A review of open-source machine learning algorithms for twitter text sentiment analysis and image classification

Conor Lynch
Nimbus Research Centre
Cork Institute of Technology
Cork, Ireland
conor.lynch@cit.ie

Christian O'Leary
Nimbus Research Centre
Cork Institute of Technology
Cork, Ireland
christian.oleary@cit.ie

Gary Smith
Nimbus Research Centre
Cork Institute of Technology
Cork, Ireland
gary.smith@cit.ie

Rose Bain
Nimbus Research Centre
Cork Institute of Technology
Cork, Ireland
rose.bain@cit.ie

Jacqueline Kehoe
Nimbus Research Centre
Cork Institute of Technology
Cork, Ireland
jacqueline.kehoe@cit.ie

Alex Vakaloudis
Nimbus Research Centre
Cork Institute of Technology
Cork, Ireland
alex.vakaloudis@cit.ie

Richrd Linger
Nimbus Research Centre
Cork Institute of Technology
Cork, Ireland
richard.linger@cit.ie

***ABSTRACT* - Sentiment analysis (SA) plays an important role in inferring sentiment or emotion from text and visual contents, such as images and videos to determine the overall contextual polarity of a document. Today, image recognition and classification are rapidly growing fields in the area of machine learning (ML). This paper presents a review of open-source machine learning algorithms, built using neural network-based frameworks such as TensorFlow and Keras, to serve as a benchmark for bespoke SA algorithms. This research also advocates open-source scikit-learn models for text tweets and image classification.**

Two prominent, publicly available and manually annotated benchmark text and image datasets were used to enable and assist in the correlation of this work with existing, present and future avant-garde and innovative methods. Quantitative results across four statistical criteria, including precision, recall, F1-score and accuracy compare favourably to the often complicated and tailored state-of-the-art methodologies developed. For SA, empirical results suggest deep-learning model frameworks to outperform scikit-learn algorithms. All experiments were conducted on computer hardware comprising 64GB of RAM and a NVIDEA GeForce RTX 2080 Ti GPU.

Keywords: Image classification, sentiment analysis, open-source, TensorFlow, Keras and CNN

I. INTRODUCTION

Twitter is a fast-growing enhanced online SMS platform where people can create, post, update and read short multimodal messages called tweets. Through tweets, users can share their opinions, views and thoughts.

Over the past decade, an interesting and popular research area in artificial intelligence (AI) called sentiment analysis (SA) is emerging [1]. SA, also known as "opinion mining" or "emotion of AI", may be useful in the cybernetic design of futuristic emotional and cognitive-based AI in humanoids. SA refers to the use of natural language processing (NLP), text mining, computational linguistics and bio measurements to methodically recognise, extricate, evaluate and examine emotional states and subjective information [2].

The main computational steps in this process are to determine the polarity or sentiment of the tweet and categorise them into a positive or a negative [3]. In general, SA is a way of identifying and categorising the polarity of a given text at document, sentence and phrase level [4]. Thus, social media messages, like tweets, have been polled to analyse user satisfaction on product quality and services [5] and could be useful for subject emotional state analysis. Therefore, a gradual practice has grown to extract the information from data available on social networks for the prediction of an election result, for use in educational purposes, or for the fields of business, communication and marketing [6].

Text-driven SA has been widely studied in the past decade on both random and benchmark textual Twitter datasets [7]. Only a few pertinent studies have reported on visual analysis of images to predict sentiment. Visual content analysis has always been important, although challenging. Given social network popularity, images have become a convenient carrier of information and content among online users

Due to the characteristics of Twitter data, hashtags, slang, emoticons, mentions etc., the primary issue with Twitter SA is the identification of the most suitable sentiment classifier that can correctly classify the tweets. Generally, heavily-tailored classification techniques like Naive Bayes classifiers [8], Random Forest classifiers [9], SVMs [10][11], Logistic Regression [12][13], statistical and lexicon weighting models [14], as well as combination or Hybrid models [15][16] are being used. The main objective of this paper is to investigate a

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

catalogue of existing open-source algorithms, including Google TensorFlow and Keras SA methods to benchmark text and image datasets and to provide theoretical comparisons to act as a baseline for emerging state-of-the-art approaches.

TensorFlow is an open source library created for Python by the Google Brain team [17]. TensorFlow offers a flexible, low-level API for building neural network models. These models are composed of nodes that form a graph which can be executed lazily or (as of version 2.0) eagerly. Importantly, TensorFlow models can be paralleled and can be easily executed on a Graphical Processor Unit (GPU), which dramatically decreases training times for deep learning models. In terms of **Keras**, it is a high-level API written in Python, capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, or Theano [18]. Keras uses these libraries as a backend to handle low-level operations. The Keras API is simpler to use but is a little less flexible. Keras simplifies code for deep learning, which in turn reduces development costs.

The paper is organised as follows: Section I comments on definitions, motivations and classification techniques used in sentiment analysis. Sections II, III and IV detail the benchmark datasets and evaluation metrics used. The results are tabulated and discussed in Section V, while Section VI describes high-level conclusions and recommendations.

II. DATASETS USED

This research used four publicly available and manually annotated datasets. To benchmark Twitter Text Sentiment Analysis (TSA), the SS-Tweet [19] and the STS-Gold [20] text datasets were used. As a standard for image classification, a two-class publicly available dataset by You et al. [21] and an eight-class annotated image sentiment classification dataset by Machajdik et al. [22] were used to evaluate an extensive selection of open-source algorithms for dual polarity classification, i.e. positive and negative sentiment.

Table 1 presents the SS-Tweet dataset [19] consisting of 4,242 tweets, manually labelled with their positive and negative sentiment strengths. i.e., a negative strength is a number between -1 (not negative) and -5 (extremely negative). Similarly, a positive strength is a number between 1 (not positive) and 5 (extremely positive). The dataset was constructed by [19] to evaluate SentiStrength, a lexicon-based method for sentiment strength detection. In this paper, as per Saif et al. [20], the tweets in this dataset were re-annotated with sentiment labels (negative, positive, neutral) rather than sentiment strengths, which will allow the use of this dataset for subjectivity classification in addition to sentiment strength detection. To this end, a single sentiment label was assigned to each tweet based on the following two rules inspired by the way SentiStrength works: (i) a tweet is deemed neutral if the absolute value of the tweet's negative to positive strength ratio is 1, (ii) a tweet is positive if its positive sentiment strength is 1.5 times higher than a negative, and is negative otherwise.

Table 1: Details of the SS-Tweet Twitter text dataset[a] in [19]

Dataset	No. of Tweets	Positives	Neutral	Negatives
SS-Tweet	4242	1252	1953	1037

[a]All 4242 images are available for download at: http://sentistrength.wlv.ac.uk/documentation/

Described in [20] and presented in Table 2, authors Saif et al. constructed the STS-Gold dataset. The goal of this dataset is to complement existing Twitter sentiment analysis evaluation datasets by providing a new dataset where tweets and targets (entities) are annotated independently, allowing for different sentiment labels between the tweet and the entities contained within it.

Table 2: Details of the STS-Gold Twitter text dataset[a] in [20]

Dataset	Total Tweets	Positives	Neutral	Negatives
STS-Gold	2034	632	0	1402

[a]All 2034 images are available for download at: https://github.com/pollockj/world_mood

One of the most popular image sentiment benchmark datasets was created by You et al. [21]. This image dataset is generated from image tweets, where an image tweets refer to those tweets that contain images. For their candidate testing images, the authors selected a total of 1,269 images. They employed the crowd intelligence services, Amazon Mechanical Turk (AMT) [23], to generate sentiment labels for these testing images. For this process, each image was passed through five AMT workers. Table 3 shows the statistics of the labelling results from the AMT system. In the Table below, "5 agree" indicates that all five of the AMT workers gave the same sentiment label for a given image. Only a small portion of the images, 153 out of 1269, had significant disagreements between the 5 workers.

Table 3: Details of the Twitter image dataset[3] in [21]

Sentiment	5 agree	At least 4 agree	At least 3 agree
Positive	581	689	769
Negative	301	427	500
Sum	882	1116	1269

[3]All 1269 images are available for download at: https://www.cs.rochester.edu/u/qyou/DeepSent/deepsentiment.html

Assembled by Machajdik et al. [22], the second image test set comprises a compilation of 807 artistic photographs downloaded from an art sharing site. The photographs were obtained by using eight emotion categories as search terms, thus the emotion category was determined by the artist who uploaded the photo. These photos are taken by people who attempt to evoke a certain emotion in the viewer through the conscious manipulation of the image composition, lighting, colours, etc. This dataset therefore allows us to investigate whether the conscious use of colours and textures by the artists improves the classification. To derive experimental results for two-point classification, Table 4 illustrates the classifying of the ARTphoto dataset into two categories as per Song et al. [24].

Table 4: Details of the ARTphoto image dataset[a] in [22]

	Amusement	Excitement	Contentment	Awe
Positive	101	105	70	102
Sum	**378**			
Sentiment	**Disgust**	**Anger**	**Fear**	**Sad**
Negative	70	77	115	166
Sum	**428**			

[a]All 806 images are available for download at: https://www.imageemotion.org/

III. EVALUATION PROTOCOLS

Using the datasets, as per Tables 1 to 4, four evaluation protocols were utilised, i.e., Precision, Recall, F1 score and Accuracy, which are widely used in [25][21] for text and image sentiment classification tasks. To facilitate greater transparency of model performance, the Mathews correlation coefficient (MCC) was also included. This metric is widely used in ML as a measure of the quality of binary (two-class) classification. It is regarded as a balanced measure, applicable despite class distribution, as it considers true and false positives and negatives. Considering a conventional positive-negative confusion matrix, as per Table 5a, Table 5b presents and overview of the modus operandi for the statistical metrics.

Table 5a: Typical positive-negative confusion matrix

		Predicted/Classified	
		Negative	**Positive**
Actual	**Negative**	True Negative (TN)	False Positive (FP)
	Positive	False Negative (FN)	True Positive (TP)

Table 5b: Details of the statistical metric formulae used

Measure	Formula	
Precision	$\frac{TP}{TP + FP}$	(1)
Recall	$\frac{TP}{TP + FN}$	(2)
F1 Score	$\frac{2 * TP}{2 * TP + FP\ \ FN}$	(3)
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$	(4)
MCC	$\frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$	(5)

Table 6 and 7 summarise model performance results, based on three runs, using the SS-Tweet and STS-Gold twitter text datasets respectively. Overall, the quantitative results across all five evaluation metrics compare favourably to the state-of-the-art methodologies discussed in [24]. Similarly, the performance metrics reporting model competence on the Twitter and ARTphoto image datasets are detailed in Table 8 and 9 below.

IV. MODEL PARAMETERS

In order to perform rigorous model testing and validation, and to facilitate a performance comparison of each model with reciprocal results elsewhere in the published literature, data from four recognised benchmark data sets was used. Each data set was shuffled and split into a training, validation and test component using a 60/20/20 split. Model output was based on the mean of 30 iterations and evaluated using k-fold cross validation during the test phase (k=5) to provide a stringent and robust appraisal process. Keras Neural network (NN) models were trained for a maximum of 50 epochs (forward passes) using an early stopping criterion with a patience value set to 10 epochs - based on validation accuracy. Checkpointing was used to preserve models with the lowest validation loss. Adam, the popular stochastic gradient descent-based method, was used as an optimiser and sparse categorical cross entropy was used as a loss function. Classification was done using Softmax activation functions and the mini batch size was set to 32.

Text models: Pre-processing steps for text data were selected based on changes to micro F1 score averaged over 30 iterations. The operations used are: (1) expand contractions, (2) apply Porter stemming, (3) convert to lowercase, (4) remove non-ascii characters (5) remove hashtags, (6) remove hyperlinks (7) remove @ mentions, (8) remove punctuation, (9) remove stop words, (10) Correct malformed HTML encodings, (11) Strip accents**.** N-grams, lemmatization, resampling and spell checking did not improve scores further. After pre-processing, text was tokenized and vectorized by token counts. Term Frequency-Inverse Document Frequency (TF-IDF) vectorization was also attempted but did not improve F1-scores. Standard scaling was applied after vectorization.

Bernoulli Naïve Bayes, Decision Tree, Gaussian Naïve Bayes, Logistic Regression, Linear Support Vector Classification (SVC) [1], k-nearest neighbour (KNN), Passive-Aggressive, Perceptron, Random Forest and SVC algorithms were implemented using scikit-learn. After some initial experimentation, Gaussian Naïve Bayes, KNN and Random Forest were disused as having low F1-scores and large memory requirements. The scikit-learn models were trained using k-fold cross validation during the training phase (in addition to the aforementioned cross-validation during the testing phase). A randomised grid search was used to select model parameters over 30 iterations. Convolutional Neural Networks (CNN), Long-Short-Term-Memory (LSTM) and Gated Recurrent Unit (GRU) models were implemented using Keras. Model architectures for reported scores are recorded in Table 10 Appendix I. Word embeddings with varying dimensions: 50, 100, 200 & 300 were used. Pre-trained embeddings used the GloVe word vectors[2].

[1] Linear SVC is a Support Vector Machine (SVM) variant that minimises *squared* hinge loss instead of hinge loss and penalizes the intercept.

[2] Downloadable at: https://nlp.stanford.edu/projects/glove/

Table 6a: Precision, Recall, F1 and Accuracy scores of state-of-the-art **open source deep learning methods** using the SS-Tweet Twitter text benchmark dataset

Model[a,b,c,d]	Positive Classification			Negative Classification			Overall		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Accuracy	Micro F1	MCC
LSTM-6-50-GloVe	0.629	0.418	0.502	0.560	0.589	0.574	0.614	0.614	0.390
LSTM-1-100-GloVe	0.546	0.566	0.556	0.571	0.522	0.545	0.610	0.552	0.296
LSTM-4-50-GloVe	0.645	0.390	0.486	0.528	0.633	0.576	0.602	0.602	0.377
LSTM-1-100-GloVe	0.532	0.622	0.574	0.558	0.420	0.479	0.601	0.601	0.374
LSTM-2-100-GloVe	0.614	0.462	0.527	0.541	0.478	0.508	0.601	0.601	0.364
LSTM-5-100-GloVe	0.618	0.355	0.451	0.541	0.671	0.599	0.601	0.601	0.377
LSTM-9-100-GloVe	0.527	0.594	0.558	0.510	0.633	0.565	0.595	0.595	0.390
LSTM-0-50-GloVe	0.547	0.530	0.538	0.522	0.585	0.551	0.594	0.594	0.371
CNN-5-50-GloVe	0.574	0.450	0.504	0.522	0.464	0.491	0.589	0.589	0.346
LSTM-3-50-GloVe	0.490	0.665	0.564	0.605	0.488	0.540	0.589	0.589	0.371

[a] Model naming structure: Model-no. of embedding layers-no of dimensions-word embedding
[b] Ranked according to the overall accuracy; Neutral results omitted to facilitate comparison purposes.
[c] CNN – Convolutional Neural Network, [d] LSTM – Long Short-Term Memory

Table 7a: Precision, Recall, F1 and Accuracy scores of state-of-the-art **open source deep learning methods** using the STS-Gold Twitter text benchmark dataset

Model[a,b,c,d,e]	Positive Classification			Negative Classification			Overall		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Accuracy	Micro F1	MCC
LSTM-6-100-GloVe	0.856	0.802	0.828	0.913	0.940	0.926	0.897	0.897	0.755
LSTM-3-100-GloVe	0.839	0.825	0.832	0.922	0.929	0.926	0.897	0.897	0.758
LSTM-8-100-GloVe	0.908	0.706	0.795	0.880	0.968	0.922	0.887	0.887	0.729
LSTM-5-100-GloVe	0.829	0.810	0.819	0.915	0.925	0.920	0.889	0.889	0.740
LSTM-9-50-GloVe	0.833	0.794	0.813	0.909	0.929	0.919	0.887	0.887	0.733
LSTM-7-50-GloVe	0.806	0.825	0.816	0.921	0.911	0.916	0.885	0.885	0.732
LSTM-8-50-GloVe	0.847	0.746	0.793	0.892	0.940	0.915	0.880	0.880	0.712
CNN-10-100-GloVe	0.835	0.762	0.797	0.897	0.932	0.914	0.880	0.880	0.713
LSTM-1-100-GloVe	0.832	0.746	0.787	0.891	0.932	0.911	0.875	0.875	0.700
CNN-7-100-GloVe	0.848	0.706	0.771	0.877	0.943	0.909	0.870	0.870	0.686
GRU-4-50-GloVe	0.808	0.667	0.730	0.861	0.929	0.894	0.848	0.848	0.631

[a] Model naming structure: Model-no. of embedding layers-no of dimensions-word embedding
[b] Ranked according to the Negative F1-score
[c] LSTM – Long Short-Term Memory, [d] CNN – Convolutional Neural Network, [e] GRU – Gated Recurrent Unit

Table 8a: Precision, Recall, F1 and Accuracy scores of state-of-the-art **open source deep learning methods** using the Twitter image benchmark dataset

Model[a]	Positive Classification			Negative Classification			Overall		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Accuracy	MCC	Run Time (sec)
Xception	0.855	0.855	**0.855**	0.828	0.828	**0.828**	0.843	0.683	181.13
Mobilenet	0.826	0.862	0.844	0.827	0.785	0.805	0.827	0.650	57.94
Densenet 121	0.809	0.884	0.844	0.847	0.750	0.794	0.823	0.645	115.16
Resnet 50	0.852	0.754	0.800	0.742	0.845	0.790	0.795	0.600	106.20
Densenet 201	0.864	0.732	0.792	0.730	0.862	0.791	0.791	0.594	174.68
Inception v3	0.772	0.870	0.815	0.834	0.690	0.747	0.787	0.582	136.98
Resnet 152	0.889	0.696	0.780	0.712	0.897	0.794	0.787	0.597	191.89
Nasnet Large	0.814	0.783	0.797	0.753	0.784	0.767	0.783	0.567	583.77
Inception Resnet v2	0.850	0.710	0.768	0.725	0.853	0.780	0.776	0.569	272.95
Resnet 101	0.885	0.667	0.760	0.693	0.900	0.782	0.772	0.571	145.74
Resnet 50 v2	0.763	0.841	0.800	0.784	0.690	0.734	0.772	0.539	98.67
Densenet 169	0.778	0.804	0.788	0.766	0.724	0.740	0.768	0.536	147.88
Mobilenet v2	0.837	0.703	0.764	0.703	0.836	0.764	0.764	0.539	66.05
Resnet 101 v2	0.854	0.594	0.701	0.646	0.879	0.745	0.724	0.486	139.88
Resnet 152 v2	0.840	0.609	0.706	0.649	0.862	0.741	0.724	0.480	185.56
Nasnet Mobile	0.820	0.594	0.689	0.636	0.845	0.726	0.709	0.448	119.32
VGG16	0.000	0.000	0.000	0.457	1.000	0.627	0.457	0.000	103.06
VGG19	0.000	0.000	0.000	0.457	1.000	0.627	0.457	0.000	110.64

[a] Ranked according to the overall accuracy score, see Table 8b Appendix I for scikit-learn model results.

Table 8b: Excerpt of Precision, Recall, F1 & Accuracy scores of state-of-the-art **open source scikit-learn methods** with the Twitter image benchmark dataset

Model[a,b]	Positive Classification			Negative Classification			Overall		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Accuracy	MCC	Run Time (sec)
Log. Reg. SVC	0.782	0.878	0.827	0.771	0.618	0.683	0.776	0.524	795.03
Linear SVC Log. Reg.	0.785	0.855	0.819	0.744	0.639	0.687	0.770	0.511	445.08
Log. Reg. Linear SVC	0.761	0.902	0.825	0.788	0.559	0.652	0.767	0.502	53.29
Log. Reg. Pas. Agg.	0.758	0.898	0.822	0.778	0.556	0.647	0.764	0.493	20.77
Perceptron Log. Reg.	0.765	0.883	0.820	0.760	0.578	0.656	0.764	0.492	407.48
SVC Log. Reg.	0.771	0.867	0.816	0.746	0.600	0.664	0.762	0.491	400.06
Bernoulli NB Bernoulli NB	0.741	0.923	0.822	0.811	0.500	0.617	0.757	0.483	7.97
Linear SVC Pas. Agg.	0.747	0.910	0.820	0.785	0.521	0.624	0.757	0.478	46.79
Perceptron Bernoulli NB	0.725	0.951	0.823	0.854	0.439	0.576	0.750	0.474	15.91
Linear SVC Bernoulli NB	0.720	0.940	0.815	0.830	0.432	0.563	0.741	0.451	46.53

[a] Table ranked according to overall accuracy score, see Table 8b Appendix I for additional scikit-learn model results.
[b] Log. = Logistic, Reg. = Regression, SVC = Support Vector Classifier, Pas. = Passive, Agg. = Aggressive, NB = Naïve Bayes

Table 9a: Precision, Recall, F1 and Accuracy scores of state-of-the-art **open source deep learning methods** using the ARTphoto image benchmark dataset

Model[a]	Positive Classification			Negative Classification			Overall		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Accuracy	MCC	Run Time (sec)
Densenet 169	0.786	0.715	**0.744**	0.696	0.756	0.720	0.734	0.477	820.00
Inception v3	0.756	0.719	0.728	0.628	0.700	0.648	0.698	0.401	94.92
Resnet 152	0.508	0.849	0.634	0.892	0.604	0.720	0.683	0.426	139.93
Densenet 121	0.605	0.770	0.673	0.772	0.621	0.686	0.682	0.384	83.71
Mobilenet v2	0.568	0.801	0.657	0.818	0.617	0.700	0.682	0.401	45.54
Nasnet Large	0.561	0.781	0.650	0.811	0.611	0.695	0.675	0.381	408.49
Inception Resnet v2	0.608	0.750	0.668	0.750	0.617	0.675	0.673	0.362	190.78
Mobilenet	0.568	0.755	0.642	0.777	0.607	0.678	0.664	0.353	37.04
Resnet 101 v2	0.629	0.714	0.664	0.703	0.622	0.657	0.663	0.333	100.12
Resnet 152 v2	0.576	0.773	0.626	0.757	0.620	0.664	0.658	0.360	135.48
Xception	0.546	0.768	0.633	0.784	0.589	0.670	0.654	0.342	120.17
Densenet 201	0.545	0.742	0.620	0.764	0.589	0.661	0.645	0.320	128.83
Resnet 101	0.424	0.851	0.562	0.901	0.568	0.696	0.642	0.368	102.38
Resnet 50	0.629	0.691	0.651	0.658	0.605	0.623	0.642	0.291	71.08
Resnet 50 v2	0.621	0.708	0.639	0.667	0.616	0.621	0.642	0.305	68.27
Nasnet Mobile	0.674	0.666	0.665	0.595	0.617	0.598	0.638	0.275	95.18
VGG16	0.000	0.000	0.000	1.000	0.457	0.627	0.457	0.000	64.47
VGG19	0.000	0.000	0.000	1.000	0.457	0.627	0.457	0.000	71.46

[a] Table ranked according overall accuracy score, see Table 9b Appendix I for scikit-learn model results.

Table 9b: Excerpt of Precision, Recall, F1 & Accuracy scores of state-of-the-art **open source scikit-learn methods** with the ARTphoto image benchmark dataset

Model[a,b]	Positive Classification			Negative Classification			Overall		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Accuracy	MCC	Run Time (sec)
SVC Perceptron	0.743	0.652	0.695	0.732	0.809	**0.769**	0.737	0.468	53.54
Linear SVC Logistic Reg.	0.752	0.609	0.671	0.716	0.831	0.769	0.729	0.454	158.47
Logistic Reg. Pas. Agg.	0.698	0.676	0.686	0.731	0.749	0.739	0.716	0.427	14.07
SVC Linear SVC	0.699	0.645	0.670	0.718	0.765	0.741	0.710	0.414	24.78
Linear SVC- Pas. Agg.	0.719	0.604	0.656	0.703	0.798	0.747	0.709	0.412	20.96
SVC Logistic Reg.	0.702	0.630	0.664	0.710	0.772	0.740	0.707	0.407	154.87
Bernoulli NB Bernoulli NB	0.713	0.605	0.654	0.702	0.792	0.744	0.706	0.406	7.14
Pas. Agg. Logistic Reg.	0.705	0.614	0.655	0.703	0.778	0.738	0.702	0.399	150.41
Bernoulli NB Perceptron	0.643	0.710	0.672	0.726	0.654	0.685	0.680	0.367	51.00
Logistic Reg. Perceptron	0.640	0.681	0.659	0.720	0.679	0.698	0.680	0.360	57.96

[a] Table ranked according to overall accuracy score, see Table 9b Appendix I for additional scikit-learn model results.
[b] Log. = Logistic, Reg. = Regression, SVC = Support Vector Classifier, Pas. = Passive, Agg. = Aggressive, NB = Naïve Bayes

Two Image classification model types investigated:

- The above mentioned scikit-learn algorithms were also used to perform classifications based on features extracted from images passed though Google's Vision API. These features included facial features, safe search labels, Optical Character Recognition (OCR), web search text and landmarks. One model was trained on the text output of Vision API while another with facial and safe search features. Separating the features prevents the vectorized text from dominating the prediction due to its dimensionality which is equal to the size of the data set vocabulary. Class probabilities from each model were combined as a weighted sum where weights were derived from the respective model's F1-score. The max class probability was taken as the final classification.

- The raw image data was passed through the deep neural networks listed in Table 8a and 9a. These networks were fine-tuned using 3 dense layers with dimension of 1024, 512, and 256 respectively. Leaky ReLu activation functions and dropout rates of 0.5 were used.

V. Results

A 2016 study by Tao [26] conducted a systematic and thorough empirical study on the machine learning algorithms for tweet sentiment analysis utilising the SS-Tweet and STS-Gold Twitter text data sets. Based on their experiments, they found that the Support Vector Machine (SVM) algorithm combined with POS (Part-Of-Speech), Bi-Grams (B), Senti-WordNet (Se) and Stop-Word (St) pre-processing steps achieved an accuracy of 0.612±0.013 for SS-Tweet. For negative classification on the STS-Gold data, an F1 score of 0.9017 was acquired using an SVM model merged with B-Se-St procedure.

Our experimental results pertaining to the SS-Tweet and STS-Gold Twitter text data sets, detailed in Table 6a, 6b, 7a and 7b respectively, present the Long-Short-Term-Memory (LSTM) algorithm as being the optimal performing model architecture. For SS-Tweet, a LSTM architecture encompassing 1 embedding layer, 100 dimensions and a 'Glove' word embedding (LSTM-1-100-Glove) obtained the highest overall accuracy score of 0.614. Whilst a LSTM-6-100-Glove model realised an F1-score of 0.926 for negative sentiment analysis on the STS-Gold data. As illustrated in Tables 6a and 7a, this study considered numerous deep learning frameworks, including numerous variations of Convolutional Neural Network (CNN) and Gated Recurrent Unit (GRU) models. Table 6b and 7b, in Appendix I, provide a cursory overview of selected open source scikit-learn methods applied to the SS-Tweet and STS-Gold Twitter text data sets.

A publication by Song et al. [24] in 2016 presented Sentiment Networks with visual Attention (SentiNet-A) architecture which explored visual attention to enhance image sentiment analysis. The Twitter and ARTphoto datasets, detailed in Table 3 and 4, were used to evaluate model performance. Considering Twitter image data where at least three annotators agreed, the developed SentiNet-A model achieved an F1-score of 0.814 and 0.718 for positive and negative polarity respectively. Whereas their tailored model effected a positive and negative F1-score of 0.699 and 0.746 for the ARTphoto image dataset.

The authors present in Table 8a, detailed research results from open source deep learning models relating to the Twitter image dataset. This presents the Xception algorithm as the most fitting model for positive sentiment classification – achieving an F1-score of 0.855. Other models that outperformed results in [24] included Densenet-121 (F1-score 0.844) and Mobilenet (F1-score 0.844). For negative polarity, Xception attained an F1-score of 0.828. Other effective algorithms included Resnet-152 (F1-score 0.794) and Resnet-50 (F1-score 0.790) – where 152 and 50 refer to the number of hidden convolutional layers. Table 8b, continued within Appendix I, summarises a range of investigated scikit-learn methods applied to the Twitter image dataset. With the best, in terms of overall model accuracy, being a Logistic Regression Support Vector Classification model.

Table 9a and 9b, summarises a comparable analysis of open source deep learning and scikit-learn models for the ARTphoto image dataset. In this instance, the Densenet 169 algorithm is the optimal model for positive sentiment classification – with an F1-score of 0.744. The other model that outperformed results in [24] included the Inception v3 model - F1-score 0.728. For negative polarity, the SVC Perceptron and the Linear SVC Logistic Regression models, both with an F1-score of 0.769, surpassed the bespoke model by Song et al. [24]. Table 9b, results are continued within Appendix I and capture the inferior results for a catalogue of scikit-learn methods applied to the ARTphoto image dataset.

VI. Conclusions & Recommendations

Social media has seen unprecedented growth in recent years. Users often express their views and emotions regarding a range of topics on social media platforms. As such, social media has become a crucial resource for obtaining information directly from end-users. While the benefits of using a resource such as Twitter include large volumes of data and direct access to end-user sentiments, there are several obstacles associated with the use of social media data. These include the use of non-standard terminologies, misspellings, short ambiguous posts and data imbalance, to name a few [27]. Consequently, Machine learning approaches have become an effective tool in performing meaningful message-level sentiment classification on Twitter data. This paper analyses a range of open source algorithms applied to two text and image benchmark datasets. Due to the unique characteristics of the tweet data, choosing machine learning classifiers and adjusting the parameters of algorithms are the essential tasks in the process of tweet sentiment analysis. It is hoped that the paper will act as a new benchmark for the field of TSA.

Primary results and findings indicate that for both a relatively evenly distributed and a negatively skewed tweet dataset, an LSTM-based model produced the best results for positive and negative TSA. Whilst for image sentiment analysis, the Xception, Densenet 169 and the SVC Perceptron algorithms all produced favourable results. Future work by the authors will investigate the use of open source machine learning algorithms, applied to the field of multimodal classification for video sentiment analysis.

Acknowledgments

The authors would like to thank FastModel Technologies [28] for project funding and the Cork Institute of Technology for its continued support.

REFERENCES

[1] P. Tyagi and R. C. Tripathi, "A Review Towards the Sentiment Analysis Techniques for the Analysis of Twitter Data," *SSRN Electron. J.*, 2019.

[2] A. Alsaeedi and M. Z. Khan, "A study on sentiment analysis techniques of Twitter data," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 2, pp. 361–374, 2019.

[3] Ankit and N. Saleena, "An Ensemble Classification System for Twitter Sentiment Analysis," *Procedia Comput. Sci.*, vol. 132, no. Iccids, pp. 937–946, 2018.

[4] L. F. S. Coletta, N. F. F. De Silva, E. R. Hruschka, and E. R. Hruschka, "Combining classification and clustering for tweet sentiment analysis," *Proc. - 2014 Brazilian Conf. Intell. Syst. BRACIS 2014*, pp. 210–215, 2014.

[5] B. J. Jansen, M. Zhang, K. Sobel, and A. Chowdury, "Twitter power: Tweets as electronic word of mouth," *J. Am. Soc. Inf. Sci. Technol.*, vol. 60, no. 11, pp. 2169–2188, 2009.

[6] A. Hasan, S. Moin, A. Karim, and S. Shamshirband, "Machine Learning-Based Sentiment Analysis for Twitter Accounts," *Math. Comput. Appl.*, vol. 23, no. 1, p. 11, 2018.

[7] A. Kumar and G. Garg, "Sentiment analysis of multimodal twitter data," *Multimed. Tools Appl.*, 2019.

[8] P. Ganesh, "Twitter Sentiment Analysis Using a Modified Naïve Bayes Algorithm," in *International Conference on Information Systems Architecture and Technology*, 2018.

[9] P. Nergis; KELEŞ, "Sentiment analysis using a random forest classifier on turkish web comments," *Commun. Fac. Sci. Univ. Ankara*, vol. 59, no. 2, pp. 69–79, 2017.

[10] M. Ahmad, S. Aftab, and I. Ali, "Sentiment Analysis of Tweets using SVM," *Int. J. Comput. Appl.*, vol. 177, no. 5, pp. 25–29, 2017.

[11] D. B. Savita and P. D. Gore, "Sentiment Analysis on Twitter Data Using Support Vector Machine," *Int. J. Comput. Sci. Trends Technol.*, vol. 4, no. 3, pp. 365–370, 2016.

[12] A. Tyagi and N. Sharma, "Sentiment Analysis using logistic regression and effective word score heuristic," *Int. J. Eng. Technol.*, vol. 7, no. 2, pp. 20–23, 2018.

[13] S. T. Indra, L. Wikarsa, and R. Turang, "Using logistic regression method to classify tweets into the selected topics," *2016 Int. Conf. Adv. Comput. Sci. Inf. Syst. ICACSIS 2016*, no. October, pp. 385–390, 2017.

[14] O. Kolchyna, T. T. P. Souza, P. Treleaven, and T. Aste, "Twitter Sentiment Analysis: Lexicon Method, Machine Learning Method and Their Combination," no. July, 2015.

[15] P. P. Balage Filho and T. A. S. Pardo, "NILC USP: A hybrid system for sentiment analysis in twitter messages," **SEM 2013 - 2nd Jt. Conf. Lex. Comput. Semant.*, vol. 2, no. SemEval, pp. 568–572, 2013.

[16] H. V. Thakkar, "Twitter Sentiment Analysis using Hybrid Naïve Bayes," no. June 2013, pp. 1–47, 2013.

[17] "TensorFlow," 2019. [Online]. Available: https://www.tensorflow.org/.

[18] "Keras," 2019. [Online]. Available: https://keras.io/.

[19] M. Thelwall, K. Buckley, and G. Paltoglou, "Sentiment Strength Detection for the Social Web1," *Bulg. J. Agric. Sci.*, vol. 23, no. 5, pp. 739–742, 2017.

[20] H. Saif, M. Fernandez, Y. He, and H. Alani, "Evaluation datasets for Twitter sentiment analysis a survey and a new dataset, the STS-Gold," *CEUR Workshop Proc.*, vol. 1096, pp. 9–21, 2013.

[21] Q. You, J. Luo, H. Jin, and J. Yang, "Robust Image Sentiment Analysis Using Progressively Trained and Domain Transferred Deep Networks," 2015.

[22] Machajdik; J.; Hanbury; A, "Affective Image Classification using Features Inspired by Psychology and Art Theory," in *ACM International Conference on Multimedia*, 2010, pp. 83–92.

[23] Amazon, "Amazon Mechanical Turk." [Online]. Available: https://www.mturk.com/.

[24] K. Song, T. Yao, Q. Ling, and T. Mei, "Boosting image sentiment analysis with visual attention," *Neurocomputing*, vol. 312, pp. 218–228, 2018.

[25] J. Yuan, "Sentribute : Image Sentiment Analysis from a Mid-level Perspective Categories and Subject Descriptors," 2013.

[26] H. Tao, "An Empirical Study on Machine Learning for Tweet Sentiment Analysis," THE UNIVERSITY OF NEW BRUNSWICK, 2016.

[27] A. Sarker, A. Nikfarjam, D. Weissenbacher, and G. Gonzalez, "DIEGOLab: An Approach for Message-level Sentiment Classification in Twitter," no. SemEval, pp. 510–514, 2015.

[28] "FastModel Technologies." .

APPENDIX I

Table 6b: Precision, Recall, F1 and Accuracy scores of state-of-the-art **open source scikit-learn methods** using the SS-Tweet Twitter text benchmark dataset

Model*	Positive Classification			Negative Classification			Overall		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Accuracy	Micro F1	MCC
Bernoulli Naïve Bayes	0.442	0.349	0.389	0.473	0.249	0.325	0.506	0.506	0.195
Decision Tree	0.441	0.270	0.333	0.461	0.238	0.312	0.505	0.505	0.184
Linear SVC	0.414	0.245	0.304	0.428	0.246	0.307	0.492	0.492	0.163
Logistic Regression	0.440	0.307	0.361	0.415	0.266	0.323	0.491	0.491	0.169
Passive Aggressive	0.414	0.399	0.406	0.394	0.339	0.364	0.478	0.478	0.175
Perceptron	0.416	0.202	0.260	0.345	0.176	0.227	0.473	0.473	0.115
SVC	0.402	0.397	0.399	0.343	0.340	0.341	0.454	0.454	0.147

* Ranked according to the overall micro F1-score

Table 7b: Precision, Recall, F1 and Accuracy scores of state-of-the-art **open source scikit-learn methods** using the STS-Gold Twitter text benchmark dataset

Model*	Positive Classification				Negative Classification				Overall
	Precision	Recall	F1 Score	MCC	Precision	Recall	F1 Score	MCC	Accuracy
Bernoulli Naïve Bayes	0.710	0.458	0.556	-	0.790	0.916	0.848	-	0.774
Decision Tree	0.597	0.321	0.416	-	0.747	0.904	0.818	-	0.723
Linear SVC	0.696	0.508	0.586	-	0.802	0.899	0.847	-	0.777
Logistic Regression	0.691	0.469	0.558	-	0.791	0.904	0.844	-	0.769
Passive Aggressive	0.693	0.520	0.594	-	0.805	0.896	0.848	-	0.779
Perceptron	0.538	0.521	0.528	-	0.786	0.796	0.791	-	0.711
SVC	0.707	0.390	0.501	-	0.772	0.928	0.843	-	0.761

* Ranked according to the overall micro F1-score

Table 8b: Continued: Precision, Recall, F1 and Accuracy scores of state-of-the-art open source scikit-learn methods using the Twitter image benchmark dataset

Model*	Positive Classification			Negative Classification			Overall		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Accuracy	MCC	Run Time (sec)
Pas. Agg. Bernoulli NB	0.744	0.531	0.62	0.679	0.844	0.752	0.700	0.398	4.96
Pas. Agg. Pas. Agg.	0.725	0.56	0.63	0.685	0.815	0.744	0.698	0.392	6.64
Pas. Agg. Linear SVC	0.714	0.565	0.625	0.69	0.807	0.742	0.696	0.387	22.58
Pas. Agg. Perceptron	0.707	0.556	0.621	0.682	0.807	0.739	0.691	0.375	52.57
Linear SVC Bernoulli NB	0.693	0.57	0.623	0.688	0.792	0.735	0.69	0.371	15.09
Logistic Reg. Bernoulli NB	0.685	0.589	0.634	0.688	0.77	0.726	0.687	0.366	11.96
Linear SVC Linear SVC	0.707	0.543	0.608	0.677	0.805	0.734	0.684	0.365	38.31
Perceptron Bernoulli NB	0.718	0.517	0.601	0.668	0.827	0.739	0.684	0.364	13.16
Logistic Reg. Linear SVC	0.655	0.652	0.653	0.705	0.708	0.707	0.682	0.36	29.74
SVC Bernoulli NB	0.688	0.558	0.616	0.676	0.784	0.726	0.68	0.352	6.56
Log. Reg. Log. Reg.	0.676	0.556	0.608	0.673	0.774	0.719	0.673	0.339	153.15
Linear SVC Perceptron	0.692	0.522	0.595	0.664	0.802	0.726	0.673	0.339	64.47
Pas. Agg. Decision Tree	0.647	0.628	0.637	0.691	0.708	0.699	0.671	0.337	201.1
Bernoulli NB Log. Reg.	0.642	0.599	0.619	0.678	0.716	0.696	0.662	0.318	152.6
Decision Tree Log. Reg.	0.649	0.58	0.612	0.672	0.733	0.701	0.662	0.316	174.31
Perceptron Pas. Agg.	0.674	0.512	0.582	0.656	0.79	0.717	0.662	0.316	15.52
Perceptron Linear SVC	0.698	0.459	0.551	0.647	0.835	0.728	0.662	0.318	31.09
Decision Tree Linear SVC	0.616	0.667	0.639	0.701	0.65	0.673	0.658	0.317	42.21
Decision Tree Pas. Agg.	0.615	0.676	0.644	0.697	0.638	0.666	0.656	0.313	25.8
SVC Decision Tree	0.638	0.572	0.602	0.666	0.722	0.692	0.653	0.299	198.4
Bernoulli NB Pas. Agg.	0.614	0.633	0.622	0.678	0.658	0.667	0.647	0.292	5.71
Bernoulli NB SVC	0.614	0.623	0.618	0.676	0.667	0.671	0.647	0.29	146.59
SVC SVC	0.658	0.449	0.528	0.634	0.802	0.707	0.64	0.271	148.43
Bernoulli NB Decision Tree	0.627	0.542	0.58	0.647	0.719	0.68	0.637	0.267	205.26
Perceptron Perceptron	0.627	0.514	0.565	0.642	0.741	0.688	0.637	0.262	59.35
Log. Reg. Decision Tree	0.601	0.614	0.606	0.667	0.654	0.66	0.636	0.268	211.61
Linear SVC SVC	0.648	0.459	0.535	0.631	0.786	0.699	0.636	0.261	158.63
Decision Tree Decision Tree	0.632	0.493	0.553	0.635	0.753	0.689	0.633	0.256	226.58
SVC Pas. Agg.	0.635	0.471	0.539	0.633	0.772	0.695	0.633	0.255	8.98
Decision Tree Bernoulli NB	0.606	0.565	0.585	0.65	0.687	0.668	0.631	0.254	24.15

* Ranked according to the overall accuracy. Log. = Logistic, Reg. = Regression, Pas. = Passive, Agg. = Aggressive, NB = Naïve Bayes

Table 9b: Continued: Precision, Recall, F1 and Accuracy scores of state-of-the-art open source scikit-learn methods using the ARTphoto image benchmark dataset

Model*	Positive Classification			Negative Classification			Overall		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Accuracy	MCC	Run Time (sec)
Log. Reg. Bernoulli NB	0.776	0.846	0.809	0.724	0.622	0.669	0.758	0.484	419.50
SVC Pas. Agg.	0.757	0.889	0.813	0.768	0.536	0.609	0.750	0.468	837.09
Log. Reg. Perceptron	0.763	0.850	0.804	0.717	0.589	0.646	0.748	0.459	400.12
Pas. Agg. Pas. Agg.	0.733	0.914	0.813	0.784	0.484	0.597	0.745	0.453	18.04
Linear SVC Perceptron	0.733	0.914	0.813	0.784	0.484	0.597	0.745	0.453	18.04
Bernoulli NB Log. Reg.	0.803	0.768	0.784	0.667	0.707	0.684	0.744	0.472	402.94
Pas. Agg. SVC	0.760	0.851	0.801	0.715	0.572	0.623	0.742	0.447	777.43
Linear SVC SVC	0.724	0.931	0.813	0.806	0.444	0.565	0.739	0.444	808.04
Pas. Agg. Log. Reg.	0.744	0.882	0.803	0.755	0.510	0.579	0.736	0.435	796.89
SVC SVC	0.714	0.941	0.812	0.819	0.416	0.550	0.735	0.436	16.67
Pas. Agg. Linear SVC	0.725	0.905	0.804	0.757	0.464	0.573	0.732	0.421	117.72
Log. Reg. Log. Reg.	0.710	0.940	0.809	0.813	0.404	0.539	0.730	0.424	7.07
Pas. Agg. Bernoulli NB	0.718	0.918	0.805	0.774	0.439	0.558	0.730	0.418	42.07
Perceptron SVC	0.724	0.903	0.803	0.756	0.462	0.569	0.730	0.418	21.18
Perceptron Pas. Agg.	0.720	0.906	0.802	0.757	0.455	0.567	0.729	0.415	148.08
Linear SVC Linear SVC	0.719	0.903	0.801	0.752	0.455	0.566	0.727	0.410	83.33
Decision Tree Bernoulli NB	0.738	0.850	0.790	0.696	0.531	0.602	0.725	0.407	27.82
Perceptron Linear SVC	0.711	0.918	0.801	0.764	0.419	0.540	0.722	0.400	52.21
Decision Tree Linear SVC	0.739	0.835	0.784	0.684	0.542	0.603	0.720	0.399	63.63
SVC Perceptron	0.713	0.903	0.796	0.741	0.435	0.548	0.719	0.392	116.61
SVC Bernoulli NB	0.699	0.940	0.801	0.799	0.370	0.501	0.716	0.392	14.45
SVC Linear SVC	0.699	0.924	0.795	0.751	0.378	0.496	0.710	0.367	48.70
Decision Tree Log. Reg.	0.745	0.796	0.769	0.648	0.578	0.610	0.710	0.383	427.94
Bernoulli NB-perceptron	0.806	0.686	0.739	0.608	0.742	0.666	0.708	0.420	108.12
Decision Tree SVC	0.735	0.807	0.768	0.645	0.545	0.588	0.704	0.365	802.79
Perceptron Perceptron	0.698	0.899	0.786	0.712	0.396	0.505	0.701	0.347	116.50
Bernoulli NB Pas. Agg.	0.818	0.652	0.722	0.595	0.774	0.670	0.700	0.420	10.72
Decision Tree Perceptron	0.719	0.830	0.770	0.651	0.495	0.561	0.699	0.346	127.93
Decision Tree Pas. Agg.	0.721	0.818	0.767	0.639	0.507	0.564	0.696	0.342	32.13
Pas. Agg. Decision Tree	0.721	0.799	0.757	0.624	0.518	0.565	0.689	0.330	579.91

* Ranked according to the overall accuracy. Log. = Logistic, Reg. = Regression, Pas. = Passive, Agg. = Aggressive, NB = Naïve Bayes

Table 10: Deep learning model architectures for text classifiers

Model	Architecture
LSTM-0	Embedding layer, LSTM layer (16 units), Softmax layer
LSTM-1	Embedding layer, LSTM layer (32 units), Softmax layer
LSTM-2	Embedding layer, LSTM layer (64 units), Softmax layer
LSTM-3	Embedding layer, LSTM layer (128 units), Softmax layer
LSTM-4	Embedding layer, LSTM layer (64 units), Dropout layer, LSTM layer (32 units), Softmax layer
LSTM-5	Embedding layer, LSTM layer (128 units), Dropout layer, LSTM layer (64 units), Dropout layer, LSTM layer (32 units), Softmax layer
LSTM-6	Embedding layer, Bidirectional LSTM layer (64 units), Dropout layer, Softmax layer
LSTM-7	Embedding layer, Bidirectional LSTM layer (64 units), Dropout layer, Dense layer (64 units), Softmax layer
LSTM-8	Embedding layer, Bidirectional LSTM layer (32 units), Dropout layer, Dense layer (32 units), Softmax layer
LSTM-9	Embedding layer, Bidirectional LSTM layer (64 units), Dropout layer, Bidirectional LSTM layer (32 units), Dropout layer, Dense layer (64 units), Dense layer (32 units), Softmax layer
GRU-4	Embedding layer, GRU layer (64 units), Dropout layer, GRU layer (32 units), Softmax layer
CNN-5	Embedding layer, Convolutional layer (64 filters, kernel size 3), Pooling layer, Dense layer (64 units), Softmax layer
CNN-7	Embedding layer, Convolutional layer (64 filters, kernel size 5), Pooling layer, Dense layer (64 units), Softmax layer
CNN-10	Embedding layer, Convolutional layer (64 filters, kernel size 5), Convolutional layer (32 filters, kernel size 5), Pooling layer, Dense layer (32 units), Softmax layer