Natural Language Processing: Chatbot and Resume Deep Learning Analysis and Comparison

Michael Childs
CS-6890 Applied Deep Learning
Utah State University
Logan, United States
a02279536@usu.edu

***Abstract*—This report discusses the processes and results of analyzing a Therapy-Chatbot and Resume Response dataset by using Natural Language Processing neural networks and deep learning practices.**

Keywords—NLP, Classification, Natural Language Processing, Analysis, Preprocessing

I. Introduction

This project expresses the purpose of utilizing Natural Language Processing in analysis and classification of text-based datasets commonly found and produced in a variety of fields. For this project in particular, Natural Language Processing is used for classifying the responses of an AI Therapy Chatbot as well as the responses of Indeed resumes. For both sources, the classification of either "flagged" or "not flagged" as useful responses is provided for the researcher to perform experiments on. Both sub-datasets required extensive preprocessing and preparation before being used in Natural Language Processing neural networks.

To provide a brief introduction to Natural Language Processing, this field (commonly denoted as NLP) is a deep-learning oriented ability and process of allowing computer programs to both understand and manipulate human language in the form of both text or voice data. By utilizing neural networks, the machine is capable of comprehending, to a degree, large amounts of human-created data and the perform a variety of tasks on it. In the case of this project, Natural Language Processing is used in classification—i.e. the act of denoting a point of data as some category, in this case in a binary manner—and allows for a simplification of otherwise time-consuming tasks in an automated and efficient manner.

In this project, it is proposed that Natural Language Processing may be effectively and efficiently utilized in classifying two fields which contain a variety of data values yet necessitate extensive human labor otherwise for analysis: therapy and resume responses. Both fields are diverse and thus pose considerable potential for extension to both related and distantly associated fields in the future.

II. Related Work

In an effort to build upon previously published studies and research projects, the researcher for this project performed thorough research to identify such cases that might provide for interesting and useful insights to further expedite and benefit this project. This section covers a series of such prior works that thus were utilitarian in advancing this project.

A. Comment Usefulness Classification on Youtube using Artificial Neural Networks [1]

This paper discusses the applications and development of Natural Language Processing in analyzing and classifying the usefulness of YouTube comments. By using Natural Language Processing, features are derived of comments which are constructive or insightful (and thus useful) while comments which are superficial, inapplicable, or otherwise not directed at the creator are classified as useless. This paper covers related works to social media deep learning classification, the "core pipeline" of classification modeling, a case study on classifying useful YouTube comments, and finally a summary and analysis of the findings, as well as future research directions. The value of this paper In analyzing a Therapy Chatbot using Natural Language Processing, is hoped that the application of determining whether or not comments made by a chatbot are "useful" can be seen. Likewise, the analysis of resumes and their respective elements for what parts dictate a "useful" addition to a resume are critical and invaluable. Overall, this paper provides an excellent example of Natural Language Processing being utilized in this field and will be extremely valuable to the final project.

B. A recent overview of the state-of-the-art elements of text classification [2]

This paper discusses the general process utilized by researchers to proceed in classifying text-based datasets. Although not explicitly focused on Natural Language Processing, this article provides phenomenal insight into the process of not only classifying data, but also acquiring data for a dataset, the process of labeling such data, and then deriving features and data representation to perform classification and evaluation thereon. It is a fascinating insight into the entire procedure that would otherwise be inapplicable to this project, as this project jumps straight into the preprocessing stage of the dataset rather than the entire acquisition process behind it. Thus, excellent insights and appreciations are derived from the article to therefore benefit this project in turn, and the significance of some of the suggestions provided by the paper should not be disregarded.

C. *A Complete Process of Text Classification System Using State-of-the-Art NLP Models [3]*

Following the general outline of the previous work, a more detailed and deep learning-oriented article was prudent and necessary to advance the project forward. Although several Natural Language Processing pipelines gravitate towards using models such as KNN, Naïve Bayes, or Decision Trees, this article delves into the formulation of neural networks and deep learning systems to achieve classification goals as well as some of the advantages and disadvantages of utilizing these neural networks over alternative models. As such, this paper became a driving factor for this project to focus on neural networks rather than some of the other less complicated methods that have been vastly utilized previously. Additionally, this article provides phenomenal insight into the "Benefits and limitations" of various feature extraction methods (such as Bag-of-words, TF-IDF, and Word2vec) to allow for the researcher of this new project to make informed decisions for the preprocessing and encoding without needing to test such methods additionally and extensively before even proceeding to neural network comparison. Overall, this article was more than paramount to the foundation of this project, and the information and knowledge gained from it was more than invaluable to allow for this project to even exist.

III. Proposed Method

Using the information and knowledge gained from the sources, the researcher began identifying the various elements that needed to be completed before actual experimentation could begin. Firstly, preprocessing for the dataset was necessary, which ended up necessitating more hands-on interaction than was planned originally. For strange reasons still unknown, the dataset was encoded in a format other than UTF-8 (a default and standard) and finding an alternative method for decoding (Latin-1) was not immediately obvious or understood.

Once the encoding issue was resolved, the researcher was able to then proceed with data exploration of the dataset to better understand what kind of neural networks could be utilized in the experimentation for the project. Following the understanding of the "flagged" vs "not flagged" class identifier, it was here that the clear differentiation between the Resume and Chatbot subsets in their text-fields was understood. Unfortunately, the end result was that both subsets would require very different and, in their own unique ways, challenging preprocessing.

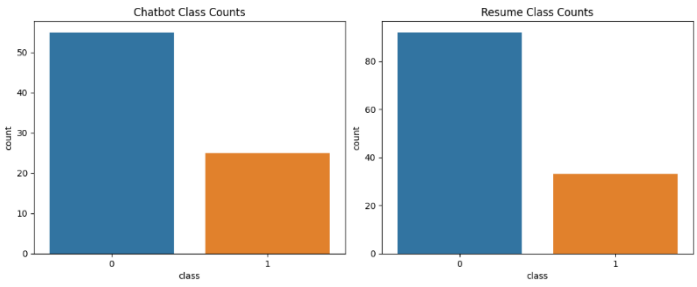

Fig. 1. Comparison barcharts of Chatbot and Resume class counts.

For the Chatbot subset, several columns containing mostly empty values was identified and required thorough investigation to the purpose of them as well as more delicate methodology for removing the obsolete columns as well as the associated commas appended to the end of each text field. For the Resume subset, it became immediately clear that the greatest challenge would be to remove the formatting escape-characters and punctuation that was present due to the text responses being raw text extracted from the sample population. This process of removing all punctuation and obsolete metadata was rather time-consuming and challenging, but in turn enabled the researcher to use the strategies prior discovered in the overall text preprocessing for both data subsets.

This aforementioned text preprocessing is a series of several combined strategies to clean and optimize the data present in the Natural Language Processing procedure. Before in-depth preprocessing could be performed on these text-responses, tokenization was performed to split each response into smaller items (i.e. tokens) by either punctuation or space delimiters. This allowed for the added bonus of also removing all useless metadata formatting when used in combination with the removal of words smaller than 3 letters long (as such words would be either irrelevant to Natural Language Processing, or the remnants of the metadata purging). Additionally, the practice of lowercasing all values and removing common "stop words" (such as "the", "with", etc.) were some of the additional steps taken for sanitizing the text for more decisive and optimized results.

In conjunction with this, the process of removing irrelevant or overly-unique words (such as names or other nouns) was investigated and attempted. Here, a Frequency Distribution (otherwise known as FreqDist) was used to identify tokens that were not commonly used across all the various datapoints. Initially, with this it was used to remove all words that don't appear more than once. This would usually be ideal, however the Chatbot dataset contains an overabundance of such words that would be removed due to it being a smaller subset than Resume in terms of both response length and response tally.

Where a sentence, after previous cleaning, might appear as "try avoid sort conflict", using this method results in the almost comical result of just "try". Nothing else; which is not more useful for Natural Language Processing purposes. As such, this methodology is still present in the code but is not utilized for either subset in the dataset in an effort to preserve it for future datasets that may be larger and more thorough. Although it may be utilizable in the Resume subset, a decision was made to also not pursue this practice in this case to maintain similar preprocessing results and procedures in both experimental cases. That said, this would be an interesting and likely effective avenue to pursue in future extensions of this project.

For the final step of preprocessing before experimentation could begin with neural networks, the act of labeling and vectorizing the dataset was necessary to allow the data to be used in Natural Language Processing methods. Since computers can not understand human language but are capable of interpreting numbers, the first step for this process was to change the labels of "flagged" and "not flagged" into the binary class of 1, for

flagged, and 0, for not flagged. With the labels now encoded, the process of vectorizing the words found within the responses was investigated to identify the ideal or optimal vectorizer method.

As mentioned previously [3], the process of extracting features by using such concepts as Bag-of-words or TF-IDF were immediately considered. After some testing and research, a decision was made to use a TF-IDF (term frequency-inverse document frequency) vectorizer to transform the collection of "raw documents" i.e. text-responses into a matrix of TF-IDF features, thus numbers. One-Hot Encoding was also considered, but due to the nature of the responses having "sparse" occurrences of words across a large dataset, the usage of TF-IDF resulted in much more appropriate vectorization. However, this does present an interesting possibility for analysis of various encoding methods as the studied variables rather than the neural networks (as is the case in this project), and with more time or an extension of this project, interesting results may be found here that are not found within the current scope for this project.

With preprocessing completed, the researcher then identified the three neural network formations to be utilized in the experiments for this project. Using recommended formations from various sources, the final three Natural Language Processing neural network models were decided on to be Recurrent Neural Network, a Long Short-Term Memory, and Gated Recurrent Unit (hereafter denoted as RNN, LSTM, and GRU respectively). With these models decided upon, the researcher could now proceed with the experiments for the dataset.

IV. EXPERIMENTS

For all three models, a Train/Test split was performed to allocate the majority of the data to be used in a training subset (respectively for both the Chatbot and Resume subsets) and then validated against the smaller yet, importantly, isolated testing subset. This allowed for an analysis of the over or underfitting that might be found within the results of each respective model. Additionally, a variable was identified to be providing recurrent dropout to the models versus utilizing them in a more simplistic form to identify if one version of a model was better than another of the same type, rather than solely between overall model types.

A great deal of iterative testing with different settings (such as dropout rate, batch size, epoch total, etc.) was performed for each individual model. But, in the end, a similar decision to the previous one of not utilizing the FreqDist preprocessing was used to promote balanced performance between all three model types. This being that all models would use a fixed random state (42, the meaning of life) on a Sequential model with one Embedding layer (with parameters of input size at 1000 and output size at 32), one main model layer (RNN, LSTM, or GRU, each of which with 100 units), and then one Dense layer of activation type sigmoid. All models used the "adam" optimizer in compilation with loss being calculated through binary cross-entropy and analyzed against the metric of accuracy. Finally, all models were fitted with an epoch value of 5, a batch size of 32, and a droupout/ recurrent dropout value of 0.2. Any discrepancies from these values were due to a necessity of the kernel crashing otherwise without them for these individual values (specifically, the batch size being 32, instead of a prior attempted value of 64).

This mention of the kernel crashing is an introduction to a problem that persisted and plagued almost the entire process of experimentation with the neural networks. In the original setup for the experiments, all three types of models were located within a single code file to be run in sequence, one after the other. This is mentioned here to further identify a the aforementioned recurring issue with the choice for using neural networks, as the researcher quickly found out that in utilizing these three models, it was a very common occurrence for the entire kernel to crash at random when attempting to fit the models to the Resume dataset specifically.

For some attempts (such as when using RNN), the models were capable of being fit more often than not. However, when utilizing dropout against LSTM and GRU specifically, it became nearly impossible to complete a fit without the kernel crashing. After extensive tweaking of settings (both within the code/ model parameters and in the kernel's environment itself) and by isolating the various models into their own respective, independent code files, the researcher was finally able to consistently run all model fits with the Resume subset without crashing. The researcher only regrets that it could not have been fixed sooner into the experimentation process.

Thus unfortunately, this recurring issue led to a rather limited scope of both what the researcher could experiment with in time as far as model type and structure, as well as how much variety could be found in the parameters themselves. Potentially in the future, it would be ideal to work in a different environment with GPU capabilities to not only enhance the speed at which models could be fit, but also to perform a wider variety of parameter selection and model construction. This noted, the act of using more rudimentary and more similarly constructed models also provides for an increased stability as a foundation for future projects with differing datasets which may not have functioned as effectively without a baseline comparison that is found with these harmoniously constructed neural networks.

Regardless, the results found within the experiments posed fascinating results as an analysis of these model types in light of the selected dataset. To begin, the results found from the Chatbot were acquired significantly faster than for the Resume. As may be seen previously, the Resume subset is significantly larger than the Chatbot subset, in terms of both data quantity (almost twice as many datapoints for Resume than Chatbot) and data value length (where some datapoints for Resume had dozens of tokens post-preprocessing, the Chatbot text-responses often retained only a few words in each datapoint).

This does demonstrate an intriguing result and relationship between both subsets, as the Chatbot model-fitting exhibited generally worse average accuracy of classification with significantly more rapid model fitting time than that of Resume text-responses. And of course, the inverse for Resume text is true as well: several power's worth more of seconds taken to fit, yet likewise much better loss and accuracy results. These results are displayed as such:

	Chatbot Averages			Resume Averages		
Model Type	Avg. Loss	Avg. Acc.	Fitting Time	Avg. Loss	Avg. Acc.	Fitting Time
RNN w/o Dropout	0.5776	0.7578	1.2s	0.61995	0.685	144.4s
RNN w/ Dropout	0.75245	0.3203	1.1s	0.6967	0.55	157.7s
LSTM w/o Dropout	0.5687	0.7578	2.7s	0.6532	0.685	130.9s
LSTM w/ Dropout	0.71185	0.35155	1.2s	0.664	0.595	103.0s
GRU w/o Dropout	0.6313	0.6875	2.4s	0.53915	0.775	124.1s
GRU w/ Dropout	0.71005	0.42185	1.3s	0.6497	0.7	101.8s
Average of Averages	0.658658333	0.549466667	1.65s	0.637116667	0.665	126.9s

The above table displays the average between training and validation results for each model and their resulting loss and accuracy values. Further detailed tables and graphs will follow hereafter to provide more intricate results. The purpose of this table in particular, however, is to provide a generalized overview of the results for each model's performance without regards to overfitting or underfitting. Of course, this is an incomplete demonstration of the more telling details of each model's strengths and weaknesses, but it does provide a satisfactory general story from a glance of which models excelled in which datasets for the various metrics.

Looking at the averaged values marked in green to denote the best results for each metric (ties included), it can be seen that LSTM w/o Dropout provides the best loss and accuracy results for the Chatbot subset (at the cost of the worst fitting time) while GRU w/o Dropout is the overall best in averaged loss and accuracy for the Resume subset with the median fitting time. Additionally, by looking at the Average of Averages, we can compare the vastly different values of 1.65s on average for Chatbot versus 126.9s on average for Resume—almost 77 times the average! It is a serious testament to the vastly different structures and quantitative values of the respective subsets.

With these preliminary results considered, the next prudent step is to consider the original, unaveraged results to identify any standout models for their respective datasets to see what information or knowledge might be gained from the varying models.

First assessing the detailed results of the Chatbot subset, an interesting divergence from the generalized results occurs. Primarily, the exhibition of GRU w/o Dropout as the best training values to loss and accuracy is visible. However, in support of the generalized data, we see that LSTM w/o Dropout boasts a higher validation accuracy and better validation loss value than that of the averaged values. Thus, we beneficially learn that LSTM without Dropout is, indeed, the best model for the Chatbot subset with even better results than would be expected from a cursory glance at the original table.

A validation accuracy of 0.875 exhibits more than satisfactory capabilities of classification in this scenario, and the loss value of 0.4825 gives a far superior loss value than any of the other models (the highest of which being 0.795, 0.3125 worse than the LSTM w/o Dropout model). Of course, the 2.7 second fitting time for the model is in fact the worst fitting time for this table. However, in comparison to the Resume fitting times, this is almost irrelevant in this given case. Of course, it is prudent to remember that, given a much larger Chatbot dataset with the same purpose and structure as the current Chatbot subset, this increased time would become incredibly relevant and exponentially exacerbated, so it is worth noting for future analyses.

With these various results considered and analyzed, it becomes relevant, of course, to consider the results and possible findings from that of the Resume subset and how its more specific products might be of interest to the researcher:

Chatbot Results					
Model Type	Loss	Accuracy	Validation Loss	Validation Accuracy	Model Fitting Time
RNN w/o Dropout	0.6565	0.6406	0.4987	0.875	1.2s
RNN w/ Dropout	0.7099	0.5156	0.795	0.125	1.1s
LSTM w/o Dropout	0.6549	0.6406	0.4825	0.875	2.7s
LSTM w/ Dropout	0.694	0.5781	0.7297	0.125	1.2s
GRU w/o Dropout	0.6352	0.6875	0.6274	0.6875	2.4s
GRU w/ Dropout	0.7139	0.5312	0.7062	0.3125	1.3s
Avg:	0.6774	0.5989	0.6399	0.5	1.65s

Fig. 2. Individualized Chatbot Results in a Table.

Resume Results					
Model Type	Loss	Accuracy	Validation Loss	Validation Accuracy	Model Fitting Time
RNN w/o Dropout	0.5645	0.77	0.6754	0.6	144.4s
RNN w/ Dropout	0.7184	0.5	0.675	0.6	157.7s
LSTM w/o Dropout	0.5447	0.77	0.7617	0.6	130.9s
LSTM w/ Dropout	0.6522	0.59	0.6758	0.6	103.0s
GRU w/o Dropout	0.6187	0.71	0.4596	0.84	124.1s
GRU w/ Dropout	0.6743	0.56	0.6251	0.84	101.8s
Avg:	0.6288	0.65	0.6454	0.68	126.9s

Fig. 3. Individualized Resume Results in a Table.

In similar fashion to the analysis of the Chatbot results, identifying the difference between the averaged table and the individualized table becomes prudent, especially when focusing on the potential for usage in classification on a larger scale. Thus, an immediate focus is given to the GRU w/o Dropout model results, as this model provides the highest validation accuracy (tied at 0.84) and lowest validation loss (0.4596) in the table. This does, indeed, correlate to the knowledge gained from the original generalized averages table, where previously similar results were recognized. However, it is incredibly important to note at this point that this validation accuracy is, in fact, lower than the best validation accuracy for the Chatbot results.

This discovery changes the resulting findings and comparison considerably, as although on average the models used with the Resume subset may yield higher accuracies, on an individual level when solely comparing validation accuracy results, the models used for the resume dataset not only take significantly longer to fit, but also in no way can provide a result that gives an advantageous or even equivalent production in comparison.

This entire analysis between both the averaged values and the individualized training-separate-from-testing values gives interesting findings to the topic of data analysis and metric comparison for machine learning results. It does, additionally, produce an interesting question of whether or not the Chatbot data, if increased to that of the size and quantity of the Resume subset, would continue to produce models with superior validation accuracy, or if it would result in better results from a different model or display results that were no longer superior to that of Resume. In future projects, this would be a very interesting consideration.

To conclude the findings for the experiments, graphs demonstrating the individual results for the models, as well as the average accuracy and loss values across all models, are provided to increase the visualization of the differences between these respective models.

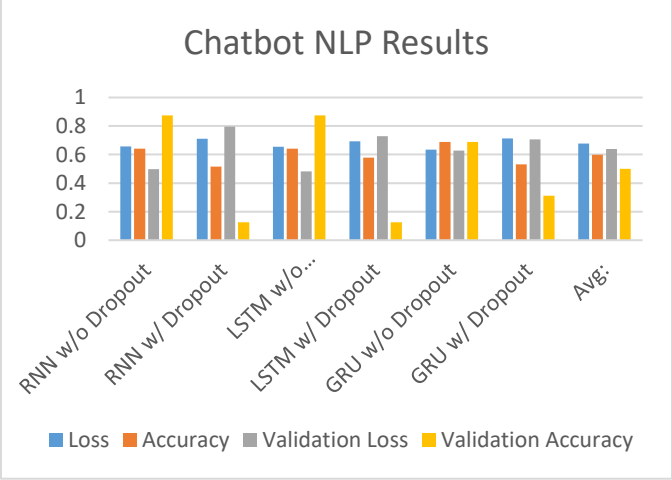

Fig. 4. Chatbot Results in a Graph.

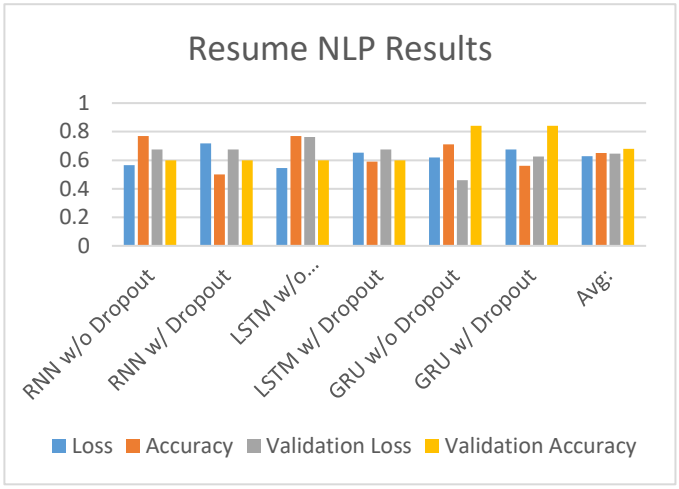

Fig. 5. Resume Results in a Graph.

Something of note, especially visible after seeing these graphs, is the significance of Dropout, especially to the Chatbot subset. Where the Resume subset sees little if any difference in the validation accuracy between the Dropout vs No Dropout models, the Chatbot subset displays immense decreases in validation accuracy when utilizing dropout. It is believed that this is due to the considerable difference in data size between the two subsets, and that although impactful for Chatbot where it has far fewer samples, Resume has an increased buffer thanks to its far larger sample size.

Between both subsets, the training accuracy is worse, as well as the loss values which, unfortunately, also increase (which is bad) between no dropout and when it is present. In future projects, it would be interesting to vary the dropout amounts (including between regular dropout and recurrent dropout) as well as see if the varying dataset sizes or model constructions lead to any different results.

V. Conclusion and Future Works

At the conclusion of this project, a great number of findings and discoveries have been made. From the initial research to the thorough exploration of the provided dataset to the preprocessing and the experimentation and its results, various topics of what could be utilized in future works and done differently in additional extensions or continuations of this project have been found.

In future projects, it would be primarily prudent to utilize far larger datasets to maximize the potential for Natural Language Processing using deep learning neural networks. The Chatbot analysis results demonstrate a variety of weaknesses due to both its lower number of datapoints and its shorter responses in general. With larger datasets in the future, the capacity for more preprocessing and more refined cleanup would likely be invaluable, and it is entirely possible that this by itself could be an entire project worth pursuing. Furthermore, pursuing methods for rectifying data imbalances, such as there being far more “not flagged” responses for the dataset in comparison to the “flagged” responses would likely change the results significantly. Regrettably, such methods for preprocessing were

outside the scope of this project, and their impacts for the presented findings are currently unknown.

Additionally, with a larger time frame it would be surely worthwhile to consider a higher variety of models, in both type of model itself as well as varying a model's parameters and network structure, both of which were out of the scope to be fully explored and analyzed in this project. Two of these parametric variables, dropout and its associated recurrent dropout, necessitate more study as well to better understand their impact on Natural Language Processing as a whole.

Aside from solely focusing on neural networks, as this project aimed to do, a comparison of the accuracy of more basic Natural Language Processing models, such as that of KNN or decision trees, might also provide interesting differences than that of the neural network-oriented models. Of course, these would provide their own series of variable and parametric modifications, and thus would likely pose a great amount of work and discovery when compared to the findings demonstrated in this project.

As for other focuses, such as that of varying the encoding type or the train/test split (or even considering a train/dev/test split) percentage, both present interesting considerations. These potential changes could provide potentially fascinating results when compared to the chosen variables for this project.

Considering any advances or shortcomings of this project, it is hoped that this project may serve as an excellent foundation for such further studies. With additional time and resources, such as utilizing the power of GPU in model fitting, an extensive number of variables and modifications to this project could be accomplished to advance the already-presented findings further.

To conclude, the topic of previous proposals for this project are worth revisiting and evaluating to identify whether this project yielded the insights that it initially set out to pursue. Indeed, a fine comparison between three of the common neural networks used for Natural Language Processing has been accomplished, as was set out to do. One goal was to present models that might be able to be used in the subjects of AI analysis in a therapeutic field, as well as the processing for the validity of resumes when presented. It is proposed that, indeed this too was accomplished.

Using the tools developed here to not only preprocess a dataset but also to analyze it using Natural Language Processing is, by the opinion of the researcher and backed by the results of satisfactory validation accuracy, accomplished in this project. Finally, the question of whether these models could be extended to usage in further fields, adjacent or distant in nature, to the chatbot and resume subjects was posed. It is yet unknown without more datasets, but it is believed that through comparison to the current findings of this project and study, results could easily be produced to validate or deny this question.

REFERENCES

[1] Takhom, P. Chirawat and P. Boonkwan, "Comment Usefulness Classification on Youtube using Artificial Neural Networks," 2023 18th International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP), Bangkok.

[2] Mirończuk, Marcin Michał, and Jarosław Protasiewicz. "A Recent Overview of the State-of-The-Art Elements of Text Classification." Expert Systems with Applications, vol. 106, Sept. 2018, pp. 36–54, https://doi.org/10.1016/j.eswa.2018.03.058

[3] Dogra V, Verma S, Kavita, Chatterjee P, Shafi J, Choi J, Ijaz MF. A Complete Process of Text Classification System Using State-of-the-Art NLP Models. Comput Intell Neurosci. 2022 Jun 9;2022:1883698. doi: 10.1155/2022/1883698. PMID: 35720939; PMCID: PMC9203176.

[4] "Jan Kirenz - Text Mining and Sentiment Analysis." Www.kirenz.com, www.kirenz.com/blog/posts/2021-12-11-text-mining-and-sentiment-analysis-with-nltk-and-pandas-in-python/.

[5] Brownlee, Jason. "Sequence Classification with LSTM Recurrent Neural Networks in Python with Keras." Machine Learning Mastery, 25 July 2016, machinelearningmastery.com/sequence-classification-lstm-recurrent-neural-networks-python-keras/.

[6] "Deep-NLP." Www.kaggle.com, www.kaggle.com/datasets/samdeeplearning/deepnlp/data

Access the code here:

https://github.com/kaisermikael/NLP_Deep_Learning_Analysis