11-2018

Predicting User Interaction on Social Media using Machine Learning

Chad Crowe
University of Nebraska at Omaha

Predicting User Interaction on Social Media using Machine Learning

A Thesis

Presented to the

College of Information Science and Technology

and the

Faculty of the Graduate College

University of Nebraska at Omaha

In Partial Fulfillment of the Requirements for the Degree

Master of Science in Computer Science

by

Chad Crowe

November 2018

Supervisory Committee

Dr. Brian Ricks

Dr. Margeret Hall

Dr. Yuliya Lierler

Abstract

Predicting User Interaction on Social Media using Machine Learning

Chad Crowe, MS

University of Nebraska, 2018

Advisor: Dr. Margeret Hall

Analysis of Facebook posts provides helpful information for users on social media. Current papers about user engagement on social media explore methods for predicting user engagement. These analyses of Facebook posts have included text and image analysis. Yet, the studies have not incorporate both text and image data. This research explores the usefulness of incorporating image and text data to predict user engagement. The study incorporates five types of machine learning models: text-based Neural Networks (NN), image-based Convolutional Neural Networks (CNN), Word2Vec, decision trees, and a combination of text-based NN and image-based CNN. The models are unique in their use of the data. The research collects 350k Facebook posts. The models learn and test on advertisement posts in order to predict user engagement. User engagements includes share count, comment count, and comment sentiment. The study found that combining image and text data produced the best models. The research further demonstrates that combined models outperform random models.

Contents

Abstract **i**

1 Introduction **1**

1.0.1 Problem . . . 2

1.0.2 Thesis Structure . . . 4

1.1 Scope of Study . . . 4

1.2 Scenario . . . 5

1.3 Significance of Study . . . 5

1.4 Motivation . . . 6

2 Related Work **8**

2.0.1 User Interaction Studies . . . 8

2.0.2 ROI Studies . . . 9

2.0.3 Other Research . . . 10

2.0.4 Convolutional Neural Networks . . . 13

2.1 Current Theory and Practice . . . 14

2.1.1 Computer Vision . . . 14

2.1.2 Natural Language Processing . . . 18

2.1.3 Feature Extraction . . . 20

2.1.4 Deep Learning with Images . . . 23

3 Research Gaps and Questions **25**

3.1 Research Gaps . . . 25

3.2 Research Questions . . . 25

4 Research Methodology **27**

4.1 Data Context . . . 27

4.1.1 Data Origin . . . 28

4.1.2 Collection . . . 29

4.1.3 Analysis Methods . . . 30

4.2 Text Processing . . . 33

4.3 Open Sourced Data . . . 33

5 Model Methodology **35**

5.1 Data Verification . . . 35

5.1.1 Approach . . . 36

5.2 Data Mining 37
5.3 Sentiment Analysis 41
5.4 Algorithm Reliability 42
5.5 Evaluation Criteria 44
5.5.1 Experiments 45
5.5.2 Word2Vec Experiment 46

6 Results 47
6.1 Summary of Findings 47
6.2 Linear Models of Scraped Data 49
6.3 Text-based NN Models 52
6.4 Image-based CNN Models 53
6.5 Combined Models 54
6.6 Model Prediction Problems 54
6.7 Word2Vec Models 55
6.8 Use Case 56

7 Discussion 57
7.1 Addressing Research Questions 57
7.2 Linear Models 58
7.3 Combined Model 59
7.4 Model Training Time 59
7.5 Issues with CNN and NN Networks 60
7.6 Metric Prediction 60
7.7 Model Loss and Outperforming Random Guessing 61
7.8 Unexpected Findings and Restructured Hypotheses 62
7.9 Study Contributions 62

8 Conclusion 65
8.1 Limitations 66
8.2 Future Work 66

Bibliography 68

Bibliography 68

Chapter 1

Introduction

The majority of Americans engage on social media (Smith 2018). This is due in part to social media's low barriers (Azizian et al., 2017). For example, anyone can scroll through a Twitter or Facebook feed. Yet, there is another side of social media. Not all users on social media consume content. Social media platforms provide interfaces for advertisers. On these platforms, companies pay a small fee to display their ads. Social media is having a larger influence on businesses and consumers (Fisher 2009). This creates a massive conglomerate of marketing. Each of these companies wants to reach customers with their products. They trust that social media is an effective means to reach target audiences. Often, social media platforms let companies specify advertisement demographics. Targeting information ensures that advertising money is well spent. Investors spend billions of dollars to advertise on social media (Statista 2018). These investors care about the efficacy of their social media spending.

This creates a large challenge for social media platforms. Social media platforms need to provide the ability to connect companies with users. The field of advertising on social media is new. Platforms want to justify their effectiveness to potential advertisers. Measuring this effectiveness after advertising is possible. Yet, companies prefer a way to calculate return on investment (ROI) before investing. One way to calculate ROI is to measure user engagement. This study predicts user engagement for advertisement posts. The user engagement metrics include share count, comment count, and comment sentiment. This study explores predicting user engagement with different types of machine learning models. The models are unique in their incorporation of text and image data. The goal is to understand which model-types best predict user engagement.

The research applies these models to a real-world use-case. The research uses models, which can predict user engagement, in order to vet for better performing ads. Platforms could use these models to inform advertisers which of their ads will perform best on the platform. This allows advertisers have their ads vetted. The vetting could prevent advertisers from spending a lot of money showing worse ads. Moreover, the vetting would allow advertisers to only show ads that will perform best.

1.0.1 Problem

Almost 70% of adults use Facebook (Gramlich 2018). Running such a large site for many users is expensive. Yet, users pay no fee for using Facebook. Google is like Facebook in that it also provides free services to users. The services are free because platforms generate revenue from advertising. Google generates more than 70% of its revenue from advertising (Statista 2017). Facebook generates more than 80% of its revenue from advertising (Statista 2018). Advertising is not without difficulties. Most people do not like advertisements (Shi, X. 2017). In fact, people generally feel annoyed by advertisements (Shi, X. 2017). Users will even avoid advertisements (Shi, X. 2017). This results in users using social media less. Fewer users mean that advertisements receive less engagement. Lower user engagement means the platform makes less money. Fewer users also mean that platforms are unable to reach as many people. In this way, advertisements can hurt the platform and advertisers. The challenge is to deliver relevant content to users. The content might agree with user interests, hobbies, and preferences. If advertisers can deliver relevant content, then advertisers, users, and the platform benefit. The users buy content. The advertiser's sell merchandise and the platform make money. Yet, there are millions of users on social media and many advertisements (Smith 2018). This thesis aims to explore this problem by predicting user engagement. This helps companies better understand how their advertisement will perform on social media. It also gives advertisers feedback. More confident advertisers will invest more on the platform.

The thesis performs its analysis on the Facebook post with image and text data from a Facebook post as input. The research performs computations on the data and predicts user interaction. The aim of the thesis is to predict user interaction. The thesis will explore this prediction with different machine learning models. The analysis used the Facebook post since it contains image and text data. Its data is also available via Facebook's Graph API. The study collected and trained on exactly 350k posts. The Facebook post is generic. Other platforms have similar representations of the Facebook post. These include tweets and Pinterest posts. The posts in this study are from advertisers. This limitation might simplify the models.

Previous studies have modeled user interaction on social media (Straton 2015). The paper refers to post metrics as post-performance. Facebook refers to the metrics as user interactions, or engagements (Facebook API). The Facebook post may include text, an image, and meta-data. The meta-data includes shares, comments, and data from the post's Facebook page. The metrics denoting user interaction include the number of shares and the comments. The scope of the research will consist of predicting these metrics. The research creates many types of machine learning models. The research will uncover which can best predict user interaction. One goal is to model user interaction with both text and image data. The study will gauge how well models can predict user interaction.

1.0.2 Thesis Structure

The rest of Chapter 1 will explain the motivation for the research. Chapter 2 will explore the related work. Chapter 3 will address the research gaps and define the research questions. Chapter 4 will present the research methodology. Chapter 5 will explain machine learning model methodology and data processing. Chapter 6 will present the results and chapter 7 will discuss the results. Finally, chapter 8 will present the study's conclusions.

1.1 Scope of Study

The study explores three types of models. These are text-based, image-based, and those that combine text and image data. The scope of the study limits both its input and output data. Its input data includes the post's text and an image if present. The output is user interaction. The user interactions considered include share count, comment count, and comment sentiment. These outputs were the available user interactions from the Facebook API. These inputs are ubiquitous on Facebook and common on other social media platforms. There was enough Facebook data to train all but one type of the machine learning models. Even more, each post always has some share count, comment count, and comment sentiment. The study has interest in the benefits and effects of combining image and text data. The study will also determine if the model is useful in practice. The study will compare the model's performance with a random model.

1.2 Scenario

This thesis will be dealing with social media data. Specifically, Facebook posts and its associated data. This thesis will use this data to build machine learning models. These models will predict post data. This research is most interested in what features improve predicting aspects of user interaction. This study defines user interaction number of shares, comment count, and the sentiment of those comments. This paper hopes to use these metrics to generate predictive models.

The techniques will include data transformation, post-analysis, and the creation of machine learning and statistical models to fit the data. The goal is to create a generative predictive model for any social media post. The problem under consideration is one of regression. An overarching goal of the thesis is to predict user interaction. Given a Facebook post and its contextual information, predict the number of shares and comments received by that post. This contextual information includes the post's accompanying text and photo.

1.3 Significance of Study

The study contributes to understanding user engagement on social media. The study also highlights which models best predict user interaction. The research will uncover how well models can measure user interaction. The paper will also discuss how models can predict with both image and text data. On a practical level, the models could be a tool for platforms and help advertisers. The study also serves as an overview for approaching user interaction prediction.

1.4 Motivation

There exist a great number of algorithms which interact with social media. Kaggle hosted a competition for the company Avazu for predicting click-through rate using ad data (Kaggle 2015). Yet, a download of their dataset showed that the provided input variables focus on where the ad was displayed, where the ad was hosted, the type of ad that was displayed, and the device the ad was viewed on. While these can provide a great deal of explanation, they lack the most important information, i.e. the ad itself. Other studies predicting click-through rates have focused on the text content of the ad (Li et al., 2015). Yet, these studies oversimplify the data. The vast majority of advertisements on social media consist of text and image data. When users interact with ads, they are mainly interacting with the content of the ad. It seems like the best way to approach predicting user interaction must include this ad data. Moreover, it is commonly said that a picture's worth a thousand words. Any analysis which eclipses this data is likely short-sided and suffer in precision. New analyses should include both text and image data.

It is no surprise that the analysis of images on social media is largely under-addressed. Images are notoriously difficult to analyze. They are wrought with noise, colors, and might contains many objects at many angles. Moreover, images might contain text and often convey meaning. This represents a massive amount of variance in image data. This degree of complication makes images both interesting and notoriously difficult for analysis.

In recent years, more APIs and tools have been created to perform object identification and feature detection on images. Their algorithms perform object identification and identify similarities between images. Many a paper has performed sentiment analysis on images (Wang et al., 2015). The ability to discover similar images, their objects, and its sentiment provides an opportunity for using this data in social media analysis.

The increase in machine learning libraries includes implementations for image-related algorithms. Convolutional neural networks are more accessible due to libraries like Keras and TensorFlow. These convolutional neural networks give their users the ability to perform better image analysis. They have been most widely used in image classification. They also have applications in image clustering and textual synthesis. These are new tools for researchers within the domain of image-analysis. They provide the ability to improve upon current social media analysis by also analyzing image data.

Social media provides an optimal environment for such research. A great deal of its data is publically available through social media platform APIs. Such APIs provide textual and image data at a massive scale. This data can be downloaded, transformed, and organized for large-scale machine learning analysis.

These new tools, techniques, and availability of data provide an exceptional opportunity to improve upon current social media analysis. These also provide opportunities to develop new techniques and discover important features for social media analysis. This paper seeks to append to this largely under-addressed field within academia.

Chapter 2

Related Work

2.0.1 User Interaction Studies

Li et al. predicts click-through rates on Twitter (2015). Click-through prediction estimates the likelihood for users to click on advertisements. For example, a user might view some merchandise on Amazon. Amazon then stores a cookie in that user's browser. Later, while on Twitter, Amazon will pay Twitter to reshow this merchandise on the user's feed. The study itself predicts the likelihood of ad clicks on user feeds. This likelihood is difficult to calculate since clicks are few and far between. The probability of ad clicks is often a fraction of a percent. The goal is to model the likelihood of user clicks. The study made predictions by correlation user interests with ad relevance. The study also modeled twitter sessions for the study.

Stranton et al. predicts user interaction on Facebook with post, text, and time data (2015). The study predicted page likes, shares, and comment counts from this data. The analysis categorized all posts into categories of low, medium, and high engagement. A neural network trained on this data. The study was successful at predicting for low user engagement. The neural network performed poorly at predicting higher levels of engagement. The particular study sampled from public health data. The study's sample size was 100k posts. The study did not incorporate images or comment text in their predictions.

Ohsawa and Matsuo investigated predicting user interaction for Facebook pages (2013). The study predicted paged likes based on the page description. The study creates a neural network of pages based on their entities. The page like prediction relies on the number of likes on similar pages. The model used like counts from Wikipedia pages for its prediction. The final model could predict Facebook page likes with a high degree of precision.

Text sentiment on social media is the subject of many studies. Liu performs opinion mining from social media (2012). Opinion mining works with keywords that are sentiment indicators. Existing sentiment lexicons are available for predicting sentence sentiment. Wang et al. focus on image sentiment analysis by clustering images by sentiment (2015). An unsupervised model trains on the clustered images. Image sentiment is also classified using sentiment banks. An existing model classifies object images. A sentiment bank uses the set of image objects to classify the image's sentiment.

2.0.2 ROI Studies

Fisher cites that ROI is the Holy Grail of social media (2009). Social media has a growing effect on consumer behavior (Fisher 2009). The study polled for user behavior. 34% of participants post products about opinions on blogs. 36% of participants better rate companies with blogs. Moreover, traffic to blogs increased 50% that year, compared to 17% at CNN, MSNBC, and the New York Times. 70% of consumers visit social media sites for information. 49% of the 70% buy based on social media content. 60% of users pass along social media data to other users.

ROI is difficult to track (Schacht, Hall, Chorley 2015). Most companies are unable to get revenue or cost savings from social media (Romero 2011). Romero calculates ROI for non-profits using the increase in service-use. Romero calculates service use differences between new and old users. However, Romero provides no ROI numbers. Schacht also measures ROI by user consumption. The study performed a cross-platform analysis of ROI on Facebook, Twitter, and Foursquare. Schacht proved that tweets can predict rising Foursquare check-ins.

Tiago polls which social media metrics marketing managers care about most (2014). The results are percentages of marketing managers who consider the metrics important. The most important metrics are brand awareness (89%), word-of-mouth buzz (88%), customer satisfaction (87%), user-generated content (80%), and web analytics (80%). Managers prefer metrics that promote engagement. Such metrics include page views, cost per thousand impressions and click-through rate. 18% of the surveyed companies plan to increase their investments in social media.

2.0.3 Other Research

Image analysis often includes denoising techniques. Using denoising creates an image that is easier to analyze. One such techinque is principal component analysis (PCA). PCA is especially helpful for image denoising. It works on image gradients. An image gradient is the change in pixel intensity from pixel to pixel. A change from black to white represents a large pixel gradient. Such a large gradient generally represents some kind of image edge. An image comprises many pixel gradients in many directions. PCA eliminates pixel gradient directions that contain smaller gradients. The result is an image that preserves strong contrast areas and edges.

Another important aspect of image denoising is simplifying colors. Colors are often simplified into a single pixel intensity. The final result is a single grayscale pixel intensity. This pixel intensity is the mean weight of the red, green, and blue channels (Kanan and Cottrell, 2012). Kanan and Cottrell explore other methods for calculating grayscale intensity. Kanan and Cottrell explore if other grayscale intensity calculations produce better image descriptors. The study used machine learning model performance to measure the grayscale method's performance. Kanan and Cottrell thought that the mean color intensity might misrepresent features. Kanan and Cottrell point to examples where color-blindness hide image details. By simplifying colors, the final image might lose important information. The study applies feature detection to images. The study ranks the quality of the produced image descriptors. The authors found that methods based on human brightness perception performed worse. Methods that incorporated a form of gamma correction performed best. One such gamma correction algorithm is the Gleam algorithm. This algorithm averages each color after applying a gamma transformation to each channel.

Lowe discusses how to create and collect distinctive image keypoints (2004). The paper presents the Scale Invariant Feature Transform approach (SIFT). The goal is to identify important image keypoints and describe these keypoints. The SIFT algorithm is special because it is invariant to size or rotation. The SIFT algorithm recognizes important keypoints. The algorithm detects these keypoints with image gradients. The algorithm determines keypoint direction of reference by its greatest gradient direction. The keypoint descriptor describes the gradients around the keypoint. The final keypoint descriptors are distinctive. Algorithms can match them to other images. Matching keypoints have a high probability of containing similar objects.

A related algorithm is the Harris detector (Derpanis 2004). The Harris specializes in corner-detection. The algorithms detect image corners with image gradients. The algorithm uses image contours and boundaries to identify corners. The algorithm applies Gaussian blurring to identify corners. The Gaussian blurring emphasizes edges by eliminating noise. The algorithm applies the second derivative to identify curves. The algorithm selects gradient curves as optimally stable points for keypoints. Matas et al. (2002) have shown that such features are maximally-stable regions.

Digital image correlation (DIC) is a technique for measuring shape, motion, and image deformation (Chu 85). The technique works by comparing and matching grayscale images from different views. The DIC technique employs a correlation criterion. This detects a best match group of pixels at some keypoint. Chu uses DIC in conjunction with SIFT to detect keypoints. Chu employs an algorithm known as iRANSAC. This algorithm detects false-positive matching keypoints from the SIFT algorithm.

Mandhyani et al. looks into techniques for classifying images (2017). Mandhyani et al. uses a bag-of-visual-words representation. The algorithm works by identifying objects in the image. The algorithm identifies objects with feature detection and keypoint clusters. The features map to databases of object features. The algorithm classifies objects by matching feautes to a feature databbase of objects. The algorithm performs classification with these set of objects. Each object, once identified, maps to words. The algorithm uses these bag-of-words to classify the image. The algorithm uses k-means clustering on image vectors. The created models can classify with accuracies from 50-75

This research explored synthetic textures. Synthetic textures are ways to transform images with convolutional neural networks. Algorithms can apply synthetic textures to images to style them. This research wonders if certain synthetic textures would cause advertisements to perform better. Aittala, Aila, and Lehtinen explore the current work on synthetic textures (2016). Studies apply synthetic textures to recreate images with modified color and patterns. Texture synthesis uses a base for its transformation. An image descriptor minimizes image differences, which produces the synthetic image. One large disadvantage is that textures must be manually created. They hold promise for transforming images. Yet, they might be too work-intensive for this study.

Generative modeling has shown promising results in texture synthesis (Li et al., 2017). There is generally a trade-off between efficiency and generality. Efficient algorithms often produce similar and non-diverse images. A goal is to perform synthesis with many textures using a single generative model. There are two types of texture synthesis: non-parametric and parametric. The assumption is that two images are visually similar when image statistics match. The synthesis procedures start with random noise. They gradually coerce an image to have the same relevant statistics. Alternatively, non-parametric models focus on growing an image from an initial seed. Li et al. uses neural network to create new textures with the generative model. This seemed less applicable to transforming social media images.

2.0.4 Convolutional Neural Networks

Studies use Convolutional Neural Networks (CNN) for image analysis. They have become a sort of standard within the realm of social media. Hassner uses CNNs in age and gender classification (2015). Chen et al. and Xu et al. produce visual sentiment classifiers with CNNs (2014). You et al. classified image polarity with CNNs (2015). Poria et al. detected sarcasm on twitter with CNNs (2016). Lin et al. identified stress within social media images with CNNs (2014). Sengalin, Cheng, and Cristani performed social profiling with CNNs (2017). Gelli et al. performed sentiment and estimated social media popularity with CNNs

(2015). Khosla, Das Sarma, and Hamid predicted image popularity with CNNs (2014). This thesis aims to improve upon these findings.

This study is different from others because it targets different social media metrics. Galli et al. have predicted which types of images are popular on social media (2015). Khosla, Das Sarma, and Hamid have predicted which posts will receive the most clicks (2014). This study examines the relationship between text and image on social media. This study also predicts social media metrics with text and images.

2.1 Current Theory and Practice

2.1.1 Computer Vision

Computer vision is the automated extraction of information from images. The most widely used and mainstream library for computer vision is OpenCV. Its authors implemented OpenCV in C++. Existing software wrap OpenCV in Python. Other well-known libraries include PIL, SciPy, pydot, and hcluster. These libraries support image blurring, resizing, grayscaling, image-clustering, feature detection, and more. ”Programming Computer Vision with Python,” ”Learning OpenCV Computer Vision with the OpenCV Library,” and ”OpenCV 3.x with Python By Example - Second Edition” explore these techniques.

Each of these books follows a pipeline in image processing. Each book takes an image and over successive chapters performs further processing. The ultimate goal is the extraction of information via image processing. Many of the subjects covered include clustering similar images. These can in turn classify image content. There are widely accepted guidelines for the initial steps in image processing. These include image resizing, denoising, and the creation of image descriptors. Resizing is generally performed to simplify later analysis.

Often neural networks need a fixed number of pixels for processing. It is also easier to reason about image similarities when images are the same size. Such reasoning would include geospatial specific information. These concern feature locations within an image. These include the location of keypoint descriptors, curves, and objects within an image.

Zelinsky implements seam carving (2009). Resizing can blur an image when increasing size. Reducing image size can reduce in lost image pixel information. Seam carving is a way of resizing an image without blurring more important pieces of the image. These more important pieces of an image include image gradient. The image gradient is the color intensity change of an image from one pixel to the next. This is important information when identifying curves and interesting points within an image. Seam carving is a dynamic programming, brute force attempt at discovering vertical and horizontal seams. These seams contain small amounts of gradient change. It performs this calculation by brute force discovery of seams across the image. These seams are those with the least amount of gradient change. The algorithm then duplicates or removes these seams to change the image size. The final result preserves the most interesting image points.

Denoising simplifies analysis. One way to denoise is to convert the image to grayscale. Grayscaling is often an averaging of the color pixels. This can reduce image size by a factor of three. Zelinskly demonstrates dilating and eroding images for image denoising. Dilation removes small bright regions but preserves and isolates larger bright regions. Eroding joins bright regions but retains their basic size. Both techniques help remove image intensity outliers. These transformations are helpful in emphasizing edges and general shapes within images. Other types of denoising include Gaussian blurring (Zelinsky 2009). Many images contain unimportant noise for image identification. Gaussian blurring can denoise a lot of this information. Gaussian blurring applies a 2D-kernel with some standard deviation. This kernel is then run through the entire image.

The process of applying kernels or filters to images requires explanation. When a CNNS apply kernels to an image, it is often applied at the pixel level, often to single pixels. The algorithm applies the kernel across the whole image. This changes pixel intensities across the image. These filters and kernels blur images or detect patterns within images. The filter determines the intensity of that pixel. The filter determines the intensity by examining its surrounding points. The algorithm uses the surrounding points to determine the pixel's intensity. In the case of Gaussian blurring, it includes pixels within some standard deviation. The algorithm uses the mean pixel value as the pixel's intensity. The larger the Gaussian standard deviation will consider more pixels. Gaussian blurring eliminates detailed noise from an image. The result of Gaussian blurring identifies edges and curves (Lowe 2004).

As seen with Gaussian blurring, a great deal of image analysis focuses on the big picture (Lowe 2004). Image analysis focuses on big changes within the image, curves, and edges. One of these big changes is the change in image intensity along an image. One unfortunate challenge is that this intensity change can occur in many directions. The image intensity variance requires massive amounts of computer resources and time. For example, a 100 x 100-pixel grayscale image has 10,000 dimensions. Each dimension has its own variance of pixel intensity. One way standard practice simplifies this is with Principal Component Analysis (PCA).

Principal Component Analysis is a useful technique for dimensionality reduction. It is useful for simplifying an image into the most important dimensions (Lowe 2004). PCA can order these dimensions in the order of importance. One can then choose to use the most important ten or twenty dimensions. This preserves the most important information while discarding superfluous dimensionality. This also decreases the time it takes to discover interest points within images. Interest points within images are also known as keypoints. Keypoints are places where interesting things are occurring within an image. Mainly, keypoints are places within an image where edges meet. These are also points where color intensity varies most. Many algorithms exist to find these keypoints.

Image descriptors describe the image at particular keypoints. These descriptors record pixel gradients around the image. Descriptors are also invariant to rotation. The rotation chooses the direction containing the greatest image gradient. All other points are in reference to this direction. This is extremely convenient when comparing image descriptors between images. It makes it simple to find images containing similar features. These image descriptors can find similar objects from other images. These findings are very accurate. Other applications include clustering images with similar content or geolocation of content. ”Programming Computer Vision with Python” includes an example of collecting many images based on their geolocation and then using image descriptors to create a cluster of other images of the white house.

Image clustering is another common topic across these books. Clustering helps with recognition, dividing data into sets of images, and organization. A common implementation of clustering is k-means clustering. The algorithm is like k-nearest-neighbors, but unsupervised. The unsupervised algorithm begins with k randomly distributed points. The algorithm calculates each image’s centroid using averaging of pixel intensities. The algorithm assigns images to their nearest cluster. The cluster’s location is then updated to be the average of all keypoints assigned to that cluster. Sometimes the algorithm is rerun a few times. This is because the initial selection of cluster points affects algorithm convergence. This classifies each image in one cluster.

A similar clustering algorithm to k-means is hierarchical clustering. The main idea is to build similarity trees based on pairwise distances between keypoints. Each added node finds its closest pair. The algorithm extracts clusters traversing the tree. It stops at nodes with distances smaller than some threshold. Another type of clustering algorithm is spectral clustering. Spectral clustering takes a similarity matrix for each image and constructs image clusters. The benefit is that the algorithm can construct similarity matrices cwith any criteria. A complicated Laplacian matrix formula uses the similarity matrix. The algorithm can then calculate eigenvectors. These eigenvectors can cluster images.

”Learning OpenCV Computer Vision with the OpenCV Library” covers common image descriptor algorithms. All these rely on, in one way or another, the first and second derivative of image gradients. The Laplace and Canny methods rely on the second derivatives. This makes them good at identifying curvature within images. Hough transformations are helpful at finding simple lines and curves in an image. The SIFT algorithm is also used to discover image keypoints and their descriptors. Many of these algorithms are not open source, but there exist open-source alternatives. Surprisingly, OpenCV includes built-in machine learning algorithms for image processing. These take feature vectors as arguments and apply these algorithms. The included algorithms include k-means, Mahalanobis, Naive Bayes classifier, Decision trees, Boosting, Random trees, Haar classifier, Expectation maximization, and Neural Networks.

2.1.2 Natural Language Processing

Some of the most popular NLP libraries include NLTK, TextBlob, Stanford CoreNLP, spaCy, and gensim. TextBlob is a quick tool for NLP prototyping and has simple tools for sentiment analysis. The creators of Stanford CoreNLP wrote it in Java. There are Python wrappers for Stanford CoreNLP. It includes part-of-speech (POS) tagging, entity recognition, pattern learning, and parsing. Spacy is a great general NLP library that largely replaces NLTK. Genism is a library for topic modeling and document similarity analysis.

"Natural Language Processing with Java" explores some of these libraries. The books also present a pipeline for natural language processing. There is a greater breadth of research on this topic. Much of which largely agrees with basic NLP practices. These practices include text processing tasks and building NLP models. Text processing tasks include finding parts of the text, finding sentences, finding people and things, detecting parts of speech, classifying text and documents, and extracting these relationships (Morgan 2015). Current algorithms optimize this process for many general use cases. Finding parts of the text is synonymous with identifying tokens. The classification of words can include simple words, morphemes, prefixes and suffixes, synonyms, abbreviations, acronyms, contractions, and numbers. Other techniques include stemming and lemmatization. They can break down words and find roots.

Finding entities (people and things) can be important for identifying major concepts and subjects in the text (Morgan 2015). Entity recognition can complete this task. Tokenizers and entity sets help with this identification. There are problems with misclassifying objects due to ambiguities in language. Some techniques rely on lists, regular expressions, or use trained models to detect the presence of entities. Another way of classifying parts of the text is by detecting parts of speech. Parts of speech tagging (POS) is useful for extracting relationships between words (Morgan 2015). These can be helpful in determining the meaning of the text. Parsing is the process of determining these relationships. This is within the realm of text analytics. This has difficulties. Many tags may associate with more than one tag. Such associations could be independent of their context. Classifying documents can then aid with sentiment analysis. It is common to use supervised training to create sentiment analysis models.

2.1.3 Feature Extraction

”Feature Engineering for Machine Learning” covers a lot of current theory and practice in feature extraction. Feature engineering is the process of extracting features from input data. These features are usually a numeric representation of the data. These engineered features are then suitable for machine learning models. Feature engineering is a crucial step in building any machine learning pipeline. Good features tend to be simple and make are easy for the model to ingest (Casari 2017). Bad features may require a much more complicated model to achieve the same level of performance. More complicated features tend to consume more training time and tend to have diminishing returns. Also, irrelevant features make the model more expensive and trickier to train. This leaves an opportunity for odd things to occur and ruin the model’s performance.

The number of features is also important. Few features might not contain enough information to model the task at hand. Too many features can largely overfit the training data and cause poor performance on similar data sets (Casari 2017). It is important to be cognoscente of the magnitude of numeric data. For models like Relu, not normalizing the data could lead to frequent false positives. With linear models, when input data varies by factors, the model might end up extrapolating. With K-nearest neighbors, when measuring Euclidean distance, large data might cause misclassification. Controlling the input data will prevent these situations. This control often takes the form of normalizing the input data. Though, in the case of decision trees and random forests, these decipher value boundaries and are not sensitive to the scale of the data.

In a more interesting example, linear models often assume that the error is normally distributed among the values (Casari 2017). Yet, when the order of magnitude varies by multiple factors, this assumption likely does not hold true. Log transformations can control the magnitude of growth. These largely reduce the degree of magnitude and can make inputs more normalized. A popular form of feature extraction is the combination of inputs into single features. At an extreme, features may be the output of other machine learning models. This concept is model stacking.

Feature pruning is another common technique (Casari 2017). There are ways of automating some types of feature pruning. Some of these seek to eliminate inputs least correlated with the model's output. Others seek to eliminate the variables that most reduces standard error. Others seek to compare the model with and without the feature to see if the feature is statistically significant.

The input to a model is often represented as vectors (Casari 2017). More often than not, these vectors might be extremely sparse. Often, the model can transform scalar values into binary values. These represent the presence or absence of the value. One other interesting technique is the intermediate application of naive Bayes to combine sparse vectors. Feature engineering can represent a combination of events as a binary value in a smaller vector. In this way, a smaller vector can represent a combination of events from very large sparse vectors. Models can miss input correlation if their combination never occurs in the training data.

Binarization is another way to deal with counts (Casari 2017). Bins can simplify input data into fewer categories and still retain a lot of the magnitude information. This can aid in models like k-nearest-neighbors because binning helps control the magnitude of the data. Exponential binning is often applied to largely varying magnitudes of data, while still retaining the same bin size on the main axis. Binning can also reveal data skews for how many instances of each bin type occur. One general conclusion is that visualizing the relationships between input and output data is always important when building models.

Types of feature scaling exist. One of these is min-max scaling. This essentially squeezes all the values into the range from 0 to 1, without normalizing the values. The great part about min-max scaling is that it doesn't shift the data to be closer, i.e. take a sparse vector and create a dense vector. Since many models perform well and optimize on sparse vectors, this can help with model performance and training times. Example of this might include bag-of-words, where sparse inputs are useful for sentiment classification. Standardization is similar to min-max scaling, but it divides by the variance. If the original distribution was Gaussian, this transformation will output the standard Gaussian. L2 normalization, also known as L2 scaling, is a technique to control the magnitude of the input data by dividing by the data's magnitude.

The book also touched on classification using bag-of-words and tf-IDF. Bag-of-words can be creates using n-grams. Yet, most n-grams are only of size two to three, because of the space complexity of larger n-grams. Surprisingly, it is also helpful to filter out rare words. Many of these rarer words only appear in one or two documents are more noise than helpful. These rare words tend to be unreliable as predictors and also generate computational overhead. Tf-IDF uses a normalized count. Each word count divides by the number of documents the word appears in. This creates more normalized numerical data and is faster for training models.

2.1.4 Deep Learning with Images

Regularization is an important way to prevent overfitting in deep neural networks (Buduma 2015). There are multiples types of regularization. One popular type with stochastic gradient descent (SGD) models change the objective function by adding additional terms that penalize large weights. In other words, the objective function becomes an additive function of the error and the original objective function. This is similar to lasso and ridge regression in linear modeling where the goal is to minimize the vector weights. The common type of regularization is L2 regularization. It adds the square of the weights to the error function. L2 regularization has the effect of heavily penalizing large peaks in weight vectors, instead preferring spreading out the vector weights. L1 regularization adds a function of the weight to every weight in the neural network. This causes very sparse vectors, which implies that only the most important nodes are being used as outputs.

Minibatching is a common technique for training (Buduma 2015). Singular inputs become the average of a batch of values. This input is then fed to train the network. This ensures that incoming data is more averaged and robust. Hopefully, the transformation prevents model confusion from outlying data.

Input data and its types will control the model's architecture. Some models do classification, probabilities, or even regression. The number of hidden layers is important within deep networks. MNIST only needs three or four hidden layers, but Facebook's DeepFace uses nine hidden layers. The size of the dataset helps determine this. The number of neurons in progressive layers should also shrink. No hidden layer should have fewer than a quarter of the input layer's nodes. For training, a good initial idea is to remove parameters (or neurons) to solve overfitting issues. Practice tends to use more neurons than necessary (Budama 2015).

For convolutional neural networks, setting stride above two is uncommon. Stride determines the number of skipped pixels between filters (Buduma). Often it is best to pad the input volume with zeros in order to not change the spatial dimensions of the input. This also preserves border information. The general practice uses fewer filters near the input. As the convolutional neural network progresses towards the output there tends to be more filters. The example in the book moves from sixty-four filters too well over a thousand filters near the output. Also, smaller filters in multiple stacked layers tend to perform better as opposed to big filters in fewer convolutional layers.

Chapter 3

Research Gaps and Questions

3.1 Research Gaps

There are research gaps in predicting user engagement on social media. The gaps include which models perform best. The gaps include how well models can predict user engagement with image and text data. The analysis uncovers which models perform best at predicting user engagement. The research delineates which data is most helpful for machine learning.

3.2 Research Questions

This thesis introduces the following high-level question: Can text and image data be used to predict user interaction on social media via machine learning? Further, we introduce the following hypotheses:

1.Text-based data can be used by machine learning models to predict user interaction on social media with a mean squared error that is less than the distribution's variance.

2.Image-based data can be used by machine learning models to predict user interaction on social media with a mean squared error that is less than the distribution's variance.

3. Text and image-based data can be used by machine learning models to predict user interaction on social media with a mean squared error that is lower than the mean squared error produced by either the text or image-based machine learning models.

4. Text and image-based machine learning models can statistically outperform random models when predicting for greater user interaction between Facebook advertisements.

Some of the reserach questions are with reference to the distribution's variance. This relates to model training. Models perform regression and train with a loss function. The study used mean-squared-error to calculate model loss. The formula for the mean squared error is similar the variance formula. The variance is the mean squared error with respect to the mean. This makes the variance a good baseline for understanding model performance. A model with a loss less than the variance is performing better than predicting the mean.

Chapter 4

Research Methodology

4.1 Data Context

The data context consists of an online post and its metadata. The general form of content that users interact with is the online post. Posts generally contain text and might also contain an image. These posts also contain metadata. Metadata about these posts consists of likes, shares, reactions, tags, and timestamp data. A lot of this metadata are ways users have interacted with the post, e.g. by manually leaving a comment, like, reaction, or sharing the post with other Facebook users (Azizian 2017). The collected Facebook post data was drawn from advertiser Facebook pages. These pages include page data like ratings, the number of followers, and the number of persons actively speaking about the page. The combination of page data, its posts, the post data, and its comments were all collected for the study.

It should be noted that not all of these interactions are publicly available, as they are

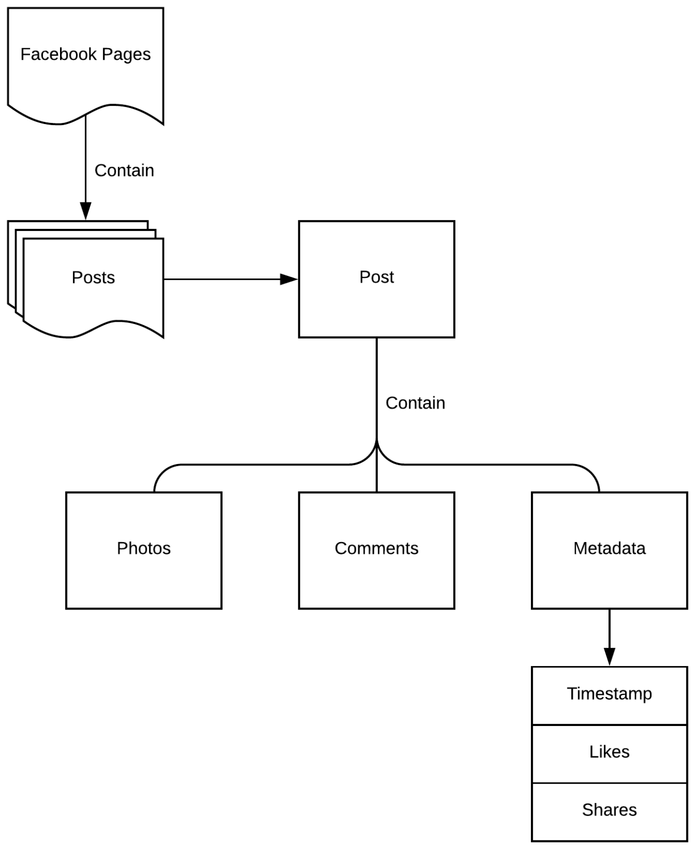


FIGURE 4.1: Data Extraction Pipeline

restricted by Facebook's privacy policy (`https://www.facebook.com/policy.php`). Restricted interactions include likes and reactions. User Facebook pages are by default restricted. This left the app without access to user data.

4.1.1 Data Origin

The Facebook post URLs were obtained from AdEspresso. The website is owned by Hootsuite, a social media management platform that was created in 2008 (Hootsuite). The company manages many platforms, including Twitter, Facebook, Instagram, LinkedIn, Google+, and YouTube (Hootsuite). The AdEspresso website provides a management platform for social media advertising. It allows marketers to create and promote advertisements. The website pushes these advertisements to multiple social media platforms. The result is a single place where marketers can create and promote advertisements. The website also offers consulting for creating successful advertisements. The website features over one-hundred-thousand demo Facebook advertisements. This paper scraped these demo advertisements.

4.1.2 Collection

The research created a Python web scraper to crawled through these Facebook pages. All Facebook crawling was done with the Facebook's publicly available graph API. Each Facebook page contained anywhere from a dozen to a few thousand posts. A maximum of one-thousand posts was scraped from each page. The maximum of one-thousand posts was chosen in order to control the machine learning bias to any one Facebook page. The maximum was also chosen because Facebook limits the number of API requests per day per Facebook page. Requesting content for 1000 posts and their comments can reach this limit. The full number of posts collected was over three-hundred thousand. The image URLs are stored, rather than downloaded, to save space. The text data is collected and stored in a database. Figure 4.1 shows the entire data extraction pipeline. Figure 4.2 shows a histogram of the number of posts scraped from each Facebook page.

The Facebook API is simple to use decode. Moreover, the API is hierarchical, so that the URL can be extended in order to access children elements. For example, from a Facebook page, its posts can be accessed by appending "/Posts" to the URL. From a post URL, its comments can be accessed by appending /Comments to the URL. This made collecting data easy and less error-prone. The entire URL is also a unique hash to the page's

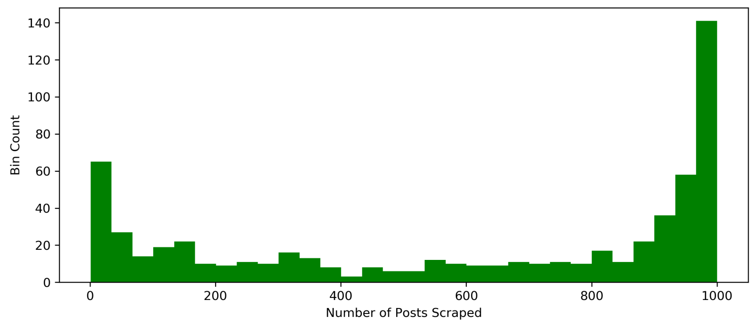


FIGURE 4.2: Histogram of the Number of Posts Scraped Per Facebook Page

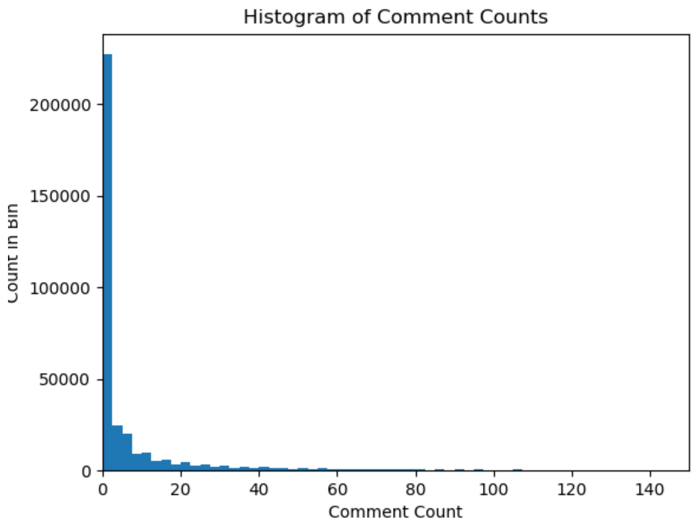


FIGURE 4.3: Comment Count Histogram

element. This makes the URL a useful primary key in the database. The sample size for share and comment counts is about 350k, and 50k samples for comment sentiment. Below are histograms in Figure 4.3-4.5 of the user interaction metrics collected for this study. The histogram sample sizes were chosen to help visualize the data.

4.1.3 Analysis Methods

Gathering data included image processing. Part of image processing is denoising images in order to emphasize important image features. The features deemed important vary for each denoising algorithm. Generally, the denoising algorithms emphasize edges, gradient

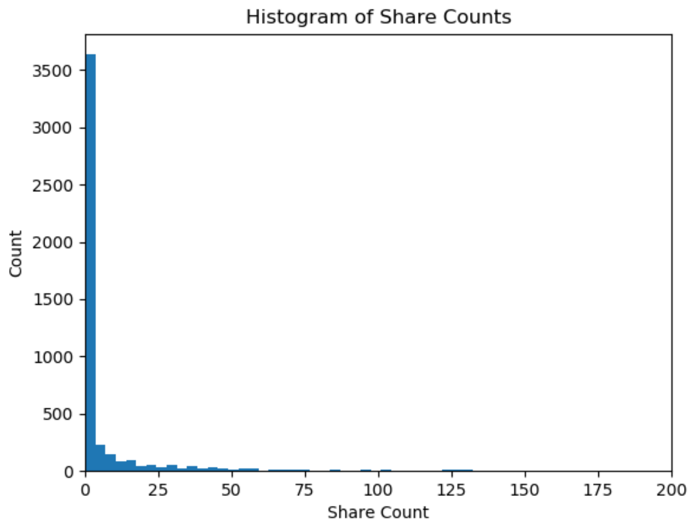


Figure 4.4: Share Count Histogram

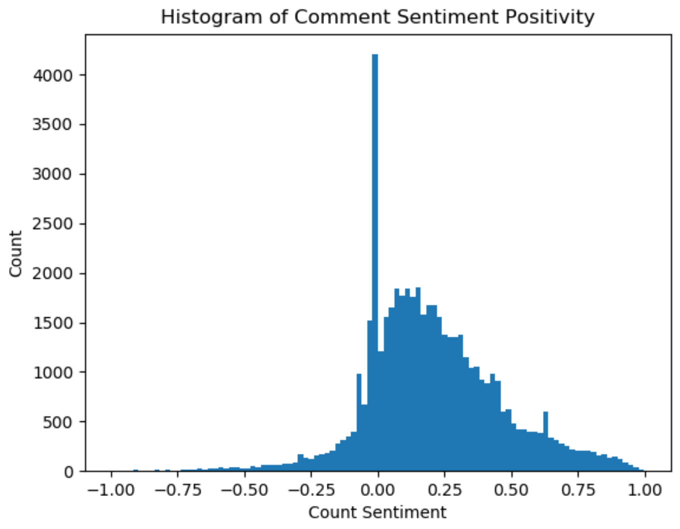


Figure 4.5: Comment Sentiment Histogram

contrasts, and curves (Derpanis 2004). One result of denoising can be a decreased in the amount of image data, which saves computer space. The reduction in image size makes storing all the images on a machine more practical. While reducing data size is helpful, the ultimate goal is not reducing image size. The ultimate goal is eliminating image noise that might interfere with image-based model training. By eliminating noise, the models can focus on image aspects that denoising has emphasized.

Part of denoising is preparing the image for analysis. Many of the machine learning models require for all images to be the same size. Therefore, reducing image size is important for the final machine learning model. One goal is to reduce image size without losing important data. One way to reduce image size while preserving important features is via seam carving (Zelinsky 2009). Seam carving is the process of resizing an image by duplicating or removing the least descriptive parts. The least descriptive parts are determined by finding seams across the image where the gradient changes the least. This seam is duplicated or removed as a way of changing the image's size. This process is a very intensive computer operation. The conclusion is that this operation was too time intensive. A fifty-pixel increase or decrease in size required a few minutes. This is far too much time when considering hundreds of thousands of images. A less sophisticated resizing method from OpenCV was instead used. Python contains a wrapper around the OpenCV C++ library for image processing. The cv2 library performs image resizing and grayscaling. The grayscale image is still noisy and needs to be processed.

A step to eliminate unimportant data is to reduce the number of image dimensions via dimensionality reduction. This can be done via principal component analysis. The number of dimensions to keep was set at 20 since the image variance is lower after denoising. This reduces noise, and data size.

Image denoising can further reduce noise. The most common type of image denoising is a Gaussian blurring. This applies image blurring and emphasizes edges. The research applied a Gaussian blur with a standard deviation of five to each image. A dilation and erosion are also applied to the image.

Basic feature detection was then applied to each image. The algorithm detects image keypoints. These keypoints are based on an image's grayscale intensity gradient. These keypoints were stored within a database and used to find images with similar keypoints. The final image data consists of denoised photos and their keypoints descriptors.

4.2 Text Processing

Neural networks take feature vectors as input. Yet, plain text data can not be processed by the neural network. Therefore, the research must transform the text data into vectors that are usable by the neural network. The process for transforming the text data into a word vector follows examples from ”Deep Learning with Keras” (Antonio et al., 2017). The program split the text data into words using whitespace as a delimiter. Word tokens are create from the split strings. The program grouped these tokens into sentences. The words are then lowercased and the stopwords are removed. Words with length at or below three are removed. A port stemmer is used to create stems for all the words. A POS tag library was used to perform parts-of-speech tagging. The program then extracted stems with a word lemmatizer, which takes as input the stem and the POS tag. The program fed the lemmatized text sentences to a td-IDF vectorizer. This created word vectors. The word vectors form the features for the neural network.

4.3 Open Sourced Data

The code and data have been open-sourced. The scripts have been written in Python and stored on GitHub at `https://github.com/cpluspluscrowe/Success-Predictor-for-Social-Media-Ad` Hopefully, this encourages data access and use. The raw data has been included too, which should enable readers to rerun and duplicate experimental results from their computer. The desire is that making the code available allows other researchers to adopt, replicate, and expand upon this paper's research.

Chapter 5

Model Methodology

5.1 Data Verification

With the data collected, this research can move to the next step of building the machine learning models. This research used the collected data as model input. The input features consist of transformed text and image data. The transformed text data are the word vectors and processed images. The goal is to use the word vectors and processed images as model features to train the machine learning models.

This research tested inputs features to maximize training results. Word vector lengths from 100-100,000 are tested on text-based NNs. Word vector sizes of 10,000 produced the lowest losses. This word vector size is used throughout the study. Rectangular image sizes from 30 to 360 pixels are tested. Similar training losses on CNNs are produced for 60x60 images with regards to larger images. 60 pixels produces a CNN that trains far faster, so this size is used in all CNN models. The features were tested on smaller subsets of data that ranged from 20k-50k images. Smaller training sets were used since they are faster to train.

Once the features are selected, the model's size and characteristics can be trained. The characteristics of a model are known as hyperparameters (Casari 2017). Badly tuned hyperparameters might produce models that do not train well (Casari 2017). For example, a large learning rate or momentum could hamper a model's ability to converge to the solution. A high number of hidden layers could cause training overfitting and poor performance. A lack of regularization could also lead to model overfit or an odd distribution of weights within the neural network and correlated inputs. A large filter size for convolutional neural networks can cause a failure to extract image features.

Each Facebook page has auxiliary page metrics that describes how many persons follow the page and its popularity. This research used linear models to test the usefulness of these features. The wonderful thing about linear models is that they are much faster to train. The better features will increase model performance. The quick feedback is useful for identifying performant features. Yet, the Facebook page metrics were not strongly correlated with user interaction. Due to the low correlation with user engagement, the Facebook page metrics were not included as input features.

5.1.1 Approach

The goal is feature extraction for machine learning. This application includes the scaling, normalization, and categorization of data. After processing the input data, it is possible to create basic models. These models may range from linear to random forests to convolutional neural networks. Certain data are better processed by particular models (Krizhevsky 2012). For example, convolutional neural networks work well with images. Text-based models can represent Text by the frequency with bag-of-words or tf-IDF. The model can use word vectors to predict with Naive Bayes. The general approach will be to create simple features from input data. Likely, creating features from machine learning models. This model stacking can create simple features. Other models can incorporate these features. The general goal will be to keep models and data simple. Simple data is easier to reason about, and simple models process data faster. The described verification techniques will produce and verify better features.

5.2 Data Mining

The program utilized the Facebook API to perform data mining. This data mining required creating a Facebook app, obtaining tokens, and querying the data. The tokens are Facebook's form of authentication. The token is necessary for all Facebook queries. This allows Facebook to control content access. The application requests are throttled on an hourly and daily rate, so as to limit how many API requests occur at once. The API also throttles based upon the number of users on the application. The application created for this study was not for public use, so it had more strict API limits.

The application to collect AdExpresso links is Selenium based. It navigated the website by executing JavaScript to navigate to new pages. The Selenium application collected advertisement links. The website is not navigable via URLs, and all requests have authentication tokens. The application manually navigated through one-thousand pages on AdEspresso. The application stored the list of visited pages and skipped these whenever the application crashed. The application pickled these page numbers into a shared file and up to 20 Selenium instances were ran at once. The database only stores unique ad links, so duplicate urls causes no harm to the data. Examples of the pickle files are on GitHub with the .pkl file extension.

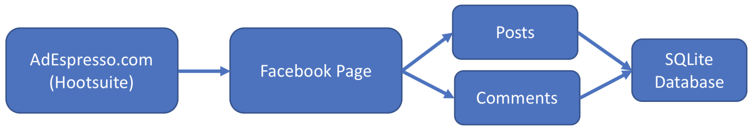


Figure 5.1: Data Collection Pipeline

The next application converted gathered Facebook links from Expresso pages. The first application gathered links to Expresso pages, which contain Facebook links. This application visited those pages and extracted the Facebook page links. This application did not rely on executing JavaScript. Instead, Python's urllib library retrieved the data. Python makes requests, which return the page data. The Python program then parses the request's HTML data. The program extracted links from the HTML page. Requesting is largely I/O, so this researchers threaded the application. This application was able very stable and would run for days. The application obtained 281,090 unique Facebook page links. Figure 5.1 shows the data collection pipeline.

Once the application obtains Facebook links, it can scrape the Facebook pages. The third application ran through all the Facebook URLs and scraped the post data from each Facebook page. Collected post data included the post's text, image, share count, and the number of comments. The program stores the data in a database. This is the brunt of the application and represents most of the data in the analysis. The only data missing were the Facebook comments for each post. The last Facebook post data collection application obtained comment data for each Facebook post. The application requested the comment data for each Facebook post. The comments were then stored on a database. The machine learning models used this data to learn the training data. Figure 5.2 depicts the inputs to the machine learning models.

Mark Zuckerberg testified before congress concerning the Cambridge Analytica data breach while this research was collecting Facebook data. Facebook's immediate response

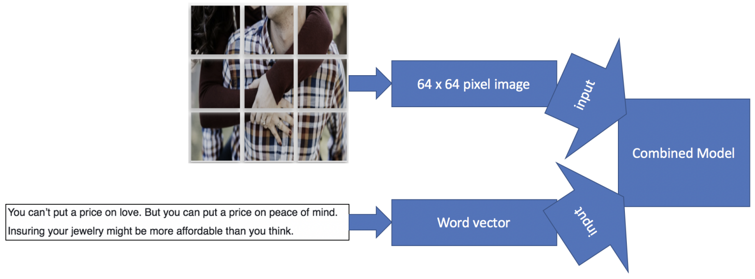

64 x 64 pixel image
input
Combined Model
You can't put a price on love. But you can put a price on peace of mind. Insuring your jewelry might be more affordable than you think.
Word vector
input

FIGURE 5.2: Combined Model Inputs

was to permanently disable API tokens and severely decrease the API limit threshold. These changes became large hurdles for this research to overcome. This research spent a great deal of time retrieving temporary API tokens via selenium and working with new API limitations. The study was able to overcome API throttling through threading. A single thread-safe queue stored URLs to scrape. Threads would pull from the thread-safe queue, perform work, and sleep if they hit API limitations. This allowed applications to continually run and process data as Facebook allowed. The application found that fifty threads were a good balance of scraping enough data while avoiding request throttling.

The application ran into other issues. Sometimes requests returned characters that SQLite could not store, though SQLite supports UTF-8. All these issues caused the application to run many times. The program incorporated a lot of logic to skip over already-processed data. Many of the data database layers would run queries to only return unprocessed Facebook links. Each application had this preprocessing layer to prevent re-obtaining old data. Moreover, if the program reprocessed old data, the storing algorithms included the insert or replace queries to exclude duplicate rows from forming within the database.

The Facebook post data included image URLs. One of the applications downloads these Facebook images. This program utilizes the urllib Python module to retrieve the image. The library implements image-processing in C and is very fast and even thread-safe. Python releases the GIL (Global Interpreter Lock) on I/O operations. The allowed the request to perform multithreading. This makes collecting downloading many images possible in Python. Unfortunately, these images are large and require too much space. The process of downloading all the Facebook post images is impractical from a space perspective. Rather, the program must download each image. The program must immediately be processed and only store the necessary data. This process consists of three scripts. The scripts downloaded the data, denoised the data, and obtained keypoint data.

The study collected and stored user interaction data in an SQLite database. This made it possible to SCP the database between servers. The stored analysis data included Facebook post and comment data. This research stored the Facebook post data in a flat table. It included basic post information like the image's URL, UUID, and the post's text. The Facebook post table also included other auxiliary data so that all the data could be easily fed into a machine learning model without any joins. Some auxiliary data includes page metrics like fan count, the number of page ratings, the overall average page rating, and the number of persons talking about the page. Other calculated metrics within the table include comment count and the post's text subjectivity. This research generated the sentiment metrics using Python's Vader library. The library performed on social media posts. The comment table was simpler and only stored the comment's text and its Facebook post id. The models trained on this data.

The study performed initial statistic analysis on the user interaction data. Researchers used R programming to measure the data's distribution. Comment and share count closely fit a gamma or Weibull distribution. Figure 5.3 depicts some basic statistical fits to comment counts and the gamma distribution. Comment sentiment was similar to a normal distribution, as seen in Figure 5.4.

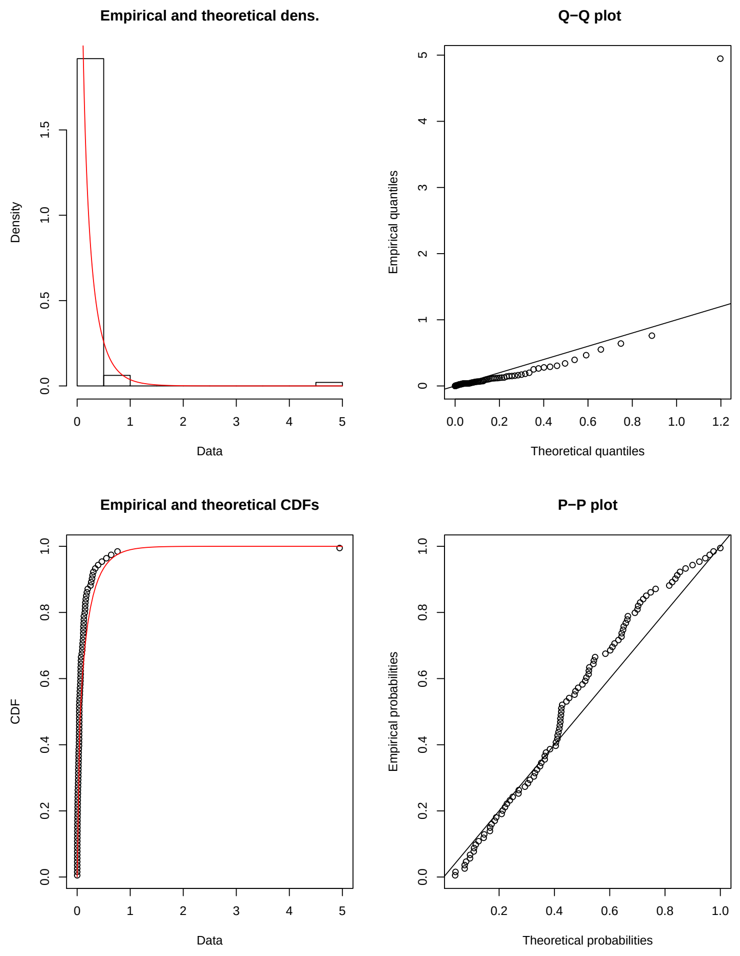


Figure 5.3: Statistical Graphs for Comment Counts vs Gamma Distribution

5.3 Sentiment Analysis

This research used existing sentiment analysis tools to calculate sentiment for the post and comment data. Initial research began with Python's Blob library but soon switched to using the VADER Python library. VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media and works well on texts from other domains (Hutton et al., 2014). The library calculated polarity scores for the post and comment data. The comment data's polarity scores were the output for the comment sentiment model.

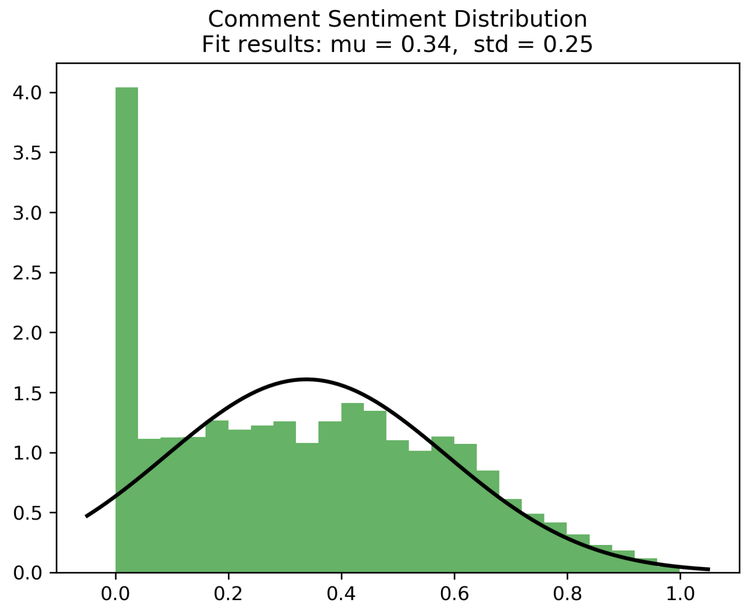


FIGURE 5.4: Positive Comment Sentiment fit to a Normal Distribution

5.4 Algorithm Reliability

Standards exist for measuring model reliability. Model reliability describes how well the model can train, learn, and handle data variances. A more reliable model will be well suited for its data. A well suited model is quicker to trains and more sturdy to variations in the data. K-fold cross validation can measure model performance and sturdiness to small data variations (Casari 2017). This procedure trains k models, where each model uses one of the k groups as the test dataset, and the other nine as the training data. The model performance is averaged across all k data sets. Using k-fold cross validation demonstrates that the model can generalize its predictions and learn on with different sets of data. Methods like cross-validation help with data quality and prevent overfitting. This experiment used a 10-fold cross-validation on all NN and CNN models. Model training will also include types of regularization. The dropout rate also regularizes the machine learning models. A dropout of 0.35 to 0.1 is incorporated after each CNN or NN layer. Type L1 or L2 regularization can change neural network weights to prevent overfitting (Srivastava 2014). The NNs incorporate L2 regularization on each dense layer. The use of cross-validation and testing final performance on a validation test set largely ensure model reliability.

Keras provides APIs for model k-fold cross-validation. Keras is an open-source machine learning library (Keras). The models use default train-test splits. Models train to 100 epochs. The models in this research tended to begin overfitting after 30 epochs. The training began with larger batch sizes of 256. The program retrained with a larger batch size every 20 epochs. The batch sizes used were 256, 128, 64, 32, and 1. Regularization was also employed. Each NN layer includes L2 regularization. The L2 regularization had a weight penalty of 0.01. This penalty is also known as weight decay, or Ridge (Keras). Each layer also included dropouts. The dropout ranged from 0.4 to 0.1. The general pattern began with a large dropout and decrease dropout through later layers. Despite this regularization, the models eventually overtrain. Future work should include adding L1 regularization.

The process evaluates the model with the validation set. This set is a separate data set from the training and test data. The API used this dataset to check model performance. In practice, it is possible to overfit to the test set by modifying and selecting the models which have the highest test accuracy. Unfortunately, this is likely overfitting to the test set and does not represent true performance. It more accurate to test the model on a new set of data. This research set aside a third validation data set to test the final model's performance.

5.5 Evaluation Criteria

Machine learning often takes one of two approaches, classification or regression. This research created regression models since the metric outputs were continuous. The values it predicts are share count, comment count, and comment polarity. The model predicts how much of these metrics the advertisement will receive. The prediction leaves advertisers with a better sense of how well their advertisement should perform on social media. The regression numbers also demonstrate that the model learned the data. Thus, the regression serves as a check to see if the model was successful at learning the data. The binary prediction for which of two ads performs best is a practical way to measure model performance.

There are existing benchmarks for predicting user interaction on social media (Straton et.all 2015). Yet, the study performs classification, not regression. The study split the Facebook posts into three ordinal groups, based on how much user interaction they received. The study then trains to classify the data into these groups. The study had a very high accuracy predicting for low user interaction. This outcome matches the zero-skewed user interaction distribution. The study did not show that their results were better than random guessing. Moreover, the classifier performed very poorly on predicting posts with more user interaction (below 10%). This study will serve as a new benchmark for performing regression on user interaction. The study will also compare the created models with a random model, to prove if the models created from this study outperform random guessing.

Although the model can predict reactions to the post, advertisers are more interested in using such a model to enhance their decision making. The models can aid advertisers to predict which of several advertisements will be the most profitable. The general assumption is that advertisements that receive more shares, comments, and more positive comments will produce greater profits than advertisements which elicit lower user interaction metrics.

This research explores how well the models can enhance decision making between several advertisements. The models are used to predict user interaction. This research measures how often the models correctly predict that one advertisement will outperform the other. This experimentation was carried out on a validation set that is mutually exclusive to the training and test data sets. Each model predicts three metrics for each advertisement: share count, comment count, and comment polarity. Advertisements were separated into combinations of size two for the comparison. The model score, for each comparison, is the total number of correct guesses divided by the total number of guesses. A successful model will perform statistically better than random guessing.

5.5.1 Experiments

This research includes thirteen experiments. The experiments seek to predict user interaction quantities. The to-be-predicted user interaction metrics are share counts, comment counts, and comment sentiment. The models are a text-based neural network, an image-based convolutional neural network, a decision tree, Word2Vec models, and a model that combines the text-based neural network and image-based convolutional neural network. Four of the models train to predict the user interaction metric. Each of the four models predicting three metrics makes twelve experiments. The final experiment consists of three Word2Vec models that try to predict comment sentiment. Four models to predict user interaction and one Word2Vec model to predict comment sentiment makes five total models and thirteen total experiments.

5.5.2 Word2Vec Experiment

The Word2Vec models required a different setup. The purpose was to train models on posts that received positive comment sentiments, so that the models could identify new sentences that would also received comments with positive sentiments. The study separated posts into three groups. The groups were those with positive (x >= 0.7), neutral (0.7 > x >= 0.3), and negative comment sentiments (x < 0.3). Each group trained a Word2Vec model. This research evaluated the model score with test data. Each model predicted the likelihood a post fell into its sentiment group. The model with the highest likelihood classifies the test post. The model's performance was scored based on how accurately it labeled posts into sentiment groups. The final score is the number of correct sentiment classifications over incorrect classifications.

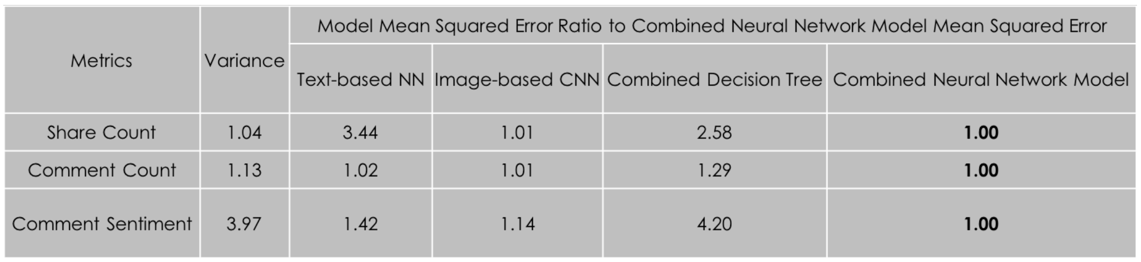
Chapter 6

Results

6.1 Summary of Findings

This research includes five types of machine learning models. The five models are text-based neural networks (NN), image-based convolutional neural networks (CNN), Word2Vec, decision trees, and a combined NN, CNN model. The models train to predict user interaction. The user interaction metrics include the comment count, share count, and comment sentiment. The experiments use different sized data sets. Comment and share count metrics utilized 350k posts in training and testing. Comment sentiment experiments include 50k posts. The code specifies these experimental sizes. The comment sentiment data set is smaller due to its greater complexity. It takes more times and more requests to gather post comments. Facebook also limits the number of API requests per Facebook page. Word2Vec models use the Python gensim library (Radim et al., 2010). The 50k comment sentiment data set was not large enough to train the Word2Vec models. Since the models did not have enough data to train on, the models predicted 0% similarity on all new sentences.

The remaining twelve models trained with the Python Keras library. The models



Metrics	Variance	Model Mean Squared Error Ratio to Combined Neural Network Model Mean Squared Error			
		Text-based NN	Image-based CNN	Combined Decision Tree	Combined Neural Network Model
Share Count	1.04	3.44	1.01	2.58	**1.00**
Comment Count	1.13	1.02	1.01	1.29	**1.00**
Comment Sentiment	3.97	1.42	1.14	4.20	**1.00**

FIGURE 6.1: Model Mean Squared Error Reported as a Ratio to the Combined Neural Network Model's Mean Squared Error

predict continuous variables, which requires the model to perform regression. A loss function evaluates the regression models. The loss function is a way of explaining how well the model fits the target data. The loss value used to train these models is the mean squared error. Keras recorded a few other loss functions. These functions include mean absolute error and cosine proximity. Figure 6.1 displays the results of the twelve experiments. The results include the variance. The variance describes the mean squared error, if each predicted value was the mean. If the mean squared error is below the loss, then the model is doing better than always predicting the mean.

Figure 6.1 displays results as a ratio to the combined model's loss. Due to this, the combined model's mean squared error (last column) displays 1. This is done because the mean squared errors are very large and difficult to compare between models. Leaving the numbers as a ratio makes comparing between columns easier.

The combined model outperformed all other models on all metrics. The combined model attained a loss less than the variance. On many metrics, the combined model far outperformed the NN and CNN. The combined model performed best on predicting comment sentiment. The combined model performed the worst on predicting share count. The combined decision tree performed poorly on all metrics. It is interesting that the CNN outperformed the NN on all metrics. The models emphasized this performance on metrics that are more difficult to predict. For example, model loss on share count was much higher. The

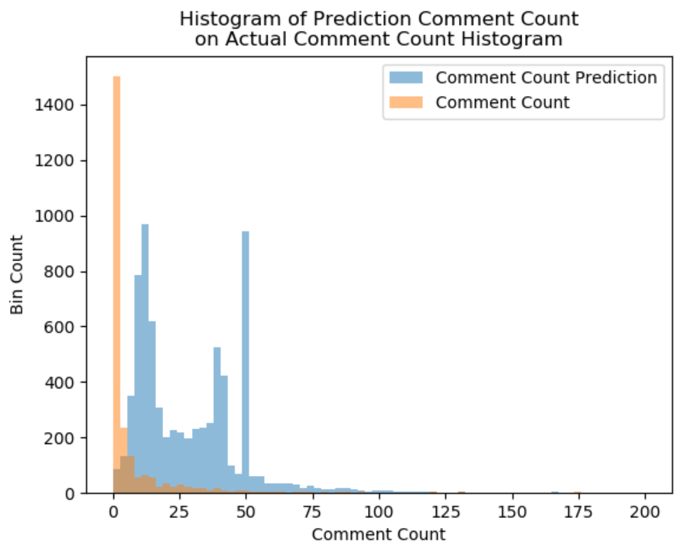


FIGURE 6.2: Histogram of Combined Model Comment Count Prediction on Actual Comment Count Histogram

share count variance was also larger. Both the NN and decision tree struggled to predict share count. Each model achieves a similar loss for predicting comment count.

The combined model performed the best of all the machine learning models. The study graphed the distribution of the combined model's predictions. There were concerns with the prediction of the initial machine learning models. Initial models often predict zero or one. This is due to the distribution's left skew. This skew also created a prediction graph that was heavily left skewed. Fortunately, this problem did not occur with the combined model. Rather, the combined model had a larger data spread. Figures 6.2-6.4 show the predicted vs actual distribution for the combined model.

6.2 Linear Models of Scraped Data

This research created linear models to better understand the data before creating machine learning models. This is done in order to identify data correlations, a lack of normality, and homoscedasticity in the data that might negatively affect the machine learning model.

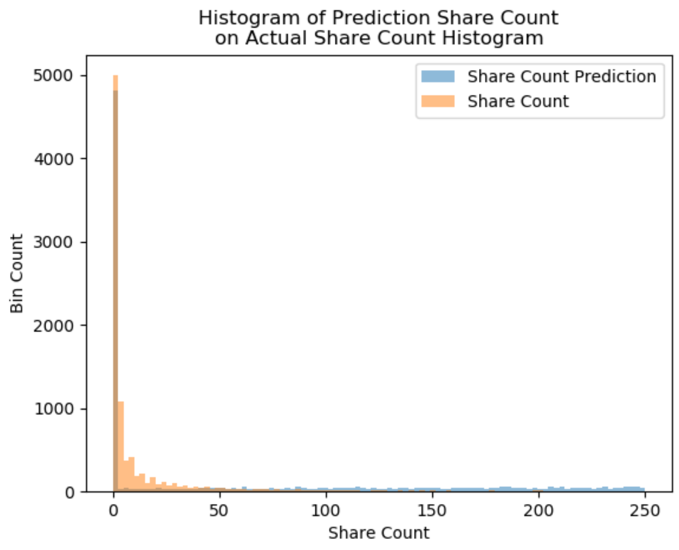


Figure 6.3: Histogram of Combined Model Comment Count Prediction on Actual Comment Count Histogram

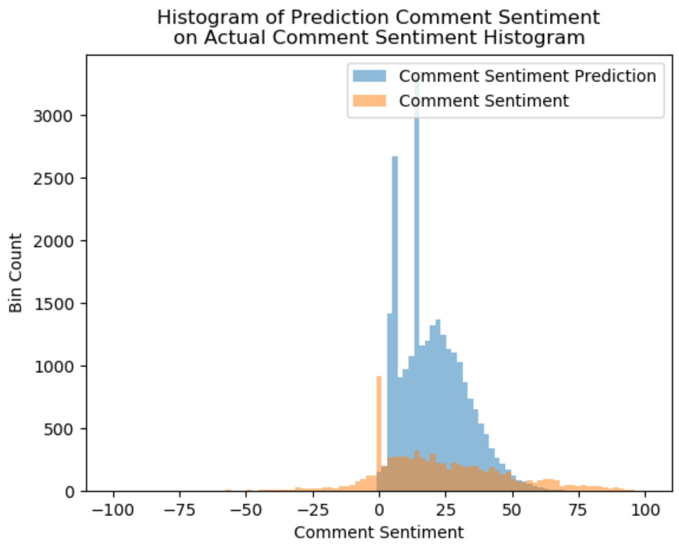


Figure 6.4: Histogram of Combined Model Comment Sentiment Prediction on Actual Comment Sentiment Histogram

The creation of linear models also helps identify which metrics are likely the most useful for machine learning models.

R programs created linear models. These models examine scraped data for data correlations. Examined data include Facebook page metrics and user interaction metrics. Strong correlations between Facebook page metrics and user metrics are important. This could affect this research's ability to compare post performance between Facebook pages. It would also affect the final model. If such correlation exists, models should include page metric data. Otherwise, it might justify using Facebook page metrics within another machine learning model.

The reported numbers are the coefficient of determination of the linear models. These numbers signify how much of dependent variable's variance can be explained by the independent variable. A value of 0.2 is fairly low, which a number above 0.4 denotes a stronger variable correlation. The linear models found an R^2 of 0.22 between talking about count and share count. The analysis found an R^2 of 0.38 between fan count and talking about count. The most significant result was an R^2 of 0.44 between comment sentiment and comment count. An R^2 of 0.38 and 0.44 are significant enough to create multicollinearity issues in machine learning features, which means that best practice avoids have two features that are strongly correlated. It is best practice to avoid variables that are correlated because they can negatively affect model weights by giving double the signal due to variable correlation.

Figure 6.5 shows a plot of comment count and comment sentiment. It is evident that both metrics do not seem correlated at either metric grows or decrease. Rather, there is a small and general correlation between the metrics. Figure 6.6 shows a plot of post sentiment and comment sentiment. The coefficient of determination is small between post sentiment and comment sentiment.

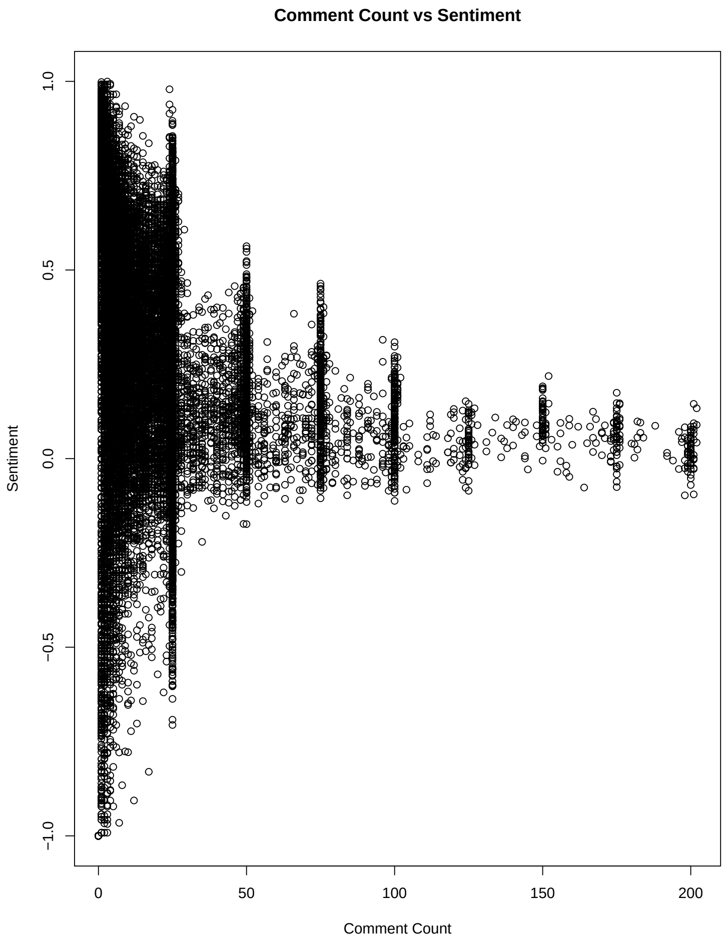


Figure 6.5: Plot of Comment Count vs Comment Sentiment

6.3 Text-based NN Models

The following sections will cover each model. The first model to cover is the text-based NN. This research utilized Keras to create, train, and test this model. Using Keras for text-based NN was beneficial. The Keras API allows flexible input vector sizes and dynamic word vector lengths. Keras allowed the creation of models with varying depth, regularization, and input. Keras also allowed quick model training. Initial training experimented with 2-8 hidden layers. The final model utilized two hidden layers, since two layers performed similarly to eight hidden layers. This research initially experimented with word vector sizes from 1k-400k. Good performance and fast training occurred with a word vector size of 10k.

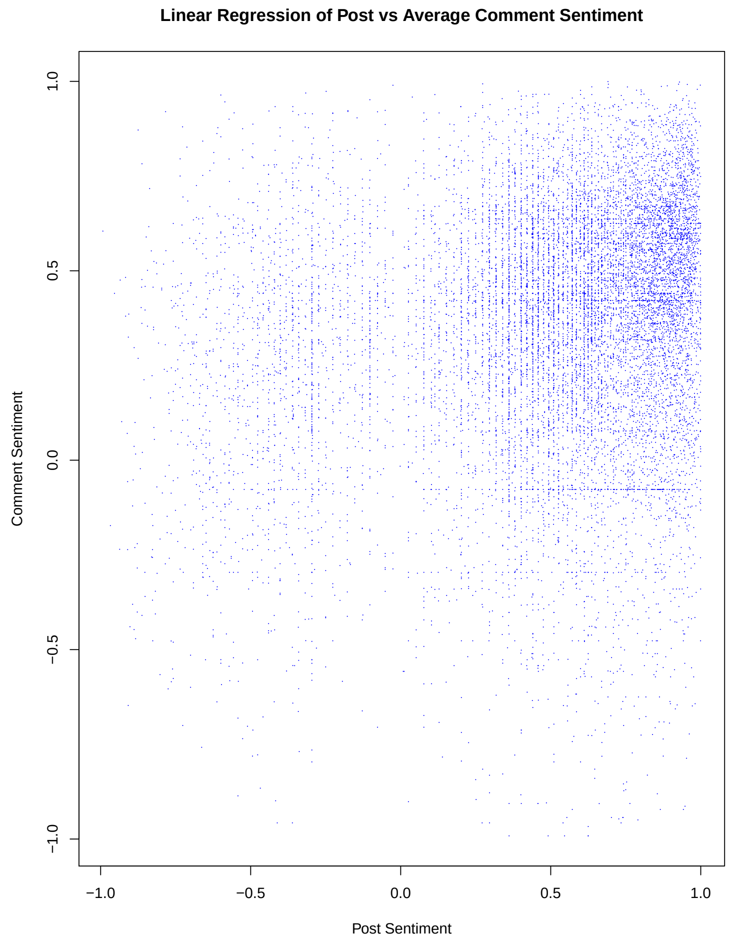


FIGURE 6.6: Plot of Post vs Comment Sentiment

The NN models used 10k as the word vector size.

6.4 Image-based CNN Models

Related work includes sections on keypoint descriptors and neural networks. CNNs detect image features and key points during repeated pooling and filter stages. This replaces the usefulness of image descriptors. The applied research uses CNNs in favor of image descriptors. Moreover, CNNs were faster to train than comparing image descriptors between images. This research trained with as many as eight layers and as few as one hidden layer. Model depths above three trained to a similar degree of accuracy. The models experimented with different convolutional size sequences. The experiments include doubling in size, usually from 64 to 128 with a 2x2 filter. The alternative reduced by size in half with a 2x2 filter. Models with a 2x2 filter that decreased their size by half produced the highest accuracies. The final model contained four hidden layers. This research used this CNN configuration for all CNN model training. The final image-based model performed with a better loss than the text-based NN model on all metrics.

6.5 Combined Models

The combined model concatenates both text-based NN and image-based CNN. The Keras API includes the ability to combine models. The program utilized this Keras feature. The API concatenated each model after the CNN flattened. At this point, the CNN portion has finished and the remaining eight nodes form a NN. At the point of model concatenation, the NN also decreased to eight nodes. The result was a NN layer of sixteen densely connected nodes. The sixteen-node NN connected to a final output node.

6.6 Model Prediction Problems

Initial models had the natural log applied to their data, because the results skewed towards zero. Applying natural log eliminated the skew and made the data more normal. The initial models that utilized the log data reached low loss values. These values were very low, around 0.5 loss with a small variance around 0.1. This prediction and variance is odd. This is especially true considering the distribution's variance is more than 3.5. This occurred because the distribution skews towards zero, and most values fall between 0 and

1. The model could produce the smallest mean squared error by predicting 0.5 with a slight variance. The problem disappears when the log transformation is not applied.

6.7 Word2Vec Models

Word2Vec models language without depending on n-grams (Radim et al., 2010). This partly explains why models like Word2Vec are a type of skip-gram model. Not depending on n-grams is useful, because increasing n-grams is computationally expensive. This is because the number of possible permutations increases exponentially with increasing n. The NN's word vector comprises from 3-gram vectors. Word2Vec is able to predict word context. The model stores which words tend to surround one another. The model remembers word context. This allows Word2Vec to predict words within phrases. Specifically, the model predicts the likelihood of a word occurring within a context. This makes the model good at modeling word relationships within contexts.

The study concerned itself to modeling comment sentiments. One of the hypothesis is that particular phrases might produce positive comment sentiments. The study built Word2Vec models and trained them on posts received positive comment sentiments. It was hoped that the model could then predict the likelihood that new posts elicit positive comment sentiments. The initial model correctly predicted for negative comment sentiment 25% of the time. There was a model mistake, which trained on letter frequency, not words. The repaired model had insufficient data to predict comment sentiment. Future work includes the need for more data to use the Word2Vec models. Despite the inability of the model to guess model similarity on new sentences, the trained model did learn from the data.

6.8 Use Case

This section will explain a real-world use case for the models created during this research. The purpose of the use case is to elucidate the model's usefulness in the real world. The experiment will also delineate the boundaries of the model's performance. The data used in the experiment is drawn from a separate validation data set that was not used during training or model testing. The data compared in the experiment differ in their magnitude of use interaction by at least one standard deviation. This was done to not penalize the model in cases where advertisements perform similarly. All predictions are performed using the combined model, since it performed best on all metrics.

The scenario is predicting user engagement for advertisements. The scenario pairs Facebook posts together. The model predicts user engagement for all posts. The program scores how often the model correctly predicted or greater user engagement. The program reports the result for each metric. The score is the number of correct predictions over incorrect predictions. This research compared the score with a random model. The random model produced the correct answer 50% of the time. The combined model scored 57%, 55%, and 53% for comment count, comment sentiment, and share count.

The model performance shows its applicability in the real world. Platforms could use the models for advertisers. The model can tell advertisers which ads will perform best on the platform. This allows advertisers have their ads vetted. The vetting could prevent advertisers from spending a lot of money showing worse ads. Moreover, the vetting would allow advertisers to only show ads that will perform best. While slightly above 50% is small, in the context of billions, 57% could make a monetary difference.

Chapter 7

Discussion

This study is similar to those before it (Stranton et al., 2015). Both studies seek to predict for user interaction. Yet, Stranton et al. binned their data and had their models perform classification. This is different, because this study used regression, since the data was continuous. This study is also different because it incorporates the post's image data, which was not a metric used by Stranton et al.

7.1 Addressing Research Questions

This thesis' research concerns machine learning with text and image data. It asks if machine learning models can predict user interaction from this data. Moreover, this thesis explores how well the models can learn the data. This research sought to create models that predict data with a loss that is less than the variance. Prediction is the mean squared error loss from the machine learning models. The variance is a very similar formula to the mean-squared-error but uses the mean. The hypotheses also ask if the models outperform a random model.

The many model types explore the hypotheses. The hypotheses ask if each model type can predict user interaction. Each model type has varying degrees of accuracy for predicting user interaction. Many models use different data. One model relies on the provided text and comments, while the other relies on post images. The Word2Vec model relied on post groups by sentiment. The combined model relied on text and image data. Yet, these models differed in their performance. Specifically, the combined model outperformed all other models. The conclusion is that a model which incorporates both text and image data is capable of outperforming other model types that only use one data type, i.e. images or text data. The result is that any prediction of user engagement on social media must incorporate both image and text data in order to achieve maximal performance.

7.2 Linear Models

This section cover the results of the linear models presented in the last section. The data correlations were sensible. The metrics talking about count and share count and correlated. It makes sense that users talk about shared content. The content is likely interesting, or worth sharing. Shared content also creates more content to talk about. Linear models found a correlation between fan count and share count. The more fans a page has, the more opportunities exist for people to share the content. Linear models also discovered a correlation between comment count and comment sentiment. Posts that receive a lot of comments are likely to receive positive comments. This might infer that happy users comment at a higher rate. Comments could also become more positive as their numbers increase on a post. Future work could include comment sentiment as a feature for the comment count. It is interesting that comment count was not correlated to share count. Page metrics are also not correlated with comment count. There was a lack of correlation between page metrics and user interaction metrics. The study used the lack of correlation to justify comparing posts across Facebook pages.

7.3 Combined Model

The combination of different-typed NN is not common practice within machine learning. Generally, new models use outputs from other models. The first attempt at creating a combined model used a decision tree. The decision tree's input was the output of both the text-based NN and image-based CNN. The decision tree was unable to generate a loss less than the data's variance. Likely, the decision tree was unable to learn from only two scalar inputs. Every single output is likely a poor and limited representation of the input data. This researchers decided to combine the CNN and NN in hopes of a better model. This mid-model data is more representative of the input data. It also includes image and text data. The final result was the concatenated CNN and NN that outperformed all other models.

7.4 Model Training Time

Regression is often calculated with equations like ordinary least squares (OLS). These methods calculated the squared difference between the predicted and actual value. The goal of each machine learning model is to decrease the model's loss. OLS is a well-accepted industry standard for fitting linear regression models (Whitenack 2017). A model that decreases the loss is doing a better job fitting the data.

The machine learning models were quick to train. The large batch sizes of 256 helped initialize model weights and prevent overfitting. Employing the Adam optimizer also helped reduce training time. This research scaled the data using min-max scaling. Each type of model required a similar quantity of training time.

7.5 Issues with CNN and NN Networks

The natural log of the comment and share counts had a variance of 3.0 and 4.3. The CNN trained to a loss below 0.05 in both models. This emphasizes the CNN's ability to train with a great deal of precision. The loss was low but was dissimilar from the test data distribution. Rather, the range of the prediction distribution was very low, not at all exponential. The CNN had predicted a small band around the data's mean. This produced a low loss, but a bad model. The NLP NN trained to 1.0 but faced the same problem of training the data's mean. This left both models unable to train to fit the test data. This problem was overcome with smaller batch sizes. The problem was also ameliorated by not applying the log-transform.

7.6 Metric Prediction

The study found that certain metrics are easier to predict than others. The share count had a lower variance than the comment count. The difference was not small. The variance was 1000x smaller. The combined model loss was similar between share and comment count. This reflects that share counts are difficult to predict. It also infers that share counts are less related to the image and text data. Share counts might be a factor of other features, like page popularity. Future work might include a third model with Facebook page metrics.

The combined model outperformed all other models for each metric. The result is even more surprising in light of the decision tree performance. The decision tree also took as input data from both the text-based NN and the image-based CNN. However, the decision tree performed far worse than the combined model. While the exact reason is unknown, there are a few differences between the two models. The combined model was tightly integrated with the text-based NN and image-based CNN. When the combined model trained, it also trained its two-parent models. This is different from the decision tree. The decision tree did not learn alongside its inputs. Likely, parent models compensate for mistakes by training together.

The model concatenation occurs in the later stages of each model. The goal was to combine these dense networks before the final output and after the CNN. The goal was a combined model that kept parent model training more-so independent. Combining models at the end keeps both models somewhat separate. This research plotted prediction vs test data for all models. The combined model best resembled the test data's distribution. The text-based NN and image-based CNN tended to skew towards zero. The combined model's predictions are more spread out.

Text-based models were poor predictors for many metrics. Text-based models did especially bad in prediction share counts. One would guess that users are sharing content they consider worthy of sharing. However, image-based models could better predict share counts. Image-based models performed better than text-based models on all metrics. This emphasizes the importance of images for predicting user interaction on social media.

7.7 Model Loss and Outperforming Random Guessing

One goal of the paper was to produce a model that performed better than random guessing. Random guessing alone is not representative of the data. A better guess is based on the input's distribution. A good random guess would consider the data's value at each point along its distribution. Such a distribution would weigh each value by the frequency of data. This calculation is the expected value. For a normal distribution, the expected value is equal to the mean. The log transform of the data is more normal. The combined model produced a loss of 4.0 on a distribution with a mean of 4.6 for the comment count. The share count had a mean of 3.5 and the loss got down to 3.1. In both cases, the combined model performed far better than the distribution's mean. Later modeling compares the results against a random model.

7.8 Unexpected Findings and Restructured Hypotheses

The poor Word2Vec performance was not expected. Word2Vec should be particularly good at performing NLP from its context. Limited data prevented enough Word2Vec training. The Word2Vec models were unable to predict comment sentiment with any accuracy. CNN with image inputs was more accurate than text-based NN. This seems to say that images influence a large degree of the comment sentiment.

7.9 Study Contributions

The study contributes to how well model types predict user interaction. The study also demonstrates increased model performance from combining NN and CNN models. This research provides model loss values for future research. This research compares model performance with existing research. The code is available on GitHub and includes the trained models, 20GB of images, and the text data on a 1GB SQLite database. This makes the data set available for future studies.

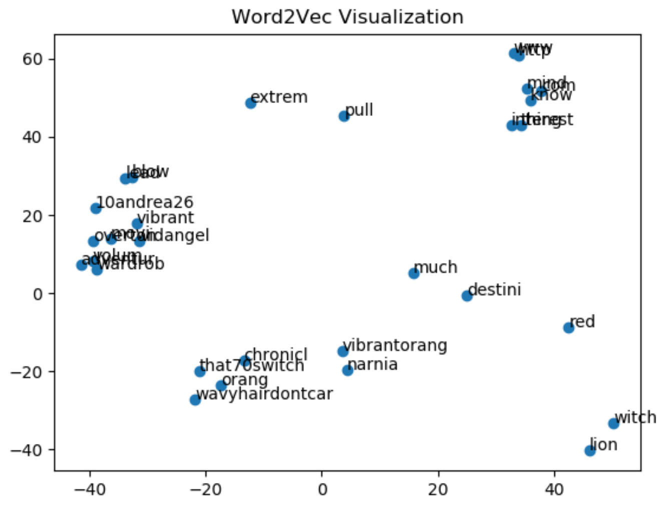


Figure 7.1: Sample of Positive Word2Vec Visualization for 20 Data Points

The study also explores other areas of user interaction data on social media. The study created many linear models with Facebook page data. These findings are further explored in the final models and model performance experiment. The benchmarks define how well research can predict user interaction. The paper explores the log-normal distribution for comment and share counts. This paper notes common errors for these models. The Word2Vec visualizations provide dimensionality-reduced images of positive and negative sentiment groups. One of these visualizations is shown in Figure 7.1. The visualization only displays a sample of the large dataset, since the image becomes unreadable as more words overlap.

This research also explored constructing models to predict user interaction. The paper provides insight on model hyperparameters. These hyperparameters include model depth, dropout rates, and word vector sizes. The GitHub project provides clear examples for creating Keras models. The study also explored the extent that NLP can predict on social media. The study again demonstrates the power of CNN for prediction. The study also explores methods to combine image and text data.

The study is a case study for user interaction on advertiser posts. The expectation is that the results will be generalizable to other studies. This research contributes to understanding advertisement success on social media. This research provides a guidepost of sorts for approaching social media data. This research will aid in the selection and transformation of social media data. Similar studies focus on modeling data to user relevance for users (Li et al., 2015). Research needs an approach from the perspective of ad content.

Chapter 8

Conclusion

This study adds to the existing body of research on predicting user interaction. This research explored five types of machine learning models. This research tested and reported the results. This research create two model that combined image-based and text-based models. The mid-model concatenation better-predicted user interaction than the other models. The combined models were then evaluated against the text-based NN and image-based CNN. The combined model outperformed the text-based and image-based model for each metric. This research also demonstrated the performance of CNN on social media data. The image-based models always outperformed the text-based models. Image-based models tended to perform far better on more complicated data.

This research applied the combined model to a real-world use case. The model took as input two advertisement post. The model then generated user engagement predictions for each advertisement. The model paired predictions for comment count, share count, and comment sentiment with each post. These predictions were compared with the actual user engagement metrics. The random model was correct 50% of the time. The combined was correct 57% 55%, and 53% of the time for comment sentiment, comment count, and share count. The model performance serves as a benchmark for future research.

8.1 Limitations

Model training time was a limitation. There were fifteen models. Each model requires many hours of training time. Machines with GPUs can only train one model at a time, so this process was very time-consuming. Time was not sufficient to train all models thoroughly. Heuristics dictated for hyperparameters and batch sizes. With more time, the models could become more performant. Model overtraining was also a result of less-than-perfect hyperparameters and model characteristics. One observation is that the models generally begin to overtrain after 30 epochs. More regularization might mitigate this problem. Such mitigations would likely lead to increased model performance. During the study,

8.2 Future Work

Future research should consider combining image and text-based machine learning models. Researchers should use image data for predicting user interaction. Moreover, image-based models performed best on complicated data. Comment sentiment is one such metric. Image-based models outperformed text-based model loss by over 40% for comment sentiment. The phenomenon repeats itself with share counts. The image-based model far outperformed the text-based model by a factor of 3.5. Text and image-based models performed similarly on simple data, i.e. comment count.

Word2Vec performs NLP with much smaller word vectors. This research hoped to use Word2Vec to predict comment sentiment. Future research should collect more comment sentiment data to train Word2Vec models. Future research could overcome existing Facebook API scraping problems. This researchers could finish collecting comment sentiments for the 350k Facebook posts. The rest of the data should be enough to train the Word2Vec models.

Future work combines all models in this study. Each model likely learns something different about the input data. This is especially true of more disparate models like the decision tree. Each model could help predict user interaction. This likely would lead to increased accuracies. Share count and comment sentiment have a coefficient of determination over 0.4. This likely means that a share count model can be input for predicting comment sentiment. Alternatively, advertisement success with one metric predicts success for other metrics. In this way, all 15 models can predict for generalized advertisement's performance on Facebook. The created models could also perform content recommendation. The program could apply filters, transformations, and rotations. The models from this research could predict for user interaction. This could lead to predicting which transformation would produce an image that would perform best on social media.

Bibliography

Ahmadi, R. et al. (2013) 'Survey of image denoising techniques', Life Science Journal, 10(1), pp. 753–755. doi: 10.1.1.107.1823.

Aittala, M., Aila, T. and Lehtinen, J. (2016) 'Reflectance modeling by neural texture synthesis', ACM Transactions on Graphics, 35(4), pp. 1–13. doi: 10.1145/2897824.2925917.

Antonio, G. and Sujit, P. (2017) Deep Learning With Keras. Packt Publishing.

Azizian, S. et al. (2017) 'Identifying Personal Messages: A Step towards Product/Service Review and Opinion Mining', *International Conference on Computational Science and Computational Intelligence* , (January 2018).

Buduma, N. (2015) Fundamentals of Deep Learning, CEUR Workshop Proceedings. doi: 10.1017/CBO9781107415324.004.

Casari, A. (2017) 'Mastering Feature Engineering Principles and Techniques for Data Scientists'. Oreilly Associates Inc.

Chen, T. et al. (2014) 'DeepSentiBank: Visual Sentiment Concept Classification with Deep Convolutional Neural Networks'. Available at: `http://arxiv.org/abs/1410.8586`.

"Click-Through Rate Prediction." Kaggle Competition (2015), `https://www.kaggle.com/c/avazu-ctr-prediction`.

Chu, T. C. et al. (1985) 'Applications of Digital . Image-Correlation Techniques to Experimental Mechanics', (August 2015). doi: 10.1007/BF02325092.

Confessore, Nicholas. "Cambridge Analytica and Facebook: The Scandal and the Fallout So Far." The New York Times, The New York Times, 4 Apr. 2018, `www.nytimes.com/2018/04/04/us/politics/cambridge-analytica-scandal-fallout.html`.

Derpanis, K. G. (2004) 'The Harris Corner Detector', pp. 2–3.

Dwivedi, Y. K., Kapoor, K. K. and Chen, H. (2015) 'Social media marketing and advertising', The Marketing Review, 15(3), pp. 289–309. doi: 10.1362/146934715X14441363377999.

Facebook's Growth Is Fueled by Mobile Ads — Statistic. (n.d.). Retrieved from `https://www.statista.com/chart/2496/facebook-revenue-by-segment/`

Fisher, T. (2009) 'ROI in social media: A look at the arguments', *Journal of Database Marketing and Customer Strategy Management*, 16(3), pp. 189–195.
doi: 10.1057/dbm.2009.16.

'Romero Nuria' (2011), 24, pp. 145–151. doi: 10.1108/08880451111169223.

Gao, B. et al. (2005) 'Web image clustering by consistent utilization of visual features and surrounding texts', *Proceedings of the 13th annual ACM international conference on Multimedia - MULTIMEDIA* '05, p. 112. doi: 10.1145/1101149.1101167.

Gelli, F. et al. (2015) 'Image Popularity Prediction in Social Media Using Sentiment and Context Features', *Proceedings of the 23rd ACM international conference on Multimedia* - MM '15, pp. 907–910. doi: 10.1145/2733373.2806361.

Gramlich, John. "5 Facts about Americans and Facebook." Pew Research Center, Pew Research Center, 10 Apr. 2018, www.pewresearch.org/fact-tank/2018/04/10/5-facts-about-americans-and-facebook/.

Graph API Engagement - Documentation. (n.d.). Retrieved from `https://developers.facebook.com/docs/graph-api/reference/engagement`

Hassner, G. L. and T. (2015) 'Age and Gender Classification using Convolutional Neural Networks', *2008 8th IEEE International Conference on Automatic Face and Gesture Recognition*, FG 2008, 24(3), pp. 2622–2629. doi: 10.1109/AFGR.2008.4813314.

Hoffman, D. D. L. D. L. and Fodor, M. (2010) 'Can You Measure the ROI of Your Social Media Marketing?', MIT Sloan Management Review, 52(1), pp. 41–49. doi: 10.1287/mksc.1120.0768.

Hootsuite Media Inc. "Social Media Marketing Management Dashboard." Hootsuite, `hootsuite.com/`.2018.08.01

VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text (by C.J. Hutto and Eric Gilbert) *Eighth International Conference on Weblogs and Social Media (ICWSM-14).* Ann Arbor, MI, June 2014.

Jan Erik Solem (2012) 'Programming Computer Vision with Python', Programming Computer Vision with Python, p. 264. doi: 10.1017/CBO9781107415324.004.

Kaehler, Adrian, and Gary R. Bradski. Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library. O'Reilly Media, 2017.

Kanan, C. and Cottrell, G. W. (2012) 'Color-to-grayscale: Does the method matter in image recognition?', PLoS ONE, 7(1). doi: 10.1371/journal.pone.0029740.

"Keras: The Python Deep Learning Library." Keras Documentation, keras.io/.

Khosla, A., Das Sarma, A. and Hamid, R. (2014) 'What Makes an Image Popular?', *Proceedings of the 23rd International Conference on World Wide Web*, pp. 867–876. doi: 10.1145/2566486.2567996.

Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2012) 'ImageNet Classification with Deep Convolutional Neural Networks', Advances In Neural Information Processing Systems, pp. 1–9. doi: `http://dx.doi.org/10.1016/j.protcy.2014.09.007`.

Li, C. et al. (2015) 'Click-through Prediction for Advertising in Twitter Timeline', *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* - KDD '15, pp. 1959–1968. doi: 10.1145/2783258.2788582.

Li, Y. et al. (2017) 'Diversified Texture Synthesis with Feed-forward Networks'. doi: 10.1109/CVPR.2017.36.

Lin, H. et al. (2014) 'User-level psychological stress detection from social media using deep neural network', *Proceedings of the ACM International Conference on Multimedia* - MM '14, pp. 507–516. doi: 10.1145/2647868.2654945.

Liu, B. (2012) 'Sentiment Analysis and Opinion Mining', (May), pp. 1–108. doi: 10.2200/S00416ED1V01Y201204HLT016.

Lowe, D. G. (2004) 'Distinctive image features from scale invariant keypoints', *International Journal of Computer Vision*, 60, pp. 91–11020042. doi: `http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94`. Mandhyani, J. et al. (2017) 'Image Sentiment Analysis', *International Journal of Engineering Science and Computing*, 7(2), pp. 4566–4569.

Morgan, R. and Garigl, R. (2015) Natural language processing with JAVA, Endeavour. doi: 10.1016/0160-9327(95)98888-M.

Ohsawa, S. and Matsuo, Y. (2013) 'Like Prediction: Modeling Like Counts by Bridging Facebook Pages with Linked Data', *Proceedings of the 22Nd International Conference on World Wide Web Companion*, pp. 541–548. doi: 10.1145/2487788.2487992.

Distribution-of-googles-revenues-by-source. Google: Ad revenue 2001-2017. Retrieved from `https://www.statista.com/statistics/266249/advertising-revenue-of-google/`

Patterson, J. and Gibson, A. (2017) Deep Learning: A Practitioner's Approach, O'Reilly Media. doi: 10.1016/B978-0-12-391420-0.09987-X.

Poria, S. et al. (2016) 'A Deeper Look into Sarcastic Tweets Using Deep Convolutional Neural Networks'. Available at: `http://arxiv.org/abs/1610.08815`.

Poushter, J., Bishop, C., Chwe, H. (2018, June 19). Social Media Use Continues to Rise in Developing Countries but Plateaus Across Developed Ones. Retrieved from `http://www.pewglobal.org/2018/06/19/social-media-use-continues-to-rise-in-developing-countries-but-`

Radim Řehůřek and Petr Sojka (2010) 'Software Framework for Topic Modelling with Large Corpora' ,Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. ELRA. p. 45–50.

Schacht, J., Hall, M. and Chorley, M. (2015) 'Tweet if you will – the real question is , who do you influence?', (June). doi: 10.1145/2786451.2786923.

Segalin, C., Cheng, D. S. and Cristani, M. (2017) 'Social profiling through image understanding: Personality inference using convolutional neural networks', Computer Vision and Image Understanding. Academic Press, 156, pp. 34–50. doi: 10.1016/J.CVIU.2016.10.013.
Tang, Y. and Hinton, G. (no date) 'Robust Boltzmann Machines for Recognition and Denoising.(2012).pdf'.

Shavitt, S., Lowrey, P. M. and Haefner, J. E. (1998) 'Public Attitudes Toward Advertising : More Favorabie Than You Might Think', *Journal of Advertising Research*, 38(4), pp. 7–22. Available at: `https://www.researchgate.net/publication/247294509_Public_Attitudes_Toward_Advertising_More_Favorable_Than_You_Might_Think`.

Smith, A., Anderson, M. (2018, March 01). Social Media Use in 2018. Retrieved from `http://www.pewinternet.org/2018/03/01/social-media-use-in-2018/`

Srivastava, N. et al. (2014) 'Dropout: A Simple Way to Prevent Neural Networks from Overfitting', *Journal of Machine Learning Research*, 15, pp. 1929–1958. doi: 10.1214/12-AOS1000.

Straton, N., Mukkamala, R. R. and Vatrapu, R. (2015) 'Big Social Data Analytics for Public Health : Predicting Facebook Post Performance using Artificial Neural Networks and Deep Learning'.

Tiago, M. T. P. M. B. and Veríssimo, J. M. C. (2014) 'Digital marketing and social media: Why bother?', Business Horizons, 57(6), pp. 703–708. doi: 10.1016/j.bushor.2014.07.002.

Wang, Y. et al. (2015) 'Unsupervised sentiment analysis for social media images', *Proceedings of the 24th International Conference on Artificial Intelligence*, pp. 2378–2379. Available at: `https://arxiv.org/abs/1604.03489`.

Xu, C. et al. (2014) 'Visual Sentiment Prediction with Deep Convolutional Neural Networks'. Available at: `http://arxiv.org/abs/1411.5731`.

You, Q. et al. (2015) 'Robust Image Sentiment Analysis Using Progressively Trained and Domain Transferred Deep Networks', pp. 381–388. doi: 10.1145/2733373.2806284.

Whitenack, Daniel, and Vipul Agrawal. Machine Learning With Go. Packt Publishing, 2017.

Wright, E. et al. (2010) 'The Lasting Effects Of Social Media Trends On Advertising', *Journal of Business Economics Research (JBER)*, 8(11). doi: 10.19030/jber.v8i11.50.

Shi, X. (2017). A Study on Factors Influencing the Click Through Rate , CTR , of Internet ads by the Charging Mode of CPM (Cost Per Mile) Based on Information Transfer.

Zelinsky, A. (2009) 'Learning OpenCV. Computer Vision with the OpenCV Library', IEEE Robotics and Automation Magazine, p. 100. doi: 10.1109/MRA.2009.933612.