training and testing which is email_train and email_test. Using label encoder we are converting the text data into numerical values to train and test our model and it has the maximum features of 50000 and maximum length of 2000 .
- Then tokenizing the text then padding them into sequences using pad_sequence function
- From Keras we import Dense, Input, LSTM, Embedding, Dropout, Activation and Bidirectional for Bi-LSTM.

  Embedding - For embedding all the words after training

  Activation function - Relu and Sigmoid for detailed scaling of each sentence which will help us to acquire more accuracy , as these activation functions help our LSTM model to check accurately each point.

  Binary entropy for loss function validation and adam for optimisor and metrics accuracy
- We are fitting our model with certain features with batch size of 32 and epoch size of 20 for validation our data we are using X_text features and test_y
- For detailed understanding we visualized the graph for models accuracy wrt accuracy, for the test we got 98.8%.
- We plotted a confusion matrix wrt true labels and predicted labels(Spam and not spam).

- From gensim models we import Word2Vec function
- We label the data separately for spam and given text
- Using label encoder we are transforming our data, we are using porterstemmer to stem all the words and compare with the words existing in the ham
- As usually splitting the data and assigning training and testing we performed a sequential method to arrange all the trained data, adding embedding layers to the words and also added LSTM for the given dropout and recurring dropout function by using sigmoid activation function
- Under embedding layer we got 3417300 parameters with shape of 300 and 100
- So we got 97% for Word2Vec finally and plotted a confusion matrix for the same.

Preliminary results:

Existing System	Our System
In the existing system they have used only one vectorisation model.	In our system we have implemented couple vectorization methods like TF-IDF and Word2vec
In the existing system there is no comparison between the performance of multiple machine learning methods.	In our system we have compared the performance of multiple machine learning methods like Naive Bayes, Random Forest, Support Vector Machine, Decision tree.
The accuracy of the existing methods achieved near to 92%	In our system we have achieved an accuracy of 98% which is better than the existing.
The time complexity of models is not implemented.	We are trying to achieve the least complexity.

Project Management:

Work completed

We have implemented TFIDF and Bag of Words vectorisation models and then used these processed data to the machine learning models like Decision tree, SVM, Random Forest. Then implemented word2vec by using LSTM for data preprocessing.

Responsibility (Task, Person):

Data Preprocessing: Saikiran Yedulla, Blessy Kuriakose

TF-IDF: Saikiran Yedulla, Blessy Kuriakose

Bag-Words : Yamuna Bollepalli, Saisri Teja Pepeti,

Decision Tree: Saisri Teja Pepeti,

Random Forest: Saikiran Yedulla,

Naive bayes: Blessy Kuriakose

SVM:Yamuna Bollepalli,

Data Preprocessing: Saisri Teja Pepeti, Yamuna Bollepalli

LSTM & BILSTM: Saisri Teja Pepeti, Yamuna Bollepalli

Word2Vec: Saisri Teja Pepeti, Yamuna Bollepalli, Saikiran Yedulla, Blessy Kuriakose

Contributions (members/percentage) :

Blessy Kuriakose - 25%

Saisri Teja Pepeti - 25%

Saikiran Yedulla - 25%

Yamuna Bollepalli - 25%

Work to be completed :

We will be implementing another word embedding technique that builds on the word2vec paradigm is fastText. FastText displays each word as an n-gram of letters rather than learning words as vectors directly. This enables the encoding to comprehend suffixes and prefixes and helps to learn the meaning of shorter words. We assume that fast text will provide better accuracy than the word2vec.

Then we will be completing some missing code parts of LSTM and BiLSTM so that we will be able to compare among the models.

Responsibilities (Task, Person) :

As we have previously worked on NLP techniques LSTM and BILSTM and word 2vec together now we will be planning further technique FastText collectively.

References

https://medium.com/analytics-vidhya/magic-of-tf-idf-202649d39c2f

https://neptune.ai/blog/vectorization-techniques-in-nlp-guide

https://www.sketchbubble.com/en/presentation-project-life-cycle.html

https://machine-learning.paperspace.com/wiki/machine-learning-models-explained

https://www.kaggle.com/datasets/benros0305/spamdetect

Recording Link : *https://www.youtube.com/watch?v=P9CH_P8qxDE*