**Evaluation of Deep Learning methods for Sentiment Analysis of Twitter Data**

Oliver D’Esposito

The Lab School of Washington, D.C.

English 12, Senior Thesis

Ms. Jacob

January 18, 2022

Contact: odesposito@dragonwork.com

**Abstract**

This paper looks into the efficacy of different deep learning methods for short-term sentiment analysis. This paper’s goal is to challenge the common notion that Recurrent Neural Networks (RNN) are more effective than Forward Feed Neural Networks (FFNN) for text processing of short-form text content. This comparison is based on the leading forms of each network structure. For RNN's, Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) will be explored. These were chosen because of their ability for taking and understanding sequential data. The FFNN network will be a Convolutional Neural Network (CNN), selected for its ability to focus on individual high weighted tokens. The first part of this paper will outline why the question is important, what challenges it comes with, and what technologies are in the field. The second section consists of the counterargument. This outlines arguments for RNN's and K-nearest Neighbor (KNN) over CNN for this processing task. That is followed by the argument that CNN with natural language processing (NLP) is the ideal method for sentiment analysis of the data. These arguments include the validity of NLP in the form of embedding layers, why older models are less effective, how hardware allows for DNN support, CNN's application flexibility, and CNN being more effective than RNN's. Then concluding with the recommended approach. This approach is the use of a CNN with complex NLP, mostly due to CNN's ability to focus on position irrelevant features.

 *Keywords:* short term sentiment analysis, sentiment analysis, lstm, gru, cnn, knn, ffnn, forward feed neural network, recursive neural network, recursive neural network, twitter, knn, ann'

**Evaluation of Deep Learning Methods for Sentiment Analysis of Twitter data**

**Introduction**

 Over the past few years, the impact of short-form content on society has skyrocketed. It has become a source of news, communication, and organization. Twitter is one of the platforms that has grown with the societal shift towards using social media as a hub of interaction. The Twitter platform allows users to post short text content limited to 250 characters. Twitter's users create short-form text content also called micro-blogs. As the platform has grown the amount of data being created on Twitter has become unparalleled. This data is an ideal corpus for testing natural language processing (NLP) tasks and specifically short-form text sentiment analysis. Many machine learning (ML) models can do sentiment analysis. However, the active debate on the most effective model has focused on Convolutional Neural Networks (CNN), Gated Recurrent Unit (GRU), and Long Short Term Memory (LSTM). The goal of this paper is to show which of these models is ideal for the sentiment analysis of short-form textual data. It also compares the efficacy of advanced NLP in this case and the validity of older ML models for the task. This paper reviews the best implementation of sentiment analysis for short-form text content. Twitter provides an ideal data source for this testing and will be used for research, but this review does not only apply to Twitter data. This paper's conclusions generalize to other domains with a similar textual structure such as e-commerce reviews. This paper concludes that because of the importance of flagging single terms and the lack of need for understanding sequential data, CNN's are the ideal option

**Background**

Machine learning, NLP, and the impact of social media are wide-ranging fields that can not be fully explored in this paper. This paper focuses on these topics only concerning the sentiment analysis of Twitter data. This section will also outline why short form text analysis is worth exploring and the usefulness of its outcome.

 Sentiment analysis is a cornerstone of NLP and modern deep learning. It is the use of NLP and computational analysis to classify or extract information about sentiment from a piece of textual data (Hussein, 2018). NLP is any process involving having a computer extract and learn information from text. In more unadorned terms, sentiment analysis is having a computer system called a model to determine the emotions of a piece of text. Sentiment analysis can be done by different model types and architectures. The best model for the task differs depending on information, such as how the text is structured and its length. NLP in general is a challenging problem domain and sentiment analysis is no exception. Take, for example, two tweets “Wow I cannot wait to go to work today, more stacking useless papers” and “I love my job.” Without context, it is even challenging for humans to derive the sentiment of these tweets. Maybe they enjoy stacking useless papers and perhaps this other user intended “I love my job” to be a sarcastic comment. Many other tweets are more straightforward than this but the challenge is clear, especially for a computer. Sentiment analysis is not only relevant to a short text, it is used on variations of text length; these are referred to as levels. These sentiment analysis levels are documented, word/term, sentence, and aspect levels (Hussein, 2018). The use of sentiment analysis has rapidly spread across fields and is used for books, advertisements, and consumer information. All this effort is worth it because of the value of the information that is produced. Sentiment analysis allows companies to know how well an ad campaign performed, politicians to know their chances in the next race, and researchers to know how political movements ebb and flow.

**Data overview**

 The data present on the Twitter platforms is unrivaled as a textual corpus for sentiment analysis. This data provides major advantages to any accessible alternative. Users on Twitter make content that is often emotion-rich, the most useful tweets being thoughts on current events or topics users feel strongly about. This massive corpus of emotion-rich data lends itself to sentiment analysis research. Due to the widespread adoption of the Twitter platform, it also presents a diverse data set representing a larger user base than just the United States (Jamal et al., 2019). Another advantage is that Twitter data is also easily grouped into topics, this is because of the widespread use of hashtags (#) and subtweets. Users grouping their tweets creates a more accurate and nuanced grouping; it allows the data to be used to train complex classification algorithms (Varol et al., 2017). Also, Twitter data is relatively accessible. Similar sources such as YouTube or the Facebook platform, although being domains where the research of this paper would be useful, are closed from access. Twitter allows for approved data collection and research openly on their platform with valid justification. Facebook specifically does not allow any form of data research on its platform and actively pursues those attempting it (Batrinca & Treleaven, 2014). This access to data also means there is far more research surrounding Twitter data. The Twitter platform's data corpus’s accessibility, emotional richness, grouping, and user posts make it the ideal platform for this paper to focus on.

Although Twitter is the ideal platform for research, it is not free of challenges. Some of these challenges are specific to the domain of sentiment analysis of tweets, like the volume of tweets. There can easily be a million tweets about one topic in a day. This is good because it provides a more useful and comprehensive dataset but working with that sheer amount of data is hardware and software-intensive. Most of the time when training Deep Learning models a labeled dataset is needed. The amount of data on Twitter makes labeling this data a major hurdle. It can either be done by hand or by some method of automatic sorting. Further complicating Twitter data is the language conventions on the platform. Many people on Twitter use slang or extra letters to add meaning. For example, many users repeat letters in words to imply an intent or excitement, “Hello” would become “Hellllooooo” (Khan et al., 2020). This makes dealing with the data more challenging as it can not just be based on a known dictionary of words. These words must be filtered out in preprocessing or implemented into the model's NLP protocol. Although outside the scope of this paper, it is worth noting that a wide range of data can be present in a tweet other than text. A tweet could link to an article, image, video, or in 2022 a Non-fungible Token (NFT). Also, all Twitter user accounts have their data, such as followers and average engagement. For many other models, this data may be useful. Switching to viewing the dataset from a macro scale, both an advantage and disadvantage are cultural differences across the corpus of text. These differences mean that picking up on subtle nuances in text is more challenging. It is hard to teach an ML model different cultural nuances from the same data corpus (Khan et al., 2020). Nuances are important because the boundary between emotions is not clear at times (Khan et al., 2020). Arguably this fuzzy border can be more of a challenge on Twitter as it is expected that tweets are short and therefore have less information from which to build context. This fuzzy boundary is also present because judging emotions is a subjective task, therefore two humans classifying data may classify it differently. These challenges will depend on the data corpus but it should be clear that some generally apply to sentiment analysis as a whole.

**Social Media Impact**

 Social media like Twitter has become the hot spot for social change in recent years. People interact with social media on an unprecedented scale. At this point, social media platforms are part of people's daily routines. Facebook for example recorded 845 million users in 2011, 10 years ago (Gundecha & Liu, 2012). Twitter sees its fair share of a massive user base. In 2013 Twitter hit an average of 500 million tweets per day, 5,700 Tweets per second. That is on average not during spikes of usage (Kirkorian, 2013). There is less data on daily tweets released now, however conclusions of usage growth can be drawn from Twitter's investor reports from 2020. With 27% growth since 2019, Twitter now has 192 million monetizable daily active users. This metric refers to the users who are signed when using Twitter and can have ads served to them. It is also worth noting that the yearly income of Twitter is now 3.2 billion dollars (Twitter Investor Relations, 2021). This massive usage has been used to make an equally massive impact. Movements like Black Lives Matter (BLM) and #MeToo have brought large social change. The BLM movement started on social media and much of its organizational force continued to originate from there. The day after the Murder of George Floyd, there were over 218,000 tweets on the #BlackLivesMatter hashtag. After the movement sped up and protests began the number of tweets with #BlackLivesMatter passed 1 million per day but this dwindles in comparison to the largest day for BLM on Twitter. On May 28th, 2020 there were 8.8 million tweets related to BLM. These tweets correlated with country-wide protests and large amounts of focus on the BLM movement (Pew, 2021). This is not to mention the massive amount of advertising and political campaigns that now focus on platforms like Twitter. The influence of Twitter and platforms like it on modern society are omnipresent. All this data on how these events transpired and what made them happen was created and cataloged on social media platforms. The analysis of this data will provide invaluable insight into a massive portion of the population. Although Sentiment Analysis is only a small part of this analysis due to the challenge of every facet of NLP and Deep Learning, it is worth exploring in detail.

**AI explanation**

 Artificial intelligence has been around for a long time and the ideas around it have existed since the invention of modern computing. The idea of modern computers and computers “thinking” for themselves was first introduced by Alan Turning in his 1950 paper Computing Machinery and Intelligence. Early models were often based on basic mathematical or logical principles. Examples of these models include expert systems, Support Vector Machines (SVM), Naive Bayes, logit boosting, decision tree, and K-Nearest Neighbor (KNN). KNN and SVM are relevant to this paper because they are still implemented and used in textual sentiment analysis (Mehedi, 2021). A KNN model functions by taking data points and mapping them into an n-dimensional space. A KNN model will decide how to classify a data input based on its distance from other data points (Mehedi, 2019). SVM is also a classification model. SVM functions by mapping datasets to a higher dimension than extracting a hyperplane by maximizing the margin between the decision boundary and the point on each side. A hyperplane can be thought of as a line, if data is on one side it is positive the other negative. These models are less used now and this paper goes on to conclude in many cases uncompetitive for a plethora of reasons.

In recent years, Artificial Intelligence has had a renaissance. At the forefront are artificial neural networks (ANN). ANNs are commonly explained as being built like the human brain. A simple explanation of these systems is that they are made up of units also known as neurons. These units when put together take in data and modify it to make a more useful representation of that data. Units take in many data points and then return only one output. This output is passed to the next unit. Units are organized into three layers, input, hidden, and output. An example of this would be taking an image, which at its base level is a matrix of numbers and making it a more useful representation in the form of a vector [0, 1] the value of which represents the chance that the image has a human in it.

A surface-level understanding of ANNs will suffice for this paper. In the simplest term, these units are organized into layers. There is an input layer, where data comes in, hidden layers, where weights are changed to learn better representations of data, and an output layer where the final output of the network is returned. Each successful layer can extend more useful information (Migwil, 2018). The model learns by testing known data and seeing how off its predictions were. Based on the error of each prediction, it changes the outputs of its units until the error is minimized. This learning loop is repeated many times until the model can make accurate predictions.

From a technical point of view each unit is a mathematical function, the cornerstones of ANNs are sigmoid and relu units.

$$y=\frac{1}{1-e^{-ls}}$$

$$s=\sum\_{i = 1}^{n}\W\_{i}x\_{i}+b\_{i}\_{(unit weigh)}$$

The first function is the sigmoid function. It is known as a “squashing” function, whatever input it gets is returned in the range of 0 to 1. In this case $W\_{i}$ are weights that the models learn. Sigmoid or whatever squashing function takes its place is called the activation function. All weights are and numbers that the model changes to represent a new equation. The variable $b\_{i}$ is the learned bias which is another way to alter more general output of a unit. The symbol $Σ$ represents the summation of all the input data $x\_{i}$ and learned weights $W\_{i}$. In more condensed language, this model takes all the outputs of the units before it and multiples them by a learned number for each new input. It then adds that all together and adds a learned number known as bais, finally whatever the output, it is squashed between 1 and 0 to normalize the output (Staudemeyer, 2019). This is the equation for only one unit and these equations get more complex as the entire system is taken into account, but that is outside of the scope of this paper.

ANNs usually learn via a process called gradient descent or some derivation of it. One of the key components of an ANN is the loss function. During the learning process each time the network computes its guess for a given input (a step) it calculates how wrong it was with the loss function. The goal of the network is to minimize its loss or error when making guesses. To learn, the model takes the error and calculates the derivative of it at the input. The derivative informs the model of which direction the weights, negative or positive, should be altered in. Due to the chain rule from calculus, the derivative of each unit, being treated as its own function, can be found therefore allowing the model to use every weight in the network to minimize the loss (Staudemeyer, 2019).

 This paper mainly focuses on the effectiveness of two types of Neural Networks. These types are Recurrent Neural Networks (RNN) and Forward Feed Neural Networks (FFNN). Forward feed neural networks are what was just explained. All the data flows forward through the layers with no way of a unit knowing what data was imputed before it. RNN's are networks that allow for the original input data to be remembered sequentially. Examples of sequential data that RNN's are effective for are sound and text data (Ghelani, 2019). This is a complex topic that will be discussed later in the paper when comparing RNN's to FFNN's.

 There are subtypes of each of these ANN structures. Convolutional neural networks are forward feed networks that are known for use in computer vision. They take advantage of convolution layers to pick up on patterns in the input data. This allows a CNN model to pick on relations to this specific use case and show lots of interesting uses in NLP (Ghelani, 2019; T. Makker, 2017). The other two model implementations that are at the forefront of this debate are both RNN's. The first is Long Short Term Memory, which is a model in which each unit consists of gates. Gates are used to control the flow of information in the network. LSTM also has a memory cell to hold the information before it is forgotten. This allows LSTM to not only recall information like an RNN but also forget that information after a learned interval (Tan, 2018). The other model focused on is the Gated Recurrent Unit (GRU) architecture, a type of RNN. GRU was created after LSTM and is somewhat a simpler version of the same idea. GRU only has two gates, a reset and update gate; it has no need for a memory block. It is a relatively new architecture but is simpler and achives much of the same goal as LSTM, making it an excellent choice for sentiment analysis.

**Counter Argument**

The ideal model type and architecture for sentiment analysis is still up for debate. There are many who conclude that simpler ML models such as KNN or RNN are the ideal model type for sentiment analysis of short form text, for their simplicity and easy application.

Many argue for the use of more basic models because they tend to be more mathematically sound. KNN and SVM due to their more basic mathematical basis when compared to ANNs are easier to understand and diagnose. Also because of the simplicity of the models the network structure is less finicky. ANNs have to be fine-tuned via educated guesswork to render ideal results (Mijiwil, 2018). KNNs are not out of the race, there is research indicating that KNNs can still be used for this sentiment analysis scenario. One such piece is from Mehdi (2021) who used KNN to analyze the sentiment towards different covid vaccines around the world. Mehdi got a satisfactory outcome when using a KNN model. Also, many of these basic ML models do not overfit or underfit. Because ANNs calculate a decision boundary to classify data they are prone to over and underfitting. Underfitting is when, after being trained on a data set, an ANN does not change its weights enough and so is incorrect on both the training and testing data. Overfitting is when the training data is learned too well; this means that the model cannot generalize to any non-training data. There are solutions to this such as dropout layers. A dropout layer randomly turns data into zeros to prevent the model from being overdependent on one network feature (Ghelani, 2019). All of these are valid arguments against using more complex models but for reasons outlined later, ANNs are worth the trade off for the far better accuracy.

The most active debate on the proper implementation of sentiment analysis for Twitter data is that of ANN architecture choice. This paper will focus on the debate between FFNN and RNN models. This leads to the question of the best choice among LSTM, GRU, and CNN. There is a large base of information that supports the of use fo LSTM or GRU over a CNN. The common thought in the AI field is that RNN's are best for NLP tasks due to the ability to understand data in a sequential context (Yin, 2017). In one paper it was found that in almost all cases of general NLP tasks GRU models outperformed both LSTM and CNN models. This test was done with the Stanford Sentiment Treebank dataset (Socher, R et. al, 2013). GRU won out with 86.32% accuracy (Yin, 2017). The sequential format inherent to text favors RNN's ands GRU's or LSTM models. Because of the structure of RNN's, the models are able to recall the words or segments from earlier in a body of text, many argue this increases their ability for NLP. RNN’s efficacy in this area is shown by the model's ability to predicate the ending of sequential data. This is a task that CNN's are not good at (Ghelani, 2019), but when working with short term text it makes up for it because of how important single items are in weighting.

**Complex NLP provides far better data representation of data.**

The challenge of communicating the nuance and complexity of human language has been an issue in the foreground of ML for years. Natural language processing is a field entirely dedicated to processing text with computers. NLP in this case will be studied in the form of the processing of data. Neural networks and ML models can only take in data in the form of matrices or vectors containing numbers. This fact makes it so that text data can not simply be fed into the model directly. This leads to the need for special methods of processing text data for ML. There is a wide range of methods all with their own advantages.

One part of NLP is cleaning text. This is where special character and information that is not helpful for the model is removed. A period is often not useful for short text processes as well as a `%` or `@` may not be useful. Cleaning text also comes up in the case of removing misspelled or repeated words. An example of this is “Helloooo” instead of “Hello”, ideally the extra letters at the end of the word are truncated so that the model sees that they have the same meaning. These decisions are heavily dependent on the use case.

 The basic flow of most NLP is first to tokenize data. As stated pure words cannot be used as input for ML models, so we need to represent these words with numbers. There are many methods of this but the easiest one to understand at first is basic word based integer tokenization. This approach pairs each word with a basic integer representation which is stored in a vocabulary. The vocabulary is a list of words mapped to integers, so for example ‘this’ could be paired with the integer `124`. This vocabulary is based on either the specific text corpus for the project or a general vocabulary is built for a given language. This provides the basis for many modern methods of NLP. Another method often used with these challenges is lemmatization, which is the process of simplifying words to their base meaning. It is not ideal to have the model learn that “excellent” and “exquisite” have a similar meaning if it can be avoided. This is why lemmatization is used: it maps words to their root meanings using a dictionary. So “excellent” would be mapped to “exquisite”. All words that have the same or similar meaning would be mapped to the same word. Along with that plurals are often removed. This minimizes memory usage and compute time while allowing the models to learn faster (Mehedi, 20201).

An older NLP method is using one-hot encoding. This is the method that shows why a more complex NLP is ideal. Once again in this case a dictionary is used containing all the words needed. This method takes a given piece of textual data and uses a vector of the length of the vocabulary to represent the text. This works because each index represents a different word, for example, index 125 might represent “basic”. If a word occurs multiple times then it is represented by the amount it occurred. This creates a few issues. One, the data is sparse, this means it is spread out in a vector. In the case of a tweet, there may only be 10 words that would need a vector for the whole length of the vocabulary to train. If the vocabulary length was 10,000 (a conservative estimate in many salutations), assuming all words were unique, 10 indexes would be used in a 10,000 element vector. This uses more memory and is harder for the models to learn. Some information is totally lost, such as what order the words were in. This information can be useful for a model and is used by RNN's.

The final and most complex but interesting method is word embedding. This is more complex as it requires a machine learning model to learn how the relationship between words before the main model processes the input data. An embedding layer in itself is a neural network. This neural network's input is tokenized words and its output is a dense vector in n-dimensional space representing that word. The advantage to this method is that the meanings of words and their relationships are better captured. If words are similar, ideally, they are closer to each other in that n-dimensional space. The grouping is somewhat like a KNN’s grouping where points' proximity represents their relation. This allows the model to recognize even more complex relationships between words. This paper will not go into the mathematics and varied implementations of Embedding in great detail, but will go into different options for pertaining embeddings.

Often Embedding layers are pre-trained, meaning that the embedding model has already learned relationships between words before being imported into the model. This saves time on training the main model. It also ensures that even if a data corpus is too small to teach proper embeddings the models still get logical input. Some of the more well-known pre-trained embeddings are GloVe, CSLM, and Word2Vec (Ghannay et al., 2015). Each of these pre-trained embeddings has strengths and weaknesses, some using more complex methods than others and using different corpora. Also, the model often trains its own embedding layer starting from random weights, like a normal neural network. As shown by Ghannay’s research the best embedding layer to choose from depends on the use case. In the case of part of speech tagging, it was shown the w2vf-deps performs best with 96.66% accuracy, followed by, GloVe 95.79%, Skip-gram 96.43%, and CBOW 96.01% (Ghannay et al., 2015). This displays how close these pre-trained models can often perform and because of this lack of major deviation in performance the choice of which one to choose is dependent on the use case.

The use case of Twitter sentiment analysis as has been established comes with its own challenges. The way the users spell is somewhat unpredictable with letters being repeated for emphasis and slang being used. This makes it more challenging to use a pre-trained model and some words not in the pre-trained model could lead to the model missing information. NLP can be any combination of leeminazation, pre-processing, and training a new embedding layer. What should be done varies based on the task at hand.

The industry has overall moved to complex NLP usually in the form of word embeddings when looking at a task like sentiment analysis. The end goal of these models is to represent as much information as possible so the model can learn from that information. Word embedding is the latest solution to that challenge, by grouping words in an n-dimensional space this method maximizes the information passed onto the model. For this reason, word embeddings should be used when classifying the sentiment of short form textual data.

**Older models like KNN and SVM, although they were useful, are now being replaced by ANN’s.**

Before the advent of the hardware and data to support ANN’s and all of their derivations, many other models took hold in the industry. Some of the earlier ones would not qualify as ML such as expert systems which ruled the early days of AI. There are many newer models that were used for classification some included KNN, logit boost, and Random Tree. There are many more early models, however, this paper can not go over all of them in detail. For that reason, the focus will be on KNN, a model which was implemented on Twitter text analysis by Mehedi(2021). In this research, KNN was used to analyze the sentiment of users towards different Covid-19 vaccines. The outcome was satisfactory for the research thesis (Mehedi, 2021). These older models are still effective and have use cases where they are the best implementation, however, for sentiment analysis of Twitter data they are not as useful as ANNs.

KNN or K-nearest neighbor is a mathematically concrete model when compared to ANNs. A KNN model is used for classification. The idea behind it is that each piece of data in the corpus of data is in an n-dimensional vector or matrix, with different elements represented by different attributes of the data. The n-dimensional data is placed onto an n+1-dimensional space. The distance between the data point and its closest neighbors is what is used to determine the prediction of the model. The radius of the circle drawn around a data point that qualifies others as close to a said point can be changed. Any of the data points in the new data points circle are combined and averaged to make a prediction. It becomes clear that the position of the data points in this N-dimensional space must correlate with its relationship to other data points for the model to work (Brownlee, 2019).

KNN can be used for sentiment analysis but it is no longer the ideal implementation. One reason for this is that KNN is a memory-based learning model. This means that it compares new data to old data to figure out how to classify the new data. This is compared to ANN methods which learn the features of the old data and those features correlate to a prediction. Also, a KNN model must have all of its data on hand to compare to new data. This means that the entire dataset must be loaded in memory for the model to function at any speed (T.Makkar 2019). The larger the dataset the larger the memory needed to train. An ANN only has to load its weights and whatever data it is using at that point into memory. This can not be applied to a Twitter sentiment analysis because the data corpus is so massive and loading that much data into random access memory (RAM) or the memory on a graphic processing unit (GPU) is not feasible. Another disadvantage of memory-based models like KNN is that they have a long prediction time. Training with a massive Tweet corpus would take a massive amount of time to make individual predictions at run time(T.Makkar 2019).

ANNs used to be comparatively near impossible when it comes to processing when compared to KNN or SVM. However, this has changed with new tools. ANNs have many units and have to do many basic multiplication and derivation problems for each. ANNs lend themselves to parallel processing; the network can train much faster if the math for each unit can be run at the same time. Before the 2000s the only options to process these basic problems were CPUs which had fewer cores and were built to be able to compute more complex problems than what was at hand. In the 2000s and going into the 2010s GPUs for gaming came into the market with far lower cost, less complex cores, for the exact same basic operations that models need to use to train. When compared to a CPU with 4 to 32 cores, a modern top of the line GPU can have 10,725 CUDA cores or more. The GPUs also have fast onboard memory which allows for faster training. This increase in hardware performance is needed as many new datasets are massive and in the use case of Twitter sentiment analysis, there is a large constant flow of data. This is a stream of live new data from the platform. This increase in the speed and size of the data makes this performance paramount. This relatively cheap GPU hardware made ANNs viable when compared to older models like KNN or SVM.

Another advantage that KNN and other simpler models used to have is the ease of programming them. A simpler model means that programmers can program it faster, this is an important metric to be considered in modern companies. Implementing these models from scratch efficiently is a major challenge. Implementation requires effective low level programming with the Cuda toolkit or another similar toolkit. The Cuda toolkit allows programmers to interface with GPUs at a low level, this allows programmers to take advantage of the GPU’s full processing capability (Nvidia, 2021). Even with Cuda ANNs were still far harder to implement and change quickly. This problem is solved by TensorFlow or a similar library like PyTorch. TensorFlow is an open-source library for deep learning made by Google. It allows for effective low level performance while abstracting much of the low level programming making it easier to implement ANNs.

The most important argument against older models is that they are less accurate. ANNs are also known for being flexible; for example, Deep Neural Networks (DNN) can both be trained on images and text. There is the disadvantage of the possibility of overfitting or underfitting with ANNs which does not exist in the same form in memory-based models. This overfitting can be combated by many methods. It is a whole area of study. One is Principal Component Analysis (PCA), in which the features with maximum deviation are calculated and normalized (Prabhakaran, 2019). A common tool in sentiment analysis, due to the tendency of a data set to be happier or more sad, is over-sampling or under-sampling. Over-sampling and under-sampling, at their most basic, is the process of repeating or removing data points to get rid of model bias.There is also Synthetic Minority Oversampling Technique (SMOTE) which uses KNN to produce near-real samples and increase the population of the minority class (Maheshwari & OpenGenus Foundation, 2021). Drop-out layers are another useful tool to prevent oversampling, they are layers in the models that randomly make a feature a zero, this prevents the model from being over-reliant on one feature of the data (Chollet, 2021).

There are many examples of the increased effects of ANNs in sentiment analysis. This is due to the fact that at the end of the day, ANNs and ANN derivatives are better at learning more complex representations (Zhuang et al., 2019). An example is the work of Jamal who showed that CNN's were 98% accurate when compared to other models that all were under 70% accuracy (Jamal et al., 2019). There are exceptions when older models do compare, but often they do not. Also, Twitter itself uses ANNs, specifically RNN's, for their data processing because of the advantages outlined (Zhuang et al., 2019). Also, the way data is processed has changed, in this use case, there is a constant flow of data. This size makes it viable for CNN. Overall, especially in the case of sentiment analysis, the argument has shifted to FFNN versus RNN.

**CNN Is the ideal method when compared to RNN networks like GRU LSTM for short term sentiment analysis.**

This section covers the most active part of this debate, FFNN’s or RNN’s for short text sentiment analysis. This paper argues FFNN is the ideal model for short text sentiment analysis and more specifically the convolutional neural network (CNN). CNN will be compared to two RNN architectures, long short term memory (LSTM) and gated recurrent unit (GRU). The view that CNN is best for sentiment analysis, is not a new one. There is a general thought that CNN is better than RNN's for classification (Yin, 2017). The debate originates from the fact that RNN's have dominated NLP recently. Due to their sequential modeling ability, they are better able to learn a representation of text. The text is a sequential form of data. A CNN is the ideal method for short term sentiment analysis because of the domain's importance of individual words without relation to the other's position.

 First RNN’s should be defined in more detail. As stated RNN is thought to be better for NLP because of its ability to perceive sequence in its data. There are many forms of sequential data outside of the text, some examples being audio, image captioning, language modeling, or any other time-based data (Ghelani, 2019). Text is a sequential form of data; it is often useful to be able to know what came before a word. This allows the model to perceive for example what adjective goes with what noun. This has proven to be exceedingly useful in complex NLP and has taken the field by storm.

 An RNN’s function on a somewhat basic idea, the units are not only given the output of the last layer but the data that was input to the first layer. Each unit takes this data into account when making its output. There are many derivations but that is the basic idea. The place where this extra data is stored is regarded as a context cell. This context cell holds data from the last step and that data feeds back into the unit (Staudenmyer, 2019). That seems easy, take in new data while remembering and taking into account the old data, but ideally, the model does not store all the data that has been passed through it in its units. Leaving this data with these units would lead to extremely high memory usage, a failure of the underlying math, and horrible performance. To solve these issues RNN's have been designed with a method of “forgetting” data. There is also the issue of the vanishing gradient. Although it will not be covered in detail, it is a very important part of RNN's and why new architectures have been created. The vanishing gradient problem entails that usually backpropagation error or the loss function tends to exponentially grow or shrink. This means that on average these basic RNN models cannot bridge more than 5 steps into the past (Staudemeyer, 2019). This leads to the failure of the network or unacceptable training times. This problem applies to both CNN's and RNN's but plays a key role in RNN's, as the vanishing gradient problem prevents them from recalling data (Dettmers, 2015). This does not discount RNN's from the argument since their introduction many architectures based on RNN’s have been created which serve to solve the vanishing gradient problem.

 One architecture proposed to solve the vanishing gradient problem is Long Short Term Memory (LSTM). LSTM takes advantage of a 3 gate system. These gates are really mathematical equations that are trained throughout learning. The first gate is an input gate that functions similarly to the input of a sigmoid unit but has an extra variable which was the original input data. The forget gate is the most important part of the LSTM unit; it determines how much to weigh the original data. Lowering the coefficient of the forget gate is equivalent to forgetting the data so that the unit will no longer take it into account. Forgetting data allows the model to avoid the vanishing gradient problem. The final gate is the output gate. The output gate takes into account the data from both the input and forgets gate combining it with its own weights to pass it to the next unit (Tan, 2018). This is a complex system when compared to a basic sigmoid unit but it is a very effective RNN.

 The newer model inspired by LSTM is the gradient recurrent unit (GRU) model. GRU is like a simplified version of LSTM. GRU only has two gates reset and update (Staudemeyer, 2019). Unlike LSTM, GRU does not use a memory block. The reset gate controls how much information is let into the unit. The update gate determines how much the last hidden value is taken into account. These combined to output a new hidden state. GRU is a comparatively simple RNN. Due to the simplicity of how GRU passes information, it often outperforms any competitor (Staudemeyer, 2019). It even beat out CNN and LSTM on sentiment analysis of Russian tweets in Mikolov’s (2014) study. This does not apply to all cases but GRU is LSTM's main competitor. GRU is also less compute-intensive than LSTM. It has fewer units to store and so takes up less memory. With those exceptions, GRU has all the other characteristics of an RNN. Its performance makes it CNN's main competitor. GRU is the leading model for textual analysis in general.

The other main ANN architecture is forward feed neural networks (FFNN). A forward feed network is closer to the ANNs introduction early. Most of the time deep neural networks (DNNs) are used as a sub-architecture of ANNs. DNNs are simply models where every unit is connected to every unit in the layer before, this leads to a far better representation. FFNN's unlike RNN's are loop free, the data only feeds forward. Each unit does not take into account any data other than what the units before pass to it (Staudemeyer, 2019). FFNN's have a simpler training method than RNN's although not by much. Both use some type of backpropagation. Backpropagation functions on the idea of finding the derivative of the error function or loss function. Having the derivative of the loss function allows the model to know which direction to go to minimize the loss. However, the model needs more than just the general derivative of the loss function to change the variables of each individual unit to minimize the loss function. This is where the chain rule comes into play. The chain rules allow the model to see the derivative of the loss function as the outcome of all of the unit derivatives chained together. From this, the model can see the derivative of not only the loss function but the derivatives of each unit's activation function. This means the model can change the attributes of each unit to minimize the error function. This is why all activation functions of units must be derivable. This training method is the basis of all ANNs.

FFNN's are an overarching design with many sub-architectures that are more performant for short form sentiment analysis. CNN's are the model that this paper argues is the ideal method of sentiment analysis. CNN's or Conventional Neural Networks are networks usually applied to image classification. The defining feature of these models is their ability to extract features from patterns in data. They extract this information using a convolutional network (Dettmers, 2015). A convolutional layer in a model is like a filter, each unit in a convolutional layer applies its own filter; an example could be a unit that filters for edges. Now going deeper the unit is really applying a learned matrix to the matrix of the image. This learned matrix is called a kernel. If the kernel is a 5 by 5 matrix and the data is a 25 by 25 matrix, the kernel is multiplied by 5 by 5 sections of the matrix. This outputs data with the kernel applied 25 times. These kernels extract information from the matrix and transform it into a more useful representation. Another layer used by CNN's is pooling layers. Pooling layers take a given *n* by *n* area of the input matrix and output the maximum or minimum value of the given area. These allow the information to be distilled more effectively (Detterms, 2015). The entity of the CNN architecture is built to recognize major features in certain patterns.

It is worth noting that CNN's are very flexible when it comes to what data they can intake. As mentioned CNN's are most often used with image data (Ghelanu, 2019). They can also take in other forms of data as long as it can be encoded in a way that works with ML models. For example, a CNN and all other FFNN's could take in user data, like how many followers a user has, to see how a tweet from the user should be recommended. An RNN is made for sequence data and so it would make no sense to input nonsequential data (Zhang, 2018). Also, CNN's can take in text data as well as other data at the same time, so only one model would be needed to process the data. An RNN would have to be put into a larger flow of models to make it possible to contribute to a system that does not only have nonsequential data. This increased flexibility makes CNN's more useful in many cases. For example, when Twitter wanted to know how to recommend tweets, it used a split model system with CNN (Zhuang, 2019). Twitter's model took many features into account and passed each data group through its own model.

A CNN's nack for recognizing major features in a pattern is what makes it useful for short sentiment analysis. This paper argues that many accurate scores can be extracted from single keywords signaling emotion. Due to the short nature of the text, there is often no space for winding ideas that intertwine both negative and positive words. Due to the size constraints, many micro-blogs have to get to the point quickly. This carries over to most sentiment analyses of short term text: it is term-heavy. This extends to Twitter.

There are cases when CNN's outperforms LSTM in NLP-related work. In certain experiments, CNN beat out LSTM, even in long-term text analysis (Yin, 2017). This research by Yin also shows that often RNN's are better for sentiment analysis challenges, but in the case of short form text, CNN's pulled ahead. Yin concluded that CNN performs best at lengths of text under 10 words. This is the spot where tweets often fall near. Ghelani (2019) supports this concussion, saying that CNN's are ideal for extracting local and position irrelevant features, while RNN's do better in long-term text. These sources argue local term and position irrelevant short term text sentiment analysis is best done with CNN's.

**Conclusion**

Due to the rapid increase in use and impact of microblogging platforms and other short form text, sentiment analysis of short form text has become an extremely important sub-section of NLP. Sentiment analysis allows for the understanding of how people operate and think when engaging with these platforms and the greater world. The importance of platforms like Twitter is rapidly expanding. Twitter is used for many purposes: entertainment, political campaigns, movements, or sharing information about an invasion. Using newfound AI expertise, we can analyze these massive amounts of data.

This leads to the need for better sentiment analysis of short text data. The ideal implementation is CNN with NLP preprocessing. Preprocessing with advanced NLP is worth the extra compute requirements as the initial representation of the text data is far more useful for models. ANNs in general are better for short term sentiment analysis than older models like KNN. Also, the advance in libraries for AI programming, like TensorFlow, have made more complex models comparatively easy to build. When it comes to ANNs, CNN's are the ideal choice when compared to RNN's like GRU and LSTM. This is because short-form text analysis prediction is heavily dependent on key terms, these key terms' positions are often irrelevant. A CNN's structure is ideal because of its ability for pattern recognition and its relative simplicity compared to GRU and LSTM. CNN's do not take into account order and instead look for known high importance patterns. CNN's are often more performant for short term sentiment analysis and are the ideal choice.

**References**

*About Keras* [Fact sheet]. (n.d.). Keras.io. Retrieved December 15, 2021, from https://keras.io/

 about/

Anderson, M., Barthel, M., Vogels, E. A., & Perrin, A. (2020, June). *#BlackLivesMatter surges on*

 *Twitter after George Floyd's death*. Pew Research Center. Retrieved October 13, 2021, from

 https://www.pewresearch.org/fact-tank/2020/06/10/

 blacklivesmatter-surges-on-twitter-after-george-floyds-death/

Assenmacher, D., Adam, L., Trautmann, H., & Grimme, C. (2020, May). Towards real-time and

 unsupervised campaign detection in social media. In *The Thirty-Third International Flairs*

 *Conference*.

Batrinca, B., & Treleaven, P. C. (2014). Social media analytics: A survey of techniques, tools and

 platforms. *AI & SOCIETY*, *30*(1), 89-116. https://doi.org/10.1007/s00146-014-0549-4

*BLAS (Basic Linear Algebra Subprograms)* [Fact sheet]. (29, June 6). netlib.org. Retrieved December

 15, 2021, from https://www.netlib.org/blas/

Browlee, J. (2019, August 12). A Tour of Machine Learning Algorithms.

 https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/

Chollet, F. (n.d.). *Dropout layer*. Keras.io/. Retrieved December 15, 2021, from https://keras.io/api/

 layers/regularization\_layers/dropout/

Dettmers, T. (2015, November 3). Deep Learning in a Nutshell: Core Concepts. *Deveoper Blog*.

 https://developer.nvidia.com/blog/deep-learning-nutshell-core-concepts/

Ghannay, S., Favre, B., Estève, Y., & Camelin, N. (2016). Word Embedding Evaluation and

 Combination. *LREC*.

Ghelani, S. (2019, June 2). Text Classification — RNN's or CNN's? *Towards Data Science*.

 https://towardsdatascience.com/text-classification- RNN's-or-cnn-s-98c86a0dd361

Gundecha, P., & Liu, H. (2012). Mining social media: A brief introduction. *2012 TutORials in*

 *Operations Research*, 1-17. https://doi.org/10.1287/educ.1120.0105

Hussein, D. M. E.-D. M. (2018). A survey on sentiment analysis challenges. *Journal of King Saud*

 *University - Engineering Sciences*, *30*(4), 330-338. https://doi.org/10.1016/J.JKSUES.2016.04.002

Jamal, Xianqiao, & Aldabbas. (2019). Deep learning-based sentimental analysis for large-scale

 imbalanced twitter data. *Future Internet*, *11*(9), 190. http://dx.doi.org/10.3390/fi11090190

Khan, M., Malviya, A., & Yadav, S. K. (2020). Big data and social media analytics- A challenging

 approach in processing of big data. *Lecture Notes in Electrical Engineering*, 611-622.

 https://doi.org/10.1007/978-981-15-7961-5\_59

Kirkorian, R. (2013, August 16). New Tweets per second record, and how! *Twitter engineering*.

 https://blog.twitter.com/engineering/en\_us/a/2013/new-tweets-per-second-record-and-how

Maheshwari, A., & OpenGenus Foundation. (n.d.). SMOTE for Imbalanced Dataset. *OpenGenus*.

 https://iq.opengenus.org/smote-for-imbalanced-dataset/

Mehedi shamrat, F. M. J., Chakraborty, S., Imran, M. M., Muna, J. N., Billah, M. M., Das, P., &

 Rahman, M. O. (2021). Sentiment analysis on twitter tweets about covid-19 vaccines usi ng NLP

 and supervised KNN classification algorithm. *Indonesian Journal of Electrical Engineering and*

 *Computer Science*, *23*(1), 463. https://doi.org/10.11591/ijeecs.v23.i1.pp463-470

Mijwil, Maad. (2018). Artificial Neural Networks Advantages and Disadvantages.

Nvidia. (2021). *CUDA Toolkit*. Nvidia.com. https://developer.nvidia.com/cuda-toolkit

Prabhakaran, S. (2019, March 23). Principal Component Analysis (PCA) – Better Explained.

 *machine Learning +*. https://www.machinelearningplus.com/machine-learning/

 principal-components-analysis-pca-better-explained/

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. (2013,

 October). Recursive deep models for semantic compositionality over a sentiment treebank. In

 *Proceedings of the 2013 conference on empirical methods in natural language processing* (pp.

 1631-1642).

Staudemeyer, R. C., & Morris, E. R. (2019). Understanding LSTM--a tutorial into Long Short-Term

 Memory Recurrent Neural Networks. *arXiv preprint arXiv:1909.09586*.

T. Makkar, Y. Kumar, A. K. Dubey, Á. Rocha and A. Goyal, "Analogizing time complexity of KNN and

 CNN in recognizing handwritten digits," *2017 Fourth International Conference on Image*

 *Information Processing (ICIIP)*, 2017, pp. 1-6, doi: 10.1109/ICIIP.2017.8313707.

Tan, W., Wang, X., & Xu, X. (2018). *Sentiment Analysis for Amazon Reviews*.

 https://cs229.stanford.edu/proj2018/report/122.pdf

Twitter Inverstor Relations. (2021, October 26). *Selected Company Metrics and Financials* [Table].

 Twitter Investor Relations. https://s22.q4cdn.com/826641620/files/doc\_financials/2021/q3/

 Final-Selected-Metrics-and-Financials.pdf

Varol, O., Ferrara, E., Menczer, F., & Flammini, A. (2017). Early detection of promoted campaigns on

 social media. *EPJ Data Science*, *6*(1). http://dx.doi.org/10.1140/epjds/s13688-017-0111-y

West, S. (2021, July 22). *Digital Activism: Social Movement on Social Media* (M. Li, Ed.). Psychology

 Today. Retrieved October 13, 2021, from https://www.psychologytoday.com/us/blog/

 understanding-the-social-world/202107/digital-activism-social-movement-social-media

Whiting, J. (2020). Tweets show what hinders reports of sexual assault and harassment on campus -

 and why the new federal title IX rules may be a step back. *The Conversation : Education*.

*Why TensorFlow* [Fact sheet]. (n.d.). TensorFlow.org. Retrieved December 15, 2021, from

 https://www.tensorflow.org/about

Yin, W., Kann, K., Yu, M., & Schütze, H. (2017). Comparative study of CNN and RNN for natural

 language processing. *arXiv preprint arXiv:1702.01923*.

Zhang, K. (2018). LSTM: An Image Classification Model Based on Fashion-MNIST Dataset.

Zhuang, Y., Thiagarajan, A., & Sweeney, T. (2019, March 4). Ranking Tweets with TensorFlow.

 *TensorFlow Blog*. <https://blog.tensorflow.org/2019/03/ranking-tweets-with-tensorflow.html>

Le and Mikolov (2014) Quoc V Le and Tomas Mikolov. 2014. [Distributed representations](https://deepai.org/machine-learning-glossary-and-terms/distributed-representation) of sentences and documents. In Proceedings of ICML. pages 1188–1196.