OSJI OPEN SCIENCE JOURNAL

RESEARCH ARTICLE

Growth and Fixed Mindset Classification Using NLP Approach

Ege Topkoc[1]*

[1]American School of Dubai, Al Barsha, Dubai, United Arab Emirates

*Corresponding author: Ege Topkoc: ege.topkoc@gmail.com

OPEN ACCESS

Citation: Topkoc E.(2024Growth and Fixed Mindset Classification Using NLP Approach. Open Science Journal 9(1)

Received: 31st January 2024

Accepted: 25th March 2024

Published: 17th April 2024

Funding: The author(s) received no specific funding for this work

Competing Interests: The authors have declared that no competing interests exists.

Abstract:

Theories regarding the growth and fixed mindsets have emerged in the last couple of decades. It focuses on how peoples' brains think and handle problems. People with a fixed mindset tend to feel they cannot improve or overcome difficult situations. On the other hand, people with a growth mindset tend to focus on the process and believe they can improve no matter where they started. We hypothesized that these mindsets can also be detected from text. Our goal was to design an NLP framework to classify sentences as growth or fixed mindsets. We used data generated by a Large Language Model (LLM) for our dataset: around 2000 sentences. We discovered a relationship between the sentiment of the sentence and the mindset type. Our model was a merged model which extracted features from words using word embeddings and used manually extracted features such as sentiment scores. A bidirectional Long Short-Term Memory (LSTM) was used to provide more context to both the beginning as well as the end of sentence. The final model had an F1 score of 0.99. The model can be improved by using a greater dataset, preferably created by humans instead of an AI.

Keywords: NLP, Mindsets, LSTM, Merged, Sentiment analysis

Introduction

Many people exhibit growth and fixed mindsets and are not aware of it. These specific terms emerged from Dweck's studies on children's responses to different challenging situations. People with fixed mindsets tend to believe they cannot improve and feel whatever knowledge they have is fixed. On the other hand, people with growth mindsets focus on improving throughout a process, embracing hard-work to overcome challenges. Growth mindset interventions show that growth mindsets can be developed through workshops [1,2]. Hence, it is important to create additional ways to detect these mindsets to support more interventions.

Ever since the growth and fixed mindset theories have been introduced, deep research has been conducted in schools, cities, and countries from a purely psychological perspective. However, there has been a lack in studies which incorporate ML relating to these mindsets. We aim to use NLP to create a growth and fixed mindset classification model to foster more intervention programs. We hypothesized that these mindsets can be detected from text since text can reflect the thought process of people.

We used data generated by ChatGPT (OpenAI ChatGPT, personal communication, 2023 August 26) to train our model. We found a correlation between sentiment scores for the phrases in our data and the type of model. Thus we followed a merged model approach using an embedding layer and bidirectional LSTM concatenated with a hidden layer of manually extracted features including the sentiment score. Our model had a F1 score of 0.99. Overall, our results suggest creating a highly accurate mindset classification model is achievable; however, using a human generated dataset may provide more realistic results.

Background

Dweck's Theory on Mindsets

Carol Dweck proposed two different mindsets in her book, Mindset: The New Psychology of Success: the fixed mindset and the growth mindset [3]. Dweck suggests that people often feel the need to prove themselves, which stems from their unconscious belief of having a fixed amount of knowledge, a fixed character, and fixed personality. People with this type of personality do not want to feel inadequate in these categories; hence, they have the constant urge to show others what they are capable of, which creates ego. The growth mindset on the other hand, is the idea that what people start with is drastically different then what people end with. Its root is the concept of process, that life is a journey. Instead of worrying about proving themselves to others, people who employ the growth mindset direct their attention to improving themselves and becoming a better person. This helps them overcome challenges and roadblock by enabling a passion to develop and grow.

Furthermore, research by Yeager et al. revealed that lower-achieving ninth-graders attained better GPAs in core classes by the end of the year after being part of the growth mindset intervention treatment, B = 0.10 grade points (95% confidence interval = 0.04, 0.16), s.e. = 0.03, n = 6,320, k = 65, t = 3.51, P = 0.001 [4]. Even though prior to the data analysis Yeager et al. hypothesized that treated students in lower-achieving schools might not have the sufficient opportunities to demonstrate their passion for learning after the growth mindset intervention, this was not the case. Therefore, the study concluded that growth mindset intervention has the ability to positively impact students' grades and sustain other benefits as well [4].

While Diener and Dweck were studying the responses of children to failure, they noticed that some bounced back quickly when faced with difficulty, while others dwelled on the failures. Diener and Dweck characterized these students as mastery-oriented children and helpless children respectively [5,6]. In the study, they discovered that the helpless children felt they didn't have the ability to overcome the failure, whereas the mastery-oriented children regarded failure as surmountable, focusing on the intrinsic motivational factors. Additionally, helpless children focused on the causes for failure, while mastery-oriented children focused on ameliorating failure. When Diener and Dweck conducted the study, they also

observed optimism from the mastery-oriented children when faced with difficulty. Succinctly, helpless children displayed negative self-cognition while mastery-oriented children showed positive self-instructions and positive body language [5,6]. Even though this effect was initially discovered in children, studies have also shown it to be prominent in adults as well [7]. Elliot and Dweck proposed two different classes of goals which individuals pursue: performance goals and learning goals [8]. Performance goals focus on receiving positive judgment on an individual's ability while learning goals focus on increasing an individual's ability. Elliot and Dweck had hypothesized that individuals who have performance goals will exhibit the helpless response to failure, whereas individuals who have learning goals will exhibit the mastery-oriented response to failure [8].

Recurrent Neural Networks (RNNs)

An RNN works like a normal forward-feeding neural network except the output of the hidden layer from the previous timestep (word in NLP context), is also fed into the current hidden layer along with the new input [9]. This creates memory and forms a connection between previous time steps. Output from the each timestep can be used in tasks such as Parts-of-speech (POS) tagging or Named-Entity Recognition, but in a classification task, the intermediate outputs from the timesteps can be ignored and the final output can be used after being put through an appropriate activation function.

An LSTM is a type of recurrent neural network (RNN), in which it overcomes the RNN's weakness in comprehending long-term dependencies and vanishing and exploding gradients. The LSTM overcomes the difficulties an RNN faces by using a different cell architecture. The cell contains different gates which are NN layers, and they learn whether to dampen or amplify features to be able to understand long range dependencies [9]. Like a usual RNN cell, the LSTM cell is inputted current x and the hidden output from previous timestep (h_{t-1}), but it also takes in a cell state from the previous time step, c_{t-1}. The cell state serves like a memory, storing information to be kept long term. Figure 1 is a model of the cell state. This model will be described in detail below.

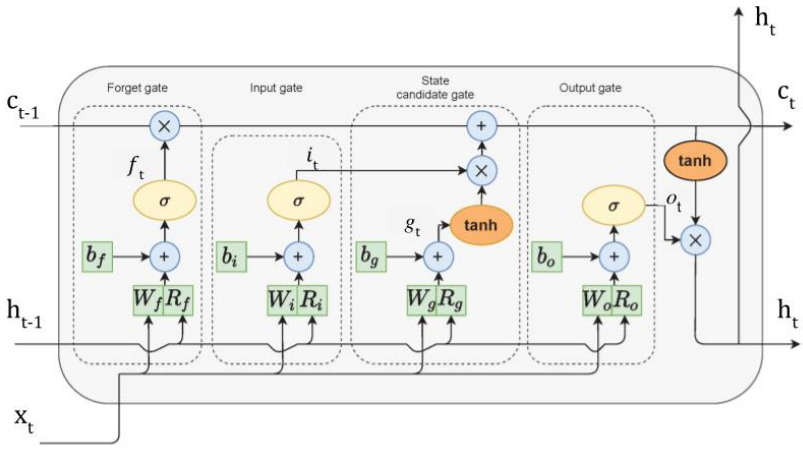

Figure 1. Model of an LSTM cell. This figure was taken from Zarzycki and Ławryńczuk [10] but modified slightly.

Forget Gate:

X and h_{t-1} are multiplied by their respective weights and added together, being input through the forget gate, which is a sigmoid function [9]. The inputs and outputs are all vectors, so the sigmoid function is applied to the elements of the vectors. This results in the output vector f_t which consists of positive numbers between 0 and 1. This vector goes through element-wise multiplication with the previous cell state. Since this vector consists of numbers between 0 and 1, it controls which elements from the cell state are passed through the cell: 0 would signify forgetting, 1 would signify keeping, and a decimal would signify keeping it in that proportion. Succinctly, this gate allows only the vital information to carry through from the previous timestep.

$$F_t = \sigma\left(W_f \cdot X_t + R_f \cdot h_{t-1} + b_f \right) \qquad [1]$$

Input Gate:

X and h_{t-1} are multiplied by their respective weights and added together, being input through the input gate, which is a sigmoid function [9]. This results in the output vector of it which also consists of numbers between 0 and 1. This determines what information to add to the cell state.

$$i_t = \sigma\left(W_i \cdot X_t + R_i \cdot h_{t-1} + b_i \right) \qquad [2]$$

Candidate Gate:

X and h_{t-1} are multiplied by their respective weights and added together, being input through the candidate gate, which is a tanh function [9]. This results in the output vector of g_t. This determines what information is available to add to the cell state.

$$g_t = \tanh\left(W_g \cdot X_t + R_g \cdot h_{t-1} + b_g \right) \qquad [3]$$

I_t and g_t are multiplied together element-wise, with the information to add to the cell state [9]. This information is added to create a new cell state, c_t, to be passed to the next timestep.

$$c_t = f_i \odot c_{t-1} + i_t \odot g_t \qquad [4]$$

Output Gate:

X and h_{t-1} are multiplied by their respective weights and added together, being input through the output gate, which is a sigmoid function [9]. This results in the output vector of o_t which also consists of numbers between 0 and 1.

$$o_t = \sigma\left(W_o \cdot X_t + R_o \cdot h_{t-1} + b_o \right) \qquad [5]$$

The current cell state is inputted into a tanh function, creating a vector with potential values. The current hidden state is calculated by multiplying this vector element wise with o_t [9]. As a result, the current hidden state contains long term information provided by the cell state.

$$h_t = o_t \odot \tanh(c_t) \quad [6]$$

The h_t output at the top can be used as the output of the current time step or can be ignored. The h_t output at the bottom is to be utilized for the next timestep.

A Bidirectional LSTM works by having another LSTM that runs in reverse order so the last token for one LSTM would be the first input for the other. As we have the full sentence available as input, a Bidirectional LSTM allows us to have full context of the sentence.

Related work

Researchers have used Natural Language Processing (NLP) and Machine Learning (ML) in the past to detect different human traits and features from text. Some of these include personality prediction and emotion detection. Yet, to the best of our knowledge, research using NLP and ML to detect different mindsets does not exist publicly. Hence, the techniques we used were heavily based on NLP and ML techniques used for emotion detection and personality prediction tasks.

Prior to using automated methods such as using ML to predict personalities, personality tests such as the Myers-Briggs Type Indicator were used [11]. The Myers-Briggs Type indicator expands on Carl Jung's work [12] relating to psychological type. Isabel Myers and her mother Katherine Briggs created different iterations of questions, testing them to be able to identify the type referred to by Jung, while keeping track of data [13]. The Myers-Briggs Type indicator has 4 different scales: extraversion-introversion, which measures how interested the person is in other people or ideas, sensation-intuition, which measures how the person perceives, thinking-feeling, which measures how the person evaluates (through objectivity with logic or subjectively through emotion), and judging-perceiving, which measures the person's tendency to jump to conclusions or acknowledge them [14].

In recent years, many novel methods have been utilized for personality detection from text. For example, Ramezani et al. introduced a different way to tackle Automatic Personality Prediction (APP) using Knowledge Representation (KR) [15]. The predicted personalities would fall under the Big Five model of openness, conscientiousness, extroversion, agreeableness, and neuroticism. For their data, they used essays which were each assigned a personality. These essays were later converted to knowledge graphs. According to Bergman [16], Knowledge Representation is a way to represent information so that computer systems can comprehend it and use it to solve tasks. Ramezani et al. used a knowledge graph-enabled text-based APP model [15]. This included preprocessing the input text through tokenization, noise removal, normalization, and named entity recognition. Next, they create a knowledge graph representation of the text consisting of three phases: graph building, graph enriching, and graph embedding. Lastly, they used four different machine learning models to predict personality traits: Convolutional Neural Network (CNN), simple Recurrent Neural Network (RNN), unidirectional Long Short-Term Memory (LSTM), and Bidirectional Long Short-Term Memory (Bi-LSTM).

Similar to personality detection, there has been extensive research on detecting emotion from text. For example, Gaind et al. used 3 different methods to create an emotion detection algorithm [17]. Their first method used Natural Language Processing (NLP) and the Emotion-Word Set (EWS), a bag of around 1500 words which are the synonyms of the six basic emotion categories: happiness, sadness,

anger, surprise, fear, and disgust. They also labeled each word in the set according to their intensity of the emotion. Their text for the emotion detection came in the form of tweets. In essence, they compared the words in the text of the tweet to the EWS, while considering different characteristics of text such as negation and degree-words, emoticons, and others. Their second method used Machine Learning (ML). To train the model, they again used text in the form of tweets. However, since tweets can contain multiple emotions which cannot be classified into one, they used the NLP approach to label the tweets. The tweets that had a 70% or greater expression of one emotion were used for training. This helped to train a more accurate model. They then applied pre-processing steps such as stop-word filtering, lower-casing, and stemming. They used SMO and J48 from Weka as classifiers. These classifiers then give the probabilities for each class, and the highest probability becomes the output class. Lastly, they combined both the NLP and ML approach into a hybrid approach. In this method, a factor of the score, the score of the emotion determined from the NLP approach, is added to the score from the classifier to add more weight to the classifier. This helps the model determine a class when two different emotions are close. The category with the highest score is determined as the emotion.

Guo created a Deep learning assisted semantic text analysis (DLSTA) model to detect human emotions using big data from questionnaires and texts [18]. DLSTA was designed to include aspects of NLP and deep learning since emotion detection through text is a context-based problem. After pre-processing is carried out, features are extracted using two different approaches: questionnaire based approach and the text-analysis based approach. Then, these two feature vectors are combined and passed into a support vector machine (SVM) based classifier.

NLP was also used for depression and suicide ideation detection from text. Jain et al. used 60,000 data points from 2 different subreddits each classified as depression or suicide watch [19]. They aim to aid counselors and other professionals in distinguishing between language used by individuals experiencing depression and individuals with suicidal thoughts. They used lower casing, tokenization, stop-word removal, and stemming for pre-processing. Then, 4 different ML algorithms were experimented with: logistic regression, naïve bayes, support vector machine (SVM), and random forest. All of the models were sufficiently accurate, varying between a 0.75 and 0.79 F1 score.

Using a merged model architecture can result in a more accurate model. Cocarascu and Toni used NLP in a merged architecture to create an argument mining (AM) model, which detects the type of argument in text: whether a sentence supports, attacks, or does neither to another sentence [20]. Therefore, this task can be simplified to a 3 class classification problem. They used a dataset consisting of topics such as movies, politics, and technology. Since each input consists of two texts they used a symmetric architecture with one text inputted into 100D GloVe vectors and a Bi-LSTM and the other text inputted to another pair of 100D GloVe vectors and Bi-LSTM. The outputs of these layers were merged using element-wise sum and concatenation. Merging through concatenation yielded better results, which they believe was because the model was able to retain more features.

Methods

Dataset

Finding an existing publicly available dataset with different phrases that exhibit the growth mindset and fixed mindset has proven to be difficult. To the best of our knowledge, an open source dataset with these phrases does not exist. Hence, we used a large language model (LLM) in ChatGPT (OpenAI ChatGPT, personal communication, 2023 August 26) to generate our dataset. There would be two columns for the data: the first column would contain the phrase and the second column would contain whether the phrase exhibits a growth mindset or a fixed mindset. The LLM was programmed to insert a "1" in the column for growth mindsets and a "0" in the column for fixed mindsets. In the first generation of the data, the LLM was biased towards generating fixed mindset phrases as it generated 467 fixed mindset phrases compared to the 163 growth mindset phrases, totalling 630 phrases. As a result, the current dataset was not sufficiently balanced or augmented. Therefore, the LLM was manually programmed to generate more growth mindset phrases. The second generation of data included 467 fixed mindset phrases and 490 growth mindset phrases, totalling 957 phrases.

However, the LLM was generating repeated data. Hence, it was required to filter the data to ensure no data repeats occurred before pre-processing. After filtering, it was determined that there were a total of 494 unique phrases: 122 unique fixed mindset phrases and 372 unique growth mindset phrases. Hence the LLM was programmed to generate a third set of data; we especially programmed it to generate fixed and growth mindset exhibiting phrases related to sports, music, and school. After this generation, there were a total 2466 phrases, 1751 of which were unique: 857 phrases for the fixed mindset, and 894 phrases for the growth mindset. An example of the data can be seen in Table 1.

Table 1. Example of data from our dataset

Phrase	Is Growth
"Challenges are opportunities for growth."	1
"I'm not naturally talented in this area."	0
"I believe in my ability to learn and improve."	1
"I'm not good at adapting to new situations."	0

Furthermore, we calculated a sentiment score, using VADER Sentiment Analysis [21], for each phrase in their respective mindset category. The scores from each category were later graphed as can be seen in Figure 2. VADER Sentiment Analysis calculates a score for negativity, positivity, and neutralness of the text, combining them into a final compound score. The compound scores were used as the x-axis for the histogram. The fixed mindset phrases graph appears to be skewed to the right while the growth mindset phrases graph appears to be skewed to the left. This establishes a potential relationship between the sentiment score and mindset type.

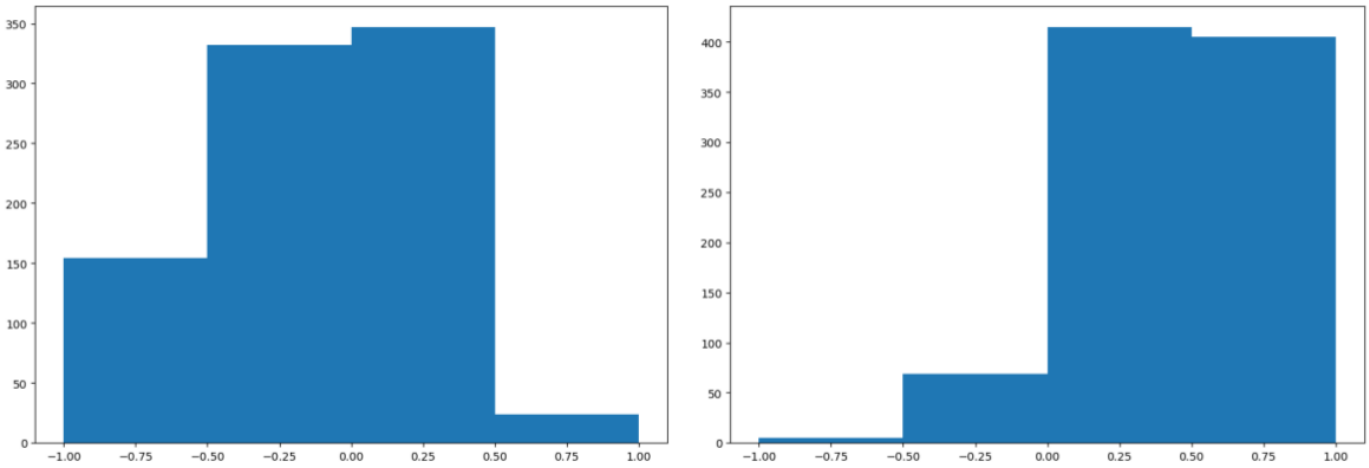

Figure 2. A comparison of the sentiment score histograms of fixed mindsets (left) and growth mindsets (right).

Pre-processing

Prior to extracting features from our data, the data needed to be pre-processed to allow the model to recognize words as usable input. The first step of pre-processing involved removing contractions. This step is necessary to create fewer token indexes in the tokenization and indexing steps. For example, "I'm" and "I am" would have different indexes if contractions were not removed. For this process, the Pycontractions library was used.

The next step was to tokenize the phrases using TenserFlow's [22] tokenization combined with indexing and sequencing. Tokenization is the process of splitting up the sentence into its words (tokens.) The Tokenizer class also assigned an integer value to each unique word for the indexing step. Tokenizer also removes any punctuation in the sentence, and case-folds, turning all characters to lowercase. These tokens were later turned into sequences to feed into the model. Sequences are the sequences of token indices that represent each sentence. These sequences were post-padded to the sequence with the maximum length, 30, to ensure they are all the same size. This was necessary as the sequences need to be the same size for each batch even though an LSTM can have inputs of different sizes.

Architecture

Figure 3 summarizes our architecture, merging a Bi-LSTM layer with a dense layer through a concatenation layer. A merged architecture was used to combine textual features extracted through word embeddings and the LSTM with manually extracted numerical features from the text. The majority of these numerical features related to the sentiment score for that sentence calculated using VADER Sentiment Analysis [21]. The 4 features extracted using the sentiment score include the positive, neutral, negative and compound score. Additionally, the 5th feature extracted was the frequency of negation words "no," "nor," and "never." The rationale behind this feature was to allow the model to acknowledge the sentences which contained these negations.

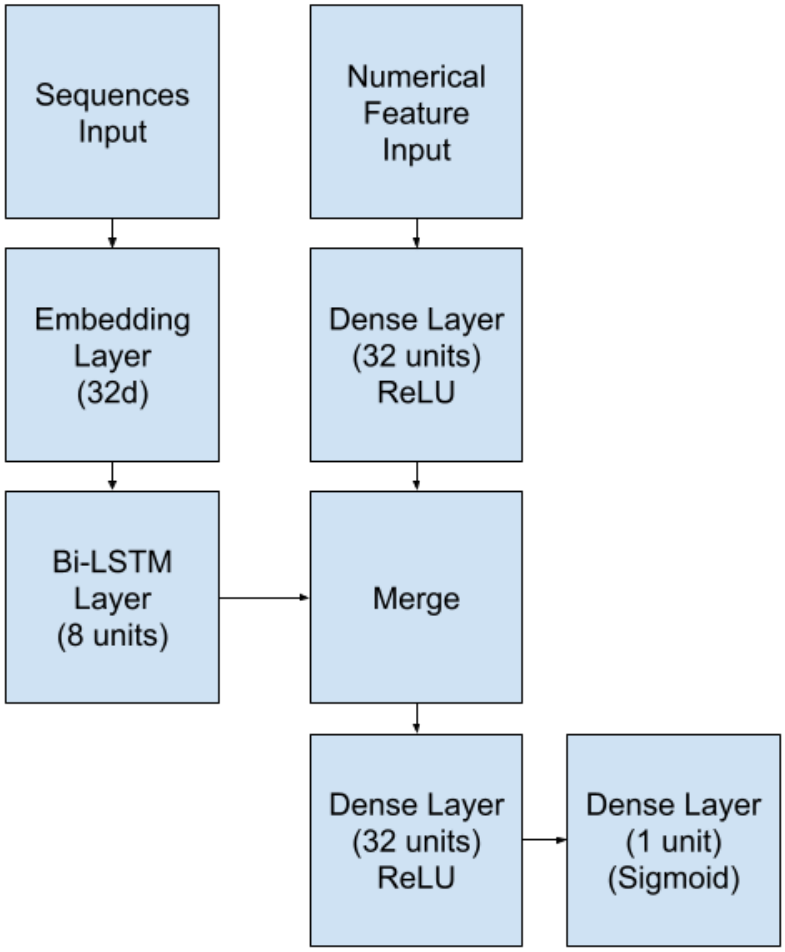

Figure 3. A summary of our merged architecture.

Embedding layers provide short and dense representations of words, capturing meaning between different words. We decided to train the word embeddings alongside our model instead of using pre-trained word vectors. Our input dimension was equal to the unique vocabulary size of 1254. We experimented with different input lengths, and found the most accurate one was 30, which is the maximum length of our sequences. Through trial and error, we landed on the embedding dimension of 32. This value is smaller than common and frequently used embedding dimensions; however, we believe this is due to the relatively small dataset size. With larger output vector dimensions, the embedding layer would be trying to extract more features from the word with loose connections to the overall classification. The Mask Zeroes parameter was set to true so the embedding layer would ignore the zeroes from the padding. Otherwise, this could create unrealistic accuracy metrics.

The word vectors from the embedding layer were input into a Bidirectional LSTM. Our Bi-LSTM layer contained 8 units, 4 for each. This value was obtained through trial and error. However, like the dimensions of the embedding layer, this parameter was lower due to the relatively small sample size. The Return Sequences parameter was defaulted to false. As mentioned previously, since this is a classification task, an output at the end of each timestep is not necessary.

The input shape for our dense layer was of shape 5 due to the 5 numerical features extracted. This layer had 32 neurons and a ReLU activation function. This layer was merged with the Bi-LSTM using a concatenate layer. Then, the output was passed through another dense layer with 32 neurons with a ReLU activation function. Lastly, the output layer has 1 neuron and a sigmoid activation function for binary classification. The model was compiled using the Adam optimizer and binary cross entropy, also known as log loss, as the loss function. The binary cross entropy loss function is equal to

$$= -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log\left(p\left(y_i\right)\right) + \left(1 - y_i\right) \cdot \log\left(1 - p\left(y_i\right)\right) \quad [7]$$

where N is the size of data being used and p(y) is the predicted probability of the sentence exhibiting the growth mindset for all sentences [23]. A summary of all layers used can be seen in Table 2.

Table 2. A summary of hyper-parameters for each layer in the model.

Hyper-parameter	Value	Hyper-parameter	Value
Embedding Size	32	Dense 1 Size	32
Sequence Length	30	Dense 2 Size	32
Bi-LSTM Size	8	Batch Size	32

Results

The model was trained with a default batch size of 32 and using Keras Early Stopping. The model would train for 10 epochs or when the loss function increased from its minimum for another 2 epochs. Hence, a patience parameter of 2 was used for the early stopping. The rationale behind early stopping is to prevent overfitting, but, due to the nature of the problem, slight overfitting is permitted. The model outputted a test accuracy of 98.5% with a loss of 0.06. Figure 4 displays a graph of Loss vs. Epochs for both training and testing. The straightening of the curve after epoch 3 for the test loss is because of the early stopping.

Additionally the dimensions and units that were determined by trial and error have resulted in their current value due to greater values causing overfitting. Greater parameter values would cause the accuracy to fluctuate around the same accuracies of lower values, but the loss would be significantly greater. Hence, we concluded that the sample size was not great enough to have a very complex model with many dimensions and units.

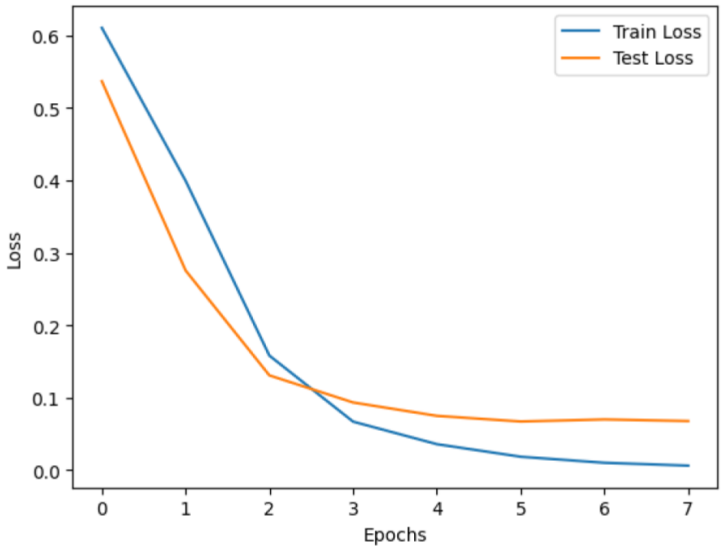

Figure 4. Loss vs. Epochs graph for both training loss (blue) and testing loss (orange).

Table 3 displays a confusion matrix of the model where 1 is for growth mindset exhibiting phrases and 0 is for fixed mindset exhibiting phrases. A confusion matrix summarizes the performance of the model in predicting correct labels. We can see that the model predicted 179 samples correctly for the fixed mindset, which would be true negatives, and 167 for the growth mindset, which would be true positives. There were 3 false negatives where the model predicted a fixed mindset and it was actually a growth mindset, and 2 false positives where the model predicted a growth mindset and it was actually a fixed mindset.

Table 3. Confusion Matrix.

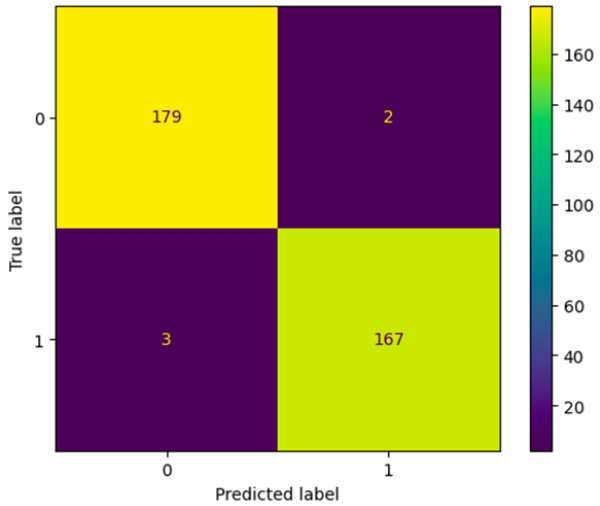

Table 4 displays summary statistics of our model. The growth mindset is 1 and fixed mindset is 0. Precision represents the ratio TP / (TP + FP) where TP is the total number of true positives and FP is the total number of false positives [24]; in this case, precision was 0.99. Recall represents the ratio TP / (TP + FN) where TP is the total number of true positives and FN is the total number of false

negatives [24]; in this case, recall was 0.98. The F1 score is the harmonic average of both the precision and recall statistics. Since $\beta = 1$, both precision and recall were weighted equally for this average. Our model's F1 score was 0.99. The support statistic represents the number of sentences from each class (fixed mindset and growth mindset) that was present in the testing data.

Table 4. Summary Statistics

	Precision	Recall	F1-Score	Support
"0"	0.98	0.99	0.99	181
"1"	0.99	0.98	0.99	170
Accuracy			0.99	351
Macro Avg	0.99	0.99	0.99	351
Weighted Avg	0.99	0.99	0.99	351

Lastly, Figure 5 displays an ROC curve. An ROC curve shows a graph of the True Positive Rate (Sensitivity) vs. False Positive Rate (1-Specificity). A no skill model is also represented where it would randomly predict a class. It has an area under the curve (AUC) of 0.5. Our model has an AUC of 0.994. The true positive rate gives us the proportion of growth mindset samples that are correctly classified as growth mindset: TP/ (TP + FN). The false positive rate gives us the proportion of fixed mindset phrases that were incorrectly classified as growth mindset. Each point represents the True Positive Rate and False Positive Rate for different thresholds to classify a sample as growth or fixed mindset. Therefore, the point (0,1) is the desired outcome for a perfectly accurate model.

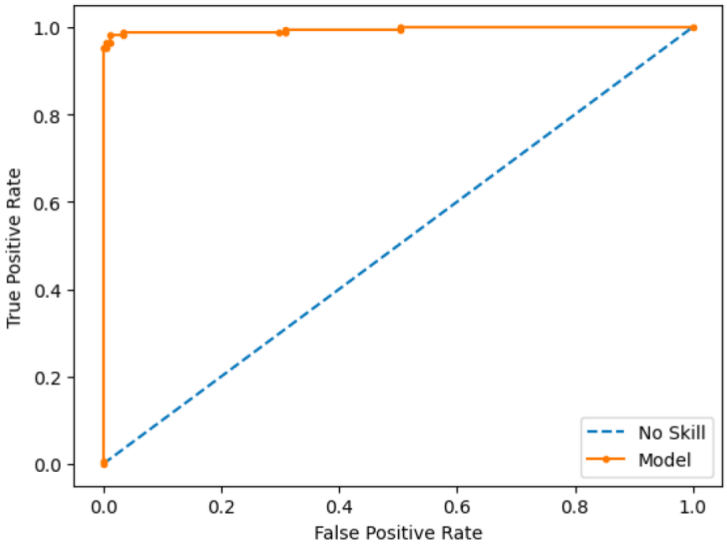

Figure 5. ROC Curve.

Explanation of results

Table 5 displays a few phrases where the model failed to classify the sentences correctly, along with the probabilities for each class. Table 6 displays the sentiment score and negation word frequency features extracted from the sentences for the incorrectly classified sentences.

Table 5. Summary of 3 incorrectly classified sentences with probabilities outputted from the model for each class.

Sentence	Probability for Fixed	Probability for Growth	Actual Class
I will never give up.	0.9993058	0.0006942	Growth
I am not naturally motivated to learn and improve.	0.4659334	0.5340666	Fixed
I will learn from my mistakes.	0.5529853	0.4470147	Growth

For the first sentence, the model outputted a 0.999 probability for the sentence to be a fixed mindset exhibiting the sentence, but it was not. Looking at Table 6 for the first sentence, we can see the sentiment was neutral with one negation word of "never." Looking at the first row of Table 6, we can see there are no extreme values in the sentiment scores, so the model must have inaccuracies in the creation of the word vectors. Figure 6 shows us the word vectors for each word in the first sentence for Table 6 using UMAP [25] to reduce the dimensionality for 32 dimensions to 3 dimensions to be able to visualize the vectors

Table 6. Manually extracted features for the 3 incorrectly classified phrases

Sentence	Negative Score	Neutral Score	Positive Score	Compound Score	Num Negation Words
I will never give up.	0.0	1.0	0.0	0.0	1
I am not naturally motivated to learn and improve.	0.2	0.565	0.234	0.1078	1
I will learn from my mistakes.	0.333	0.667	0.0	-0.3612	0

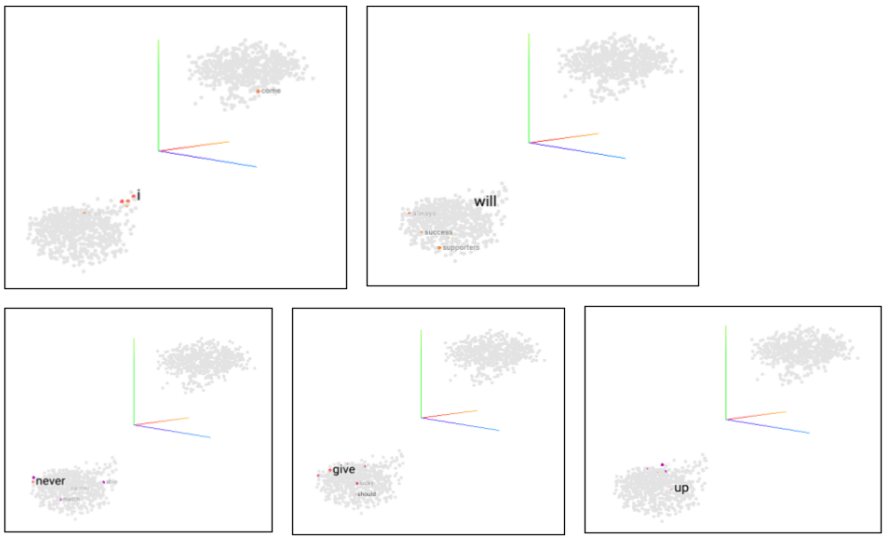

Figure 6. 3D representations of word vectors for "I will never give up" using UMAP [25] with parameters of 30 for Neighbors and 0.45 for Minimum Distribution.

All of these words are grouped together in the same cluster, explaining the model's confidence with the output probability. We can also see that the UMAP [25] visualization created two clusters, which we believe is due to the dimension reduction and the inherent binary classification task. From this pattern, the left cluster seems to be the words most commonly associated with a fixed mindset in this dataset. Also, words like "I" and "will" are very close to the center, showing it is commonly related to both types of mindsets. Additionally, there were many sentences in the data which consisted of "never" and a positive phrase, which meant the person was not able to do something that is positive. Hence, the vector of "never" was placed at the very edge of the fixed mindset cluster. Our frequency of negation words feature was not sufficient to account for the negation word in this case.

For the second sentence in Table 5, the outputted probabilities were very close. Again, from Table 6, there were no extreme values for the second sentence. Analyzing the word vectors of the sentence also revealed the words were split approximately evenly between the two clusters; words like "motivated," "learn," and "improve" were on the growth mindset side, while "not" and "naturally" were on the fixed mindset side. The word vector for a word like "naturally" is on the fixed mindset side since a person with a fixed mindset believes their qualities are from birth and cannot be improved; therefore, they would be likely to use words like "naturally." This sentence was likely classified as growth due to the positive compound sentiment score since the word vectors were balanced between both sides.

Like the second sentence, the third sentence in Table 5 also had close probabilities outputted from the model for the growth and fixed mindsets. Table 6 revealed no extreme values for positive or negative sentiment scores. Like the second sentence, the third sentence's word vectors were evenly spread out between the fixed and growth mindset clusters: "learn" was in the growth mindset side and "mistakes" was in the fixed mindset side with the other words being close to the

center. The slightly higher probability output for fixed mindset in this sentence was due to the negative compound score as sentences with negative compound scores were mostly fixed mindset exhibiting sentences. The word vectors were balanced so we believe the sentiment scores tipped it from 50%.

Discussion

Initially, we had more pre-processing steps in addition to the contraction removal, tokenization, and sequencing. After the contraction removal we tokenized each phrase in order to remove stop words and perform lemmatization. The NLTK [26] library was used for the tokenization, stop word removal, and lemmatization. After tokenization, stop words, words that do not provide any sufficient meaning to the sentence, were removed to reduce the sentence size and allow for more computationally efficient data. However, the NLTK [26] library for stop word removal also removed words such as "no," "nor," and "never," which are crucial in language related tasks; these words provide vital context to the text that follows. Hence, in our stop word removal, such negation words were not removed. Lastly, we used NLTK's WordNet Lemmatizer [26] for lemmatization, where words get reduced to their base word. For example, "jumps" would get reduced to "jump." This allows for less indexes to be created in the indexing process. These tokens were detokenized and turned back into sentences to be able to use TenserFlow's tokenization. Then the sentences underwent tokenization, indexing, and sequencing as mentioned in the Pre-Processing part.

When using stop word removal and lemmatization, the model outputted around 96% accuracy with a loss of around 0.12. We hypothesized that meaning was being lost through these processes, resulting in the word embeddings not being utilized to their full potential in forming relations between words. Additionally, we were also using a unidirectional LSTM for the training of this model.

After removing the pre-processing steps of stop word removal and lemmatization, the accuracy improved to around 98% whereas the loss improved slightly, fluctuating between 0.11 and 0.12. We decided to switch the unidirectional LSTM for a bidirectional LSTM as this would allow words at the beginning of the sentence to get as much context as the words at the end of the sentence because of the 2 LSTMs processing the data in opposite orders.

Challenges and future work

Even though our model proved to have high accuracies, these accuracies can be further improved with a greater data set. Many NLP tasks have corpuses with tens of thousands of samples. With a greater dataset, the embedding dimensions and LSTM units can be set higher without worrying about overfitting. The challenge in obtaining a dataset was due to the lack of publicly available datasets for growth and fixed mindset classification. This was the reason why our dataset was generated by a LLM. Perhaps this could create confounding as we are predicting classes using AI on a dataset created by an AI model using a similar architecture. Hence, it would be essential for future research to utilize datasets created by humans. Additionally, instead of this problem being a binary classification task, another outcome could be added such as a neutral sentence where the goal of the sentence is to simply convey information without exhibiting a mindset type. This would be

an important addition to allow the model to have practical applications. Creation of more such accurate models would aid in the dataset creation process as the models could be used to classify large corpuses into the different categories to be used for similar tasks.

Regarding pre-processing steps, parts of speech (POS) tagging could be added as a feature. This would allow the model to recognize how words are being used in relation to other words and in the scope of the entire sentence. It could allow the embedding layer to generate more accurate vector representation of words and their relationship with other words.

When using UMAP [25] to visualize our 32 dimensional embeddings in 3 dimensions, it was often the case that similar words were in opposite clusters of data points. For example in Figure 7, "job" and "jobs" are very far apart when they should be in close proximity to each other. We used UMAP parameters of 30 for Neighbors and 0.45 for Minimum Distribution to get a broader view of the data, yet these words still had a large distance between them. This is likely due to the relatively small sample size, so the word embeddings were not able to train sufficiently. This could also be caused as a result of turning a 32 dimensional matrix into 3 dimensions; however, we calculated the Euclidean distance (equation 8)

$$d(x,y) = \sqrt{\sum_{i=1}^{n} \left(y_i - x_i\right)^2} \qquad [8]$$

(where x and y are the 32 dimensional vectors, y_i and x_i are the points for each dimension, and n = 32 for the number of dimensions) between the original 32 dimension word vectors of "job" and "jobs" which resulted in 0.6689. Since this value is still quite large, it is highly unlikely the distance of the vectors of these two words in the UMAP [25] visualization was due to dimension reduction. Using pre-trained word vectors like Word2Vec or Glove can help prevent this issue, or using a greater dataset so the embeddings can train on more data if the researchers want to use embedding layers.

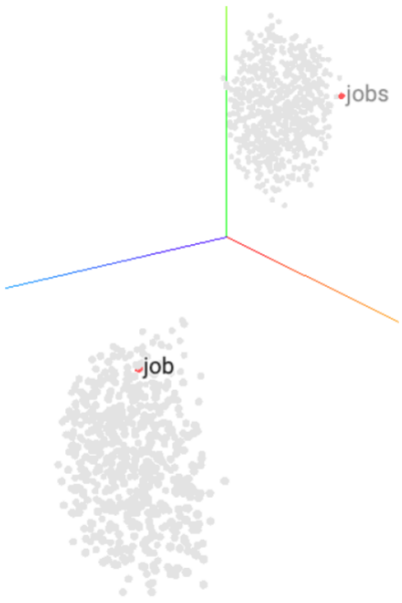

Figure 7. UMAP [25] visualization of word vectors of "jobs" and "job."

Lastly, even though we added a feature to acknowledge negation words, a more comprehensive approach to deal with negation words can be used to increase the accuracy of the model when phrases with negation words are in question.

Conclusion

Our model is highly accurate in correctly predicting growth and fixed mindset phrases with an F1 score of 0.99. Our dataset was relatively small compared to those of other NLP tasks and was also created by an LLM due to the difficulties in obtaining a publicly available dataset with growth and fixed mindset phrases. The merged architecture of our model allowed us to combine features from the phrases and words themselves with numerical features extracted from them like the sentiment score. Additionally, negation words were also accounted for using a simple frequency counter, but a more complex approach may be used in the future to improve performance.

Growth and fixed mindsets have a huge impact on a person's daily life and habits, so initially detecting phrases which exhibit these mindsets will provide valuable insight on how a person thinks. Growth and fixed mindsets are an area of psychology which has not yet been researched thoroughly using ML techniques and we hope our research acts as a catalyst for further research to come.

Acknowledgements

The author would like to thank the Lumiere Individual Research Program for their support and advising throughout the length of the research project.

References

1.Good C, Aronson J, Inzlicht M. Improving adolescents' standardized test performance: An intervention to reduce the effects of stereotype threat. Journal of Applied Developmental Psychology. 2003;24(6):645–662.

2.Blackwell LS, Trzesniewski KH, Dweck CS. Implicit Theories of Intelligence Predict Achievement Across an Adolescent Transition: A Longitudinal Study and an Intervention. Child Development. 2007;78(1):246–263.

3.Dweck CS. Mindset : The new psychology of success. New York: Ballantine Books; 2006.

4.Yeager DS, Hanselman P, Walton GM, Murray JS, Crosnoe R, Muller C, et al. Nature. 2019;573:364–369.

5.Diener CI, Dweck CS. An analysis of learned helplessness: Continuous changes in performance, strategy, and achievement cognitions following failure. Journal of Personality and Social Psychology. 1978;36(5):451–462.

6.Diener CI, Dweck CS. An analysis of learned helplessness: II. The processing of success. Journal of Personality and Social Psychology. 1980;39(5):940–952.

7.Brunson BI, Matthews KA. The Type A coronary-prone behavior pattern and reactions to uncontrollable stress: An analysis of performance strategies, affect, and attributions during failure. Journal of Personality and Social Psychology. 1981;40(5):906–918.

8.Elliott ES, Dweck CS. Goals: An approach to motivation and achievement. Journal of Personality and Social Psychology. 1988;54(1):5–12.

9.Parker N. NLP Demystified 13: Recurrent Neural Networks and Language Models [Internet]. YouTube. 2022. Available from: https://www.youtube.com/watch?v=y0FqGWbfkQw&t=1897s

10.Zarzycki K, Ławryńczuk M. LSTM and GRU Neural Networks as Models of Dynamical Processes Used in Predictive Control: A Comparison of Models Developed for Two Chemical Reactors. Sensors. 2021;21(16):5625.

11.Myers IB. The Myers-Briggs Type Indicator: Manual (1962). Palo Alto: Consulting Psychologists Press. 1962.

12.Jung CG. Psychological types. New York: Routledge; 2017.
13.Geyer P. Quantifying Jung: The Origin And Development Of The Myers-Briggs Type Indicator [MSc Thesis]. Melbourne: University of Melbourne; 1995.
14.Stricker LJ, Ross J. Intercorrelations and Reliability of the Myers-Briggs Type Indicator Scales. Psychological Reports. 1963;12:287–293.
15.Ramezani M, Feizi-Derakhshi MR, Balafar MA. Knowledge Graph-Enabled Text-Based Automatic Personality Prediction. Comput Intell Neurosci. 2022.
16.Bergman MK. A Knowledge Representation Practionary: Guidelines Based on Charles Sanders Peirce. Cham: Springer International Publishing; 2018.
17.Gaind B, Syal V, Padgalwar S. Emotion Detection and Analysis on Social Media. 2019. Available from: https://arxiv.org/pdf/1901.08458.pdf
18.Guo J. Deep learning approach to text analysis for human emotion detection from big data. Journal of Intelligent Systems. 2022;31(1):113–126.
19.Jain P, Srinivas KR, Vichare A. Depression and Suicide Analysis Using Machine Learning and NLP. Journal of Physics: Conference Series. 2022.
20.Cocarascu O, Toni F. Identifying attack and support argumentative relations using deep learning. Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. 2017;1374–1379.
21.Hutto C, Gilbert E. VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text. Proceedings of the International AAAI Conference on Web and Social Media. 2014;8(1).
22.Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. 2015.
23.Godoy D. Understanding binary cross-entropy/log loss: a visual explanation [Internet]. Towards Data Science; 2018. Available from: https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a
24.Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-Learn: Machine Learning in Python. J Mach Learn Res, 2011;12: 2825–2830.
25.McInnes L, Healy J, Melville J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. 2018. Available from: https://arxiv.org/pdf/1802.03426.pdf
26.Bird S, Klein E, Loper E. Natural language processing with Python. Sebastopol: O'Reilly Media Inc; 2009.