ANALYSIS ON THE TGA MODEL FOR STANCE DETECTION

An Undergraduate Research Scholars Thesis

by

JEFFREY XU

Submitted to the LAUNCH: Undergraduate Research office at
Texas A&M University
in partial fulfillment of the requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by
Faculty Research Advisor: Dr. Ruihong Huang

May 2022

Majors: Computer Science
Applied Mathematics

RESEARCH COMPLIANCE CERTIFICATION

Research activities involving the use of human subjects, vertebrate animals, and/or biohazards must be reviewed and approved by the appropriate Texas A&M University regulatory research committee (i.e., IRB, IACUC, IBC) before the activity can commence. This requirement applies to activities conducted at Texas A&M and to activities conducted at non-Texas A&M facilities or institutions. In both cases, students are responsible for working with the relevant Texas A&M research compliance program to ensure and document that all Texas A&M compliance obligations are met before the study begins.

I, Jeffrey Xu, certify that all research compliance requirements related to this Undergraduate Research Scholars thesis have been addressed with my Research Faculty Advisor prior to the collection of any data used in this final thesis submission.

This project did not require approval from the Texas A&M University Research Compliance & Biosafety office.

TABLE OF CONTENTS

ABSTRACT

Analysis on the TGA Model for Stance Detection

Jeffrey Xu
Department of Computer Science and Engineering
Texas A&M University

Research Faculty Advisor: Dr. Ruihong Huang
Department of Computer Science and Engineering
Texas A&M University

Stance detection, a problem concerned with finding the stance that an author takes on a specific issue, is a large subset of NLP and A.I, and its uses can already be seen in a multitude of applications. The majority of stance detection machine learning models are tested against a popular dataset called SemEval2016, which is a collection of tweets, authors, topics and stances that were derived from Twitter data and the Twitter API. Many researchers across the globe have created machine learning models to accurately predict the stance of authors based on their tweets regarding a certain topic. However, recently, researchers at Columbia university have created a new dataset called VAST along with a model called Topic-Grouped Attention (TGA), or better known as the TGANet, that claims to perform well on zero-shot and few-shot stance detection, which is a subset of stance detection that focuses on determining the stance of authors on new, never seen topics. Their VAST dataset focuses on this zero-shot and few-shot sub-problem by including a large variety of topics. This VAST dataset has many more topics than traditional stance detection datasets, which often focus on a particular subject to focus their topics around. In this thesis paper,

we analyze how the TGA model performs on the SemEval2016 dataset and determine whether the TGA model improves on the current existing zero-shot and few-shot stance detection models.

ACKNOWLEDGMENTS

Contributors

I would like to thank my faculty advisors, Dr. Ruihong Huang, and Yuanyuan Lei for their guidance and support throughout the course of this research.

Thanks also go to my friends, family and colleagues and the department faculty and staff for making my time at Texas A&M University a great experience.

The stance detection models and VAST dataset used for Analysis on the TGA Model for Stance Detection were provided by Emily Alloway and Kathleen McKeown from the department of Computer Science at Columbia University.

The SEMEval2016-Task 6 dataset used for Analysis on the TGA Model for Stance Detection were provided by Saif M. Mohammad, Parinaz Sobhani, and Svetlana Kiritchenko.

All other work conducted for the thesis was completed by the student independently.

Funding Sources

This undergraduate research did not receive any additional third-party funding.

1. INTRODUCTION

Natural Language Processing, or also known as NLP, is an ever-expanding subject within the field of artificial intelligence and machine learning. Specifically, stance detection is a sub-problem of NLP that concerns itself with determining the stance of an author on a specific topic or subject. This specific sub-problem of NLP can be applied to many industries and issues today. For example, data regarding on support for presidential candidates can be very hard to measure. However, with strong stance detection models, a general idea of what the general population thinks about a certain presidential candidate can be compiled and analyzed. This in turn can help presidential candidates better determine how to create their campaign to attract more voters. However, this is not the only use case of stance detection. Stance detection is often very useful when analyzing controversial events and the stances that the public takes on the specific event.

A quick survey on NLP and Stance Detection will be done in this section to give some background on the fundamentals of NLP/Stance Detection as well as provide some examples of datasets and models that have been developed and used for this task [3].

1.1 Stance Detection Basics

1.1.1 Machine Learning Fundamentals

At the base of artificial intelligence and its corresponding sub-fields is machine learning. Machine learning allows ideas and theories within artificial intelligence to come to life. At its very core, machine learning tries to model a specific set of data using a mathematical model. By using existing data, a model can be molded to fit the general trend and shape of the data. When new data is introduced, the model utilizes the structure of the original data to predict how the new data point should look like. This technique works very well due to the inherent structure that specific datasets and events have. For example, a model can be generated to predict annual spending amount based on yearly salary. Data can be fed to a basic machine learning model which

will create a mathematical representation of the structure of the data. Then, suppose that a yearly salary is introduced that is relatively high, the model could predict that the annual spending will also be higher. The nature of higher spending with higher salary is naturally represented within this basic model.

1.1.2 Stance Detection Dataset Structure

Data is the driving factor within machine learning and artificial intelligence. A large amount of good data will allow for high-quality models to be produced. Stance detection specifically has data that must follow a very specific structure. Since stance detection requires the prediction of a stance on a specific topic, each data point must have a topic as well as a stance on that topic. There may be additional values included for each data point within a dataset geared towards stance detection but these values are required for a traditional stance detection dataset to be properly used.

The stance can vary based on the specific sub-genre of stance detection that one is trying to use. However, the most common labels that are utilized are {*favor, against, neither*}. Other common labels include {*positive, negative, neither*}. Sometimes, the *neither* label is called *none* to emphasize that the stance isn't quite related to the topic.

Then, there needs to be a piece of data that allows a machine learning model to predict what the stance/label is on the specific topic. Thus, a document written by an individual is provided. This text is often not too long and should somewhat relate to the topic that is being discussed. These documents are often scraped from Twitter due to the convenience of the Twitter API along with the large amount of Tweets that are available for machine learning and artificial intelligence. However, many different datasets derive their documents from a variety of sources. Some other popular platforms are social media platforms, such as Reddit and blogs, news articles, and any platform where users are allowed to openly express their opinion on certain topics.

Therefore, to summarize the structure, each data point consists of a document d, a topic t, and a stance y. Putting all of these elements into a tuple/vector, a complete dataset looks like

Equation 1.1, which is shown below.

$$D = \{x_i = (d_i, t_i, y_i) | \forall i = 1, ..., n\} \tag{1.1}$$

1.1.3 Stance Detection Datasets

1.1.3.1 SemEval-2016 Task 6

Currently, there are already a lot of existing datasets for stance detection. These datasets are also open source, which means that anyone can access the data and use it. This allows for a standard to be developed across the stance detection community and the performance of different stance detection models can be directly compared. One of the most popular natural language processing datasets is the SemEval-2016 Task 6 dataset [5]. This dataset is specifically tailored for stance detection and consists of a large amount of labeled Twitter data. The researchers that put together this dataset have also given a complete breakdown of the data which allows for further analysis of performance when generating machine learning models.

The SemEval-2016 Task 6 dataset was actually originally designed as a competition amongst researchers to generate models that could achieve high performance on predicting the stance. There are two tasks associated with the SemEval-2016 Task 6 dataset.

1.1.3.2 SemEval-2016 Task 6 Sub-task A

The first sub-task of the SemEval-2016 Task 6 competition is a supervised stance detection problem. In this context, supervised simply means that each data point has a label assigned to it. Within sub-task A, there is a training dataset of 2,814 tweets along with a testing dataset that contains 1,249 tweets spread amongst 5 topics. The big advantage with supervised learning is that the labels can immediately generate a loss function that can optimize the model in a clear and straight-forward manner. The method of this optimization is called back-propagation, which involves generating a loss function that measures the performance of the model and using the gradients of the loss function, the weights within the model can be adjusted to hopefully achieve

better performance. Back-propagation utilizes partial derivatives with respect to the weights of the model in order to approximate the underlying structure that the dataset follows.

The major issue with supervised learning is that the datasets need to have labels generated for them. This process of generating labels can be a very time-consuming and tedious job as each data point needs to be individually analyzed by a human and an accurate label must be produced. Especially when working in such a vague area as natural language processing, linguistic phenomena such as sarcasm, internal references, and other literary techniques require a more in-depth analysis of what the labels should be.

1.1.3.3 SemEval-2016 Task 6 Sub-task B

The second sub-task of SemEval-2016 Task 6 involves weakly supervised stance detection. Within this task, there is a large dataset of about 78,000 unlabeled tweets along with a testing dataset of about only 707 tweets. Both of these sets are unlabeled which is why this sub-task is categorized as a weakly supervised stance detection problem. Though unsupervised learning may be to be much harder to perform since the dataset does not contain any labels, it is much easier to generate a larger dataset. In the case of SemEval-2016 Task 6, large amounts of Twitter data were able to be brought together to generate the dataset.

Often times, unsupervised datasets are used to see if models can find some underlying structure with the dataset. This often comes in the form of clustering the data to see which values are similar based on their values.

1.1.4 VAST Dataset

Another important dataset is the VAST dataset that was created by Alloway and McKeown from Columbia University [2]. VAST stands for VAried Stance Topics and is targeted for zero-shot and few-shot stance detection.

Zero-shot stance detection is another sub field of stance detection. Zero-shot learning involves training models that can predict stances for a topic that the model has not seen a lot of

training points for. Thus, it is similar to transferring a model over to a new dataset, but in this case, the only difference is that the topics are a relatively new topic compared to the topics that were used in training. Creating a strong zero-shot stance detection predictor has many important use cases. This is because there are lots of topics that could be covered by a stance detection model and it is not practical to try to train a model that covers a majority of topics. Thus, a model that is trained on a smaller subset of topics that can transfer over to other topics with a high prediction rate is desirable due to the simplicity of the design.

Few-shot stance detection comes from a similar motivation to zero-shot stance detection. However, few-shot stance detection only makes the assumption that a specific topic had very few examples in the training data, instead of never being seen in the training phase.

The VAST dataset tries to replicate these two types of learning by having a variety of topics for training and testing and trying to replicate the scenario of predicting the stance on a topic that the model hasn't seen before or hasn't seen many training samples for.

1.1.5 Other Stance Detection Datasets

In this thesis, we will be mainly focused on the SemEval-2016 Task 6 dataset as well as the VAST dataset. However, there is a multitude of stance detection datasets available for use on the internet. Many of these datasets go deeper into other sub fields of stance detection.

1.1.5.1 LIAR

For example, LIAR is a dataset specifically designed for fake news detection, another sub-genre of NLP. In this dataset, about 12.8K labeled short statements were collected from *politifact.com* and is much larger than previously generated fake news datasets. Since the dataset is labeled, it falls under the umbrella of supervised learning.

1.1.5.2 Foreign Language Stance Detection Datasets

Another interesting stance detection dataset is called x-stance, which contains large amounts of stance detection data in languages other than English. It includes languages such as German,

French, and Italian. The dataset consists of comments written by candidates of elections in Switzerland and covers a large variety of political issues. This type of dataset is especially useful when expanding models to different languages since there are a lot of documents on the internet that could help stance detection research that isn't written in English.

RuStance is another data containing foreign language data. This dataset contains tweets and news comments from Russian sources. Specifically Meduza and Russian Today, two new and media platforms, were used to help collect data about Russian political issues.

1.1.5.3 Niche Datasets

There are other more specific stance detection datasets that mainly focus on a specific topic or subcategory. For example, the CIC (Catalonia Independence Corpus) dataset has data that is soley focused on the independence of Catalonia. There is also a dataset relating to the 2020 United States presidential election by Donald Trump and Joe Biden.

Another niche dataset is CoVaxLies v1, which only contains Twitter data about the COVID-19 vaccines. However, this dataset only contains Twitter tweet id's and not the actually tweet itself. Instead, the creators of this dataset have requested everyone to individually obtain the tweets by using the Twitter API.

1.1.6 Data Preprocessing

Before the data from a stance detection dataset can be fed to a machine learning model, some preprocessing steps must be done in order for the machine learning model to be able to train and predict with it.

1.1.6.1 Remove of Specific Tokens and Characters

To begin with, the text must remove unnecessary tokens and characters. This often includes punctuation, stop-words, emojis, and other non-recognizable text that may be within the document.

Other pieces of data such as username tags in the form @username and URLs are also removed as they do not provide any NLP meaning when a model is training and predicting. There-

fore, connections and context through hashtags and usernames are lost during the preprocessing phase.

1.1.6.2 Normalization

Often times, the documents that are being used within these datasets for training and testing have misspelled words and contracted forms. These tokens may cause issues during the training and testing phase and must be removed. These errors happen most often in Twitter data due to the colloquial nature of the platform. This step within the preprocessing phase is referred to as normalization.

1.1.6.3 Case Conversion

Since entire documents are being processed, capitalization is something to be considered since words with and without capitalized letters map to different ASCII characters. Thus, in order to keep tokens consistent, all tokens are converted to either uppercase or lowercase. This allows some consistency across the data.

1.1.6.4 Tokenization

After the steps listed above, the documents must be tokenized before entering the machine learning models for the training and testing phase. Thus, the filtered text that remain is partitioned into individual tokens based on the language that is being used and mathematically mapped to vectors to feed into any specific model.

1.1.6.5 Tools for Preprocessing

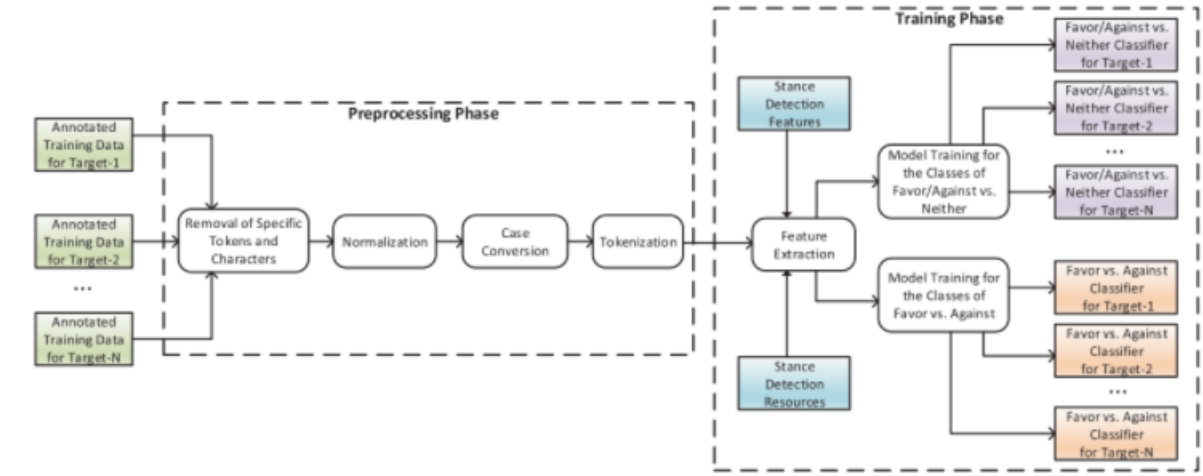
More likely than not, some open source module or library is used to perform preprocessing. This is because most preprocessing is done similarly across different stance detection problems and datasets. This allows for some consistency when comparing the standard of models since the same preprocessing process gives similar input vectors to the model.

Some popular NLP tools that are utilized during the preprocessing phase includes TweetNLP,

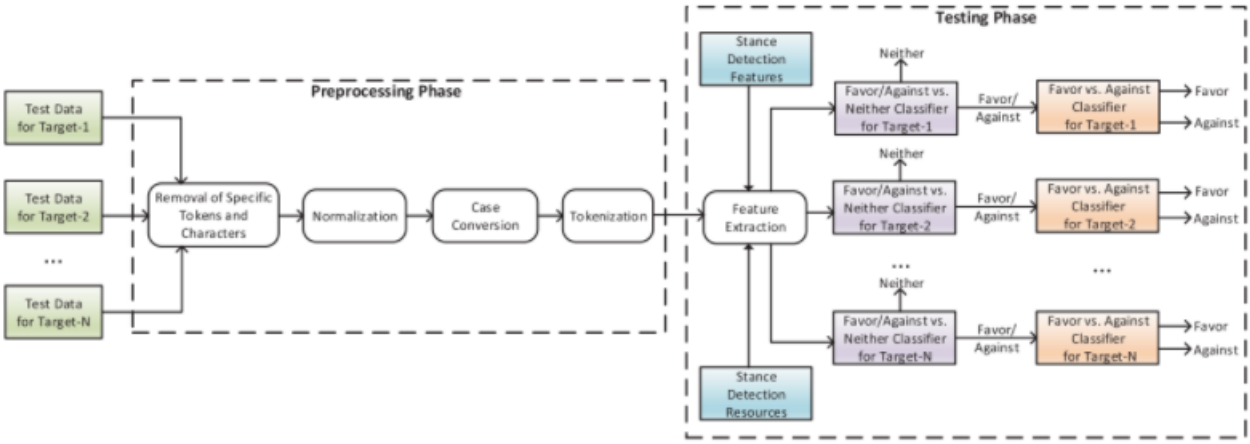
Stanford CoreNLP, and NLTK. Within this paper, we will mainly be using NLTK due to it's ability to perform NLP preprocessing on a variety of documents and it's compatibility with the system setup that is used.

1.1.6.6 Overview

A general overview of a typical stance detection model is shown in Figure 1.1. The preprocessing phase is the same for both, but the training phase includes back-propagation which allows the model to train.



(a) Training phase.



(b) Testing phase.

Figure 1.1: The architecture of a generic stance detection system and model.

These steps occur assuming that there is labelled data available. The types of models that can be trained will be discussed in more depth in the next section of the introduction.

1.2 Stance Detection Models

There are a variety of models and model types that can be utilized when performing stance detection. All the machine learning concepts are used when generating models, but we will cover

the main types of models that are popularly used within the field of stance detection.

1.2.1 Feature-based Machine Learning Models

The method of generating feature-based machine learning models is very common. In the next couple of subsections, we will discuss some of the more popular feature-based stance detection models that are used currently in the space of stance detection.

1.2.1.1 SVM

Support-vector machines are commonly used as stance detection models. For many papers, SVM's have served as the main best-scoring model or as a baseline to which other approaches and different machine learning/deep learning models are compared against. Over 40 studies actually have the SVM as a high performer for stance detection tasks. Although the SVM performs very high and is often used as a baseline, it is extremely computational expensive to use SVMs in practice since it takes lots of computational power and memory to perform gradient descent on SVM models. Thus, it is often desirable to find other models that can perform similarly to SVMs while taking less computational power and efforts to train and evaluate on datasets.

1.2.1.2 Logistic Regression

The Logistic Regression model is the second most popular feature-based machine learning model for stance detection, right after the Support Vector Machine. Despite its simplicity, the logistic regression model is used in many studies as the main classifier. Since it is a relatively simple model, it is also incorporated into ensemble learning models which will be covered in more depth in later sections.

1.2.1.3 Naive Bayes

The next most popular feature-based machine learning model is the naive Bayes classifier. Based on the current studies, it has appeared in more than 10 studies. The naive Bayes is another simple and straight-forward model which is why it is preferable to use within many studies.

1.2.1.4 Other Feature-Based Models

There are many other models that are commonly used in stance detection studies. The decision tree is also popularly used. However, the decision tree is often incorporated into more powerful ensemble learning techniques such as the random forest classifier in order to obtain better results for stance detection tasks.

The Artificial Neural Network, or better known as just the neural network, is also commonly used within stance detection research. These models utilize the multi-layer perceptron to create a network of nodes that act as a classifier. However, only a couple of studies have successfully utilized the basic neural network as there are more powerful deep learning techniques that can perform better on stance detection tasks.

Along with these models, there are even more models and techniques that are commonly used that will not be discussed in detail in the introduction of this thesis. These will be listed below.

1. Inductive Logic Programming
2. k-Nearest Neighbors
3. Log-Linear Model
4. Maximum Entropy
5. FastText
6. Stochastic Gradient Descent (Method of Optimization during learning)
7. k-Means Clustering
8. Factorization Machines
9. Multiple Convolution Kernel Learning
10. Matrix Factorization

11. Statistical Relational Learning

12. Weakly-Guided Learning

It should also be noted that some studies have utilized **active learning** for unlabeled datasets. This is very similar for the main model that will be tested in this thesis as zero-shot stance detection involves training models that can perform well with little information about the topic in the training data.

1.2.1.5 Common Features

Some of the most common features used to train feature-based machine learning models for stance detection are listed below. This list, however, is not an exhaustive list of all features that are implemented in stance detection models, rather the most common ones that are used in studies today.

- **Lexical Features:** This includes features such as bag-of-words, word and character ngrams/skip-grams, hashtags, stance indicative words, theme and context words, synonyms, punctuation marks, and post length.
- **Social Media Posts:** There are lots of metadata available for social media posts. This can include replies, likes, reposts, and other metrics that relate to the document that don't consist of the contents of the document itself.
- **Sentiment-Based Features:** These features relate mainly to the sentiment and emotion that is present within the document. This can be extracted using sentiment classifiers which try to classify the emotion/sentiment that is associated with a specific document on some particular target.
- **Word Vectors:** Word vector representations such as word2vec and GloVe vectors as well as paragraph vector representations such as para2vec. These essentially take documents and

transform the single tokens in the document to a vector representation for simple use for machine learning models to train on.

- **Topic Modeling:** These features are based on popular topics in the text. These features can be extracted using models such as the Latent Dirichlet Allocation, Latent Semantic Analysis, and TF-IDF vectors for lexical features.

- **Syntax:** Finally, there can be features that are based on POS tags, named entities, dependency relations, syntactic rules and co-reference resolutions.

1.2.2 Transform Learning

Normally, machine learning and deep learning models take in raw data and extract specific features and represent these features in some specified manner. This type of learning can be very limited since the weights for the model must be learned, but the outputs and weights are unknown during training. This type of learning is traditionally called representation learning.

1.2.3 Deep Learning Models

Similar to the feature-based machine learning models, it is very common to use deep learning models for stance detection tasks and studies. In this section, some of the most popular deep learning models used in studies will be discussed. There are two main types of deep learning techniques used when studies are done in the field of stance detection: Recurrent Neural Networks and Convolution Neural Networks.

1.2.3.1 Recurrent Neural Networks

Recurrent Neural Networks generally can characterize temporal data. Recurrent Neural Networks do this by having a form of memory when data is passed through them. Thus, these neural networks can utilize temporal data to make predictions. This is very powerful for analyzing natural language since the tokens throughout the sentence can give context on other tokens within the document.

1.2.3.2 LSTM

The most popular type of recurrent neural network that is used with the field of stance detection studies is the Long-Short Term Memory recurrent neural network. The LSTM neural network can remember values over arbitrary time intervals and has gates that regulate what information is remembered and what isn't when data is passed into and out of each gate/node. Figure 1.2 gives a visualization of the structure of a simple LSTM model.

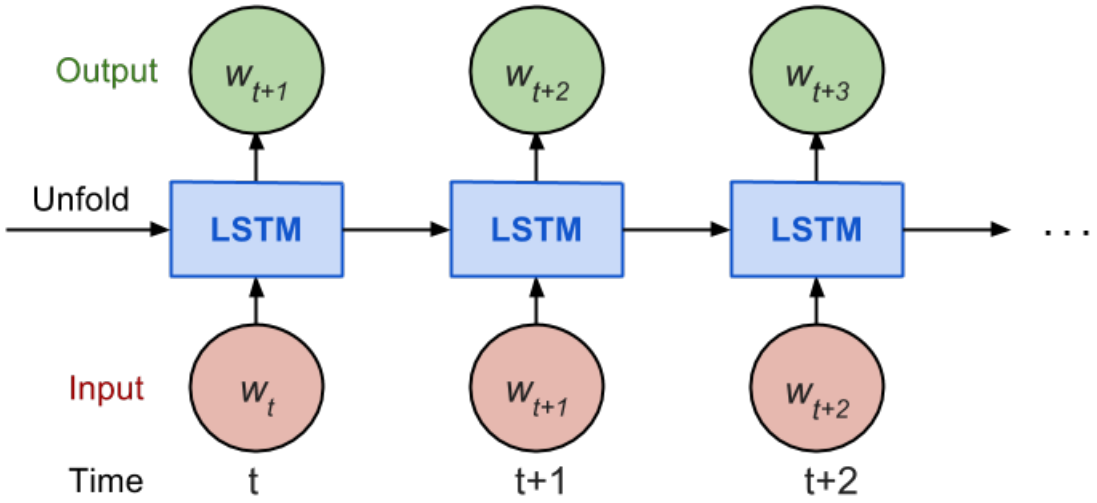


Figure 1.2: The general structure of a LSTM neural network

The LSTM model is a frequent model used in stance detection studies as it has been used in over 10 stance detection studies. Due to it's high performance overall in natural language tasks, the LSTM model is a popular one to use in stance detection tasks as well.

1.2.3.3 Gated Recurrent Unit

Along with the LSTM recurrent neural network, the Gated Recurrent Unit (GRU) is another popular that is used as the main or baseline model for stance detection studies. The GRU is actually

very similar to the LSTM recurrent neural network, but is missing some gates. Thus, the GRU is a simpler model than the LSTM recurrent neural network model and is often used in ensemble learning techniques.

1.2.3.4 Convolutional Neural Networks

A convolutional neural network focuses on image analysis. It can take in image data and determine specific objects and subjects within the image. This is useful for finding patterns in multidimensional datasets such as images which often have a 2D-matrix of pixels, each which has an RGB-value associated with the pixel.

A Convolutional Neural Network achieves these tasks by reducing images to a compressed form without losing information about those patterns of the objects in the original image. It does this by running the image through multiple filters that compress the information within the image down while preserving these properties.

The convolutional neural network is the second most frequently used deep-learning model within the field of stance detection right after the recurrent neural network models. This is mainly because convolutional neural networks aren't commonly used to analyze natural language in this regard.

1.2.3.5 BERT

The BERT model stands for Bidirectional Encoder Representations from Transformers. The BERT model utilizes a Masked Language Model (MLM) within the pre-training phase. Thus, certain tokens are masked and the objective of the BERT model is to predict the original vocabulary based on context. The Masked Language Model allows for the representation to fuse the left and right context. This allows a deep-bidirectional transformer to be trained for this task [7].

Within this thesis, a BERT model is used as a baseline model to compare the results of the TGANet model that was proposed by Alloway.

1.2.3.6 Deep Learning Features

For deep-learning models, there isn't as wide of variety when it comes to the features that are inputted into the models. This is because the model itself should be able to extract the deeper features present in the data. Thus, the main features that are used for deep-learning models are word vector representations such as word2vec, word embeddings, and word/character ngrams based on sentiment lexicons.

Transform learning is different from representation learning in that it learns a basis such that it synthesizes data from the learned coefficients. Essentially, the learned basis is used to produce the coefficients. Refer to the appendix on an introduction to linear algebra and basis. Thus, transform learning can be represented as $TX \approx Z$ where T represents the analysis transformation, X the data, and Z the corresponding coefficient matrix [6].

1.2.4 Ensemble Learning

Ensemble learning techniques combine the previous two sections (feature-based machine learning and deep learning models and techniques) together to generate more powerful classifiers. Many ensemble learning models utilize multiple models together and basically vote the outcome that the majority of the models outputted. Therefore, the strengths of each model can be utilized within ensemble learning models. This allows the weaknesses of each model to be covered up from the other models as each individual model isn't the determining factor as to what the ensemble learning model outputs during classification tasks.

1.2.4.1 Random Forest

The random forest classifier is a very popular ensemble learning model that is used within stance detection studies. A random forest is essentially a group of decision trees that are combined together to help serve as a classifier. The random forest has appeared in over 10 studies regarding stance detection and is the most popular ensemble model used in stance detection tasks and competitions.

1.2.4.2 Priority Ensemble Learning

A priority ensemble learning model is essentially a group of different learning models combined together to create a generic ensemble learning model. These class of ensemble learning can include feature-based machine learning and deep learning models. Many studies like to combine the different types of deep-learning models such as the recurrent neural networks and convolutional neural networks. A combination of deep-learning and feature-based machine learning models are also used to create ensemble learning models. There are also studies that utilize boosting and bagging for ensemble learners.

Boosting allows the generation of very strong ensemble learning models from extremely simple models, such as the XgBoost model which is a group of shallow decision trees. Bagging is a method to generate more data from the original training data. This is essentially data-sampling to generate more data and reduce the variance during predictions and classifications.

1.2.4.3 Transfer Learning

A major genre in machine learning and AI is the idea of transfer learning. The essential concept of transfer learning is training a model on a specific dataset or distribution, and then transferring that model to another dataset or distribution. This type of learning is especially useful when training a model takes a significant amount of time. Thus, training one model that performs well on a multitude of datasets proves to be extremely useful since it produces robust models while saving time and computational power.

1.3 Evaluation Metrics

Evaluation metrics are important when standardizing any task or study. Since there are many stance detection competitions, there are already many standardized metrics for evaluating the performance of models on competition datasets. In this section, popular metric functions will be reviewed.

1.3.1 Accuracy

The most simple method for an evaluation metric is the accuracy metric. This metric simple just counts the number of correct classifications over the total number of classifications that were completed. Equation 1.2 gives the exact equation for accuracy.

$$Accuracy = \frac{Correct\ Classifications}{All\ Classifications} \tag{1.2}$$

Although this method is simple, it doesn't tell the whole story of how well a classifier actually performs on some dataset. This is because it does not model label class size in-balance well. If there is one dominating label class and the classifier does a good good of classifying it but classifies the other label class wrong every time, then the accuracy would be good but the model does not actually perform well on the dataset as it essentially just outputs the same classification for all classifications. The next section will introduce a better evaluation metric that takes into account the possible in-balance of label classes.

1.3.2 F-Score

The F-score depends on the precision and recall of prediction data. The equation for precision and recall are shown below. Here, TP denotes "True Positive", TN denotes "True Negative", FP denotes "False Positive", and FN denotes "False Negative". Equation 1.3 gives the equation for precision, and Equation 1.4 gives the equation for recall.

$$P = \frac{TP}{TP + FP} \tag{1.3}$$

$$R = \frac{TP}{TP + FN} \tag{1.4}$$

The F-score is essentially the harmonic mean of the precision and recall. Equation 1.5 gives the exact equation for calculating the F-score given the precision and recall values.

$$F = \frac{2 * P * R}{P + R} \tag{1.5}$$

This computation is then done for each output class, which includes *favor, against* and *neither*. Thus, the macro F-score just becomes the average of the F-score for each distinct output label class, as shown in Equation 1.6.

$$F = \frac{F_{Favor} + F_{Against} + F_{Neither}}{3} \tag{1.6}$$

The F-score is a good evaluation metric since it accounts for unbalanced label classes amongst the dataset and solves the problem that occurred with the accuracy evaluation metric. For many studies and competitions, it is common to use the F-score metric to evaluate performance of models. Within the TGA Model and the SemEval2016-Task 6 dataset, both use the F-score metric to measure model performance.

Within this thesis, the F1-score, precision and recall scores are all displayed from each model for each label class.

2. METHODS

2.1 TGA Model

The main model that will be tested in this thesis is the TGA Model developed by Alloway and McKeown from Columbia University. Thus, this section will take a deeper dive into the TGA model.

2.1.1 Motivation

The TGANet model utilizes topic representations on the data to get more information from other topics that may be present in the document as well as try to utilize context to help during the evaluation process. By utilizing these topic representations, the TGANet model will hopefully perform well on zero and few-shot stance detection. This is a field of stance detection where the topics either have never been seen before or very few training examples were shown for the specific topic.

Currently, many models only utilize what the document and topic provides and tries to predict the stance based on these alone. However, lots of useful information is disregarded here. Thus, the TGANet attempts to utilize context and generalized topic representations (other topics that may not be explicitly included in the data point but can help predict) to perform better on zero and few-shot stance detection problems.

The goal of the TGANet model is to develop a model that can recognize and develop relations between training and evaluation topics without any supervision [2]. Thus, this model should theoretically perform well on zero and few-shot stance detection problems.

2.1.2 Generalized Topic Representations

The first component of the TGA model are the Generalized Topic Representations (GTR). For starters, the document and topic are vectorized into the form $v_{dt} = [v_d; v_t]$. Here, $v_d, v_t \in \mathbb{R}^E$

where E is some embedding dimension for the vectors. Then, a hierarchical clustering algorithm is used on v_{dt} to obtain centroids of the cluster classes, r_{dt}. The value r_{dt} is the generalized topic representation for the document and topics.

2.1.3 Topic Grouped Attention

The generalized topic representations, r_{dt} will be used to compute the the similarity between topics in the dataset. The scaled dot-product attention method is then used to compute similarity scores s_i [1]. These similarity scores are then used as weights to determine the importance of the current topic tokens $t^{(i)}$ with respect to the current topic that is being used for stance detection classification. These are combined to obtain some representation c_{dt} that quantifies the relationship between the current topic t and related topics and documents.

$$c_{dt} = \sum_i s_i t^{(i)}, \quad s_i = softmax(\lambda t^{(i)} \cdot (W_a r_{dt})) \tag{2.1}$$

Here, $W_a \in \mathbb{R}^{E \times 2E}$ contain learned parameters and $\lambda = 1/\sqrt{E}$ is the scaling value for the scaled dot-product attention.

2.1.4 TGA Model Prediction

The previous two sections were concerned with produces additional features that could help provide more insightful information about the inputted data. This section covers the actual model prediction process that the TGA model undergoes when performing classification for stance detection tasks.

2.1.4.1 Model Usage

The actual model that is used within the TGA model is a simple feed-forward neural network. The output of the feed-forward neural network are the probabilities for each label class. We denote this as $p \in \mathbb{R}^3$ where each element of p such be in the range $[0,1]$.

Thus, we can mathematically write out the process of the TGA model using the equation

that follows.

$$p = softmax(W_2(tanh(W_1[\tilde{d}; c_{dt}] + b_1) + b_2)) \tag{2.2}$$

Here, $\tilde{d} = \frac{1}{n}\sum_i d^{(i)}$. The parameters $W_1 \in \mathbb{R}^{h \times 2E}, W_2 \in \nVdash \times \eqsim, b_1 \in \mathbb{R}^h$, and $b_2 \in \mathbb{R}^3$ are learned parameters during the training phase of the stance detection machine learning pipeline. During the training phase, the cross-entropy loss serves as the loss function.

Figure 2.1, which is shown below, gives an general outline of how the TGA model is structured.

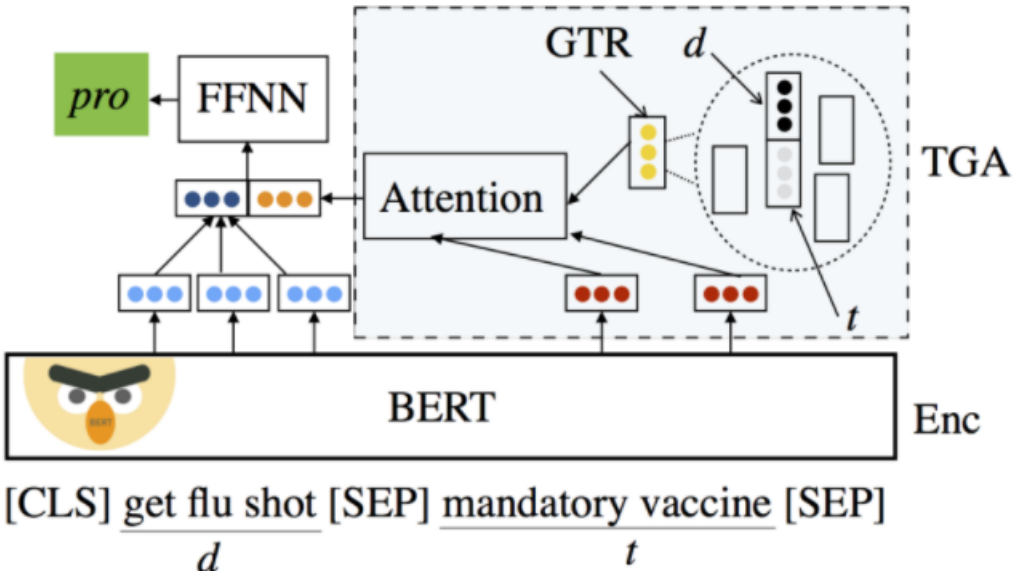


Figure 2.1: Structure and architecture of the TGANet model proposed by Alloway and McKeown. Enc indicates contextual conditional encoding. GTR stands for Generalized Topic Representations. TGA stands for Topic-grouped Attention.

2.2 VAST Dataset

The VAST dataset was also developed by Alloway and McKeown and served to test the TGA model. Within the study, the VAST dataset underwent a 70/30 split for training and testing/development data. The wide variety of topics really pushes the models to test for zero-shot stance detection.

2.2.1 Data Preprocessing

The preprocessing that occurred for the models developed by Alloway and McKeown followed standard preprocessing methods for general natural language and stance detection studies.

Alloway and McKeown started by tokenizing the docuements and topics. Then stop words and punctuation were removed. All of the data preprocessing was done using the NLTK python library.

2.3 Baselines and Models

Within their study, Alloway and McKeown utilized a variety of models and baselines. The list below highlights all of the models that were used in the study.

1. **CMaj:** Outputs the majority class that was computed from each cluster in the training data

2. **BoWV:** Separate BoW vectors were constructed for the document and topic. They were then concatenated and passed into a logistic regression classifier

3. **C-FFNN:** A simple feed-forward neural network that was trained on the generalized topic representations discussed in the previous section

4. **BiCond:** A model used for cross-target stance detection. Bi-directional encoding is used on the topic using a Bi-directional LSTM neural network to generate h_t. The document is then also encoded using a second bi-directional LSTM neural network conditioned on h_t. Pre-trained word-embeddings are used for this model.

5. **CrossNet:** Similar to the BiCond model. Encodes the document and topic using the same bi-directional encoding method as BiCond, but adds an aspect-specific attention layer before classification. Improves on BiCond in many cross-target settings.

6. **BERT-Sep:** Encodes the document and topic separately using BERT (similar to BiCond and CrossNet), then performs classification by inputting the encodings into a two-layer feed-forward neural network.

7. **Bert-Joint:** Performs contextual conditional encoding followed by a simple two-layer feed-forward neural network.

8. **TGANet:** The proposed model introduced in the previous section developed by Alloway and McKeown. Utilizes contextual conditional encodings and topic-grouped attention to better perform on few-shot and zero-shot stance detection.

2.4 SEMEval2016-Task 6 Data Conversion

2.4.1 Column Filtering

In order to use the SEMEval2016-Task 6 dataset, the format of the data had to be converted to match the input for the VAST dataset. However, there were many additional/unnecessary fields included in the VAST dataset (used for diagnostics of the VAST dataset). Thus, the first step was to determine which columns weren't needed by the TGANet model. This was done by generating null values for a specific column and determining if the TGANet model could run with that column containing null values. After filtering out the columns that weren't needed, it was determined that the remaining columns that were needed by the model could be generated from the columns given by the SEMEval2016-Task 6 dataset.

2.4.2 Data Conversion

The SEMEval2016-Task 6 dataset only contains the raw text string of each topic and document. First, the raw text from the SEMEval2016-Task 6 dataset needed to be stripped of punctua-

tion and everything must be converted to lower case. After doing so, the numeric values in the text must also be removed. Then, the text is lemmatized and tokenized. Finally, stop-words are filtered out of the list of lemmatized tokens. This process is done for the topic and the document alike.

The labels followed the formatting discussed in Alloway's paper. The raw text form of the document was also inputted into the dataset.

2.5 Model Testing

The dataset was only tested against three models. These models were the TGANet model, the C-FFNN model, and the BERT-Sep model. These models were evaluated on the SEMEval2016-Task 6 data and the metrics were recorded. The other models couldn't be evaluated on the SEMEval2016-Task 6 dataset because they required columns that the SEMEval2016-Task 6 dataset didn't provide. These columns couldn't be extracted from the raw SEMEval2016-Task 6 data since those values depended on the distribution of the VAST dataset. We can assume that the SEMEval2016-Task 6 dataset follows a different distribution compared to the VAST dataset. Thus, the only way to obtain those values for the SEMEval2016-Task 6 dataset is to replicate the process of generating those values for the VAST dataset.

As described in the section covering the technical details of the models, there are some columns within the VAST dataset that rely on the clustered topics from the VAST dataset. Thus, specific indexes and values were extracted from the VAST dataset and used to generate features with respect to the VAST dataset. These features cannot be generated from the SEMEval2016-Task 6 dataset since those clustered values and indexes were only generated with the VAST dataset and only have meaning with respect to the VAST dataset and the underlying distribution that the VAST dataset follows. Replicating these values for the SEMEval2016-Task 6 dataset might also cause issues since the models generated by Alloway were using the VAST dataset and relied on the distribution that the VAST dataset followed. By using the SEMEval2016-Task 6 distribution to generate these values, the models that utilize these values may not predict very well and reflect the

actual performance of the model.

The transferred learning metrics generated from the SEMEval2016-Task 6 dataset on the three models selected will be explored in further detail in the results section.

3. RESULTS

As noted in the methods section, the TGANet, C-FFNN, and BERT-Sep models were evaluated on the SEMEval2016-Task 6 dataset. The models were evaluated using the macro F1-scores as well as the F1-scores for each individual label classes. It should be noted that since transfer learning is occurring here, the metrics displayed are significantly lower than than the metrics obtained from the VAST dataset. This phenomena occurs because the two datasets represent two different distributions of data. This is why when training and testing a model, the same dataset is used, but it is split into a training subset and a testing subset. Thus, all of the data is obtained from the same distribution. However, transferring the model over to a new distribution will lower the metrics since the underlying assumptions about the data have changed.

Regardless of these differences in the dataset, transfer learning is a big part of machine learning and can be very useful in NLP and stance detection. If a model performs well on a dataset centered around a specific type of media, and it is realized that the same model transfers well over to a different type of media, then the model is very robust and this can save lots of time training models as the single model can be used for a variety of different tasks.

In this case, the TGANet model was trained on a dataset that derived it's sources from The New York Times, and the SEMEval2016-Task 6 dataset is a dataset of a compilation of Tweets. Since the data is derived from different social media platforms, there will be a big difference in the metrics.

It should also be noted that the SEMEval2016-Task 6 dataset contains a lot of documents that are against their target topic. Thus, there are unbalanced label classes. Therefore, the F1-score is an appropriate metric since it takes class imbalance into account. The precision and recall values for each label class are also included.

	F1-macro	F1-anti	F1-pro	p-macro	p-anti	p-pro	r-macro	r-anti	r-pro
TGANet	0.205	0.206	0.383	0.331	0.570	0.246	0.336	0.126	0.868
C-FFNN	0.243	0.728	0	0.191	0.572	0	0.333	1	0
BERT-Sep	0.246	0.332	0.371	0.490	0.558	0.246	0.338	0.236	0.760

Figure 3.1: The metric outputs for the TGANet, C-FFNN, and BERT-Sep models.

Overall, based on the results from Figure 3.1, the TGANet model gets outperformed by the C-FFNN and BERT-Sep model on the anti-classes. This resulted in a lower macro F1-score for the TGANet model. However, the TGANet model outperformed the other two models on the F1-score for the pro-class. This matches what was stated by Alloway in her paper, as she also reported a higher F1-score for the pro-class when training and testing with the VAST dataset. Thus, it can be seen that the increase in performance for detecting supporting documents for a specific topic transfers from the news media platforms to social media platforms for the TGANet model.

Looking more closely at the SEMEval2016-Task 6 dataset, Figure 3.2 below breaks down the SEMEval2016-Task 6 dataset composition with respect to the different label classes.

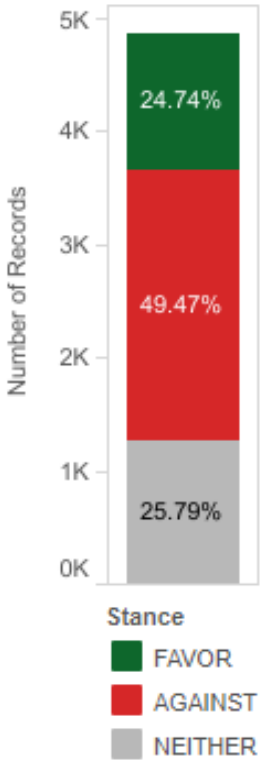


Figure 3.2: The label class composition of the SEMEval2016-Task 6 dataset.

The anti-label class almost takes up a majority of the document-target pairs in the SEMEval2016-Task 6 dataset. If the pro-label class was the largest label class, then a high F1-score for the pro-label class could just be attributed to label class imbalance. However, the opposite is the case. Thus, the TGANet model was very effective in picking up which documents were actually supporting the target topic. In fact, the supporting-label class is the smallest label class present in the SEMEval2016-Task 6 dataset.

4. CONCLUSION

4.1 TGANet Analysis

As mentioned in Alloway's paper on the TGANet model, it is indeed true that the TGANet model performs significantly better than the other models on pro labels. The TGANet model performed around the same tier or worse on the other label classes. The TGANet model did perform worse overall based on the macro F1-score. This implies that it isn't as strong on anti and none label classes overall compared to the other models. The TGANet model is very strong when supporting claims are made with related topics included in the text. However, when non-supporting claims are made, the model struggles to accurately determine the stance.

As for the other claims made by Alloway, they couldn't be verified since the SEMEval2016-Task 6 dataset contains less information than the VAST dataset with respect to their data points. Thus, it is unknown whether or not the TGANet performed well on difficult linguistic phenomena, such as sarcasm, when evaluated on the SEMEval2016-Task 6 dataset.

4.2 Future Works

This TGANet model is not the only model that was developed to tackle Zero-Shot Stance Detection. Recently, a new model, developed by Alloway at Columbia, has made advancements in the field of stance detection [4]. This new model utilizes Adversarial Learning on Social Media data for stance detection. This model was actually trained and tested on the SEMEval2016-Task 6 dataset. It would be helpful to transfer this model to other social media datasets such as the CoVaxLies v1 dataset. It would also be insightful to test this model on other sources of media such as news and maybe even the VAST dataset.

Overall, there are very few models geared towards stance detection since it is a relatively new field. Obviously, there are the traditional models that utilize straight-forward machine learning

methods without computing lots of attributes from the dataset that it's training/testing on. Thus, more models regarding stance detection and zero/few-shot stance detection are needed to improve the literature on NLP and stance detection.

Along with more models, more datasets are needed to provide researchers material to train different models on. Currently, there is only a handful of datasets for NLP and stance detection. Many of these datasets utilize Twitter data and documents from news articles and sources. However, many other social media platforms contain lots of good data for stance detection. One of these social media platforms is Reddit. Reddit contains many sub-Reddits where many users can discuss different topics and give their opinions on specific topics. Therefore, it would be very easy to find data on a specific topic. One issue with many datasets is that they don't contain a variety of topics. By utilizing Reddit data, lots of data can be collected quickly for the more niche topics. A Reddit API as exists which can help facilitate this process.

REFERENCES

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[2] E. Allaway and K. McKeown, "Zero-shot stance detection: A dataset and model using generalized topic representations," 2020.

[3] D. Küçük and F. Can, "Stance detection: A survey," *Association for Computing Machinery*, vol. 53, pp. 1–37, 2021.

[4] E. Allaway, M. Srikanth, and K. McKeown, "Adversarial learning for zero-shot stance detection on social media," 2021.

[5] G. Zarrella and A. Marsh, "Mitre at semeval-2016 task 6: Transfer learning for stance detection," 2016.

[6] B. Sick, T. Hothorn, and O. Dürr, "Deep transformation models: Tackling complex regression problems with neural network based transformation models," 2020.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2018.

APPENDIX A: LINEAR ALGEBRA BASICS

Linear algebra is often used within machine and deep learning. This section of the appendix will cover some basics of linear algebra.

A.1 Vector Spaces

A vector space consists of a set of vectors and two operations: $+$ (vector addition) and $\cdot$ (scalar multiplication). Scalars are defined for some field $\mathbb{F}$. A vector space must satisfy the following properties. Let V be some vector space, and $u, v, w \in V$ and $a, b \in \mathbb{F}$, the field that V uses.

- **Associativity:** $u + (v + w) = (u + v) + w$.
- **Commutativity:** $u + v = v + u$.
- **Identity:** There exists some element $0 \in V$ such that $0 + v = v$ for all $v \in V$.
- **Inverses:** For every $v \in V$, there exists some unique $-v$ such that $v + (-v) = 0$.
- **Scalar Multiplicative Associativity:** $a(bv) = (ab)v$
- **Scalar Multiplicative Identity:** $1v = v$
- **Distributive Properties:** $a(u + v) = au + av$ and $(a + b)v = av + bv$

A.2 Linear Combination

Suppose $v_1, \ldots, v_n \in V$ and $a_1, \ldots, a_n \in \mathbb{F}$. A linear combination can be defined as follows in Equation A.1.

$$\sum_{i=1}^{n} a_i v_i \tag{A.1}$$

A.3 Span

Suppose V is a vector space and $v_1, \ldots, v_n \in V$. Then we define the span of $v_1, \ldots, v_n$ as all vectors that can be written as a linear combination of the vectors $v_1, \ldots, v_n$. The exact definition for the span of a list of vectors is given in Equation A.2.

$$span(v_1, \ldots, v_n) = \{v \in V | v = \sum_{i=1}^{n} a_i v_i, a_1, \ldots, a_n \in \mathbb{F}\} \tag{A.2}$$

A.4 Linear Independence

A list of vectors $v_1, \ldots, v_n$ is considered as linearly independent if and only if the only way to represent 0 as a linear combination of $v_1, \ldots, v_n$ is to have all coefficients equal to 0. Equation A.3 gives the conditions needed for linear independence of a list of vectors within a specified vector space.

$$0 = \sum_{i=1}^{n} a_i v_i \iff a_1 = \cdots = a_n = 0 \tag{A.3}$$

Linear independence can also be thought of as no vector in the list can be represented as a linear combination of the other vectors in the list. If any of these conditions aren't met, then the list of vectors can be classified as linearly dependent.

A.5 Basis

Suppose V is a vector space. Then $v_1, \ldots, v_n$ is a basis for V if the list $v_1, \ldots, v_n$ is both linearly independent and spans V. If $v_1, \ldots, v_n$ is a basis for V, then every vector $v \in V$ can be uniquely represented as a linear combination of $v_1, \ldots, v_n$.