DEVELOPMENT AND DEPLOYMENT OF A ROBUST SYSTEM FOR SECURE DOCUMENT TRANSFER

2024-R24-130

Gunawardna K.P - IT20298876

B.Sc. (Hons) Degree in Information Technology

Specializing in Data Science

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

February 23, 2024

HATE SPEECH DETECTION

2024-R24-130

Project Final Report

Gunawardna K.P - IT20298876
B.Sc. (Hons) Degree in Information Technology
Specializing in Data Science

Department of Information Technology
Sri Lanka Institute of Information Technology
Sri Lanka

February 23, 2024,

DECLARATION

I declare that this is my own work and this dissertation1 does not incorporate without acknowledgement any material previously submitted for a degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to Sri Lanka Institute of Information Technology, the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Name	**It number**	**Signature**
Gunawardana K.P	IT20298876	

The supervisor/s should certify the final report with the following declaration. The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

Signature of the supervisor: _____ Date: _____

ABSTRACT

It proposes a hate speech detection model developed to respond to the rising need for automated content moderation in sensitive areas, which include military news and political discourse. In this work, several feature extraction methods such as TF-IDF, Bag of Words, and Word2Vec were considered with various machine learning models including RNN, CNN, LSTM, and BILSTM on a dataset with significant class imbalance. Therefore, with up to a 95% accuracy, an F1 score offering well-balanced detection across hate speech, offensive, and neutral categories, BILSTM was chosen as the final model after comprehensive evaluation.

Class imbalance was addressed with strategies of oversampling and under-sampling, respectively. These techniques, though effective, had some limitations in the way that several cases of false positives were recorded, and sensitiveness of the model existed regarding specific terms. Another critical issue was the nature of these deep learning models being computationally intensive, hence difficult to scale and resource-demanding. However, among the chosen models, BILSTM proved very promising and suitable for real-world application in hate speech detection.

Conclusions and recommendations were provided on integrating enhancements through, attention mechanisms, and explainability techniques, which shall further enhance the adaptability, transparency, and accuracy of the model. The recommendation will serve as suggestions for future efforts to provide a more robust, scalable, and interpretable hate speech detection solution that is resilient in the face of fast-changing trends in online discourse.

ACKNOWLEDGEMENT

This study was conducted to fulfill the academic research requirements of the subject of Comprehensive Design Analysis Project. It represents the culmination of the dedicated efforts of my group members. Deep appreciation is owed to the individuals who provided encouragement, support, and guidance throughout this endeavor.

My heartfelt appreciation is extended to my supervisor, Dr. Harinda Fernando, for his invaluable guidance, comments, and suggestions throughout the development of this research.

Additionally, gratitude is expressed to my co-supervisor, Mr. Deemantha Siriwardana, for his Assistance and knowledge.

Special thanks are also due to the evaluation panel members, Mr. Amila Nuwan, Mr.Kavinga Yapa and Mr. Deemantha Siriwardana for their invaluable suggestions and insights for the project.

Furthermore, appreciation is extended to all colleagues, friends, and family for their unwavering support, interest, and valuable advice throughout this project. Each contribution, whether direct or indirect, has been instrumental in this journey.

Table of Contents

Table of figures

1. INTRODUCTION

1.1 Background

Hate speech, as it relates to its modern definition, is much more broadly defined as communication methods through the verbal, the written word, symbols, and digital communication. For this work, the definition will extend to any communicative practice that articulates prejudice, hostility, or hatred against persons or groups based on fixed or perceived immutable attributes such as race, ethnicity, religion, nationality, sexual orientation, gender identity, and other defining personal characteristics. This is not only speech that marginalizes individuals, but rather one intended to incite or reaffirm divisions within society, leading to both discrimination and, in many cases, violence.

Hate speech includes several actions: the use of derogatory language, racial and ethnic slurs, threats, and even sustained harassment. Hate speech can also be the purposeful spread of false or misleading information as a means of dehumanizing, demeaning, or instilling fear and aggression toward certain groups. As technology grows, hate speech, through written texts, especially on social media and other online platforms, becomes increasingly worrisome. With the speed and spread that the internet covers, these destructive messages are likely to reach massive audiences, increasing their damage to individuals and communities alike.

This increase in hate speech through text brings about several serious challenges. Online spaces lack the checks that would ward off or soften hurtful content into an environment that provides license to individuals who find venting hateful opinions with relative anonymity behind computer and phone screens particularly appealing. All this has deep results: hate speech psychically destroys those it targets, erodes cohesion in society, and perpetuates inequalities. This means that quite a few countries, organizations, and platforms have put huge resources into acting against hate speech in the situation, as it is not only a social problem but could threaten peace, equality, and even the lives of those concerned.

Hate speech can lead to grave and profound harm to individuals and communities across a wide variety of dimensions. Members of targeted groups can feel fearful, insecure, and alienated upon exposure to hate language. The psychological effects create heavy burdens: hate speech can undermine the confidence and self-worth of targeted individuals, causing long-lasting distress, anxiety, and social withdrawal. Therefore, hate speech has escalated into a total disaster in society and has to be given immediate attention and action.

Modern technology has also not done any good for this problem. In this respect, digital platforms have simply increased the spread of hate speech to unimaginable audiences. More specifically, social media has become a shared space where such offensive language can spread in no time; it just becomes easy to shape an environment where hate speech becomes not only so prevalent but very hard to moderate. Contrarily, beyond social media, hate speech manifests itself in news and political domains to make certain of public opinion and to manipulate thought. In political discourse and media stories, hate speech is a powerful mechanism for influencing society, usually further polarizing it to attain some end.

Reports, such as the Amnesty International Report 2017/18, have brought to light the alarming rise of politically motivated hate speech around the world. This report shows that such cases where the use of hate speech by so-called political actors has increased, mobilize support, undermine opponents of ideas or policy positions, or intentionally exacerbate deepening societal divisions. For instance, sensationalized or one-sided news reporting may create and reinforce stereotypes with the use of hate speech, stigmatize groups, or configure the public view to make it believe in ways that perpetuate prejudice. It is a technique that is used often to divide people, consolidate one's base, or squeeze out the minority opinion by using either incorrect information or fear to foster resentment or hate.

The need for hate speech detection has gained momentum in recent times, especially from domains of authority such as news and politics. Deceitful or provocative information can also bring about the distortion of democratic discourse and trivialization of prejudiced and discriminatory beliefs in society. It is in these areas that the identification and tackling of hate speech become so vital for the building of responsible reporting and constructive political involvement. Setting clear standards and deploying state-of-the-art detection methodologies is the first attempt by societies

at stemming the tide of hate speech, protecting vulnerable groups, and fostering a more respectful, inclusive public dialogue.

Hate speech detection has become one of the crucial research and development areas; hundreds of different approaches are being considered for an effective solution to this problem. The most popular ones include ML and DL algorithms, which help systems automatically detect and classify hate speech among extensive amounts of text. These are algorithms that are pre-trained on large datasets comprising text samples that have been labeled in as much detail as possible to indicate the presence or absence of hate speech. The algorithms learn through such training to identify linguistic patterns and phrasing combined with contextual clues that define hateful language; thus, generalizing and applying this knowledge to new, unseen text.

In reality, support vector machines, random forests, and decision trees have competed reasonably well in basic hate speech detection. Deep learning methods, neural networks, and, more concretely, architectures such as RNNs and transformers-e.g., BERT and GPT-really took accuracy and scalability in hate speech detection to a new level. Advanced models capture nuanced language features, including sarcasm, slang, and contextual subtleties, which often typify hate speech.

What will make this system even more effective is if the datasets are of a wide range: representative sets of languages, cultural expressions, and social contexts. Further, the retraining of these algorithms would have to be continuous, since language evolves at a very rapid pace and new terms or coded language might be born with which to convey hate in unprecedented ways. Further helpful techniques have included transfer learning, where models are adapted to new contexts or languages by leveraging prior training on large, general datasets and fine-tuning with specific hate speech datasets.

Another promising direction involves the integration of context-sensitive NLP techniques, as hate speech is many times based on implied meanings or hidden sentiment. On this basis, the integration of NLP with ML/DL will allow systems to better detect benign statements from non-benign ones that carry a negative intention and decrease false positives.

The general approach of machine learning and deep learning is scalable and flexible in identifying hate speech on these platforms. These approaches could work even more powerfully in the future, with additional research to improve algorithmic precision, and serve as important

parts of broader societal efforts to monitor and mitigate the effects of hate speech in online and offline communications.

1.2 Literature survey

In the case of hate speech, it is an abnormally complex issue; that is why data scientists have been working hard on its detection methodology development. Research was conducted and enhanced on basically two important techniques: first, classical machine learning techniques, and secondly, more advanced deep learning models. These algorithms now form the very tools necessary in combating hate speech by providing the analytic backbone for intervention and mitigants on social media and other digital platforms.

The classical machine learning methods include logistic regression, decision trees, and support vector machines that form the lower layer of hate speech detection. These models are usually efficient and interpretable, hence analysts can interpret why certain decisions have been made, and tune the algorithm against a specific linguistic pattern. Indeed, these classical ML approaches are particularly appropriate for binary classification-hate speech versus non-hate speech-and perform quite well if they are trained on suitably curated datasets.

While the tasks of language itself were becoming increasingly complex and contextual, a number of deep learning models began to gain more attention as successful successors. The methods developed in deep learning, especially the neural network-based ones, depict patterns both linguistically and contextually that might have gone amiss in traditional ML models. While convolutional neural networks and recurrent neural networks together with transformer-based models represented by BERT and GPT have been continuously expanding the boundary of hate speech detection tasks with deep semantic and syntactic analysis of language. These models excel in contextual and subtle tasks, in that they masterfully can pick up sarcasm, indirect threats, and coded language-all common features of hate speech.

This constant reement arises not only from the magnitude but also from the complexity of the language problem at play on social media, where hate speech can shift on a dime with every cultural pendulum swing and flux particular to various platforms. Further, to be truly robust and inclusive, such algorithms must be trained with large, diverse datasets; hate speech can, after all,

vary significantly across languages, communities, and social contexts. These machine learning and deep learning models mean that the data scientist is instrumental in providing society with advanced and automated tools, which are able to identify hate speech more accurately, as well as provide a foundation for responsible moderation and response strategies on a global scale.

This combination of approaches with machine learning and deep learning, therefore, has provided a framework that is scalable, flexible for the detection of hate speech, and hence forms a very fundamental step in the attempt to make online environments much safer and respectable.

Much of the research in the detection of hate speech is based on models developed from datasets obtained from Twitter, which is, in fact, an exceptional resource for such research. Twitter's character limit, first at 140 and then extended to 280, means its data consists mainly of short and concise messages. It is this brevity that forms the nature of the data, which poses unique challenges and opportunities in coming up with an effective detection model. Because of this, many of the algorithms that have been developed to detect hate speech have been optimized for short-text formats.

Model approaches to detecting hate speech are dominated by machine learning methods, especially deep learning methods. In the case of traditional machine learning algorithms, support vector machines are very popular because they efficiently classify problems into two classes, which is really important for distinguishing hate speech from non-hate speech. The underlying ability of SVMs to map input data into high-dimensional spaces enables them to classify text with high accuracy and therefore makes them very well-suited for the analysis of short, dense texts such as tweets.

Other classical ML models, including Random Forest, k-nearest neighbors, and Naive Bayes, are also used widely in hate speech detection. Each of them has certain merits that contribute to increasing the efficiency and adaptability quotients of the entire set of hate speech detection systems. Take the case of Random Forest-an ensemble model which constructs multiple decision trees and leverages the combined outputs to reduce over fitting while improving accuracy. In contrast, KNN classifies texts depending on the 'neighbors' or similar data points, which proves to be pretty efficient, especially in environments containing short texts where some particular phrases or word patterns indicate hate speech quite frequently. Naive Bayes is a probabilistic classifier and

is much valued because of its simplicity and speed, hence proving highly useful in circumstances of large volumes of short text data with much lower computational cost.

While these models have been already deployed on Twitter data, researchers are in the process of updating their models to keep pace with how online language has been changing. Such nature of short-text content brings about a number of unique challenges-things are highly abbreviated, filled with hashtags, and emojis that can connote nuanced meanings or subtexts. By refining algorithms to grasp that nuance, among other linguistic features effectively, researchers are advancing the state of the art in research on hate speech, aiming at the reliable processing by systems able to adapt to the ever-changing patterns in Twitter-like platforms. In the final analysis, such efforts promise to add to a more general enterprise: the automated identification of hate speech and its neutralization in real time across social media environments.

LSTMs and BiLSTMs form the basis of most deep learning architectures proposed for hate speech detection, due to their ability to handle long-range dependencies in text. The LSTM model is great for processing sequences by retaining much of the contextual information spanning large portions of text; it is thereby very powerful at modeling complex patterns in language, such as hate speech. The BiLSTM models go a step further by analyzing text both ways, both ways being forward and backward; in such a way, it can yield a far more complete comprehension of how words are related and what the context is. This is quite useful in identifying subtle cues from the text that may point to hate speech, given the model's enhanced ability in grasping such nuances.

Other deep learning models that have been used for hate speech detection tasks apart from the LSTM and BiLSTM include the Convolutional Neural Networks. Traditionally used in image processing, CNN has equally achieved remarkable successes in natural language processing tasks as they find local patterns, such as specific phrases or certain combinations of words indicative of hate speech. CNN works really well with pre-trained word embeddings, such as Word2Vec and GloVe, which acquire rich contextual knowledge from big corpora of general text. The embeddings can represent semantic relationships between words; hence, the model will understand the context even if some phrases or words are changed.

Pre-trained embeddings, such as Word2Vec and GloVe, therefore give the CNN models a better initialization that can enable them to effectively recognize linguistic patterns that are indicative of hate speech within a very short span and without requiring large hate-speech-specific datasets.

This further helps the CNNs enhance the accuracy of detection by learning common patterns, phrases, and even coded languages prevalent in hate speech. Therefore, deep learning models-LSTM, BiLSTM, and CNN-with pre-trained embeddings come together as a powerful approach toward hate speech detection that ultimately assists a researcher in the development of sophisticated systems. These systems have become capable of analyzing and giving meaning to complex textual content on social media and other online platforms.

In the paper "Multi-modal Hate Speech Detection using Machine Learning," the authors discuss a holistic approach for hate speech detection, combining information from multiple modalities-text, audio, and images-to overcome the limitations posed by single-modality approaches. Most hate speech research has focused uniquely on text-based methods up until now, letting non-verbal signals often provide critical context in the identification of hateful intent fall by the wayside.

It builds on previous work that has achieved reasonable success in text-only hate speech detection. For example, prior works, such as those by Warner and Hirschberg (2012), using Support Vector Machines, focused on performing the detection based on the linguistic features present in the text. Later research included deep learning techniques as well, where Badjatiya et al. (2017) tried using LSTM and Random Forest models on Twitter data; the increase in the classification accuracy of the text data was still not considering contextual cues from other forms of media.

By expanding to a multi-modal approach, the authors address this gap. Their approach basically involves the collection of video data from YouTube or other sources and labels video segments as either hateful or non-hateful according to combined text, audio, and image data. Feature extraction specific to each kind of data is then made: images are processed to capture visual cues like anger and disgust, audio is analyzed for features like MFCC and Zero-Crossing Rate for vocal tones, and text is vectorized through TF-IDF to spot language patterns indicative of hate. These modalities separately output through a voting system to make sure that the model considers a fuller range of cues, hence addressing challenges in hate speech detection which might depend on subtle context.

This is a multilevel approach that, compared to single-mode methods, better reflects how human perception evaluates hate speech. While previous models were highly focused on textual analysis and sometimes completely failed to capture nuances or ambiguities in hate speech, this study

integrates additional layers of data into a more fine-tuned identification process. This proposed multimodal approach goes in the line of current trends within modern machine learning, which intends to make use of a wide variety of inputs in order to increase depth in contextual understanding. In fact, this model represents a serious improvement compared to single-modality detection because it integrates a richer set of indicators coming from the video content; hence, it is more adaptable toward a range of online hate speech manifestations.

In general, according to the authors' results, multimodal models seem to be a promising line for future work. Its extension with regard to source, language, and online platform would increase the chances of improving adaptability. Again, the use of advanced models, like BERT for text processing or Vision Transformers for image analysis, would further improve contextual accuracy. In addition, real deployment scenarios with real-time applications involved, such as live-streaming, may yield useful immediate benefits when real-time analysis becomes highly important in content moderation. [1]

In the paper "Hate Speech Detection Using Text Mining and Machine Learning," the authors raise concerns about the dire need to effectively carry out automatic hate speech detection on social media, where such content poses social risks. They then propose a framework and a model for detecting hate speech through text mining processes and supervised machine learning by using the Naïve Bayes classifier. In this work, the authors presented two Twitter datasets that involved text mining techniques and feature extractions to enhance hate speech classification. Their model was efficient enough to record an accuracy of 87.23% in one dataset and 93.06% in another.

This work builds on prior research involving the use of hate speech detection and sentiment analysis methods; techniques that commonly include machine learning algorithms such as SVM, Naïve Bayes, and logistic regression. The earlier work of Davidson et al., 2017, utilized Twitter data in implementing a classification of tweets into hate speech, offensive language, and neutral speech. The system struggled to make out distinctions between offensive language and hate speech. Alaoui et al. focused their attention on such distinctions by implementing advanced pre-processing steps and text mining methods in order to classify hate speech more precisely, especially in distinguishing it from general offensive content.

While the authors have relied on Naïve Bayes for probabilistic classification, as opposed to the lexicon-based approaches that have tended to misclassify neutral or offensive language as hate

speech, their performance has indeed been better in terms of precision and recall. This is also in agreement with Ruwandika and Weerasinghe's 2018 results, which showed that the performance of Naïve Bayes performs even better with text pre-processing and TF-IDF feature extraction.

For future work, the authors suggest using deep learning models that can show the highest performance in other text classification work, as they learn higher-order patterns that may appear in big datasets. Further improvements can be done by increasing the size of the dataset, using more varied data coming from a variety of social media, and multi-modal data such as audio and visual data from video content. In addition, the adoption of transformers such as BERT would fine-tune the classification, catching more subtle language and context by the model. [2]

Nicolás Benjamín Ocampo, Ekaterina Sviridova, Elena Cabrio, and Serena Villata present in their work "An In-depth Analysis of Implicit and Subtle Hate Speech Messages" a systematic study related to the detection of implicit and subtle hate speech messages, which is one of those dimensions when speaking of hate speech research. This paper tries to fill the gaps of the currently existing state-of-the-art HS detection systems that mostly aim at explicit, blatant speech and perform poorly in the case of implicit or subtle HS due to its indirect and subtle nature.

The authors extend the literature that, so far to a large extent, has focused on overt hate speech, using either simple keyword recognition or basic sentiment analysis techniques. Early work, such as that by Warner and Hirschberg (2012) and Waseem and Hovy (2016), has focused on explicit language-clear slurs or offensive terms directed toward target groups. All these methods work well on overt utterances but break when hate is conveyed implicitly, by sarcasm, metaphor, or stereotype-without offensive language as such.

Following from this lack, the authors, Ocampo et al., propose a new multilayer dataset: ISHate, which consists of messages from seven sources with explicitness, implicitness, and subtlety labels. The authors extend previous work where explicit, implicit, and subtle hate speech was defined and collected instances with properties such as sarcasm, irony, metaphor, and rhetorical questioning, which make hate intentions hard to catch. These added nuanced properties make ISHate a rich resource for exploring complex hate speech detection, hence addressing a significant gap in the existing datasets and approaches.

They further evaluate a number of state-of-the-art machine learning models' performances: BERT and DeBERTa among others. The authors find that these models, though useful in cases presenting explicit hate speech, failed with the implicit and subtle ones. Again, this brings forth the complexity that characterizes the detection of hatefulness, which relies on implicit cues and nuances in language. Further explorations by authors in deep learning models, especially in transformers, catch the required subtlety and context dependence that correctly identify nuanced hate speech.

Regarding future directions, Ocampo et al. note the scaling of ISHate and the development of more sophisticated classifiers which have been able to detect indirect forms of hate speech that often evade current systems. They further point out that the integration of contextual and extra-linguistic knowledge in the future will be important to increase rates of detection for implicit, subtle forms of hate speech-opening a path toward more vigorous hate speech moderation on digital platforms. [3]

This paper presents an investigation into the detection of hate speech in Odia using ML models. Most prior research focused on the problem of HS detection in extensively used languages, while less attention was given to Odia due to its being a low-resource language with peculiar linguistic features. The contribution of the present work is related to other existing works that have proposed different ML methods for sentiment analysis, opinion mining, and HS detection in regional languages. Based on this, the present study has created a specific HS dataset for Odia and evaluated some ML models on the same dataset.

The contribution here lies at the heart of developing an HS dataset, Odia-specific, through the collection of Odia text on social media and news platforms, classifying the text as HS and non-HS. This fills a very important gap since, compared to the more spoken languages, there are a few public resources available in Odia for language processing. Similar works have engaged in the creation of resources through annotated corpora and sentiment lexicons, which have been the driving force behind recent improvements in NLP applications for languages such as Odia.

Methodologically, the authors used machine learning techniques, namely Decision Tree, Support Vector Machine, and Random Forest. They have also discussed the performance of these classifiers based on measures such as accuracy, F1-score, precision, and recall. This is in line with other works, where they applied SVMs, Naïve Bayes, and deep learning approaches for text

classification and sentiment analysis problems in regional languages. However, here a specific HS dataset is targeted in Odia. Feature extraction through TF-IDF and n-grams extends the established strategies of the literature on HS detection by showing the effectiveness in regional language processing.

The improvements on the Odia dataset can be targeted further, more so the code-mixed text in Odia and English. It will then help solve the issues presented by multilingual nature in social media communication. Also, the work can be extended further by incorporating deep learning architectures such as transformer-based models, which show very promising performance in a number of multilingual HS detection tasks that help to improve the accuracy and generalization performance further. [4]

The paper "Hate Speech Detection in Social Networks using Machine Learning and Deep Learning Methods" reviews how both machine learning and deep learning methods have been utilized to detect hate speech over social media, mostly over Twitter. The investigation deals with an issue that is very crucial in online communication, as hate speech promotes hostility by fostering negative impacts on social cohesion. The authors look for performance comparisons between shallow models such as logistic regression and random forest with deep learning models, LSTM, BiLSTM, and CNN on hate speech detection. These comparisons are done in a manner that provides the likely benefits of deep learning when handling subtlety and context dependence of hate speech.

Their work involves developing a dataset of tweets that experts have labeled as either hate speech, offensive, or neither and further developing the dataset to make it more relevant and applicable. This practice adheres closely to previous research that emphasizes precisely how important it is for data labeling to be proper, using domain expertise in capturing subtle nuances in hate speech. Most works in the earlier times were related to simple machine learning models. They typically used some feature extraction technique, such as TF-IDF and BoW. Word2Vec and GloVe embeddings are used to extend these approaches to feed more semantic and contextualized input into the models.

Indeed, the authors made a comparative analysis where the deep learning models, especially BiLSTM, showed a superior capability in catching context and sequential dependencies of the text, beating the shallow models in hate speech detection. This further agrees with other findings, which have noted that due to the capability of the BiLSTM for bidirectional processing, it can detect hate speech more intelligibly by capturing both past and future contexts. The authors attribute this to the sensitivity of BiLSTM for temporal and contextual relationships between words, which form the most vital units necessary for the detection of hate speech in a complicated sentence.

Other potential works discussed that might serve in the future deal with more advanced architectures, such as transformers, and pre-trained models that include BERT and GPT in modeling deeper semantic structures and complex patterns of language. The authors conclude that these models are bound to further improve hate speech detection because of pretraining on large corpora of text. The paper also points out the use of explainable AI in further work, given the importance of model transparency and interpretability for ethical considerations around content moderation.

This is an important study in that it takes the foundation of hate speech detection using machine learning and really furthers the field by demonstrating, very clearly, the benefits of deep learning. They have also suggested future integration with transformer-based architectures, highlighting a need for interpretability and therefore setting out a pathway for further work on research that could have implications for the development of more robust, adaptable, and ethically responsible systems for hate speech detection. [5]

This paper, entitled "Hate Speech Detection in Twitter Using Different Models," compares different machine learning and deep learning approaches for the detection of hate speech on Twitter. The paper acknowledges the need for automated methods in times when online content is enormous, and sharing is being done at a very fast pace. The models used in this study include Support Vector Machine, Logistic Regression, Random Forest, CNN-LSTM, and Fuzzy Classification. In these models, the effectiveness and accuracy are studied. This focus on model variation ensures that the latter part sets up the efficiency comparison of a number of other approaches to detecting hate speech in real-time ecosystems like Twitter.

This study, therefore, draws its foundation from works where machine learning models have been put to work in detecting hate speech with document representation strategies such as TF-IDF and

Bag of Words for feature extraction. The authors have constructed an advanced version of this foundation in the current work, incorporating advanced models like CNN-LSTM and Fuzzy Classification. The CNN-LSTM model forms a significant advancement over prior machine learning models by incorporating both convolutional layers that capture local features and LSTM layers for capturing sequential dependencies from the context-rich content typical of social media posts. Furthermore, the Fuzzy Classification approach introduces another unique angle to this problem by trying to address the inherent ambiguity in the language that previous models may fail to consider. This technique incorporates the use of fuzzy logic; it therefore allows for nuanced classification by offering different membership degrees, which will better capture the complexity of hate speech.

In these, the most accurate model proved to be the SVM. On the other hand, some of the other well-known classifiers used in this paper are Random Forest, Logistic Regression, and CNN-LSTM. This result corroborates the findings from the previous literature, which also preferred SVM owing to its robustness in binary classification tasks, especially for high dimensional data. But the architecture of CNN-LSTM also proved substantially effective to capture both the local and global patterns and therefore showed the potential for deep learning to further improve the results of hate speech detection tasks.

The authors then indicate a future direction for deeper learning architectures, such as transformers and mechanisms of attention, which would be in a better position to deal with the intricacies of online discourse. Employing pre-trained language models like BERT or GPT-3 would further allow contextually aware detections through the inclusion of extensive prior language knowledge. Further research could integrate the development of the multimedia analysis for tweets with text but also image, video, or emoji content to further expand the model for versatile detection of hate speech across different types of content. This study, therefore, provides a foundational framework for further work in creating a much safer online environment by continuing to evolve machine learning and deep learning methodologies. [6]

The below paper, "Hate Speech Detection from Social Media Using One Dimensional CNN Coupled with Global Vector," tries to devise a methodology on detecting hate speech from social media, using a novel application of 1D Convolutional Neural Network (1D-CNN) combined with Global Vector Classifier. This research utilizes the giant leaps in the domain of deep learning to

overcome the challenges occurring during the detection of offensive language and hate speech within comments, most especially within the social media platforms like Twitter, Facebook and Instagram. The authors have shown that traditional machine learning methods and manual moderation have shortcomings; therefore, deep learning methods assure sufficient scalability and accuracy for real-time applications in social media. Thus, the peculiar structure of their model, comprising convolutional and dense layers with dropout and pooling mechanisms, outperformed predecessors in attaining remarkable accuracy over several datasets.

This is a work grounded on previous research relating to language processing and hate speech detection. The basis for the use of word embeddings and classification methods is sound. The authors specifically used paragraph-level embeddings for the detection of abusive language, whereas in earlier works, Djuric et al. used the same and Joulin et al. suggested a logistic regression model with BoW for efficient text classification. Earlier approaches using CNNs, as in those developed by Yoon Kim, showed that word vectors are promising in sentiment classification, thus preparing the ground for their application to hate speech detection, which was done by the authors in this paper. This also extends the methodological approach in leveraging character-level features, as explored in Waseem and Hovy's work, to provide this paper with a focus on fine-grained detection capabilities, taking into consideration language variations across different regions and demographics.

In this regard, the model can be further extended to more general social media content in future research, possibly by incorporating multidimensional convolutional layers that can capture complex patterns in speech and sentiments. This can be further combined with more complex transformer-based models such as BERT or GPT for an enhanced, adaptive approach toward the detection of hate speech. These extensions can also deal with a range of implicit hate speech, sarcasm, or coded language issues which the state-of-the-art models currently cannot deal with. [7]

2. RESEARCH GAP AND PROBLEM

2.1 Research gap

The ultimate goal in hate speech detection is the optimization of model performance for the best F-scores, which is defined as the harmonic mean of precision and recall. The F-score can be very challenging in this domain because a high score requires a balance between precision and recall. Precision involves correctly identifying instances of hate speech without false positives, while recall involves complete detection. In fact, this is also a kind of juggling act, as many datasets face the prevalent problem of class imbalance: the examples of non-hate speech drastically outnumber the hate speech examples. These often result in biased models towards the majority class that may fail to catch instances of hate speech.

Generally speaking, several machine learning and deep learning models have been adopted in research studies with different state-of-the-art feature extraction methods for addressing this challenge. This is typically done using an ensemble, where, instead of relying on a single algorithm, the performance improves by amalgamating strengths from a variety of different approaches. For example, typical machine learning methods, such as Random Forest or SVM, can be combined with deep learning models like LSTM, CNN, or BILSTM because these deep learning architectures are pretty good at extracting subtle patterns in text.

Methods of feature extraction are equally important in dealing with data set imbalance and improving the model's accuracy. Techniques such as TF-IDF and word embeddings-like Word2Vec, GloVe, or BERT-not only turn words into a comprehensible form but also add significance to the frequency and importance of contexts. Such representations, in turn, allow the models to grasp more logical patterns of language that denote hate speech, even when dealing with an imbalanced dataset.

It can also be approached directly by handling imbalance in the dataset, either by Over-sampling the minority class-hate speech examples, Under-sampling the majority class, or even synthetic data generation techniques such as SMOTE-Synthetic Minority Over-sampling Technique. This would

ensure that during training, the model would get a balanced dataset and could see a great improvement in F-score as it would minimize over fitting to non-hate speech instances. By using techniques such as data balancing, different models, and feature extraction, various models can be built to construct a more robust hate speech detection system. Such a design will balance precision and recall for a better overall F-score; that will lead to a more accurate and reliable detection system.

Expected method to use	**In the proposed method**	**In the selected Research paper**
Over-Sampling	✅	❌
Under-Sampling	✅	❌
SMOTE	✅	❌

Figure 1 Expected methods 1

Expected method to use	**In proposed method**	**In the selected Research paper**
Tf- Idf	✅	✅
Word2Vec	✅	✅
Bag of words	✅	✅

Figure 2 Expected methods 2

	Expected method to use	In proposed method	In the selected Research paper
Classical Machine Learning	SVM	✅	✅
	KNN	✅	✅
	Logistic Regression	✅	✅
	Naive Bayes	✅	✅
	Random Forest	✅	✅
	Decision Tree	✅	✅
	Passive Aggressive	✅	✅
Deep Learning	GRU	✅	❌
	CNN	✅	✅
	BiLSTM	✅	✅
	LSTM	✅	✅
	CNN + BiLSTM	✅	❌
	RNN	✅	❌
Transformers	GPT	✅	❌
	BERT	✅	❌

Figure 3 Expected methods 3

2.2 Research Problem

After studying several research papers, I have adopted one research paper [7] to work on further improving the performance of hate speech detection algorithms. Based on this selected paper, my objective is not only to enhance the model's accuracy but also to improve its F-score—a measure that entails the critical balance between precision and recall. While accuracy considers only a correct instance classification, the F-score is a broader view that balances recall-the model's ability to identify all instances that are hate speech-and precision-its capacity for correctly identifying hate speech out of all instances it identifies as hate speech. This balance is important in the detection of hate speech, where misclassification can imply much.

One of the main issues represented in this selected paper is the existence of an imbalanced dataset-that is, not an equal distribution of classes with examples of non-hate speech outnumbering those of hate speech. It is a formidable problem since it may result in a model being biased toward the majority class, which in this case is for non-hate speech. This bias shifts the learning of the model to a system that, while doing well overall in accuracy, may underperform in the actual identification of hate speech. Figure 1.1 in this paper illustrates visually how serious the class imbalance problem is. This is the common problem of models that are trained on imbalanced datasets; they tend to ignore the minority class, which, in this case, is crucial because the uncaught hate speech can lead to serious social consequences.

Another central problem that the research tackles is how to keep an appropriate balance between recall and precision, better known as managing false positives and false negatives. High recall will imply that the instances of hate speech are captured well, but without corresponding precision, it would end up being a high rate of false positives or everything except hate speech being labeled as hate speech. On the other hand, if the precision is too high in recall, then it generates false negatives, or the actual hate speech goes undetected. This does form a sort of delicate balance so intrinsic in developing a hate speech detection model that is not only accurate but reliable for a wide range of real-world applications.

These require strategic solutions, such as techniques to balance the dataset, tuning model parameters that can return optimized recall and precision, and advanced algorithms with adaptive

class imbalance handling. This improvement will lead me to devise a more robust and efficient hate speech detection model, which would return a high F-score, so that both the precision and recall of the process of detection may be increased further toward ensuring safer online and social media environments.

3. RESEARCH OBJECTIVES

3.1 Main Objectives

The core purpose of hate speech detection models is to autonomously identify and classify language that is harmful, derogatory, abusive, or otherwise offensive toward individuals or groups. These advanced systems are designed to analyze spoken or written input and flag any instances of hate speech that contravene legal standards, community guidelines, or terms of service agreements. By effectively detecting such content, these models serve as an essential tool in moderating online platforms, fostering an environment that discourages hostility and promotes inclusivity.

Hate speech detection algorithms typically focus on identifying specific linguistic patterns, phrases, or coded language that are often used to spread harmful ideologies or target certain groups based on race, ethnicity, religion, gender, sexual orientation, and other protected characteristics. This task is complex, as hate speech can be subtle, context-dependent, and may employ euphemisms or symbolic language to mask its intent. Therefore, these models are continuously trained on large, diverse datasets that reflect various forms of hate speech, enabling them to recognize evolving language and nuances more accurately.

Beyond flagging content for removal, hate speech detection models are integral in shaping healthier, more civil digital spaces. By automatically responding to or moderating harmful content, these tools help reduce the burden on human moderators, who often face emotional stress from exposure to offensive material. Moreover, automated detection assists social media platforms and online communities in enforcing policies that aim to maintain respectful interactions, thus contributing to a safer experience for all users.

Additionally, these detection systems align with societal and organizational efforts to combat discrimination, helping reduce the spread of harmful misinformation and prejudice online. By accurately identifying hate speech, these models support initiatives to hold users accountable for harmful behavior, encourage self-moderation, and promote empathy and tolerance in online interactions. The ultimate aim of hate speech detection models, therefore, extends beyond simple moderation—they play a pivotal role in shaping the future of online communities by establishing standards of respectful engagement and deterring behavior that undermines social harmony.

3.2 Sub Objective

- Finding a Suitable Dataset

Finding an appropriate dataset is very important as this forms the foundation upon which the hate speech model development would lie. Ideally, the suitable dataset would contain representative samples of hate speech or non-hate speech that best represent the sites where such incidents of hate speech are common-for example, social media or online forums. Datasets for research are mainly taken from Twitter, Reddit, or any other public hate speech datasets curated for academic purposes. Ideally, a dataset would be labeled with various types of hate speech and associated non-offensive content, which would help in nuanced model training. The datasets can also contain metadata useful for further analysis regarding the origin, context, or frequency of the hate speech content.

- Data Pre-Processing

Pre-processing is one of the major stages for raw text data preparation in machine learning. Since the raw form of text data contains inconsistencies, irrelevant information, and is in several different formats, pre-processing standardizes and cleans the data to enhance the accuracy of pattern detection by the model. Basic activities conducted during the steps of pre-processing include the following:

Tokenization: Breaking down text into words or tokens.

Lowercasing: All text is converted into lowercase to uniformly understand the meaning carried out by words.

Punctuation and Special Characters Removal: The removal of elements that do not add meaning to the text content.

Stop Words Removal: Emphasizing the common, unnecessary words like "and" or "the" in order to focus on meaningful information of the content.

Stemming and Lemmatization: Curing the words to their root to gather similar meanings, such as "running" as "run."

Emoticons, URLs, and Hashtags Handling: Handling such specific entities of social media requires special attention in order to keep the meaning of the context intact.

Preprocessing cleans the data by removing noise; this allows the model to pay more attention to the semantic content of the text rather than to variations that are irrelevant in many cases.

- Handling Imbalanced Dataset

The hate speech datasets are highly imbalanced as there are many more instances of non-hate speech compared to hate speech. This dataset imbalance can easily result in biased performance of the model toward the majority class, thus reducing the ability of the model to correctly identify hate speech. The strategies that can be followed to manage this include:

Oversampling: This can be done by duplicating the instances of hate speech in a dataset to a point where the classes will be balanced. Under-sampling: This is done by reducing the number of examples of non-hate speech to create a balanced dataset. The generation of synthetic data; for instance, using the SMOTE technique, which stands for Synthetic Minority Over-sampling Technique, develops new artificial instances of hate speech from the actual data. Class weight adjustment; this adjusts certain parameters of the algorithm to give it a higher priority in the detection of minority classes, in this case, the hate speeches.

The class imbalance issue must be taken care of, so that the model is fair in detecting hate speech without getting biased by the majority class.

- Converting Text into Numerical Representations

Text data needs to be transformed into numerical formats to make the data consumable for machine learning models. Some of the techniques of text-to-numerical transformation include:

Bag-of-Words: One of the methods of transforming text is by representing it based on the presence or frequency of a word within a pre-defined set of vocabulary.

TF-IDF: The weighting of words concerning their importance in the documents to emphasize unique words.

Word Embeddings: The use of pre-trained embeddings includes Word2Vec and GloVe; advanced pre-trained contextual embeddings include BERT. Both capture semantic and syntactic

relationships between words, allowing performance enhancement in a model by providing capabilities for word relationship understanding and context within sentences.

It follows then that the selection of an appropriate technique for representation will improve the interpretability capability of the model in understanding complex language patterns associated with hate speech.

- Model Training and Hate Speech Classification

Following the pre-processing, data conversion into a numerical format precedes the training of machine learning and deep learning models to classify hate speech. This includes:

Model Selection: Choose appropriate algorithms like Support Vector Machine, Random Forest, Naive Bayes for traditional machine learning, and LSTM, Bi-LSTM, CNN for deep learning models.

Training: Feed the balanced and numerically represented dataset into these selected models and optimize the parameters in a way that it increases the performance.

Cross-Validation: Techniques like k-fold cross-validation will be applied to assess the performance of the model on different subsets of the data. This will ensure the model generalizes well beyond the training data.

Hyper parameter Tuning: The process of adjusting the model parameters to obtain an optimum accuracy, precision, recall, and F-score.

A correctly trained model learns how to identify linguistic patterns and context that identify abusive or offensive content as hate speech.

- Model Evaluation

Evaluation will help in realizing the performance of the developed hate speech detection models through the following metrics, among others:

Accuracy: It shows the general correctness of the classifications; however, it might be misleading for imbalanced datasets.

It does focus on the precision of hate speech predictions, which is very important in order to reduce false positives.

It measures recall, which is about the model's ability to identify all actual hate speech-this is an important factor so that the false negatives will be at a minimum.

It balances the precision and recall of the model; especially more valuable in hate speech detection, as false positives and false negatives are each impactful in their ways.

Comparing these metrics across these various models provides insight into the strengths and weaknesses of each model, enabling further refinements toward optimal detection of hate speech.

4. METHODOLOGY

4.1 System Architecture Diagram

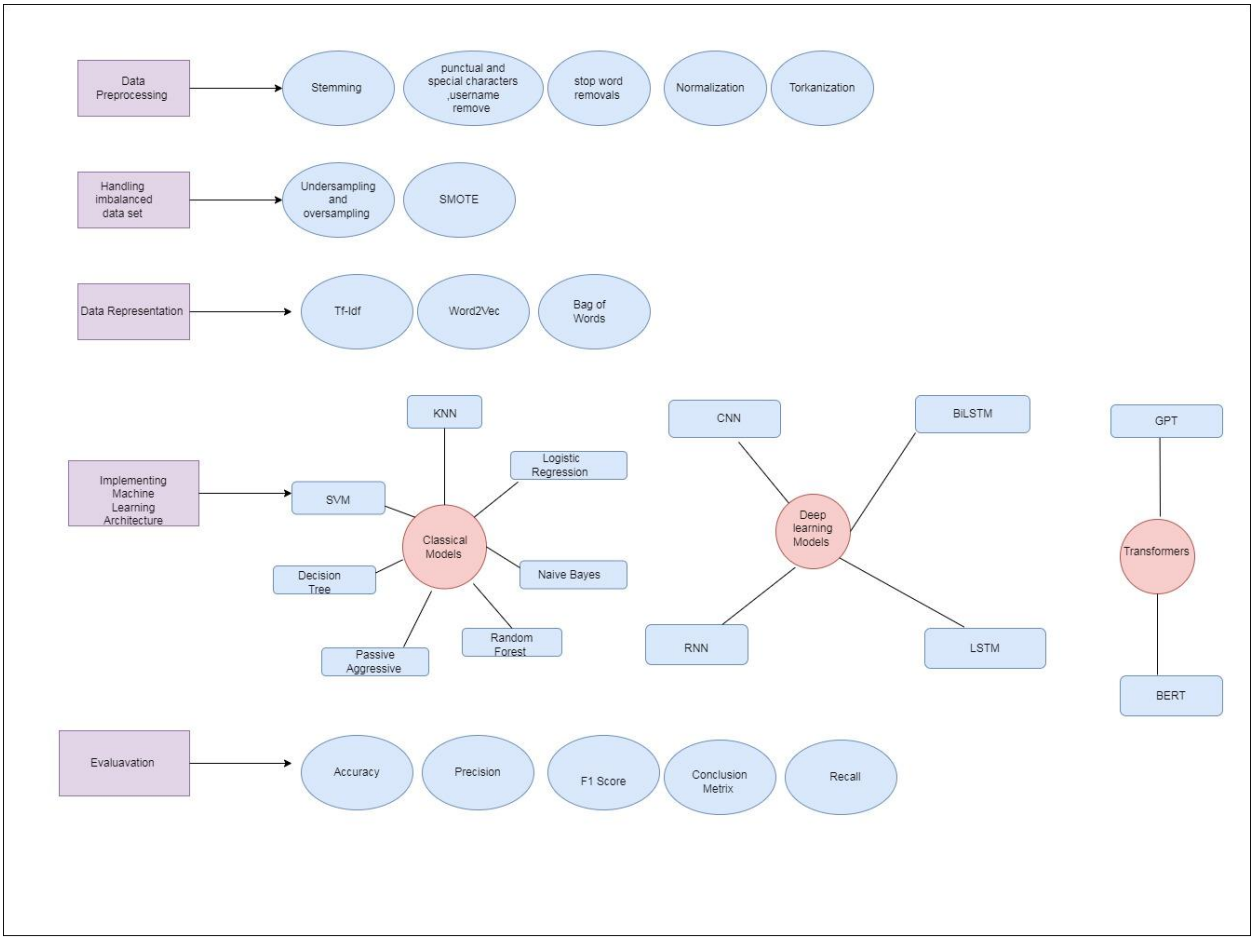

Figure 4 System Architecture Diagram

Using this application we can hate speech and offensive language written in documents. It will highlight the hate and offensive language after getting the dataset from Kaggle. It has been going through the data cleaning process. Since the dataset is unbalanced, balance the data set using oversampling and under-sampling.

To train the data set it has used classical machine learning model as well as deep learning model. The classical matching learning model used a Support Vector Machine (SVM), Random Forest, Naive Bayes, passive-aggressive, Logistic Regression, Decision Tree, and KNN. As a Deep learning model, it used RNN, LSTM, BILSTM, and CNN. For the word representation Bag of

Words and Tf-idf are used in the classical machine learning model. Karas embedding layer, pre-trained Word2vac model, and customized Word2vec model used as the word representation method in deep learning machine learning model.

So all together I have more than 36 models. After considering all those models, one model is chosen as a model to deploy. When choosing the model, it has considered the model that has the highest accuracy and as well as F1-Score.

After choosing the model. It is deploying the model using Flask which is the lightweight web framework in Python. The model will be served through an API as well as the interactive web interface. To give the most user-friendly environment it has been developed using HTML and CSS.

4.2 Data Set and Preprocessing

4.2.1 Data set

I get my data set through Kaggle. This data set has more than 29000 sentences. The data set has been collected using Twitter. This data set is classified into three categories. There is hate speech, offensive language, and neither. Each sentence in the data set has been coded by the Crowd Flower. Three minimum annotations are used in the dataset. Maximum is different. The sentence is named after the most annotated category. This data set is an unbalanced dataset witch will affect to the model accuracy directly

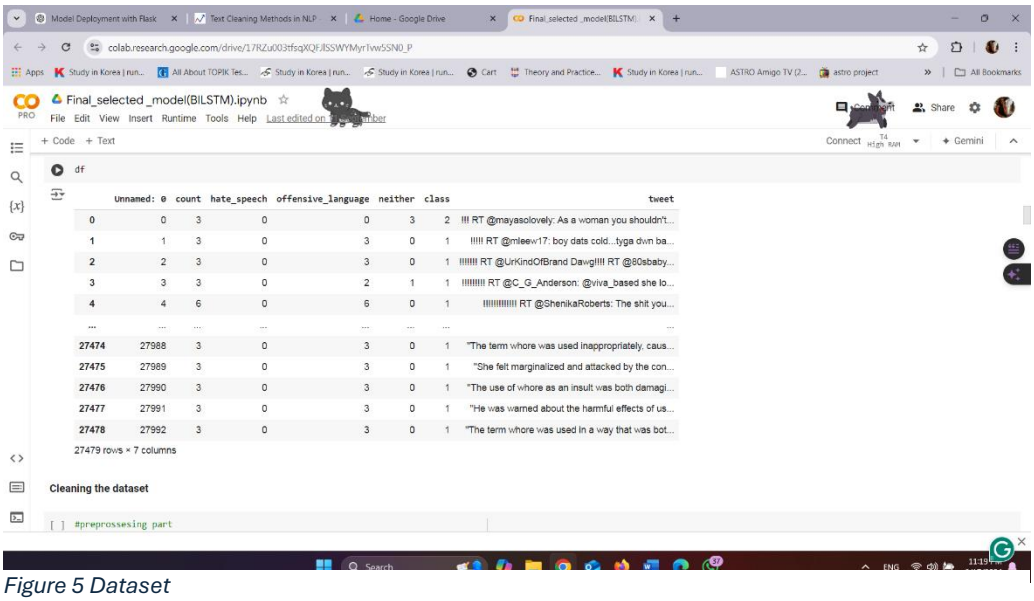

Figure 5 Dataset

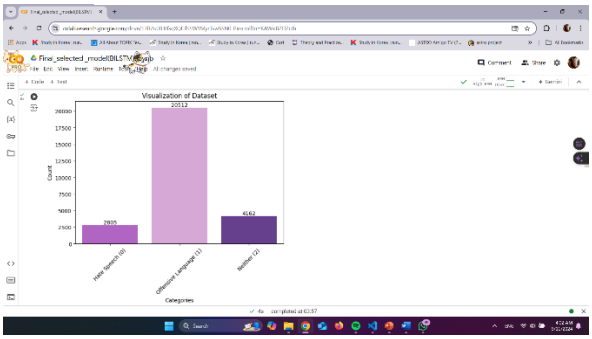

Figure 6 Dataset 1

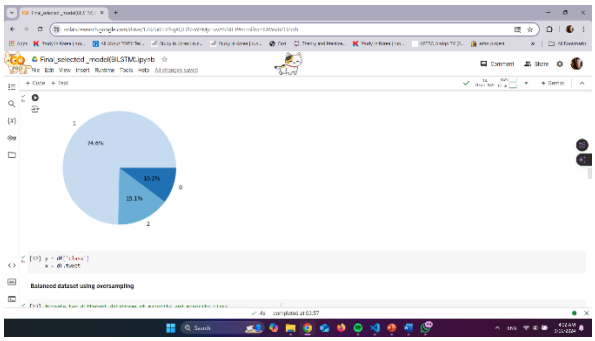

Figure 7 Dataset 2

4.2.2 Preprocessing

The raw textual data can't be used for the textual classification model. We have to preprocess the data before going through the model training. After going through these processes the dataset ensures that it is in optimal format for input into the text classification model. These steps are essential for reducing noise in the dataset. There are several steps to the data cleaning process.

Text cleaning

Normally text data contains unnecessary characters and formatting inconsistencies that hinder the models' ability to learn effectively. Some examples are lowercasing. Punctual removal, special character removal, and white space normalization. In this case, since the data has been extracted from Twitter it has to remove the username as well which tag alone with @ mark.

- Lowercasing – since this detects hate speech there is no need for capitalization. So all text can be converted into lowercase to prevent the model from treating capitalized and non-capitalized versions of the same word as different entities. So in case “DATA”, “data”, “DaTa”, and “Data” are considered the same entity as “data”. This reduces the feature sparsity and helps in model generalization.

- Punctual Removal –here it removes punctuation marks such as periods, commas, semicolons, and other symbols. These characters do not carry semantic meaning when hate speech detection is performed. So it does not need to be treated “breasted” and “breasted!” differently. This will help to reduce noise in the feature space.

- Special character removal –here it is considered Non-alphanumeric characters such as numbers emojis and symbols. Since this data is from Twitter, have to remove usernames which is easily identified since it is tagged along with @ mark.

Stop words removal

Stop words meaning the common words that are used in sentences. Such as “a”, ”and”, “the”, “is” etc. These words typically do not carry a special meaning in the text. There is a pre-defined list of stop words to use to filter out those terms from a text. This helps to reduce the dimensionality of the input data and focus on more informative words.

Stemming and lemmatization

To simplify the text we use the basic format of the word. To reduce the data to its base forms it is used to remove suffixes from words. An example is to format “running” into “run” This is called stemming. Sometimes it is used root form of the word. Normalized words into their root form is called lemmatization. For example using “good” instead of “better”.

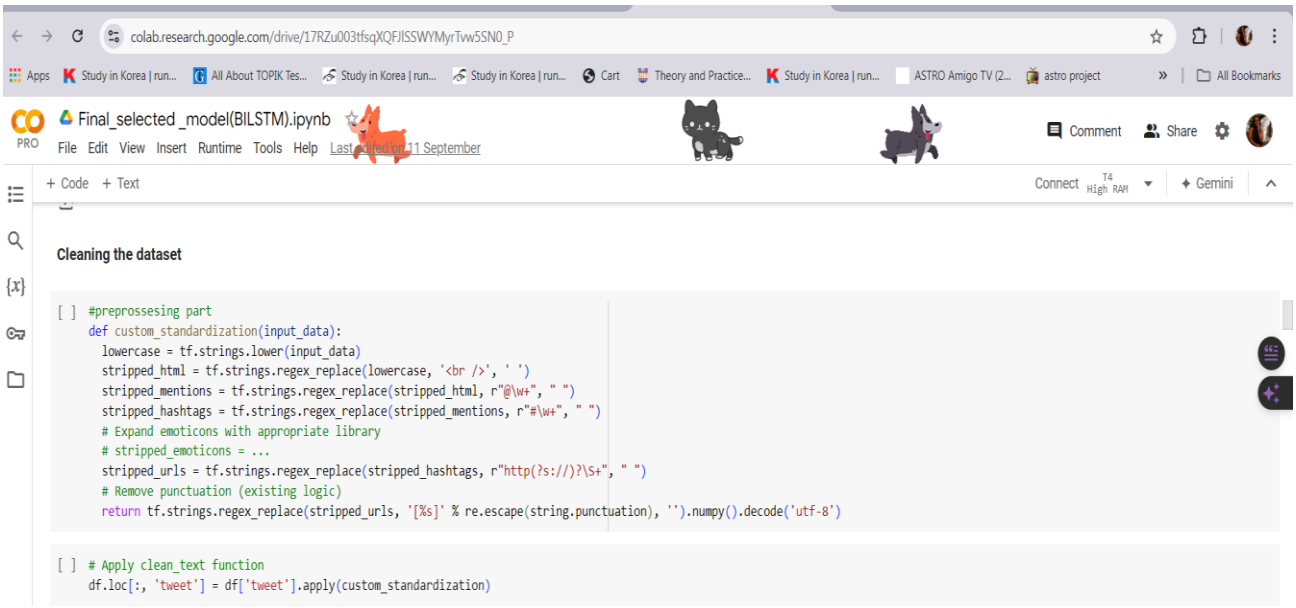

Figure 8 Stemming & lemmatization

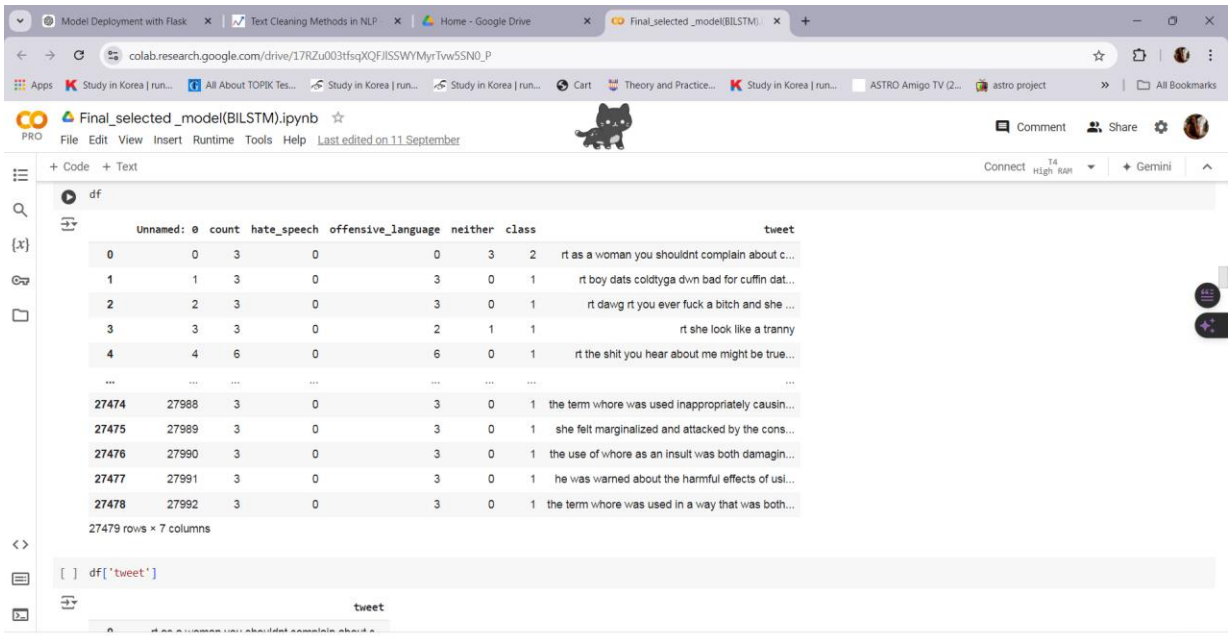

Figure 9 Stemming & lemmatization 1

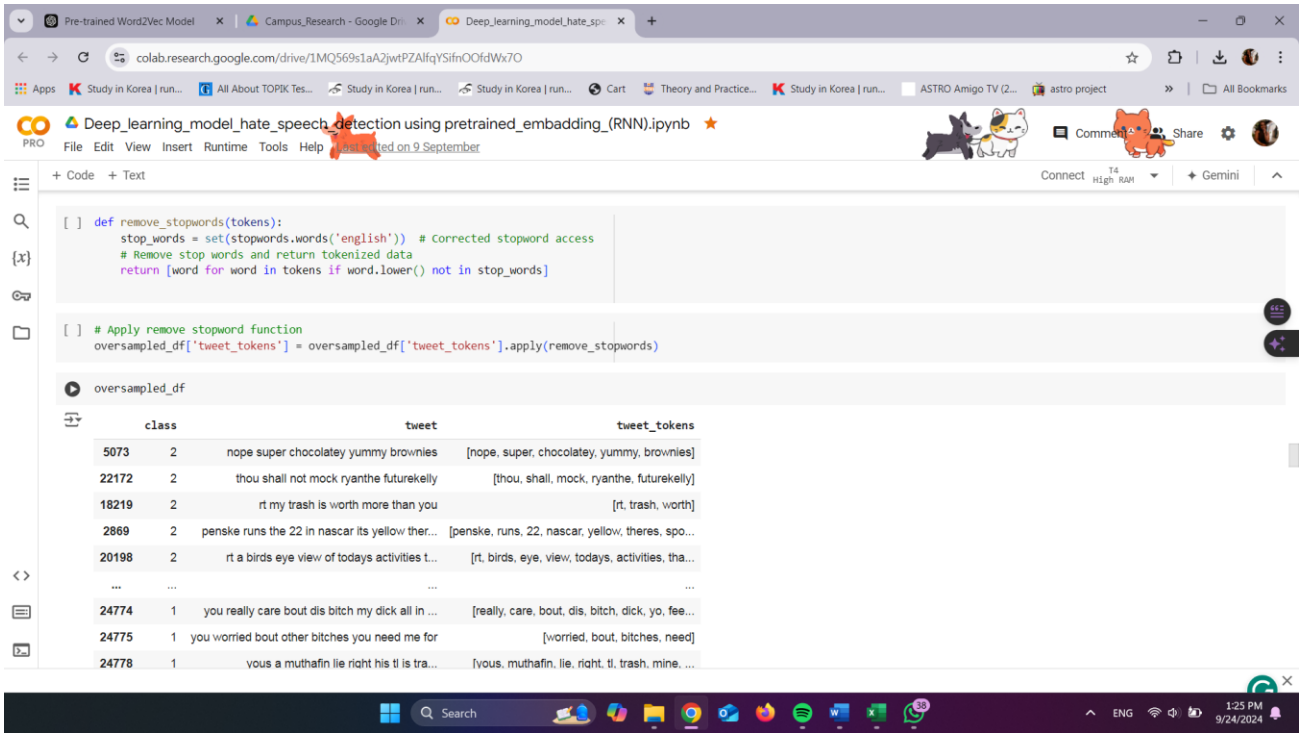

Figure 10 Stemming & lemmatization2

4.3 Data Balancing

This method helps to address the issue regarding an imbalanced dataset. This occurs when the number of instances in different classes varies significantly. This will be a clear problem when addressing the accuracy of the model. According to how data varies there is a majority class (the class that has majority instances), and one is a minority class (the class that has fewer instances) Because of this class difference the model will be biased towards the majority class and perform poorly towards minority class with effect to the model accuracy. To overcome this, various balancing methods are used in model training.

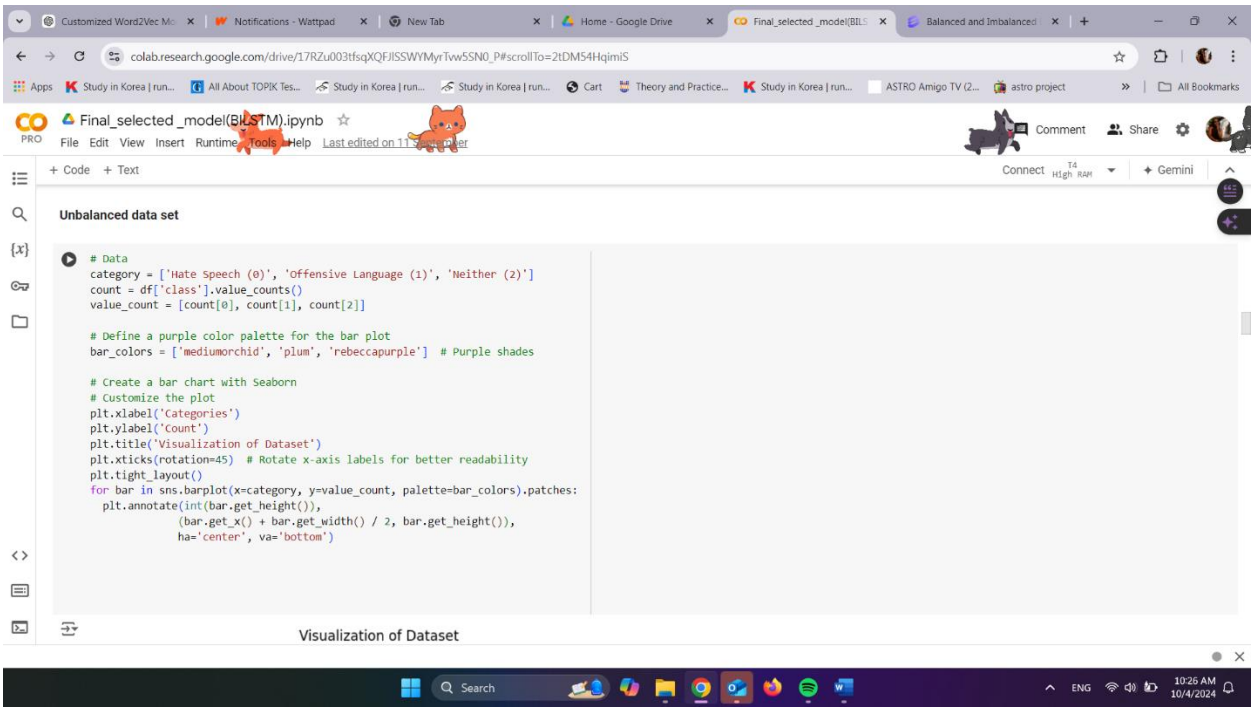

Figure 11 Unbalanced Dataset

Figure 13 Visualization of Dataset 1

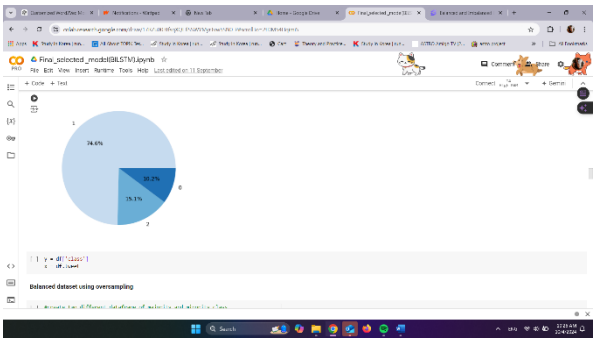

Figure 12 Visualization of Dataset 2

4.3.1 Over-sampling

The imbalanced class can be handled using the over-sampling technique. This is essential to prevent the model from drifting toward the majority class and increasing its power in identifying the examples from the minority class. Over-sampling samples in the minority class balances out the distributions in the classes. Various over-sampling techniques are used, such as Random Over-sampling and Synthetic Minority over-sampling techniques. Random Oversampling which is used in here simply replicates instances from the minority class. Random sampling was utilized to create a representative subset of the dataset, ensuring that the selected data adequately

reflects the distribution of the overall population. In Random Sampling, each data point in the dataset has an equal probability of being selected, minimizing sampling bias

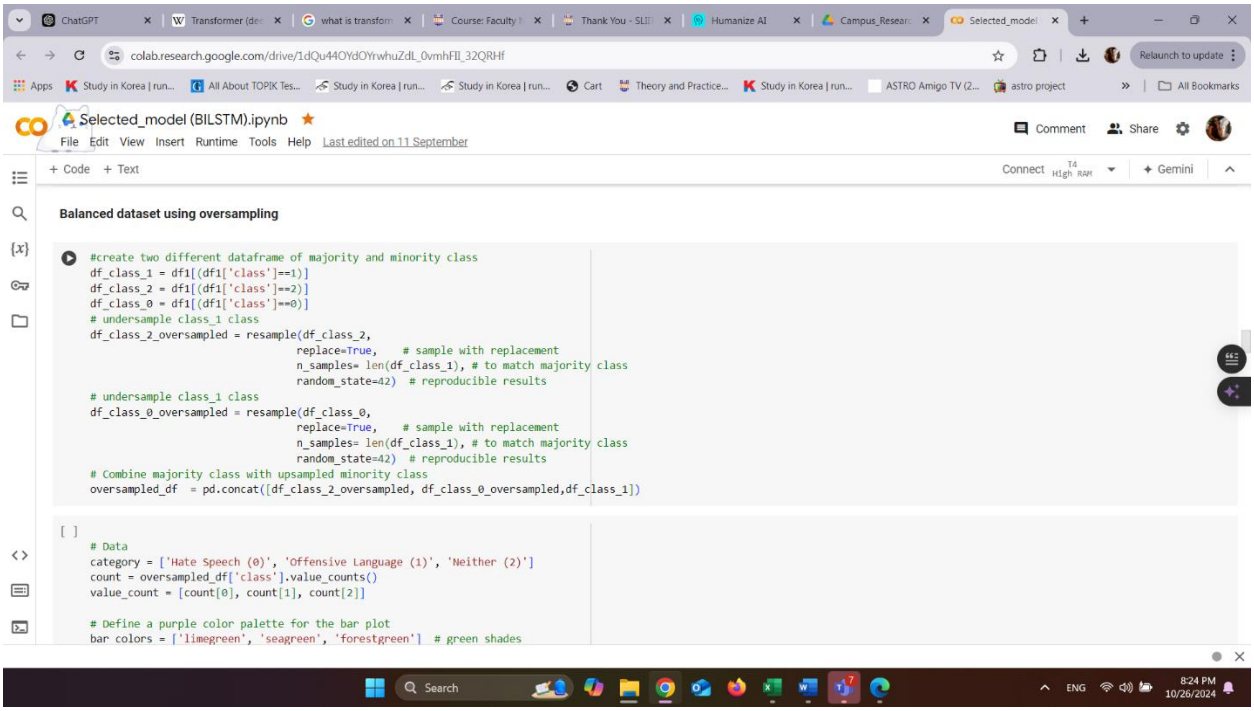

Figure 14 Balanced dataset using over-sampling

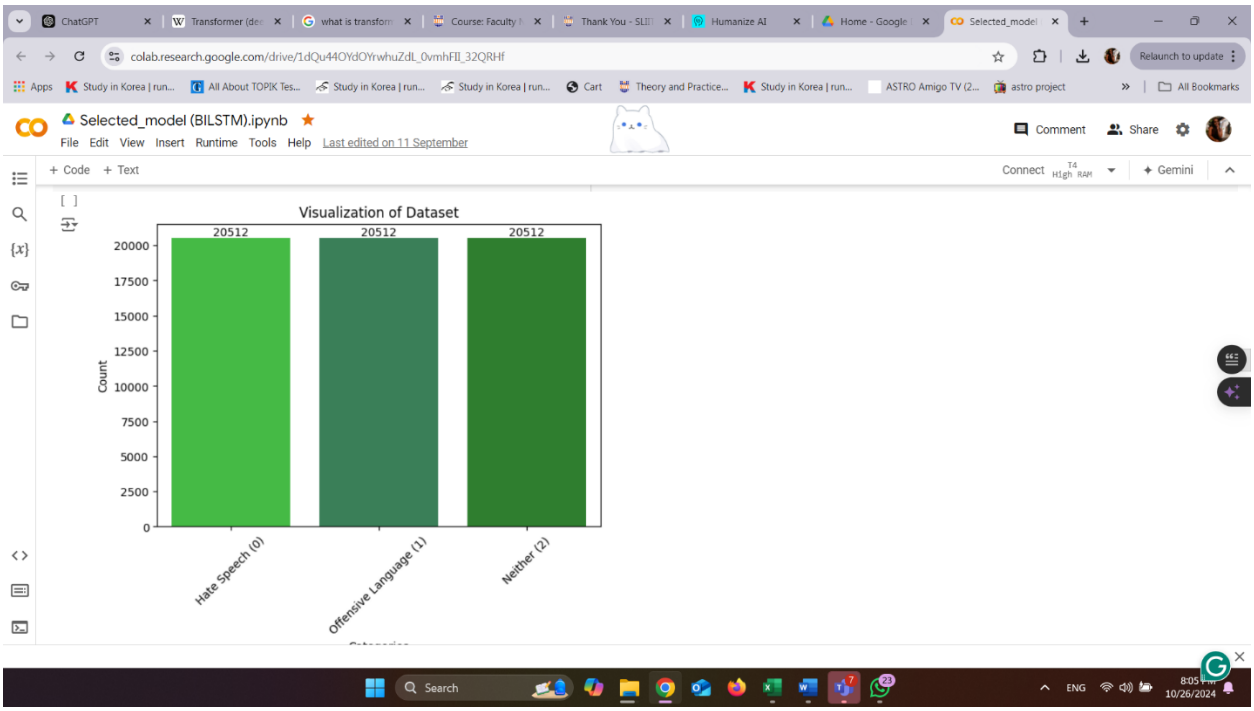

Figure 15 Visualization of dataset

4.3.2 Under-sampling

To handle the problem of class imbalance in the dataset, under-sampling technique are used.

Under-sampling involves the removal of random data points within the majority class to make

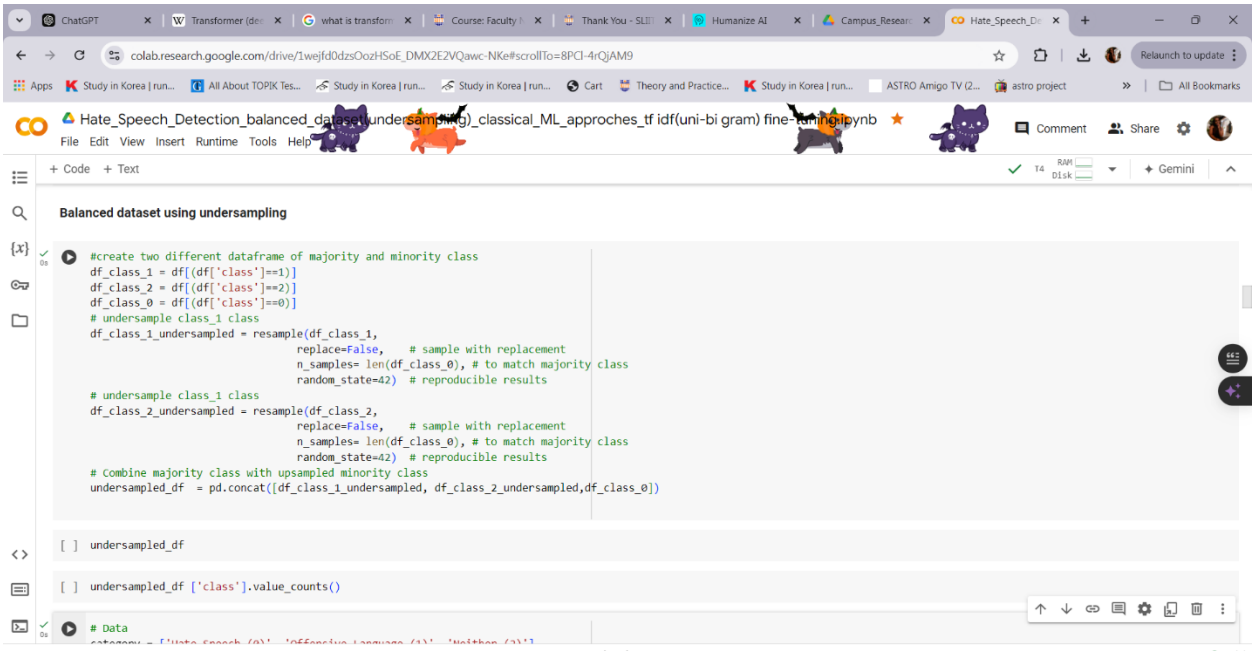

Figure 16 Balanced dataset using under-sampling

the classes balanced. It hence reduces dominancy by the majority class so that the model can work on the minority class, which often is the target of classification, more effectively. Although this may reduce some bias toward other classes, thus improving the performance of the model on the minority classes, it reduces the size of the dataset in general. Information might get lost this way, and the model may not generalize across all data points as well. Careful consideration of the ratio and of the total size of the dataset helps to make this tradeoff balanced, yielding a balanced yet informative dataset for training.

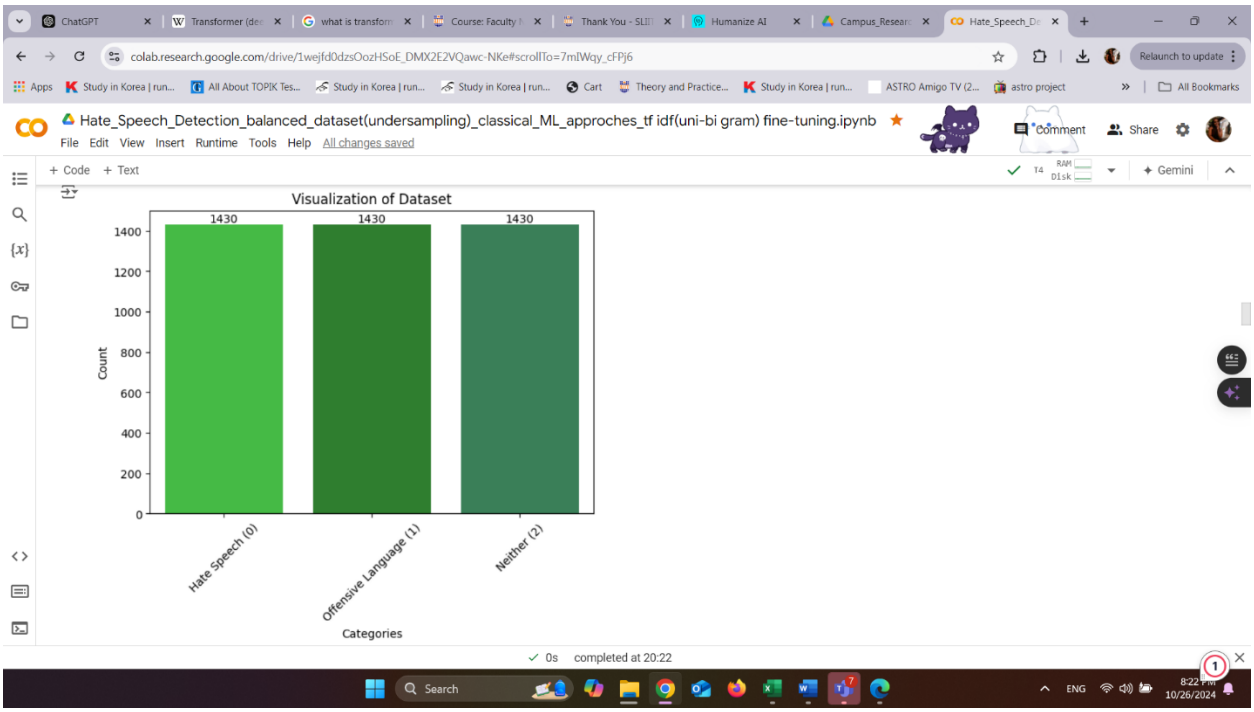

Figure 17 Visualization of dataset

4.4 Word Representation

This is called word embedding as well. This is used for the representation of words in text analysis. The words are represented in the form of real-valued vectors. The computer cannot understand text data in row format. To make them understand better, text data can be broken

down into a numeric format. This helps computers to understand text-based content better. In this process, words and documents are represented in the form of numeric vectors. The main importance is these vectors allow similar words to have similar vector representations. This helps the machine learning model preserve the semantic and syntactic information. There are various methods. Bag of Words, TF-IDF, and Word2ec model are some of them.

4.4.1 Bag of Words

This is a trendy word embedding technique used in machine learning models. This is a fundamental approach in the machine learning model. This is the simplest form of text representation in numbers. This model considers the word occurrence disregarding their order. Each text is represented as a vector of word counts where each dimension corresponds to a unique word in the corpus. So there is a limitation in capturing word importance and context. So there are some drawbacks. Such as high dimensionality and the inability to capture word order or semantic relationships between words

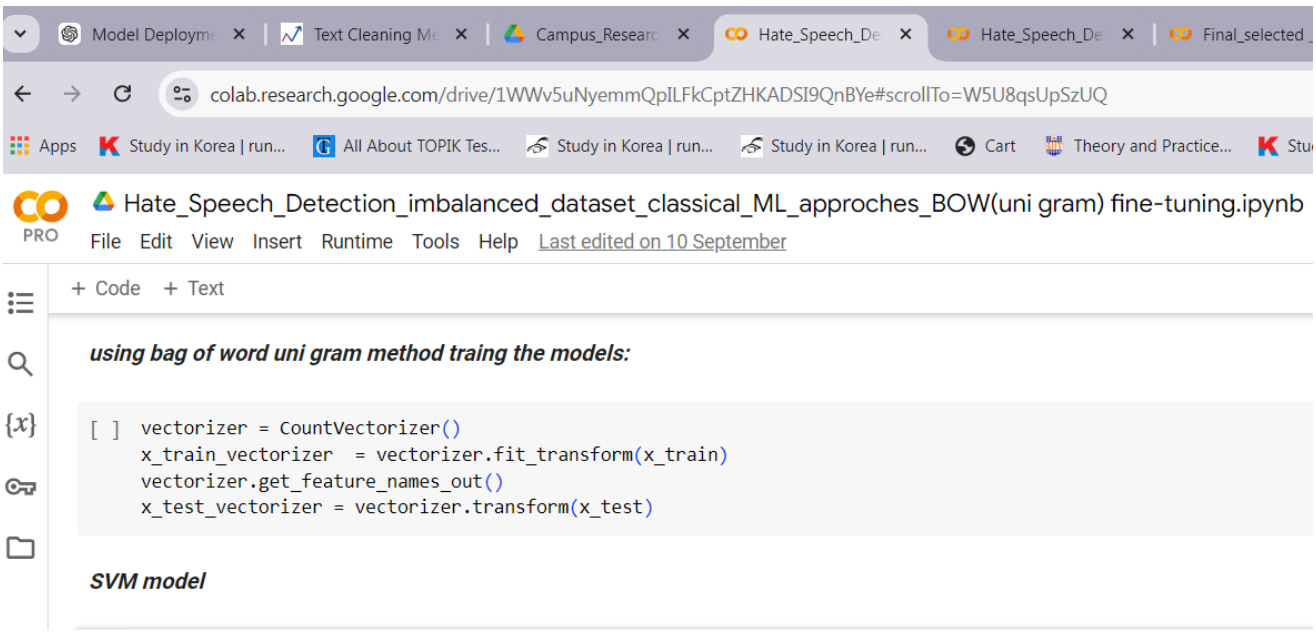

Figure 18 Bag of word uni gram method

4.4.2 Term frequency-inverse document frequency (TF-IDF)

This can be considered as a developed version of Bag of Words. TF- IDF highlighted the importance of words in the document. This balances word frequency with relevance. This gives a weightage to each word separately which helps to identify the importance of each word. This helps to focus on words that are important for the specific documents but not common in the documentation. This reduces the noise that happens due to frequent but uninformative words. I

- Term Frequency (TF): this measures how frequently a term occurs in a document. The number of items a word appears in a document divided by the total number of words in the document
- Inverse Document Frequency (IDF): this measures how the word is important to the document. so this helps to weigh down the frequent terms while scaling up the rare ones

Mathematically for word *i* in document *j* the TF-IDF score is

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

• TF (i,j) is the term frequency of word i in document j

• N is the total number of documents,

• DF (i) is the number of documents containing the word i.

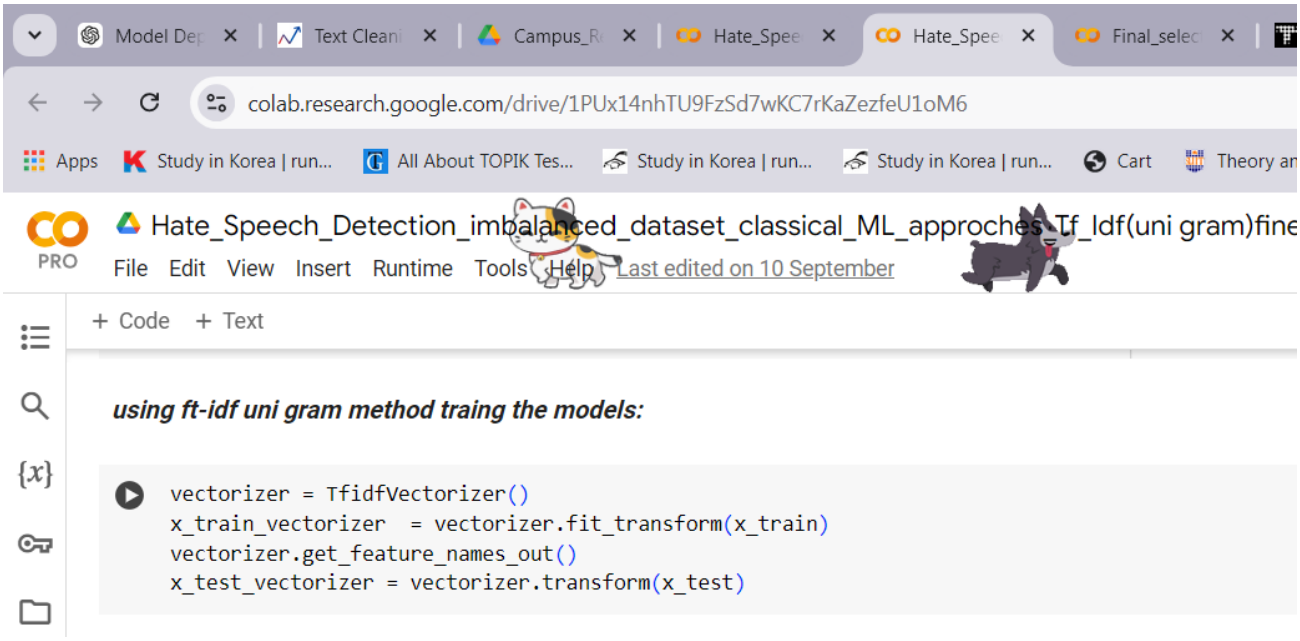

Figure 19 Ft-idf uni gram method

4.4.3 Keres Embedding Layer

This is a core component for handling word embedding in neural network models. This provides a simple and efficient way to transform words into dense vectors of fixed size, enabling the model to process textual data in a way that preserves semantic information. This converts positive integers into dense vectors of fixed size. Input to this layer is a sequence of integers each integer represents a vocabulary in the corpus. The layer is initialized with random weights by default and as the part of training process the model learns to adjust these weights the output is a dense vector representation of the word, with each word being mapped to a continuous vector in an embedding space. The vectors are updated during training to move similar mining words closer to each other in vector space

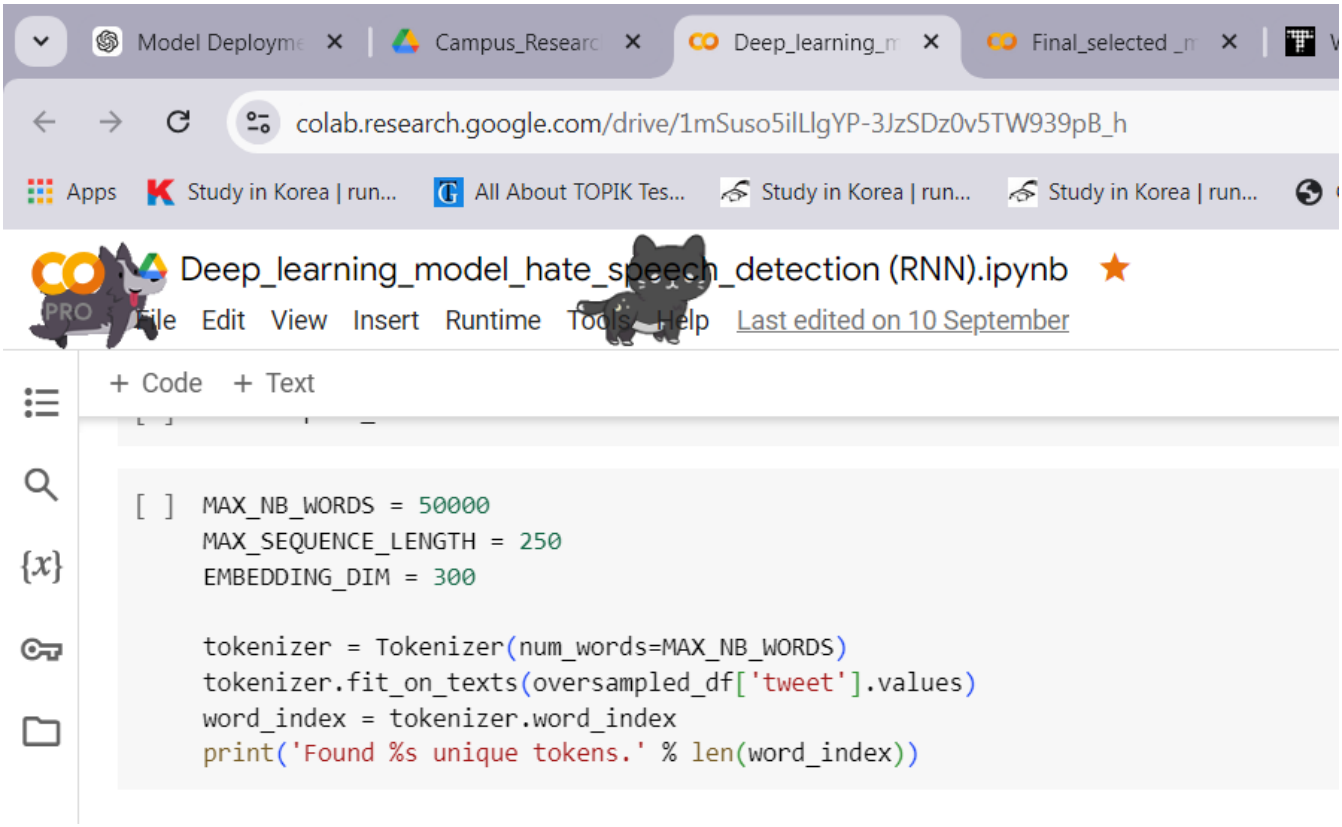

Figure 20 Keres embedding layer

4.4.4 Word2Vec model

This is also used to transform words into a dense vector representation capturing semantic relationships. This can learn a distributed representation that preserves linguistic context. The words are represented in vector space where words that occur in similar contexts are placed closer together in the space. The model outputs word embedding that encodes both syntactic and semantic relationships between words allowing for meaningful word representation This word2vec model uses two primary architectures.

Continuous Bag of Word (CBOW)

In this architecture, the model predicts the target word based on the words in the surrounding context. The methodology that it uses is a sliding window across the sentences, where context words are fed into the model to predict the middle word. This performs the best performance with smaller datasets. This is computationally efficient.

Skip – Grams

This is the opposite of CBOW. It predicts surrounding context words based on a target word. This is best when having a larger dataset. This is more effective at learning rare word representations. This is more expensive than CBOW when comes too computationally. There are two methods one uses positive input samples and the other one uses negative input samples.

When comes to the word2vec model we can use pre-trained models in our code. As well as we can customize our own word2vec model using our dataset

4.4.4.1 Pre-Trained word2vec model

This captures the semantic relationship between words, enabling the model to leverage prior knowledge learned from vast text corpora. Pre-trained word2vec model embedding from Google trained on Google news dataset. This has been trained on a dataset that has approximately 100 billion words. It contains 3 million word vectors as well. This helps to enhance the text classification task. Since this model is already trained the computational overhead involved in training embedding from scratch was avoided. So it allows for faster model training reducing training time. This has used a massive corpus of diverse text which allows for improved generalization for unseen data and better context capture. This helps to improve overall performance in the component

All tweet data is tokenized and vectorized using a pre-trained word2vec model. And then word vectors are passed as input features to a machine learning model to do the prediction, whether the given sentence is hateful or not.

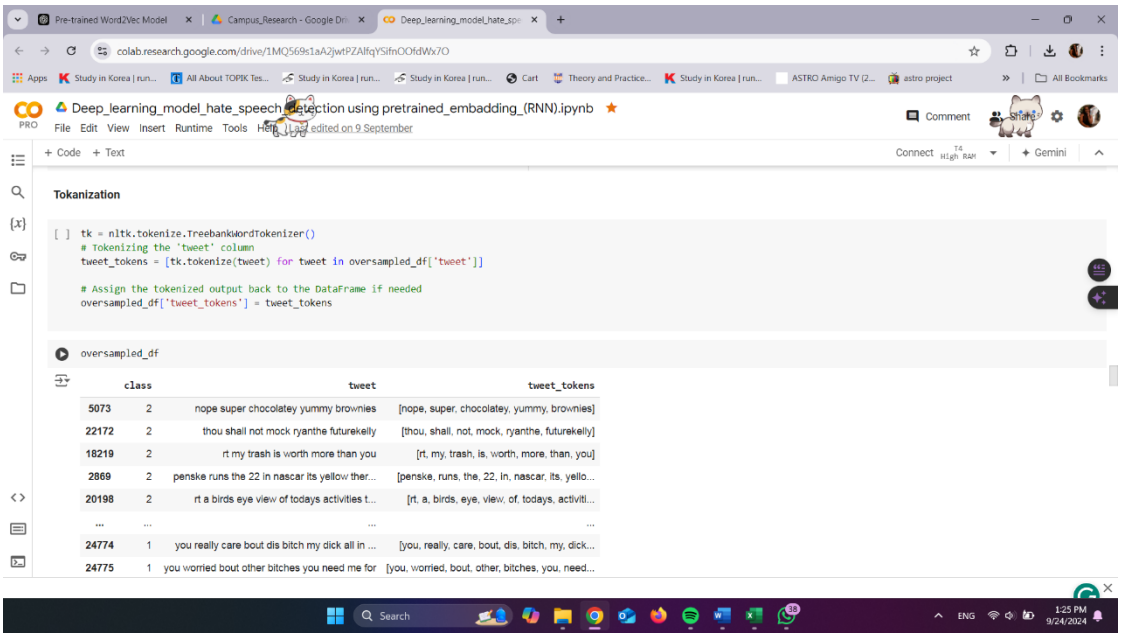

Figure 21 Tokenization

57570 rows × 2 columns

```
[ ]  model = gensim.models.KeyedVectors.load_word2vec_format('/content/drive/My Drive/Campus_Research/GoogleNews-vectors-negative300.bin.gz', binary=True)
```

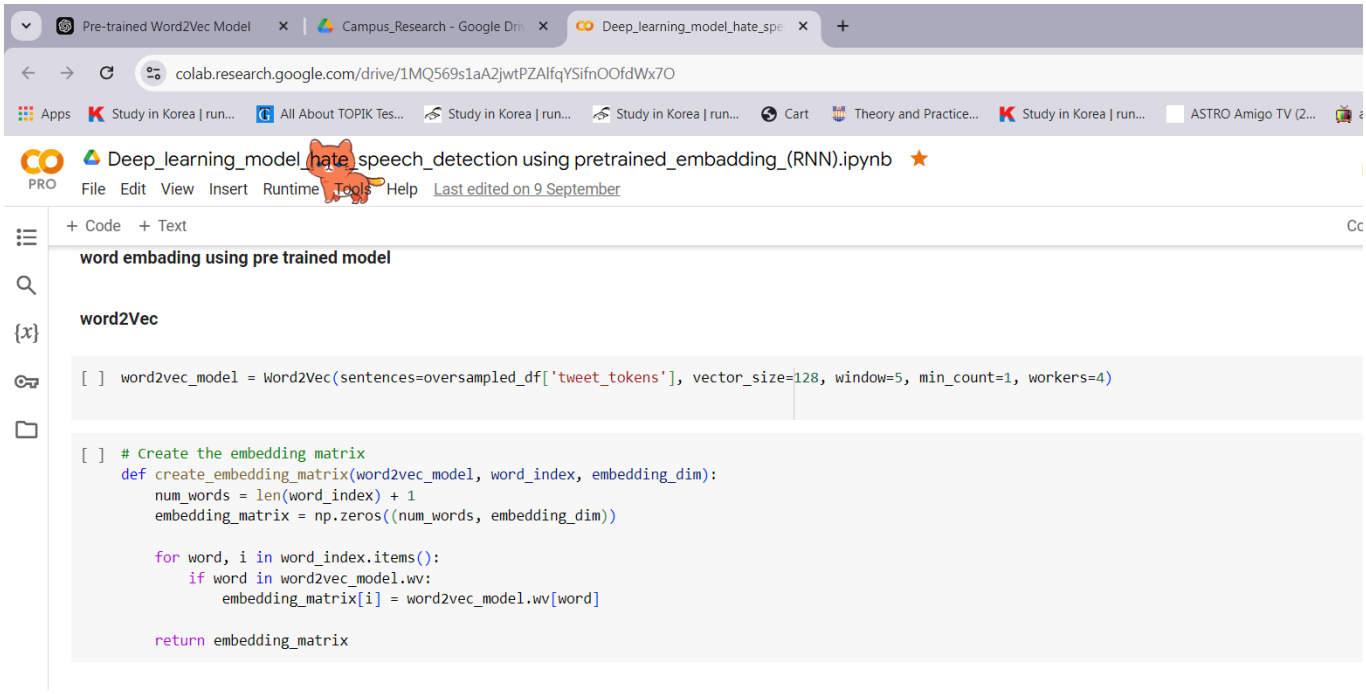

Figure 22 Word2Vec

4.4.4.2 Customized word2vec model

In addition to leveraging the pre-trained Word2vec model, a customized word2vec model was developed specifically for this research using the dataset used in the study. This was specifically trained for a specific task in that it targets hate speech using it own dataset. This will help to better capture. The specific Skip-gram negative sampling approach is used to train the customized word2vec model. This also transforms words into continuous vector representations capturing the semantic and syntactic relationship based on it own context. Since it is trained in a corpus that is relevant to a particular field it ensures that the embedding captures domain-specific words. So this improves the performance of the model training process.

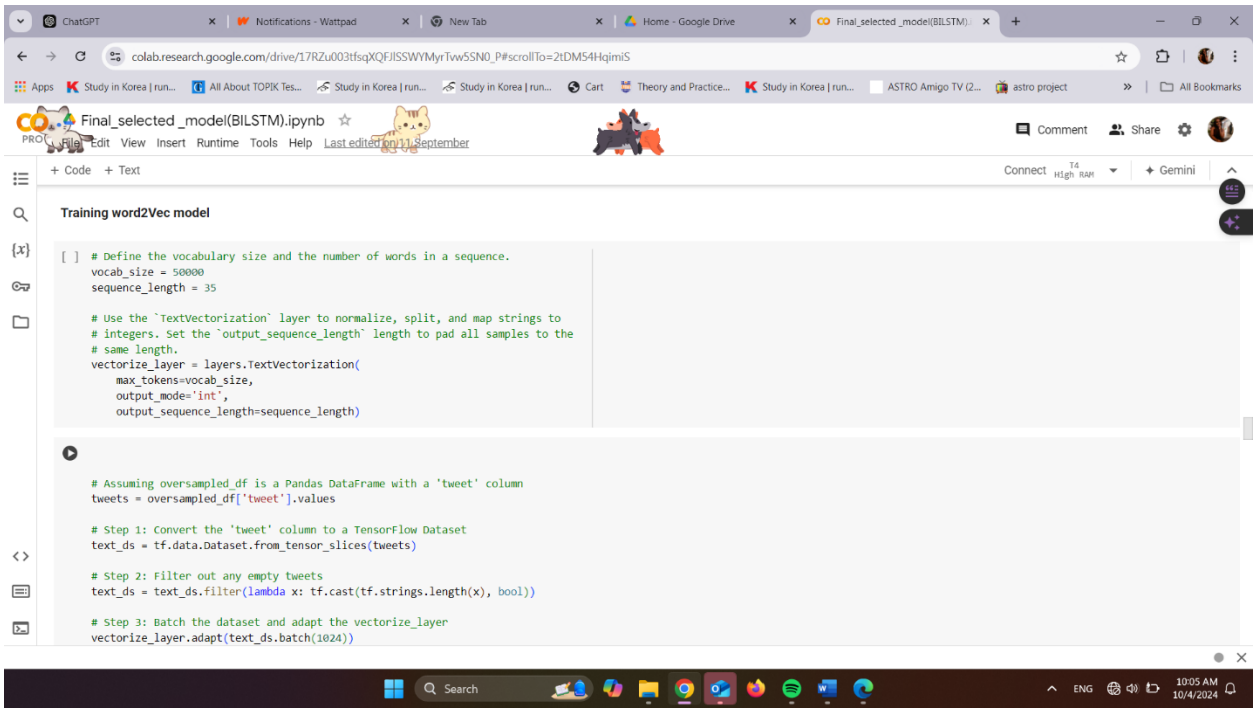

Figure 23 Training word2vec model

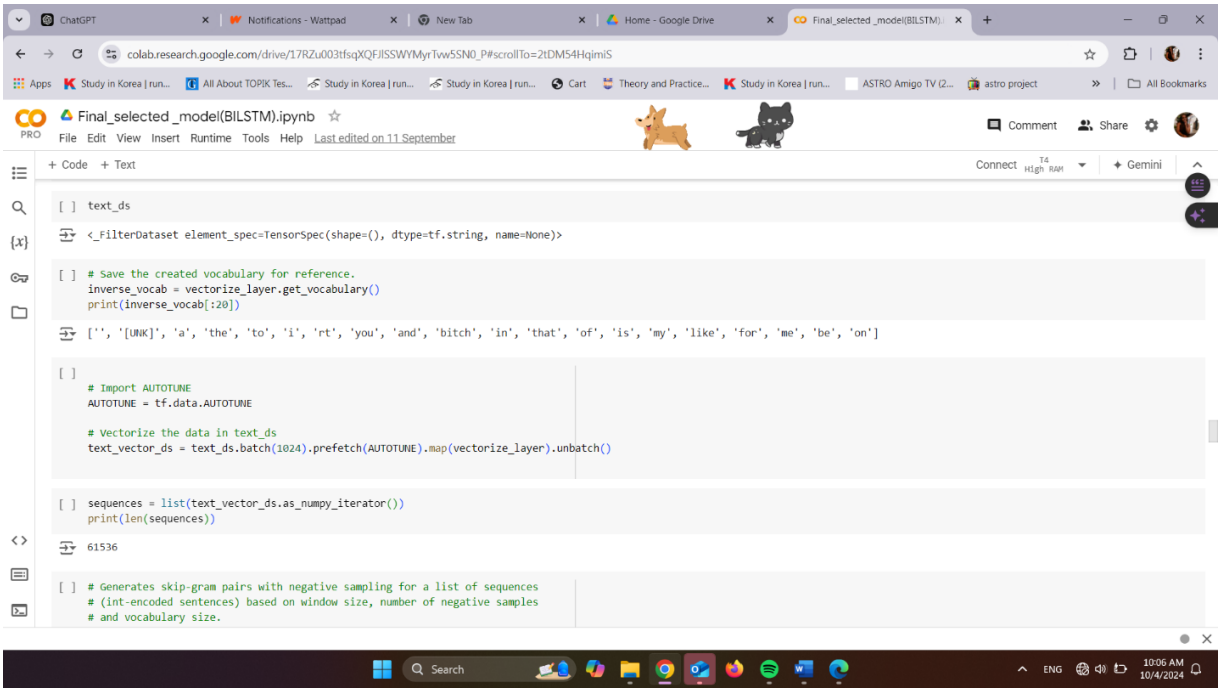

Figure 24 Training word2vec model 1

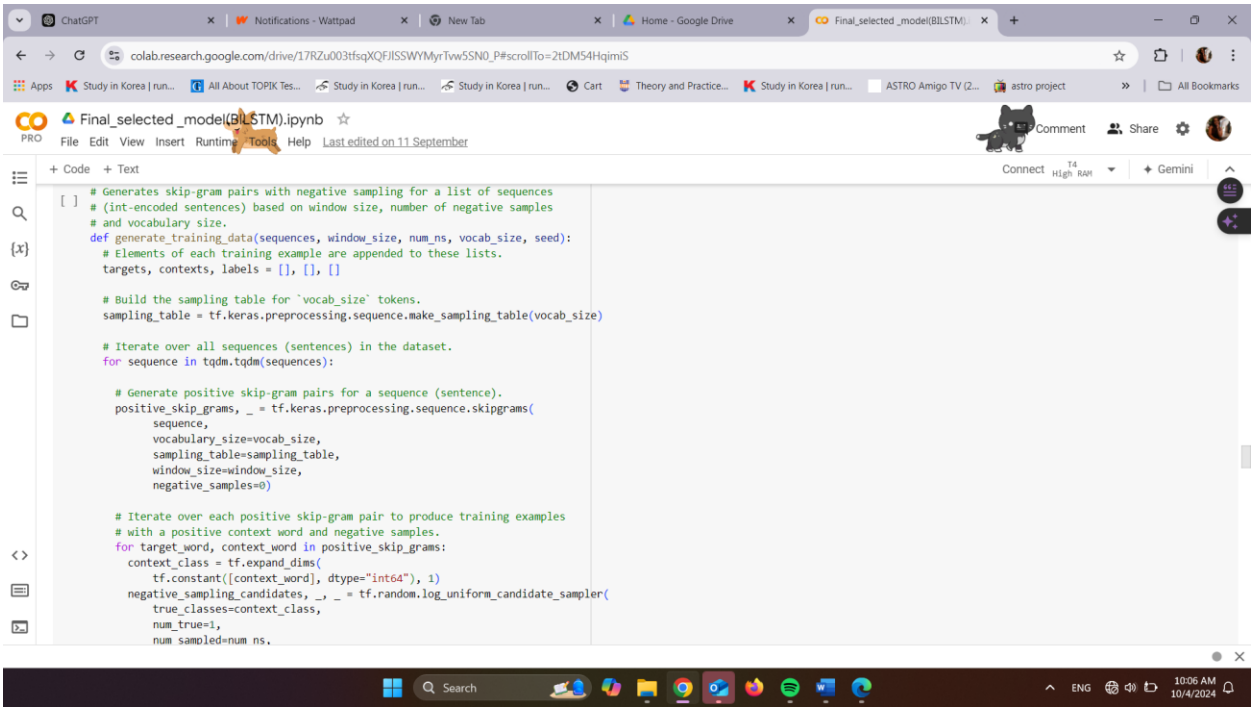

Figure 25 Training word2vec model 2

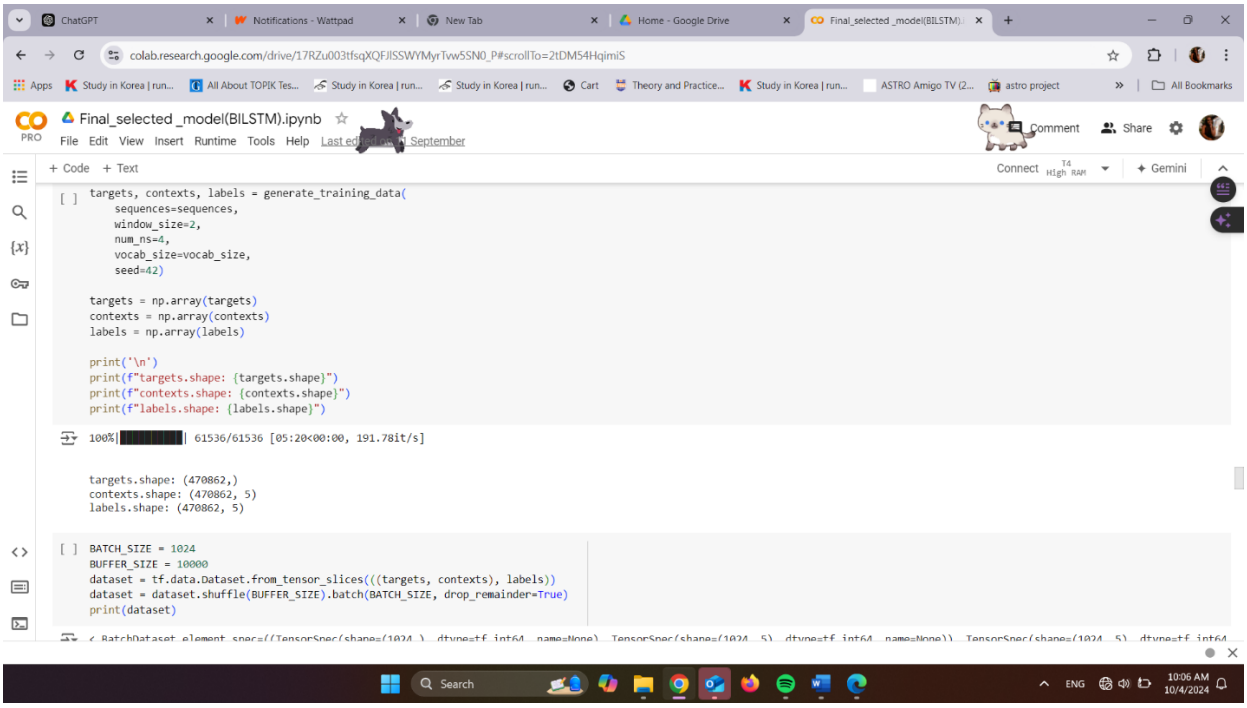

Figure 26 Training word2vec model 3

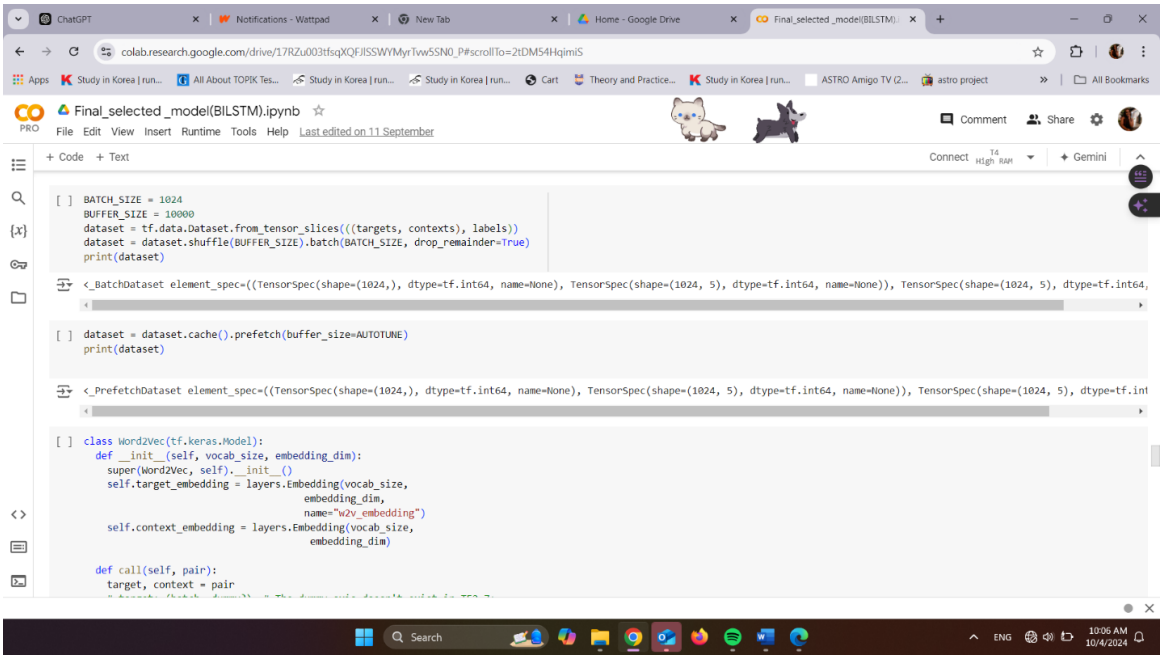

Figure 27 Training word2vec model 4

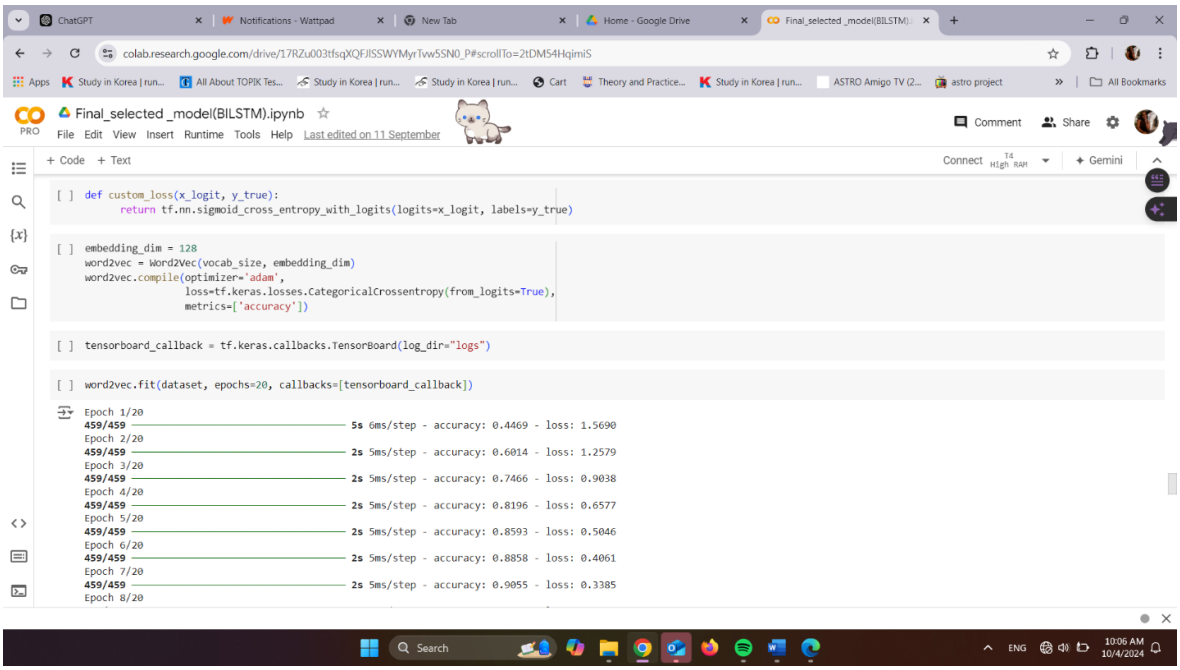

Figure 28 Training word2vec model 5

4.5 model training

Machine learning models are crucial in detecting and mitigating hate speech online. They can be broadly categorized into classical machine learning models and deep learning models, each offering unique approaches to text classification tasks.

Classical ML models - Some of the models presented here combine with the traditional text representation techniques of Bag of Words and TF-IDF, including Support Vector Machines, Naive Bayes, and Logistic Regression. Conventional models were good to go when it came to simple hate speech detection tasks based on handcrafted features and statistical ways to find out patterns of interest in text data. While they require less computational power and are easier to interpret, they may lack the capturing of complexities and nuances that characterize hate speech, such as sarcasm or context-dependent meanings

Deep Learning Model - Deep learning approaches, such as RNNs, LSTM, BILSTM, and transformers, follow complex architecture for better representation of text. Most models utilized pre-trained embeddings and huge datasets in order to contextualize relationships and learn about linguistic subtleties. Deep learning approaches have shown very promising results with regard to detecting implicit hate speech and sophisticated linguistic structures. However, most of these methods require more computational resources and extensive training data.

By leveraging both the classic models and the deep learning models, hate speech detection is better dealt with, hence contributing to the much aspired-for safe and inclusive digital spaces

4.5.1 The Models Used

4.5.1.1 Support Vector Machine

The SVMs are among the most efficient and powerful supervised learning algorithms that have been adopted in many applications, including text classification like hate speech detection. Their principle is to seek an optimum hyper plane that separates the points about different classes with a maximum margin, with the aim of increasing the generalization capability of the model. It is especially useful in cases where the data isn't linearly separable. It employs kernel functions, such as polynomial or radial basis function kernels, for mapping the input features to higher-dimensional spaces. This is highly flexible for SVM when dealing with complex and overlapping features that are very common in hate speech data. Moreover, because SVM works on the principle of minimizing the chance of over fitting, it is very useful in instances where the texts are high-dimension and the minute expressions between hate speech and neutral speech are to be determined.

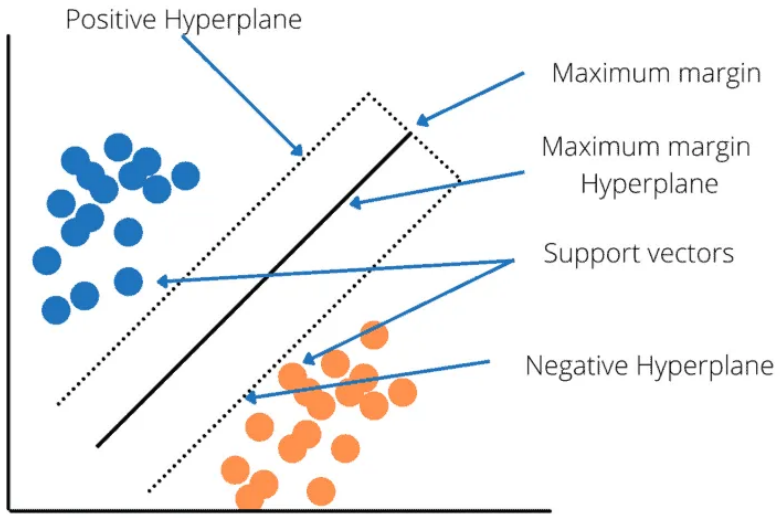

Figure 29 Support Vector Machine

SVM model

```
[ ]  parameters_SVM = {'kernel':('linear', 'rbf','sigmoid','poly'), 'C':[1] ,'gamma':(['scale','float','auto'])}
     svc = svm.SVC()
     clf = GridSearchCV(svc, parameters_SVM)
     clf.fit(x_train_vectorizer,y_train)
```

Figure 30 SVM model

4.5.1.2 Logistic Regression

Logistic regression represents one of the most well-known supervised learning algorithms, widely applied to both binary and multi-class classification tasks; hence it is quite useful in the hate speech detection task. It models the relation between input features and the probability of a target belonging to a certain class using a logistic function or sigmoid function to output probabilities. In the context of hate speech detection, the text data is first represented in numerical features using techniques like BoW or TF-IDF, then logistic regression learns the weights over these features to effectively separate the hate speech from non-hate speech. It is a relatively simple model but works well when the classes are linearly separable. Logistic regression normally serves as a good baseline in comparing more complex models, and its interpretability and efficiency make it very useful in understanding which features contribute most too classifying content as hate speech or neutral.

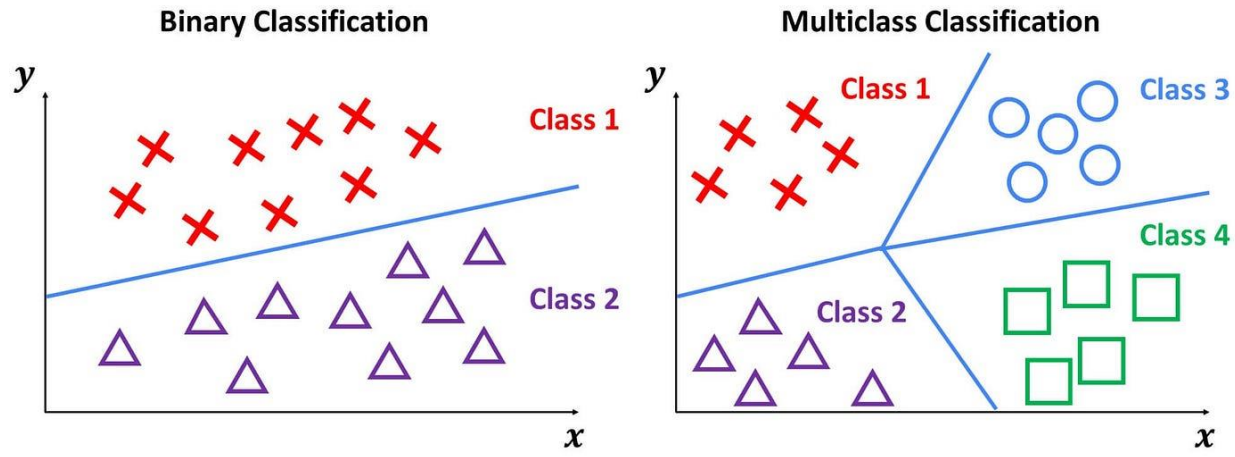

Figure 31 Binary & Multiclass classification

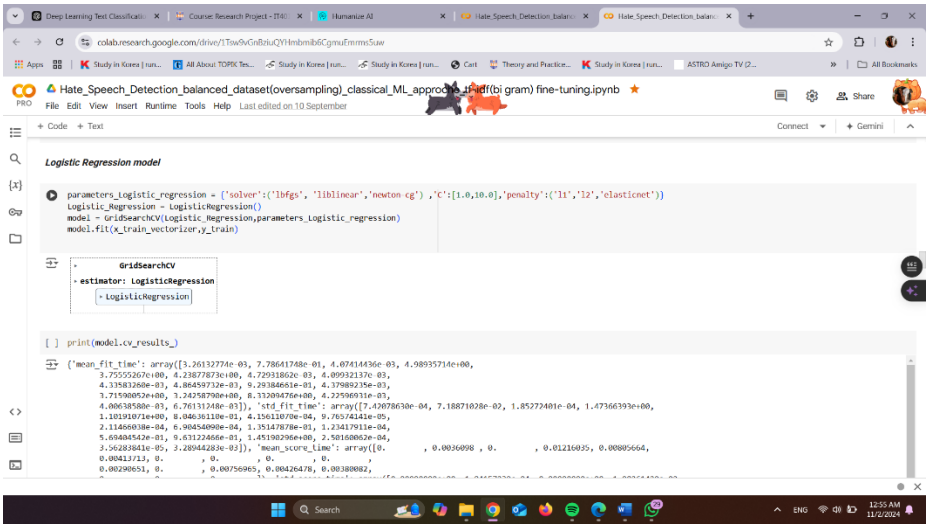

Figure 32 Logistic Regression model

4.5.1.3 Decision tree

Decision Trees represent one of the supervised learning algorithms that are widely used for classification tasks. They can be applied to hate speech detection. The working process is a recursive partitioning of the dataset into smaller subsets, using the value of the input features. In this tree-structural model, each internal node denotes a feature test, while every branch addresses an output, and every leaf denotes a class label. This algorithm recursively splits data with the goal of maximizing information gain or, correspondingly, minimizing impurity-based metrics, including Gini impurity and entropy. In the context of hate speech detection, Decision Trees are

able to discover such a pattern in text features; for instance, certain words or phrases will more likely signal either offensive or neutral speech. Although easy to interpret and visualize, Decision Trees are prone to over fitting when the dataset becomes too complex or noisy. Such methods include, among others, pruning or ensemble methods that develop more robust models, including Random Forests.

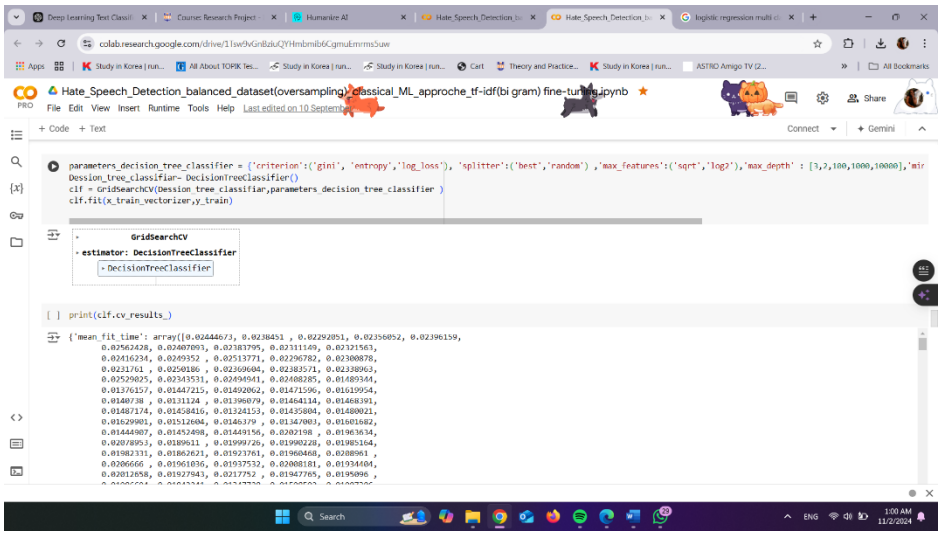

Figure 33 Decision Tree

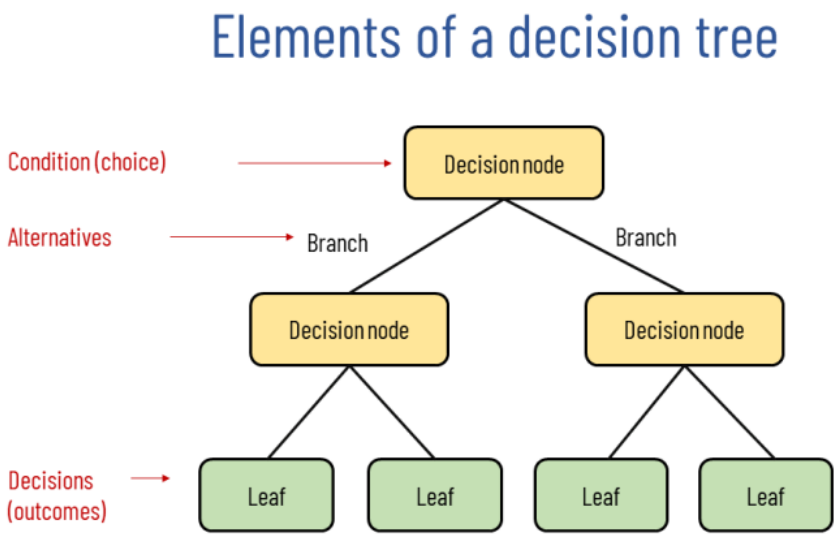

Figure 34 Elements of a decision tree

4.5.1.4 Random Forest

The Random Forest is an ensemble learning method by which numerous Decision Trees are constructed and later combined in order to raise the performance of the model as well as make it more robust. Every tree within a forest is trained on a random subset of training data, whereas at each split, features are picked randomly. This helps in reducing overfitting and enlarging the generalization capability of the model. In the case of hate speech detection, Random Forest works well with high-dimensional text data by capturing complex relationships among features, such as specific word combinations that give away hate speech or neutral language. It makes a final prediction by aggregating votes from all constituent trees, further enhancing the accuracy of the model and reducing its sensitivity to noise in the data. Besides this, another strong ability of Random Forest is ranking features according to their importance, thereby allowing one to induce which terms or patterns are most influential in identifying hate speech, hence making this model more interpretable for real-world applications.

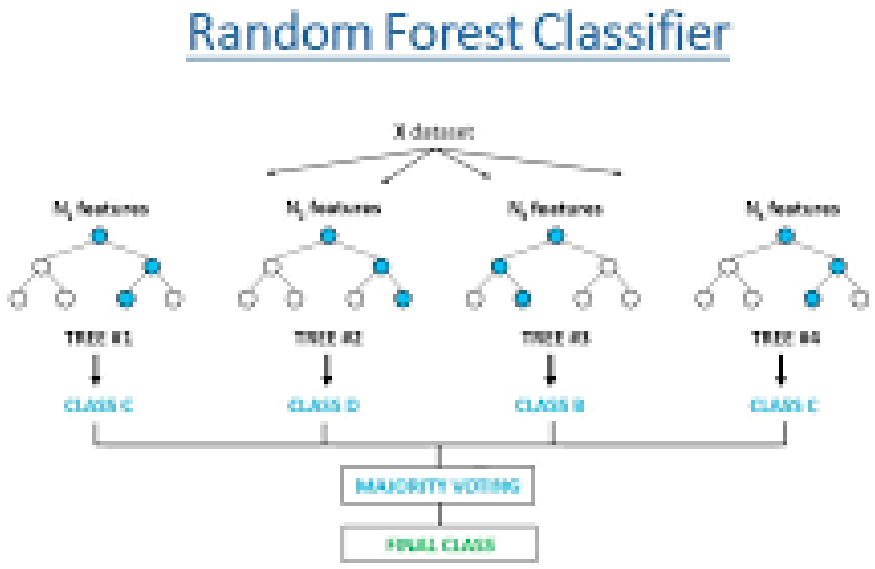

Figure 35 Random forest classifier

4.5.1.5 KNN

KNN is a simple but efficient supervised learning algorithm used for classification tasks. It has been one of the most widely used systems for hate speech detection. For classifying an unseen data point, KNN considers the majority class of its k nearest neighbors in the feature space, where k is a user-defined parameter. Distance metrics such as the Euclidean distance are common metrics used to compute closeness between data points. In the detection of hate speech, first, text data is transformed into numerical feature vectors so that KNN can compute similarities among the given text samples. The final algorithm is very intuitive and easy to implement. That means it easily can become an entry point when experimenting with most data. However, typically, the KNN involves a high computational cost in performing a large dataset because the distance of every new coming data point with all the other data points needs to be computed. It can be further sensitive to the selection of k and distribution in the data, which may therefore impact its correctness in identifying hate speech, particularly in high-dimensional text data.

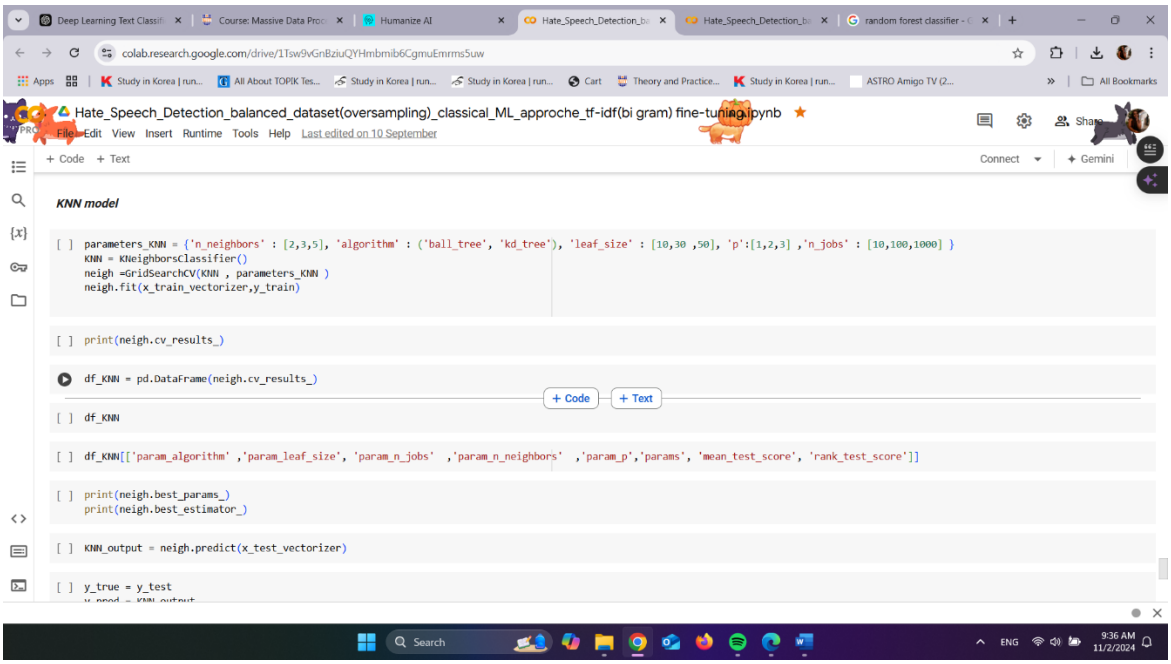

Figure 36 KNN model

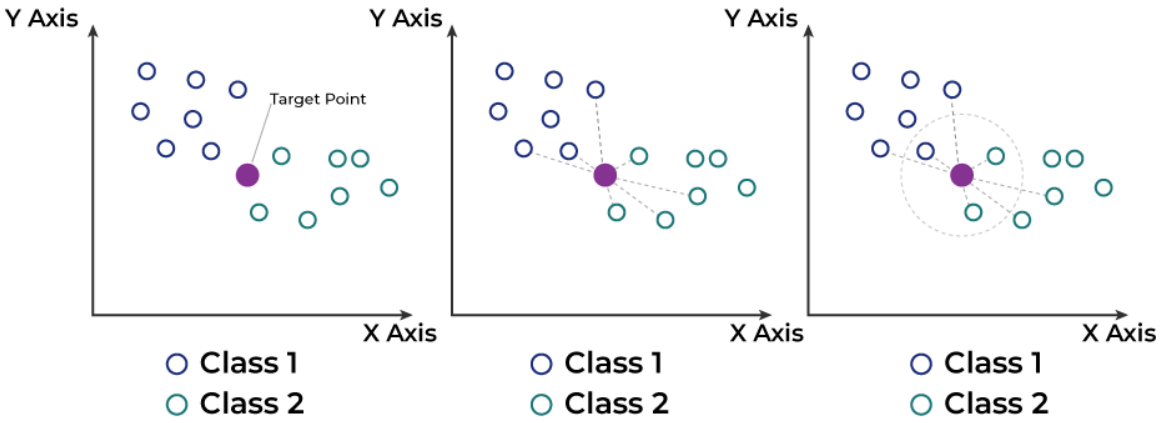

Figure 37 KNN model classification

4.5.1.6 Passive aggressive

The Passive-Aggressive Classifier is an online efficient learning algorithm for binary and multi-class classification tasks, such as hate speech detection. Unlike traditional models, which are designed to minimize mistakes while minimizing an error function, Passive-Aggressive algorithms make updates only in case they misclassify a data point or when the confidence of prediction is not enough. By "passive" here, it means that the model is not updated when the classification is correct, and "aggressive" means a considerable update in the model parameters in case of misclassification. This approach renders the algorithm quite apt for large-scale and streaming data scenarios common in text classification. For hate speech Polarity, features are usually extracted using methods such as TF-IDF, in which the Passive-Aggressive Classifier learns to make a difference between hate speech and neutral language quite efficiently. This algorithm has simplicity and can be applied on big data, which is very useful for real-time applications; however, careful tuning may be called for in order to have a good balance between performance and computational efficiency.

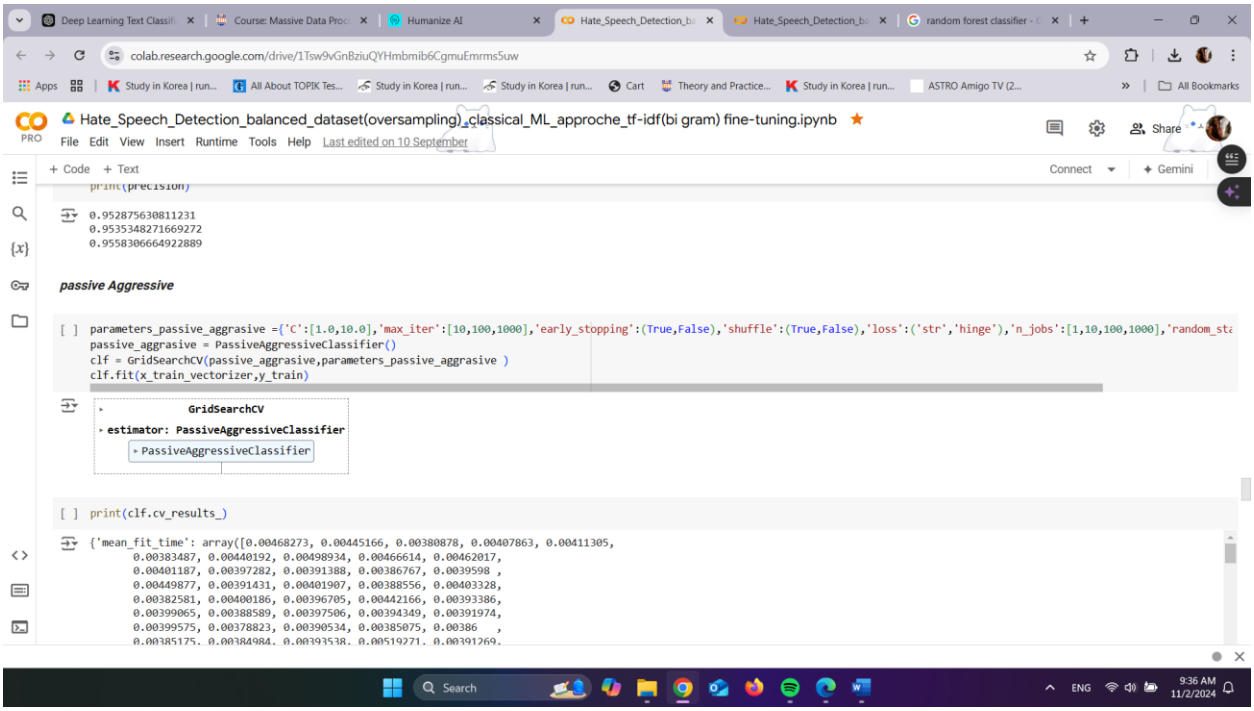

Figure 38 Passive aggressive

4.5.1.7 Naive Bayes

Naive Bayes is a family of simple but effective probabilistic classifiers based on Bayes' theorem very popular for text classification tasks, including hate speech detection. The "naive" assumption in the name means that the model assumes that all features are independent from each other, which very seldom holds true for real world data but simplifies computation by an order of magnitude. There are several types of Naive Bayes classifiers, which include the Multinomial, Bernoulli, and Gaussian. The Multinomial Naive Bayes classifier does exceedingly well on text data. In hate speech detection, features such as word frequency or presence/absence indicators are first transformed using techniques such as the Bag of Words or Term Frequency-Inverse Document Frequency. Given that this is the simplicity of Naive Bayes, it usually does relatively well in practice where the independence assumption is approximately satisfied and is hence very efficient on large datasets. The limitation might be that it would fail to capture more complicated linguistic patterns, hence probably lower performance on more subtle and context-dependent manifestations of hate speech.

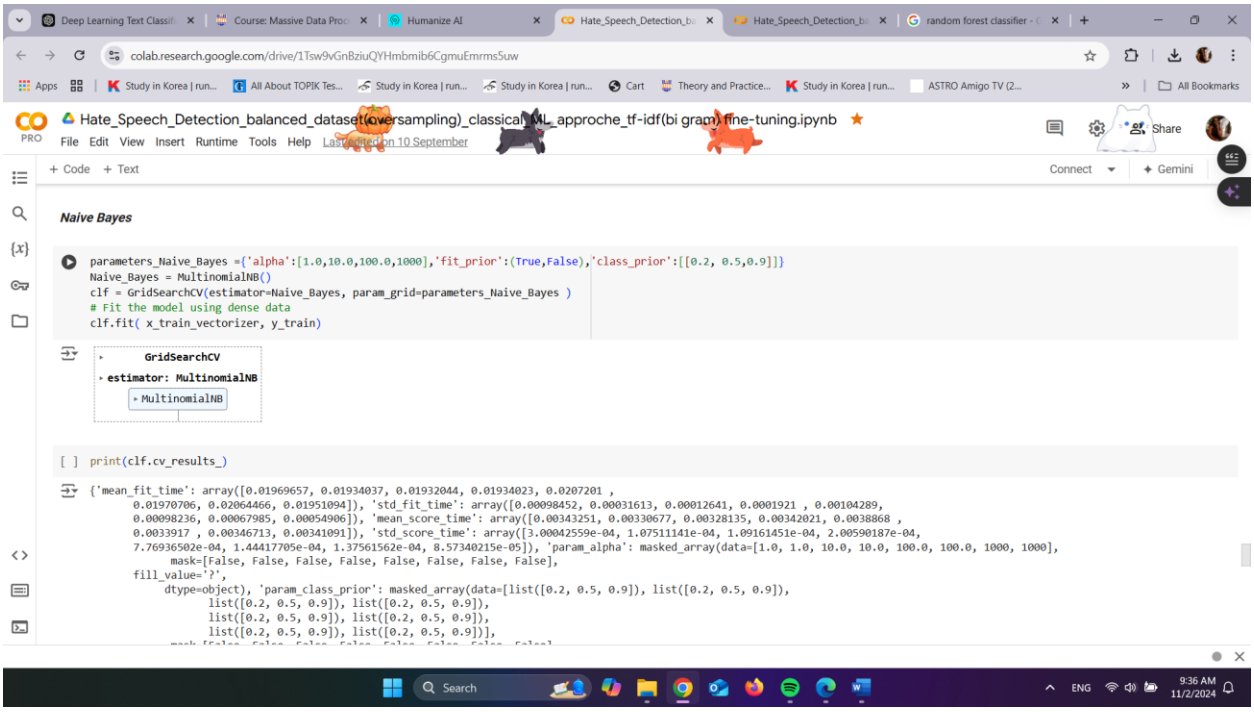

Figure 39 Naïve Bayes

4.5.1.8 Recurrent Neural Network (RNN)

Recurrent Neural Networks represent the deep learning models that are especially prepared to process sequences of data, which, in fact, makes them quite suitable for hate speech detection. Unlike other neural networks, RNNs contain loops and thus can store information from past inputs and learn from context within a sequence of words. This captures temporal dependencies, which are very important in understanding the text in both meaning and tone. More subtle language patterns arise, especially in the case of identifying hate speech, which tends to rely on subtle context. For example, variants of RNN include LSTM and GRU, both developed with the aim of overcoming issues such as the vanishing gradient problem, so that the model can keep information for longer sequences. Such RNNs identify hateful content, even in instances where it was implicit or context-dependent, either by leveraging pre-trained embeddings or by learning embeddings from scratch. However, RNNs are usually computationally expensive and may require a huge amount of training data to achieve high performance; hence, optimization becomes essential for practical use in hate speech detection.

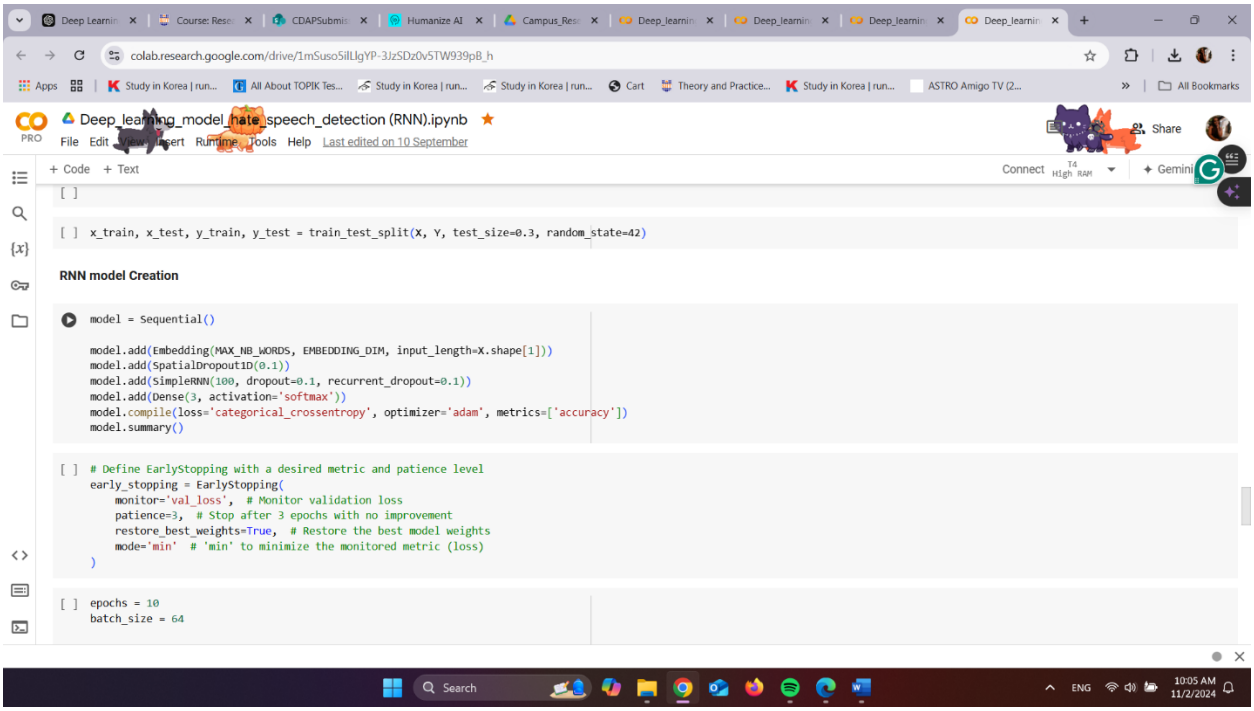

Figure 40 RNN model creation

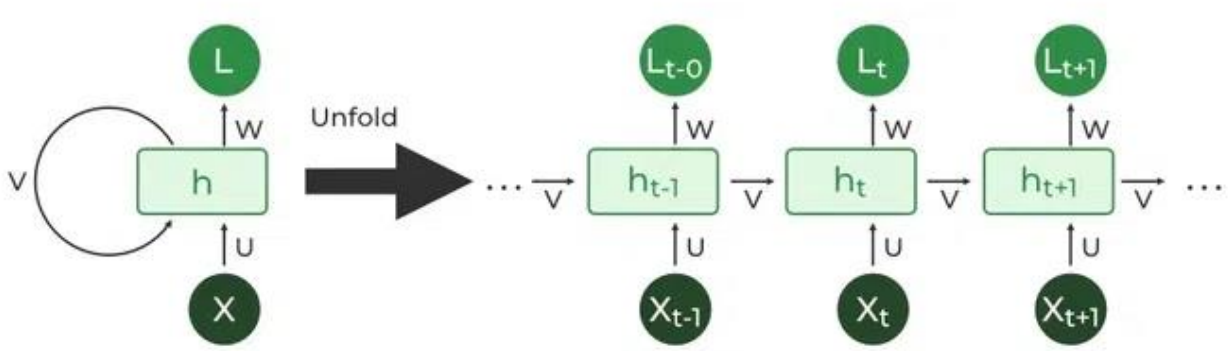

Figure 41 RNN model

4.5.1.9 Long Short-Term Memory (LSTM)

Long Short-Term Memory Networks are a special form of RNN designed to overcome some of the major traditional RNN drawbacks, including the problem of vanishing gradients. LSTMs are quite effective at processing and classifying sequences; hence, it should be an excellent choice for any application related to hate speech detection. Three main components of memory cells comprise the architecture of LSTM networks: input gate, forget gate, and output gate. These gates control how much information will flow through to the model, enabling it to remember or forget over long sequences, which is very useful in understanding contextual details and subtleties in text data. In LSTM networks for hate speech detection, complex text patterns and their dependencies can be learned, thus making the model capable of distinguishing hate speech from offense and neutral, even when the expressions are implicit or context-dependent. The LSTMs, by pre-training with large-scaled word embeddings using Word2Vec or fine-tuning over large-scaled datasets, for example, learn subtle cues and semantic meaning that may lead to more accurate and contextually richer classification. However, LSTM models are computationally intensive to train, and they do require a lot of computational resources, particularly when dealing with extensive text corpora.

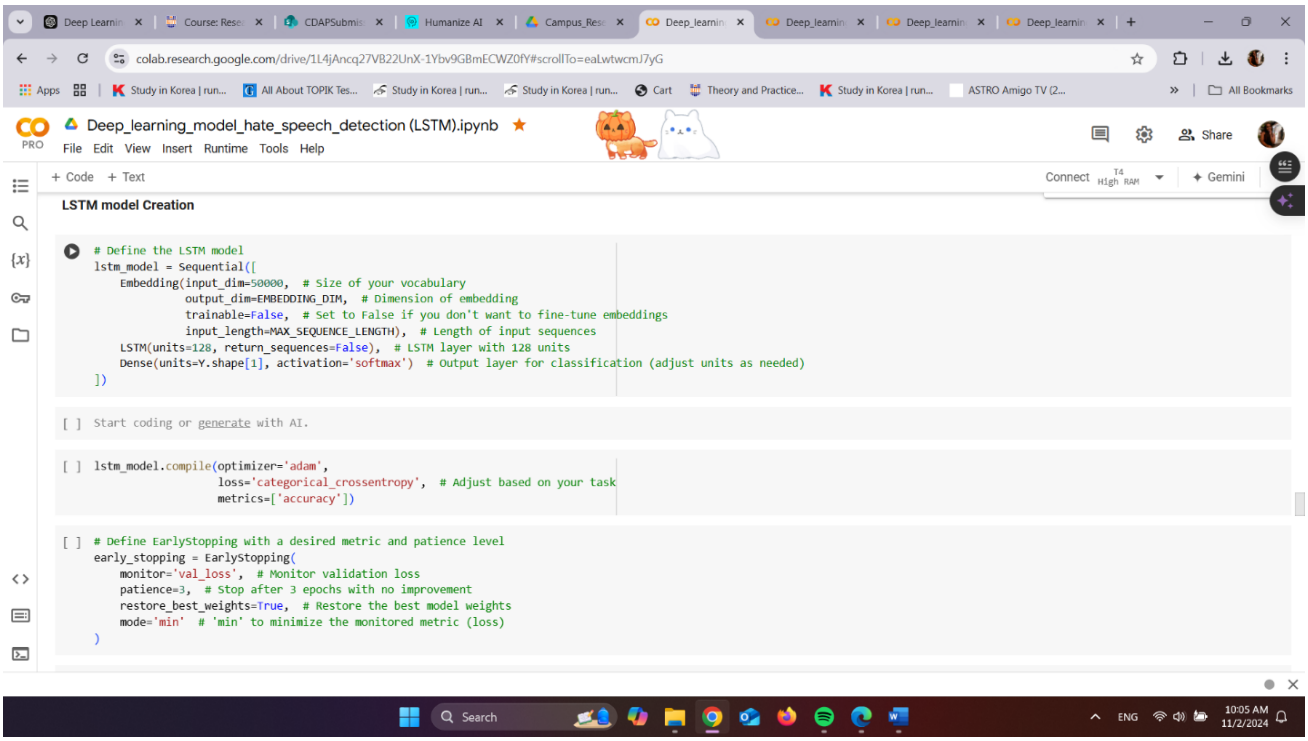

Figure 42 LSTM model creation

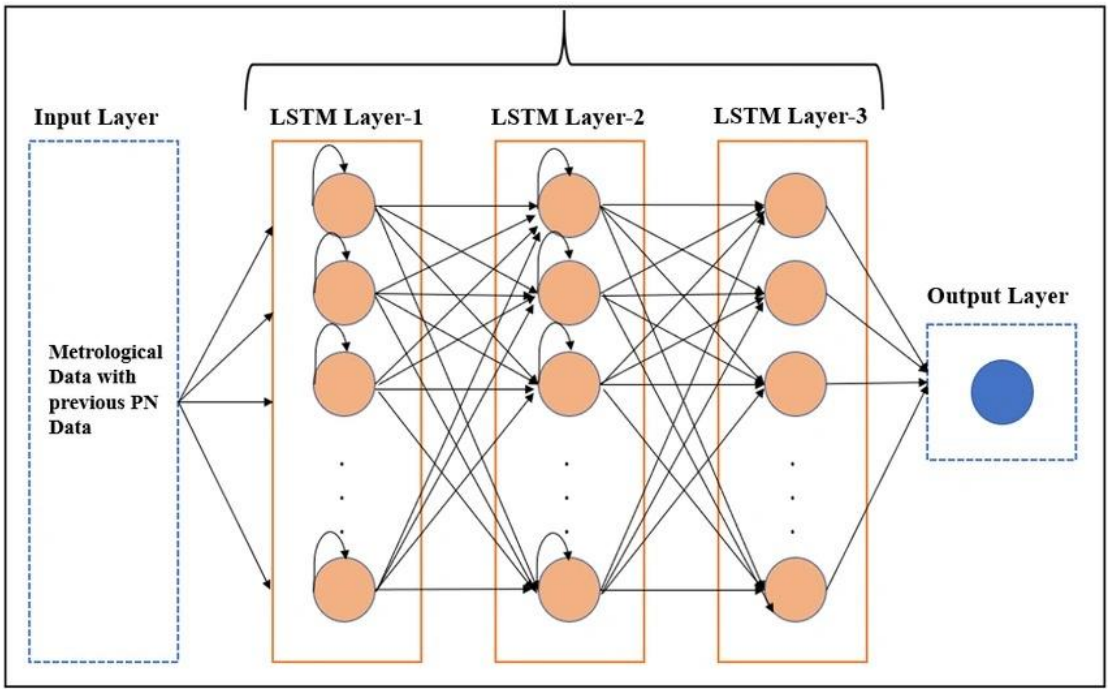

Figure 43 LSTM model

4.5.1.10 Bidirectional Long Short-Term Memory (BiLSTM)

Bidirectional LSTMs are an extension of LSTMs that process the sequence of data both in the forward and the reverse directions. A BiLSTM model combines two layers of LSTM-one reading the input sequence from left to right and another one reading it from right to left. One interesting aspect is that this perspective provides both forward and backward, which is very useful in hate speech detection since the meaning of a word or phrase often depends on what words come before or after. BiLSTM is good at understanding nuanced language, sarcasm, or implicit hate speech that requires comprehensive context in order to interpret. It becomes more powerful in learning semantic and syntactic relationships, using word embeddings like Word2Vec or fine-tuning pre-trained ones. Although the benefits of BiLSTMs over unidirectional LSTM models in improving classification tasks with enriched understanding of the text come at the cost of higher computational resources and longer training times, it makes them very powerful for capturing complex patterns in language at the cost of resources.

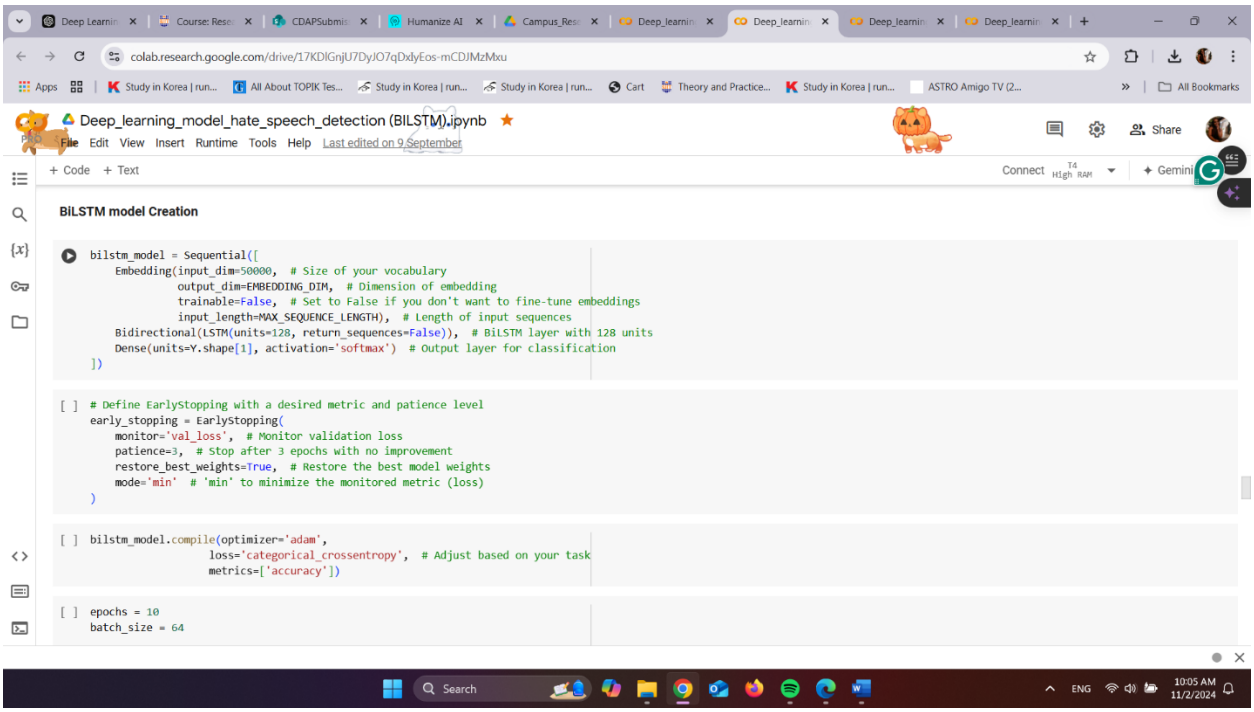

Figure 44 BiLSTM model creation

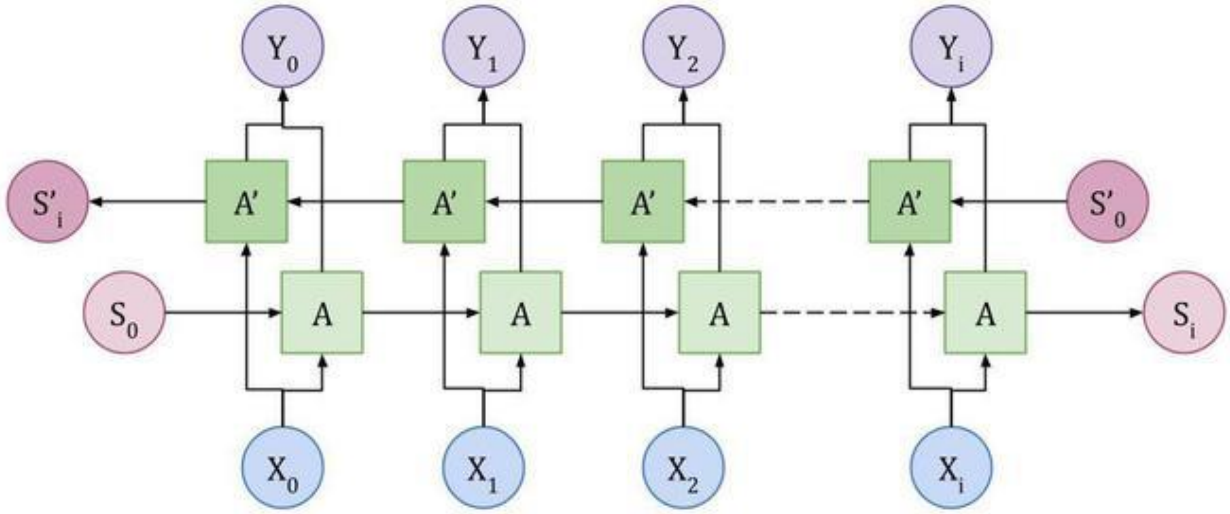

Figure 45 BiLSTM model

4.5.1.11 Convolutional Neural Network (CNN)

The Convolutional Neural Network is a kind of deep learning model employed generally in the analysis that involves grid-like data structures, which include images and text. Whereas their application had earlier been employed in image processing, recently they have been found to perform a number of NLP applications effectively, including hate speech detection. In classification tasks, particularly text classification, CNNs make use of convolutional layers to extract meaningful features from text sequences through the process of filters slipping over word embeddings. These convolution filters capture the local patterns, such as specific word combinations or phrases that may denote hate speech. Consecutives of convolution and pooling layers in CNNs can learn hierarchical features from simple word patterns to complex linguistic structures. The output is subsequently fed into a set of fully connected layers for classification. CNNs are fittingly applied with high efficiency for the task of hate speech detection, as they can easily detect discriminative patterns on small spans of text, hence suitable for dealing with huge text data. On the other hand, perhaps they cannot handle such long-range dependencies as effectively as some other models, like LSTMs, do. Therefore, CNNs will be more appropriate in the case when the local context is more important than sequential order.

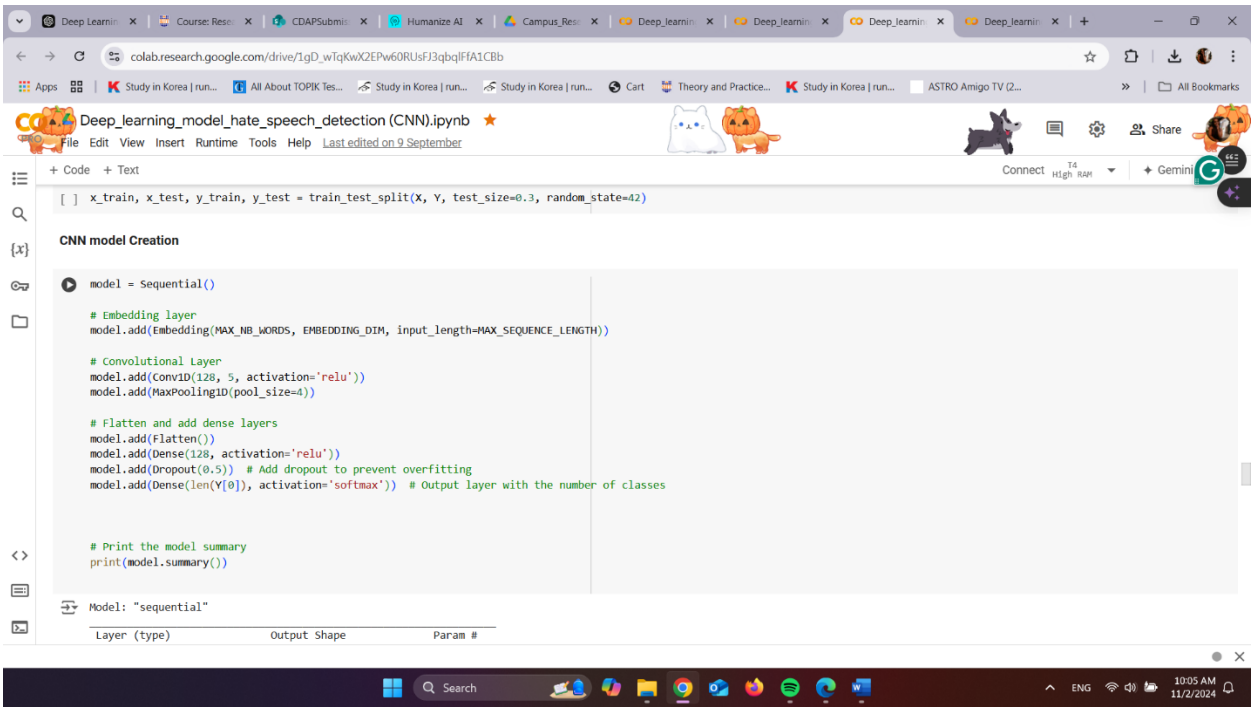

Figure 46 CNN model creation

4.5.1.12 BERT

BERT is a powerful deep learning model from Google that has taken NLP tasks to the next level, including hate speech detection. While most traditional models read their text either from left to right or from right to left, the design of BERT was geared toward reading text bidirectional: it understands the context of a word based on both its preceding and succeeding words. This bidirectional approach lets BERT learn about the subtle and complex relations in a language, crucial for identifying implicit or context-dependent hate speech.

BERT works on a transformer architecture based on self-attention mechanisms that weigh the importance of every word in a sequence relative to each other. It enables the model to give considerable weight to certain key phrases or terms that might describe hate speech while at the same time considering the context within which such phrases or terms occur. The pre-training of BERT is done in two major tasks: Masked Language Modeling (MLM), where some words are kept masked in a sentence and predicted by the model, and Next Sentence Prediction (NSP) helps BERT understand sentence relationships. Pre-training tasks like these allow BERT to have profound understanding of the language, which makes it really powerful in text classification tasks.

BERT has the ability to perform hate speech classification, which can be achieved by fine-tuning on labeled datasets. In such cases, the topmost layers of that particular model are to be tuned concerning the categorization of the text into hate speech, offensive content, or simply neutral. This points to the model's ability to understand linguistic subtlety, sarcasm, and context-very appropriate for the challenges of hate speech detection. Yet, BERT is a very computationally intensive model; it has major resource demands for both training and deployment, especially on larger datasets. However challenging, the superior performance in comprehension and complex text classification made BERT an excellent choice both for research and real-world applications of hate speech mitigation.

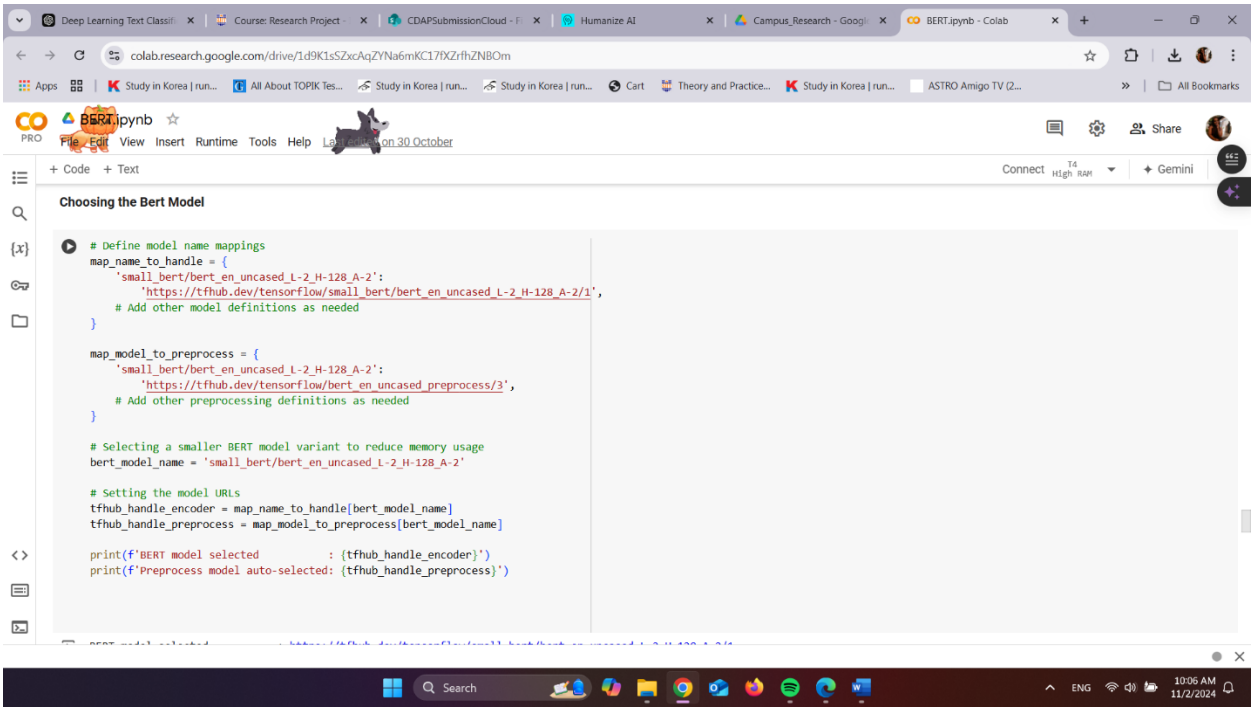

Figure 47 Choosing the BERT model

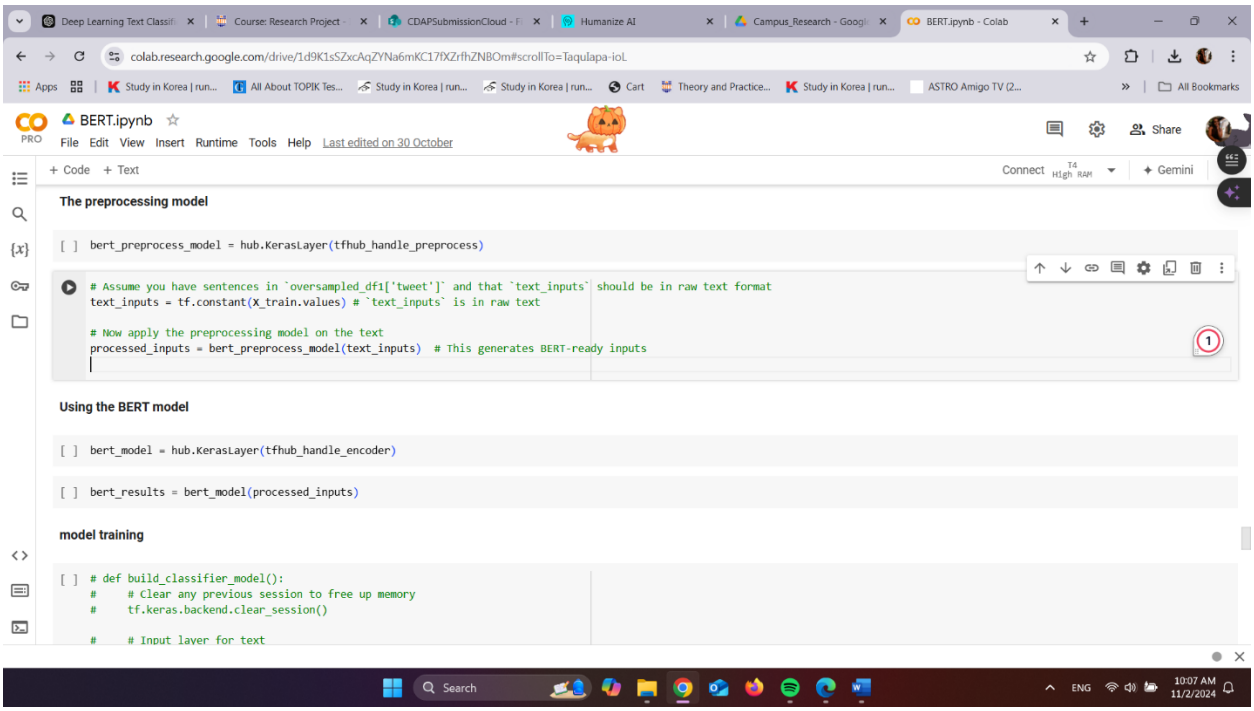

Figure 48 The preprocessing model

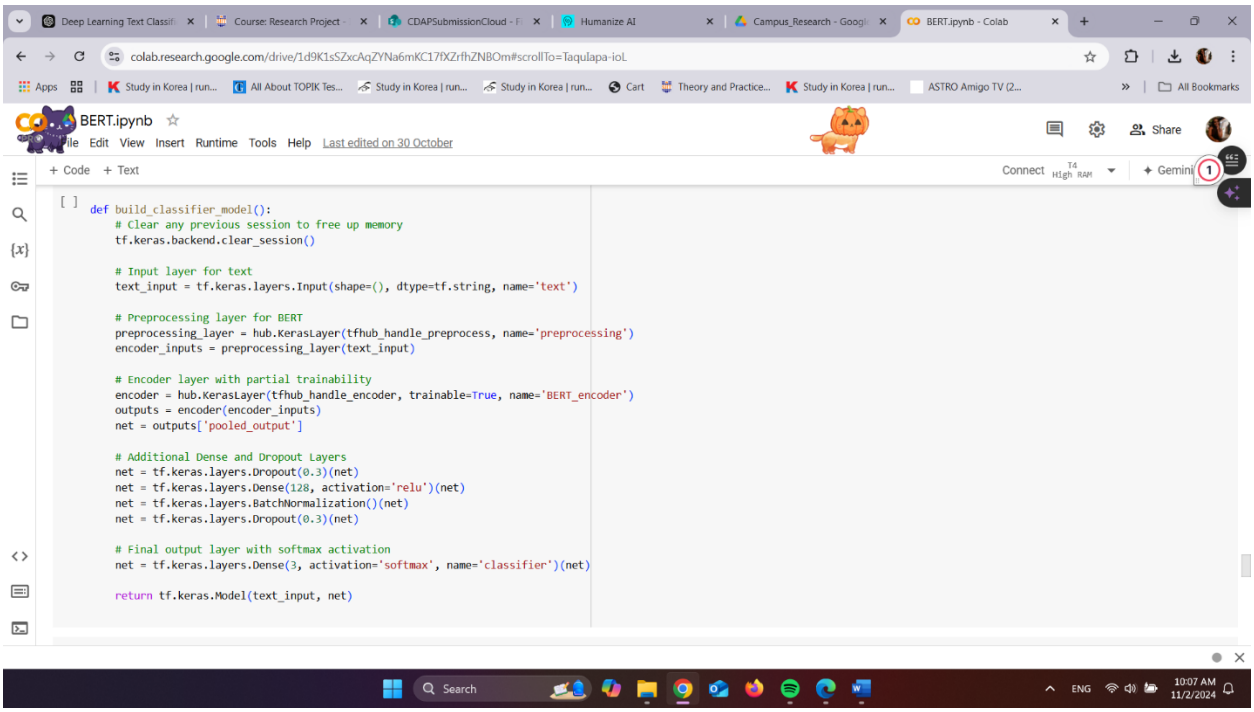

Figure 49 Model training

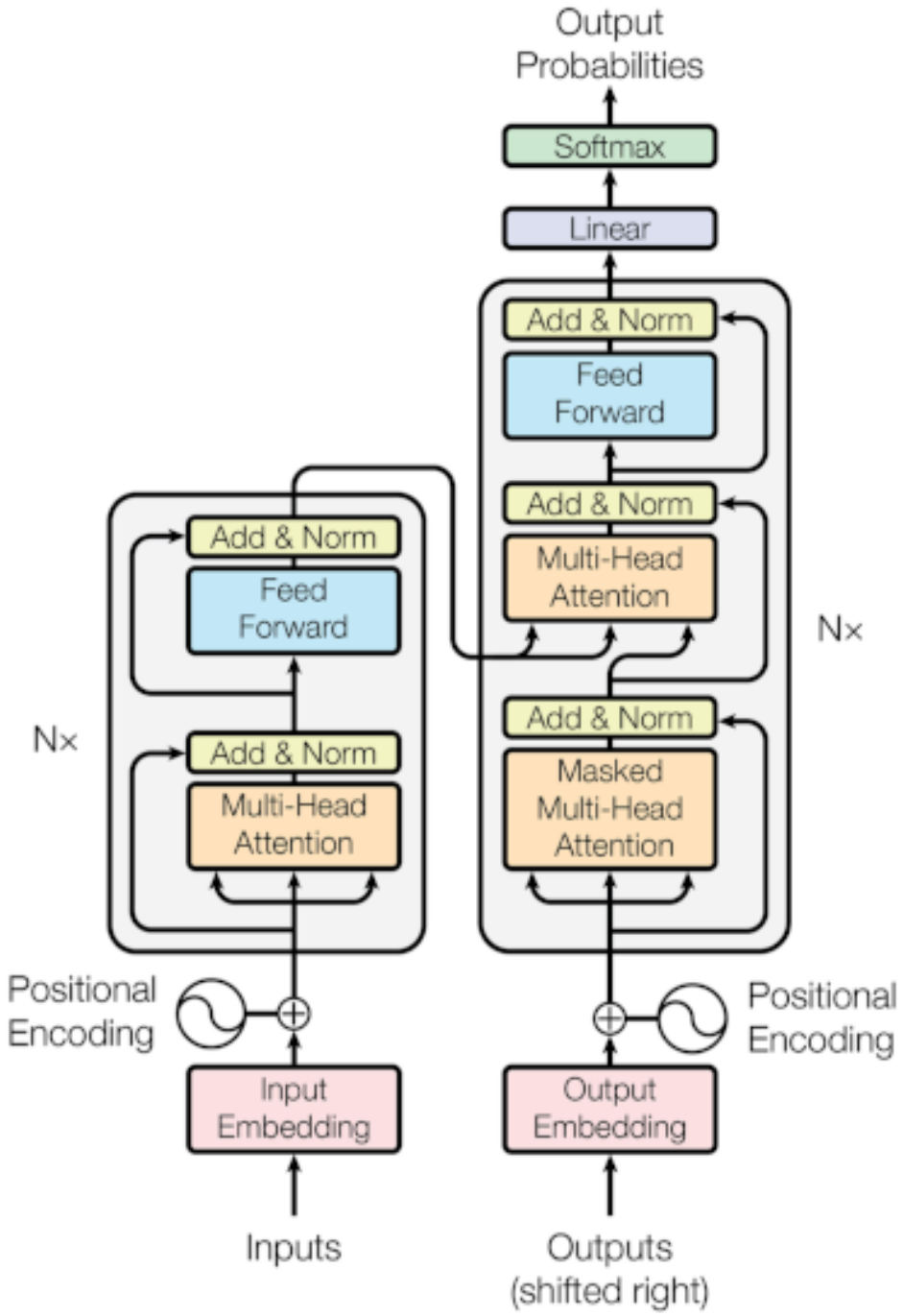

Figure 50 BERT model

4.5.2 Selected model

After a comprehensive evaluation of all the models proposed for hate speech detection, the model selected was the BiLSTM model, as it outperformed the rest with respect to quality and achieved the highest accuracy out of all the architectures tested. It captures the contextual information in a text nicely and thus works perfectly well on subtle language uses pertinent to hate speech.

Further enhancing the model's performance, we implemented a customized Word2Vec embedding system into the model, wherein the BiLSTM could make use of domain-specific word representations that were built from our dataset.

The BiLSTM model was trained using Tensor Flow, with the dataset divided into training, validation, and test sets to ensure reliable evaluation and prevent over fitting. We employed multiple epochs during training to enable the model to iteratively learn from the data, refining its weights and improving its performance with each pass. The combination of the BiLSTM architecture and the customized embeddings resulted in a robust and effective solution for detecting hate speech.

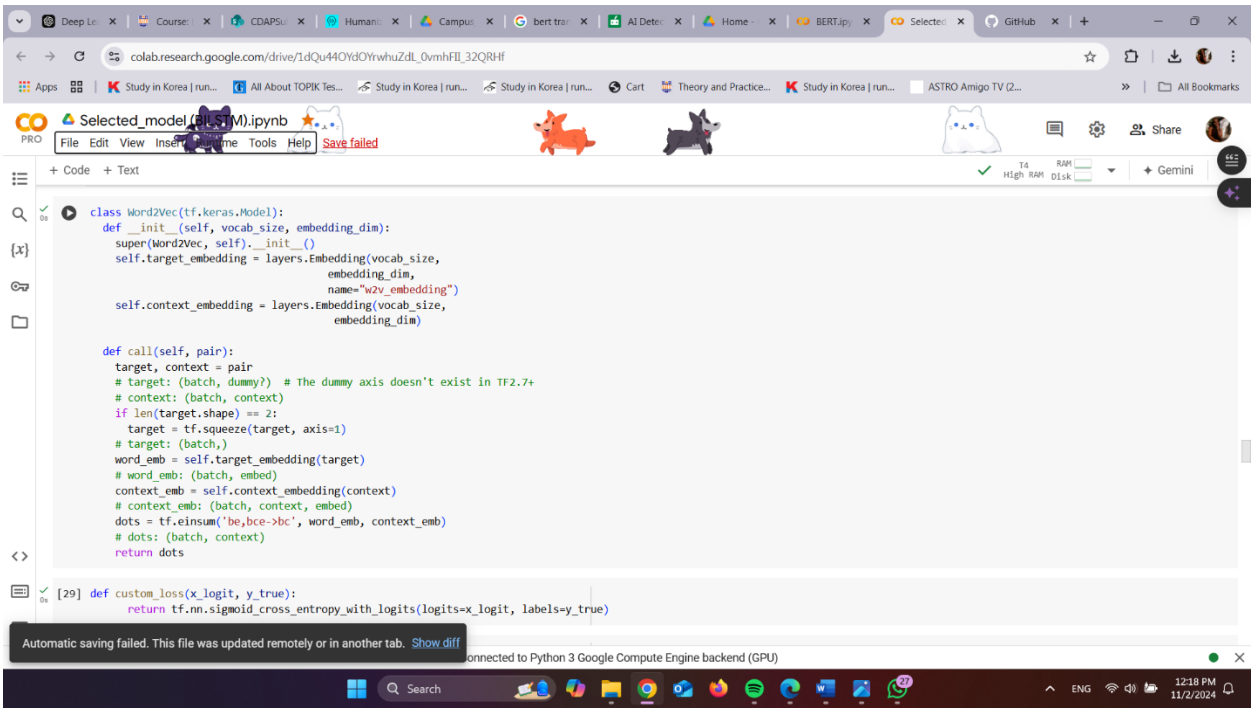

Figure 51 The selected model

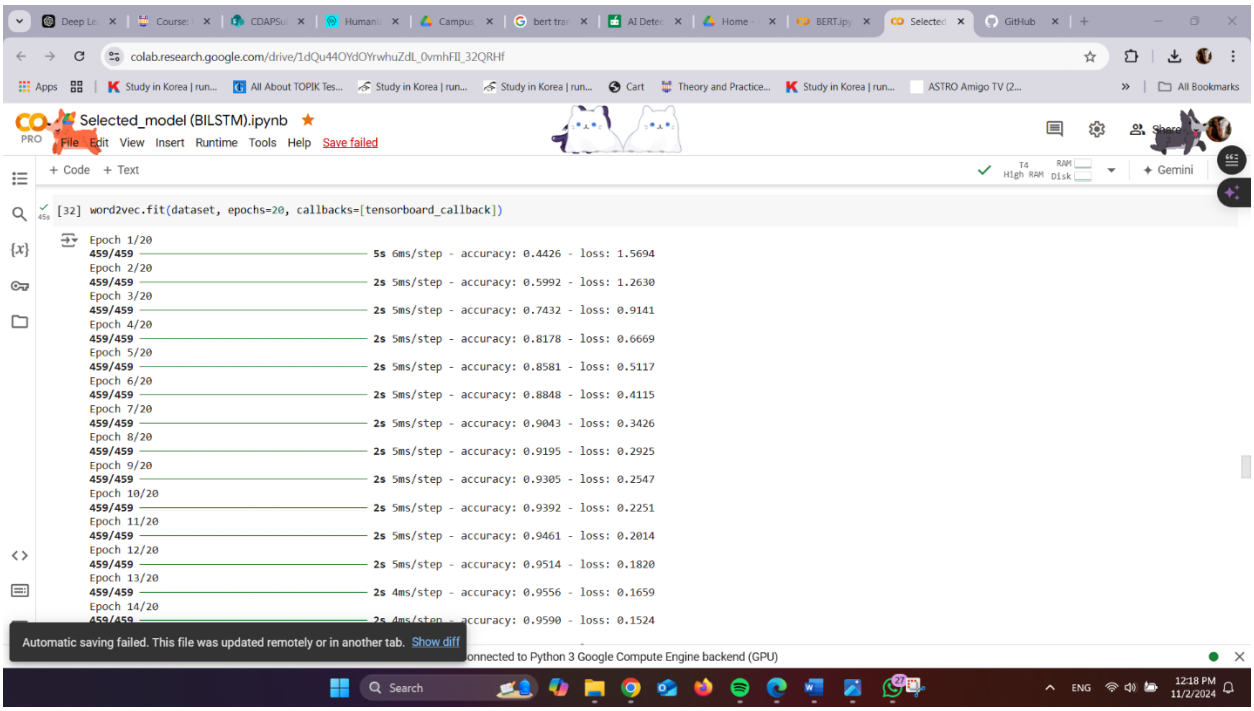

Figure 52 The selected model 1

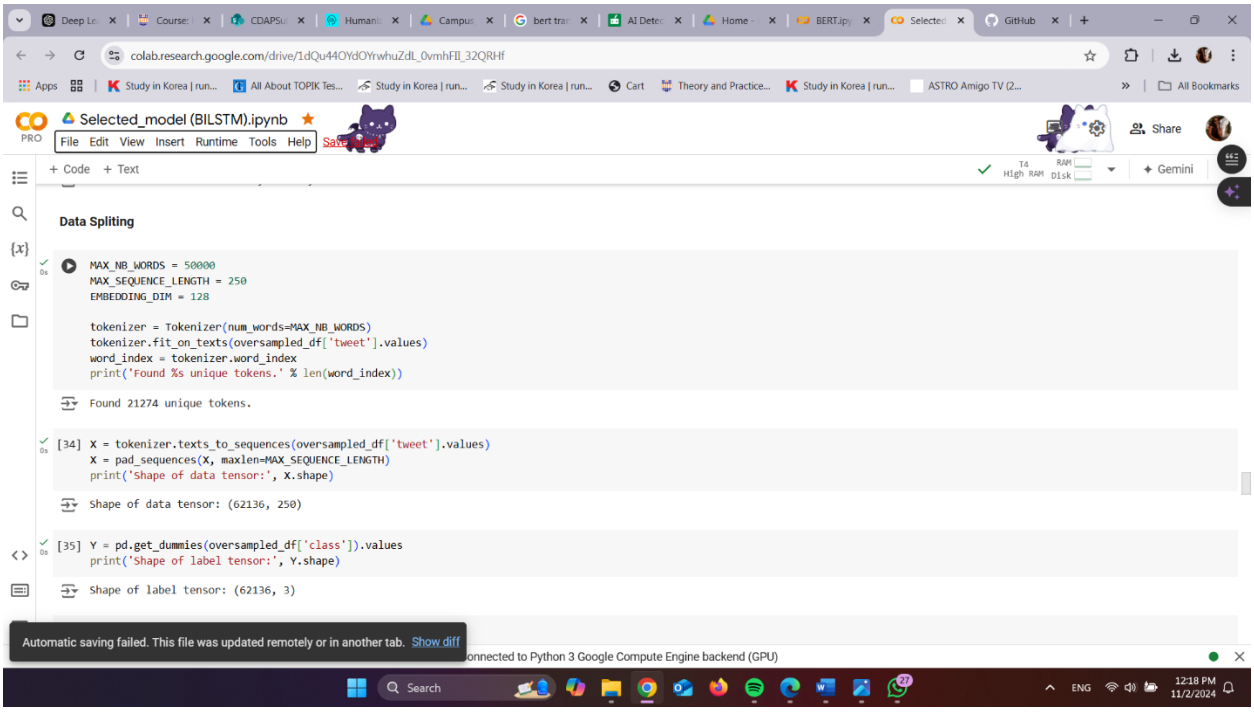

Figure 53 Data splitting

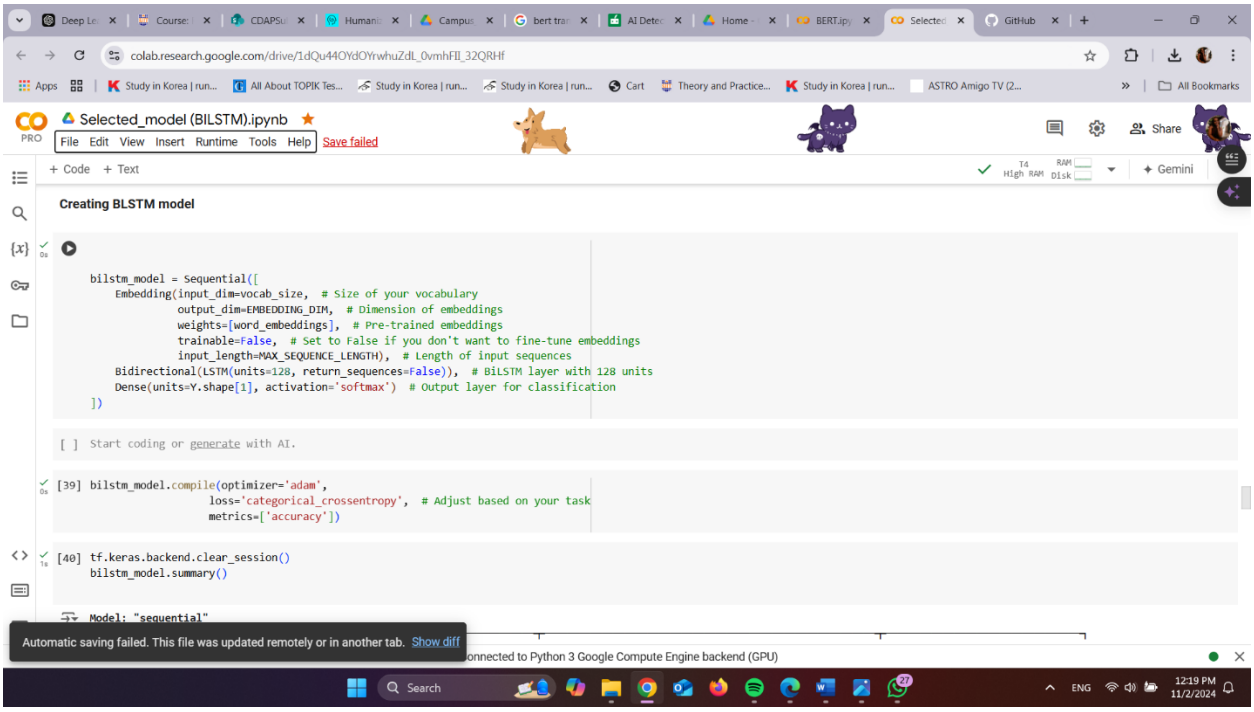

Figure 54 Creating BiLSTM model

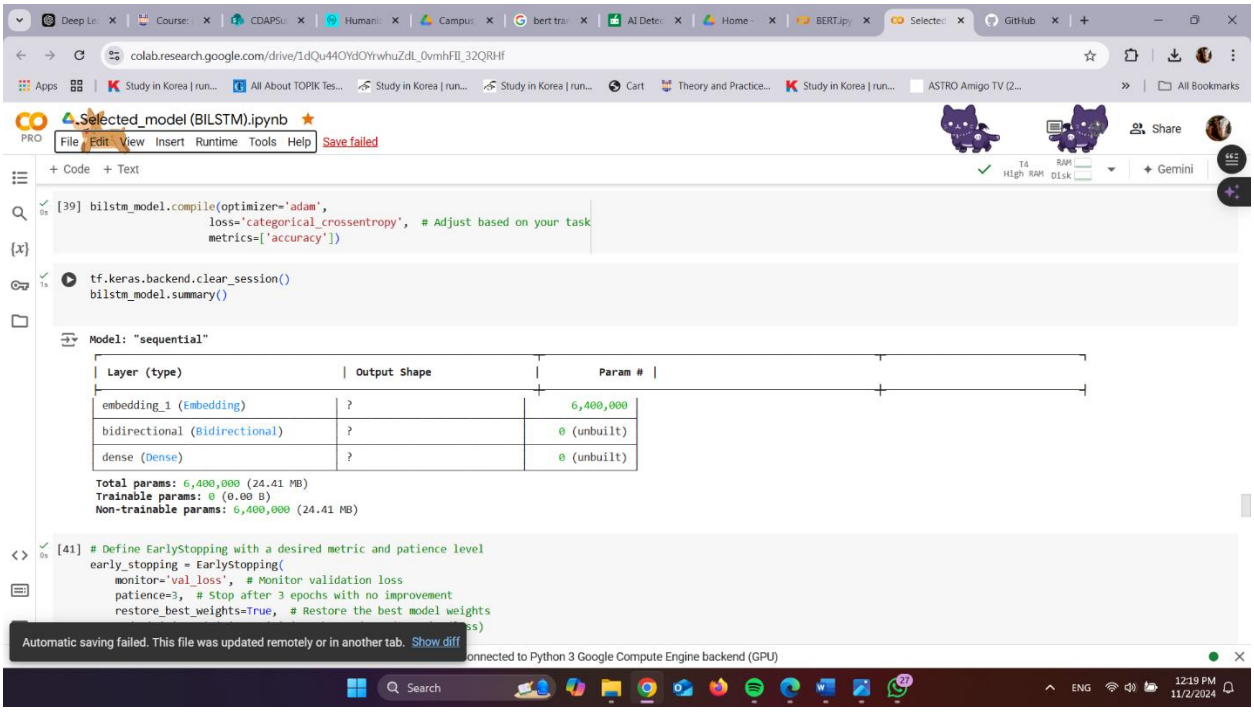

Figure 55 Creating BiLSTM model 1

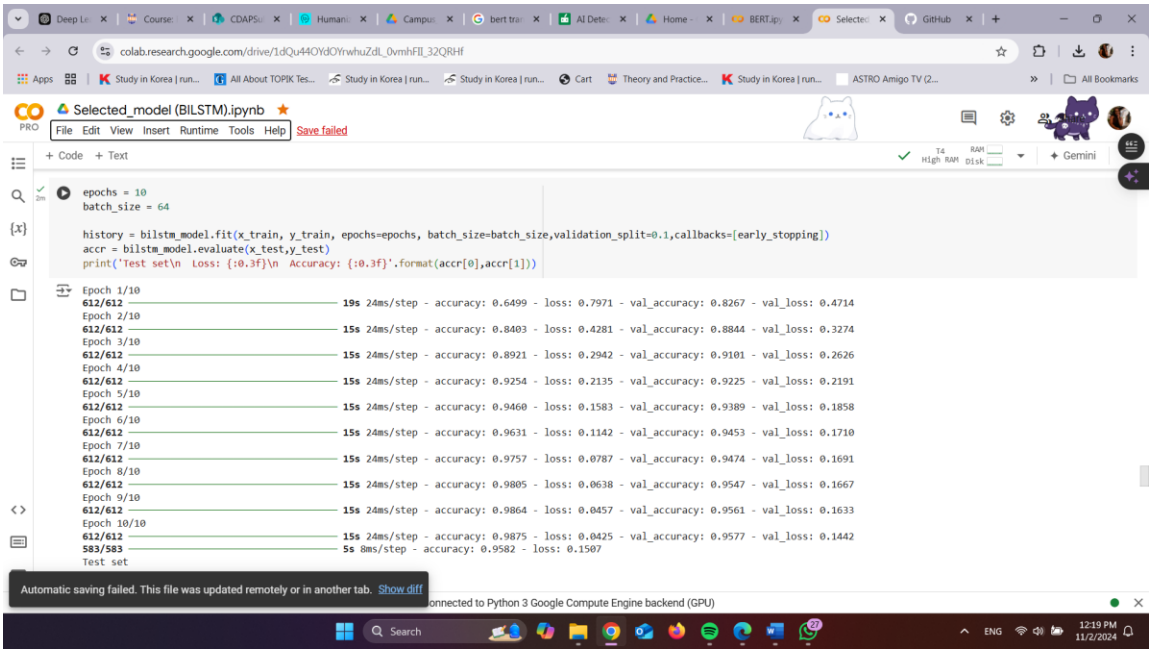

Figure 56 Creating BiLSTM model 2

4.6 Model Testing and Validation

To further evaluate the model's performance, we utilized confusion matrix metrics to analyze classification accuracy, precision, recall, and F1 score, allowing us to gain insights into the model's strengths and weaknesses in detecting hate speech across different classes.

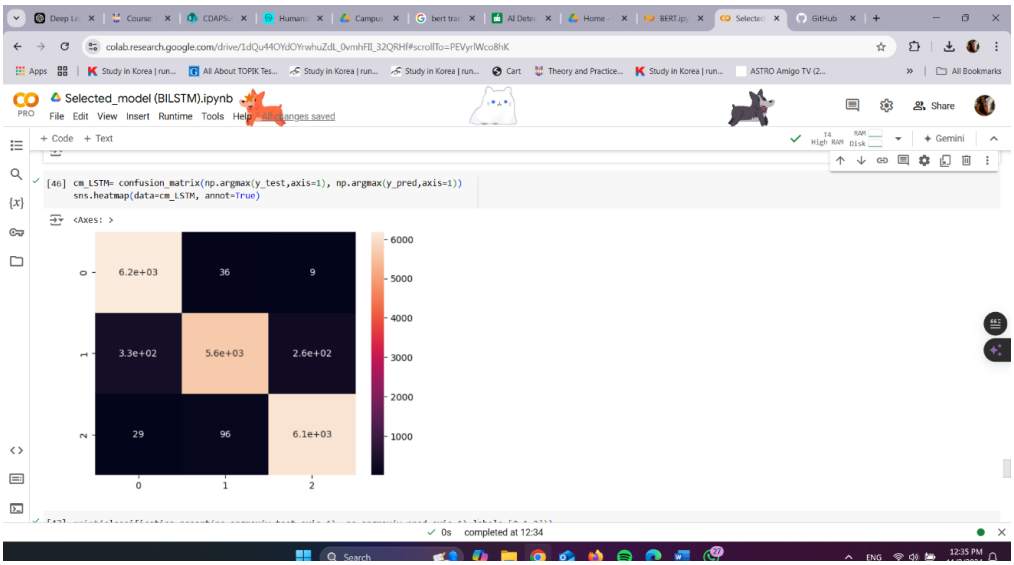

Figure 57 Model training & Validation

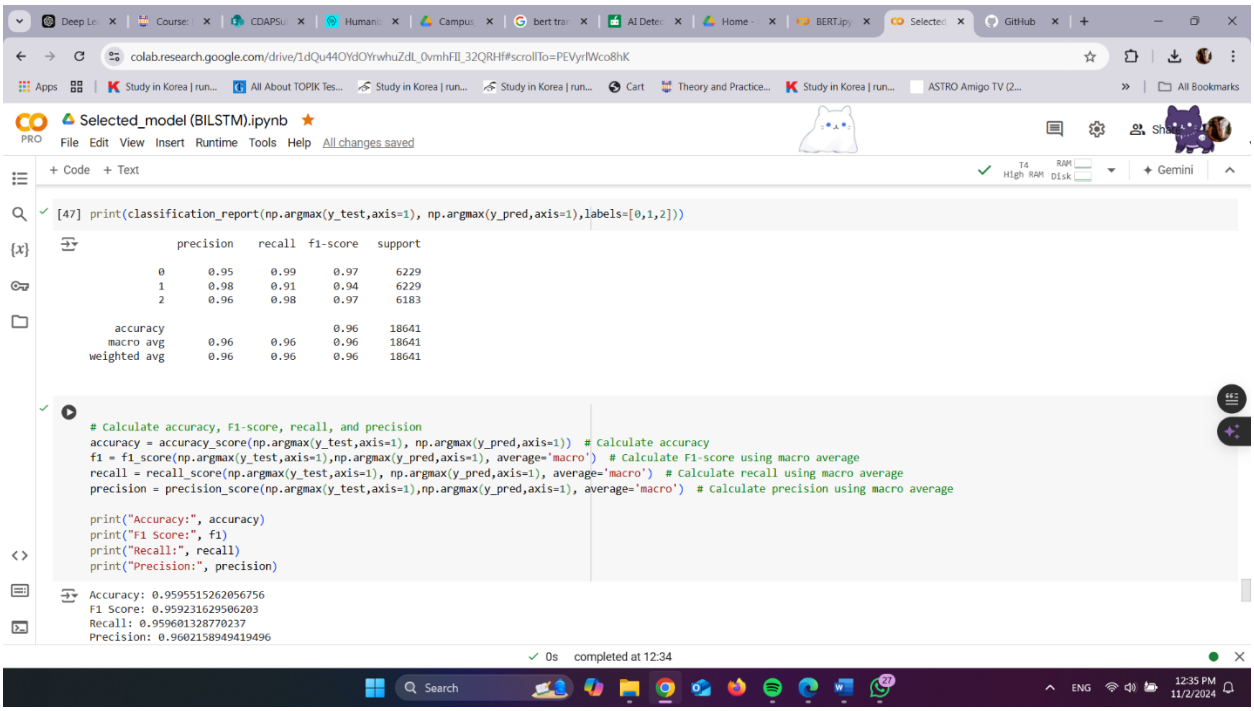

Figure 58 Model training & Validation 1

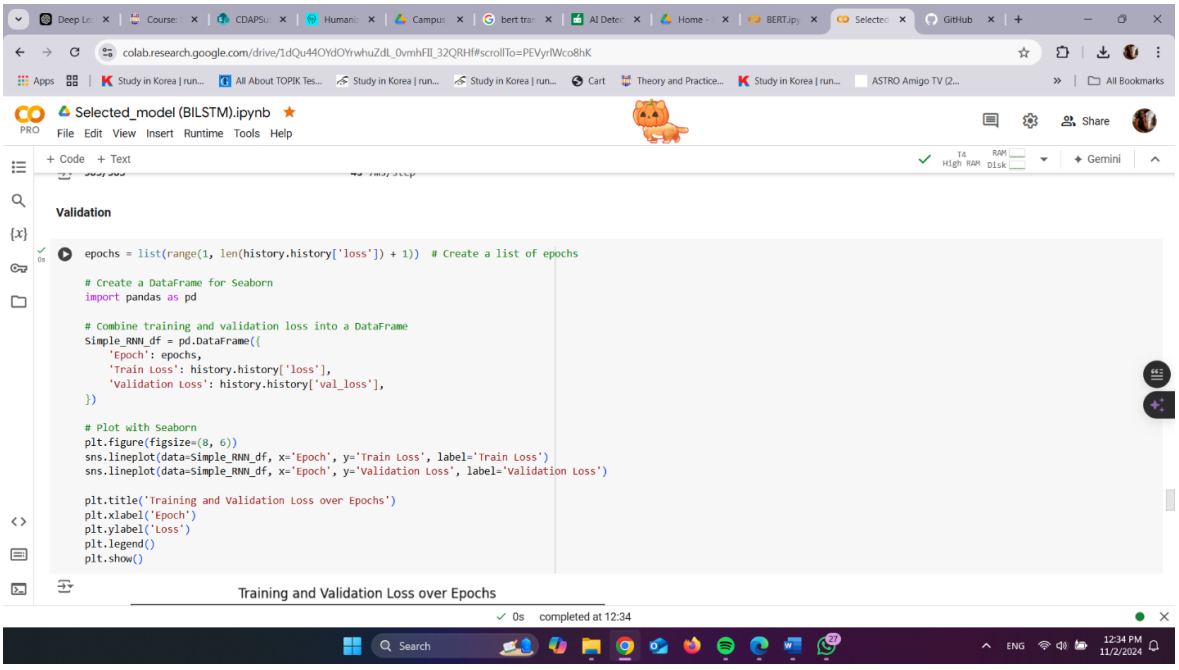

Figure 59 Model training & Validation 2

Training and validation accuracy and loss were plotted to monitor performance over epochs.

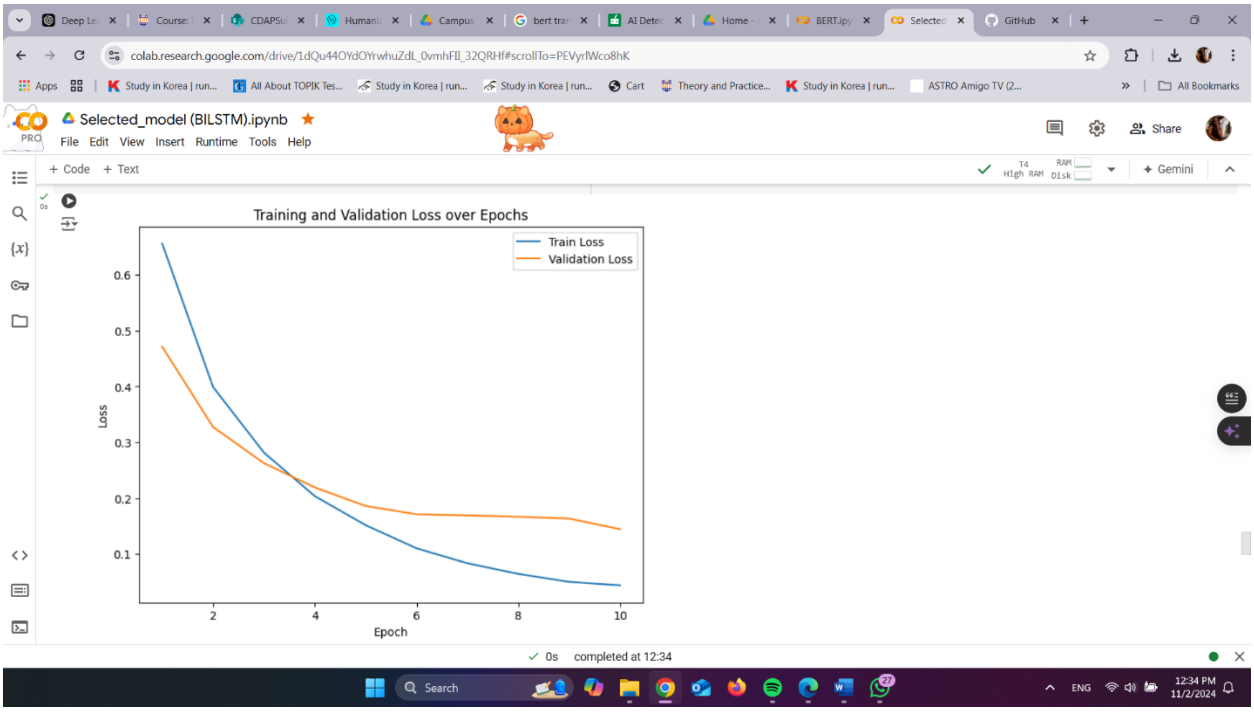

Figure 60 Training & Validation loss over Epochs

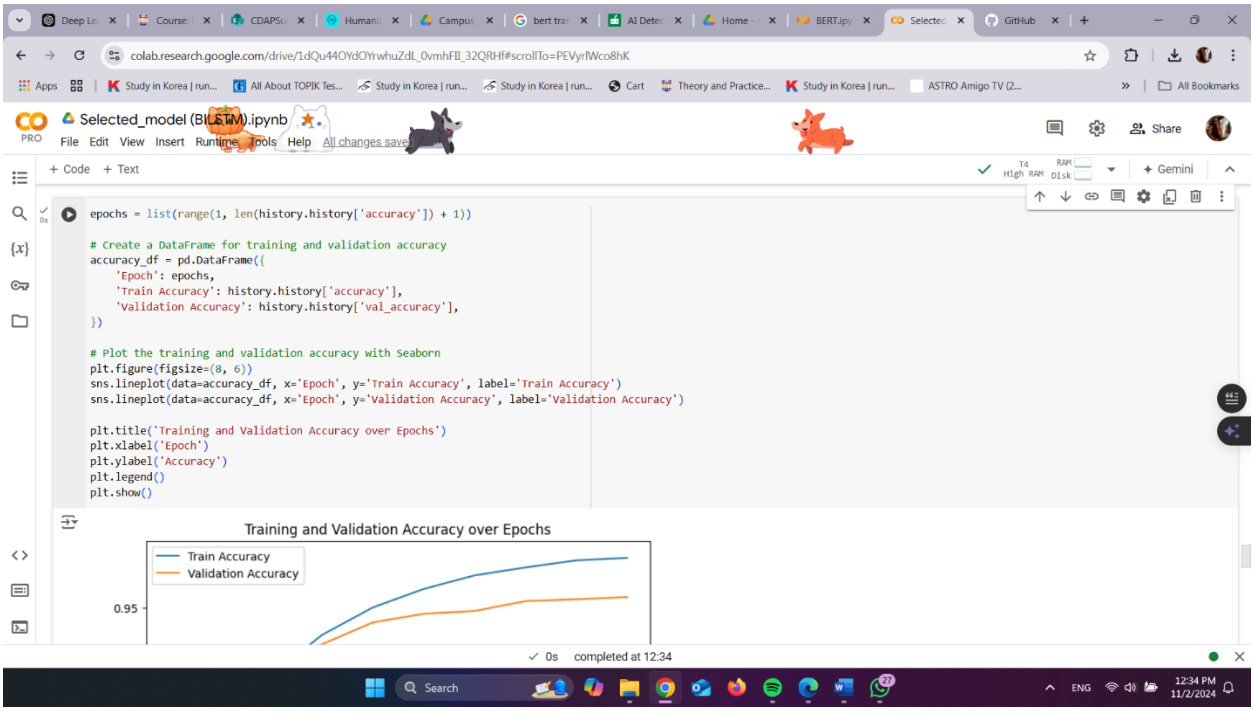

Figure 61 Training & Validation loss over Epochs 1

Figure 62 Training & Validation accuracy over Epochs

4.7 API Implementation

Implemented a Restful API for our hate speech detection system using Flask, which facilitates seamless communication between the client and server. This lightweight framework allows us to efficiently handle incoming text data, process it through our trained BiLSTM model, and return real-time predictions regarding hate speech classification

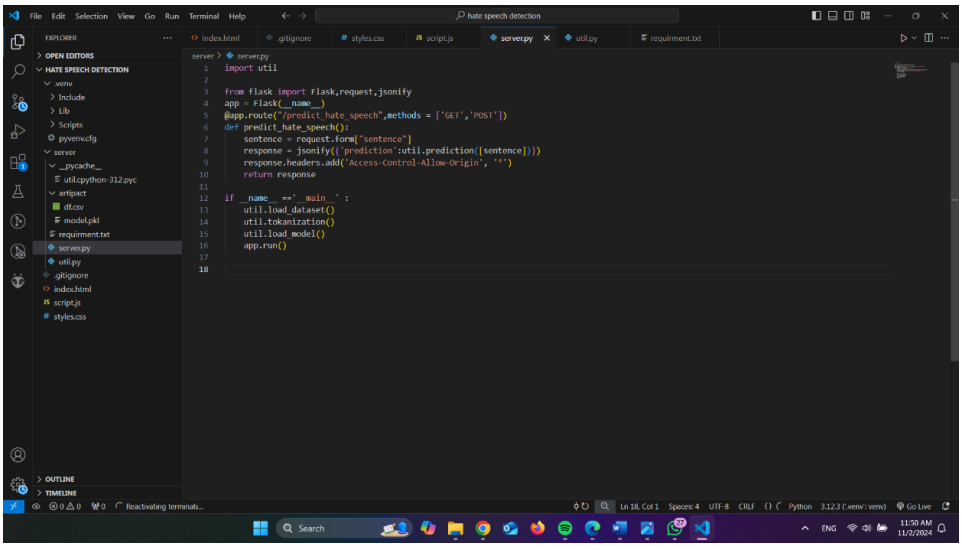

Figure 63 API implementation

5. TESTING AND IMPLEMENTATION

5.1 Testing

It is important to ensure the reliability and accuracy of our hate speech detection system. The various models, before being on boarded on our platform, were duly tested with respect to their performances. There was a wide variety in expression, tone, and context that could be evidenced from the dataset used, which contained multiple types of hate speech and neutral text samples.

We divided this dataset into training, test, and validation datasets to effectively assess the performance of the models. Cross-validation techniques established the repeatability and generalization performance of the models across subsets of data. Since real-world scenarios usually involve replication, testing models on explicit, implicit forms of hate speech, along with neutral or ambiguous linguistic variations of text samples, we have rigorously tested the modeled versions in this regard. Over the detailed process of testing, we addressed a host of challenges, such as over fitting, and that is why tuning hyper parameters was necessary to ensure optimized performance of the model. The result of paying off was that it yielded a system that could classify both hate speech and neutral text found in our dataset with a high degree of accuracy.

I tested this using non-hate speech sentences, explicit and implicit hate speech tests as well

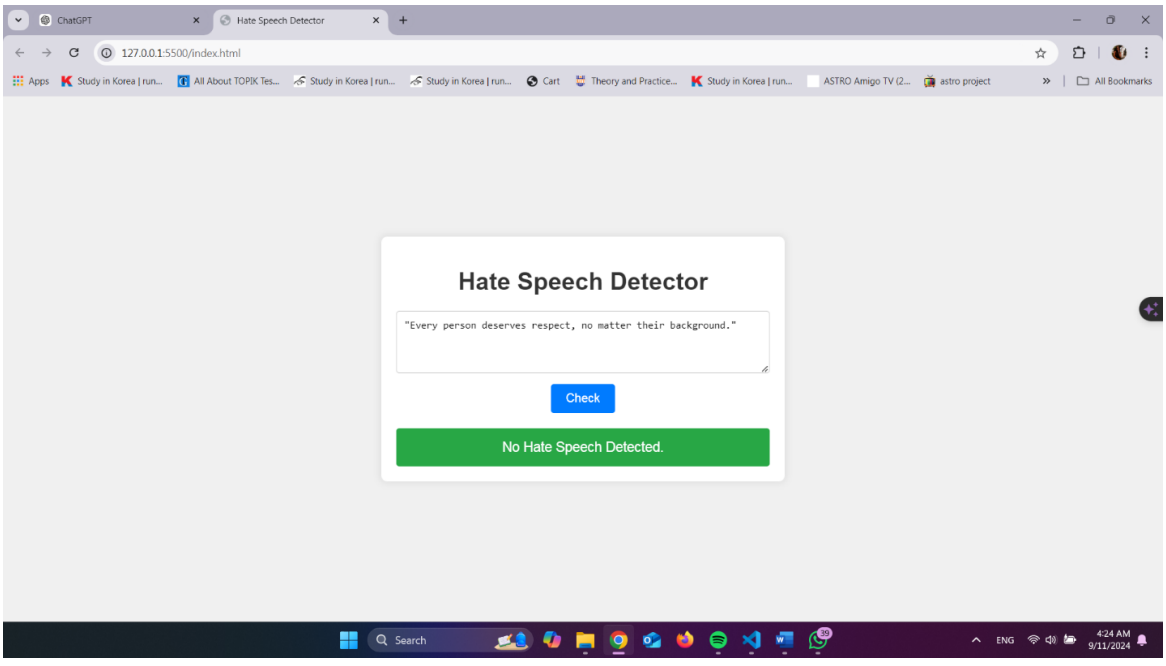

Figure 64 Testing

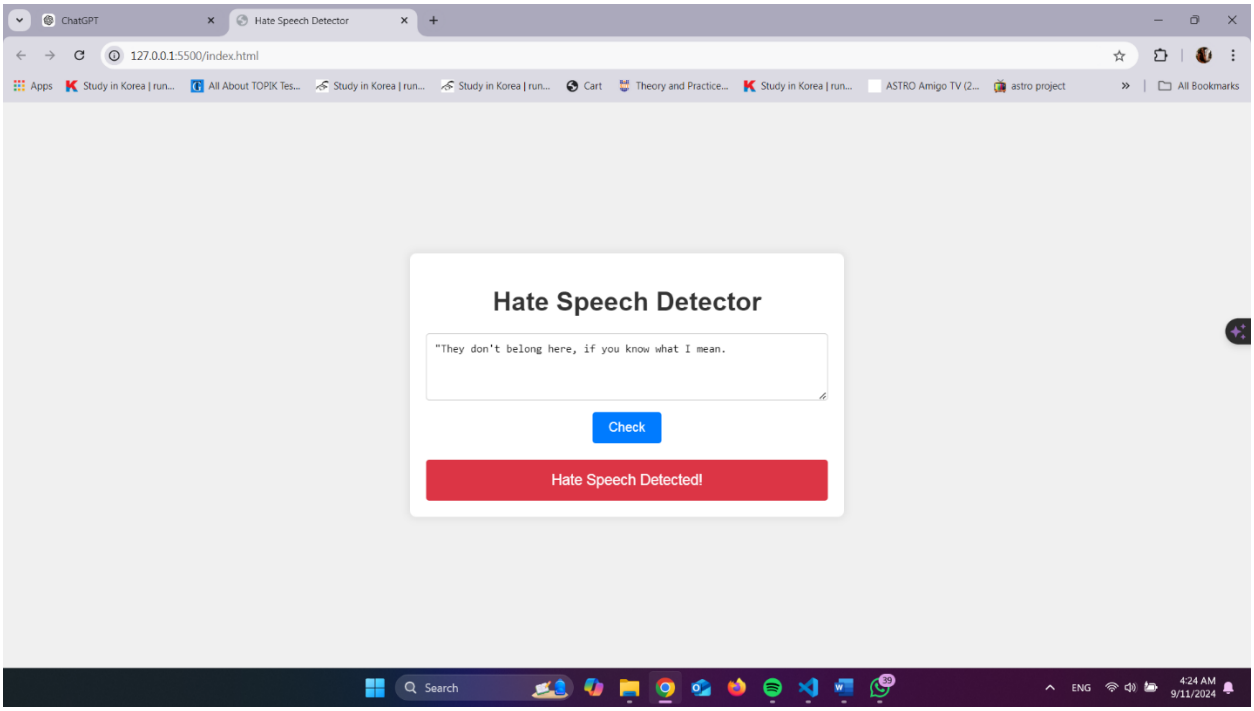

Figure 65 Testing 1

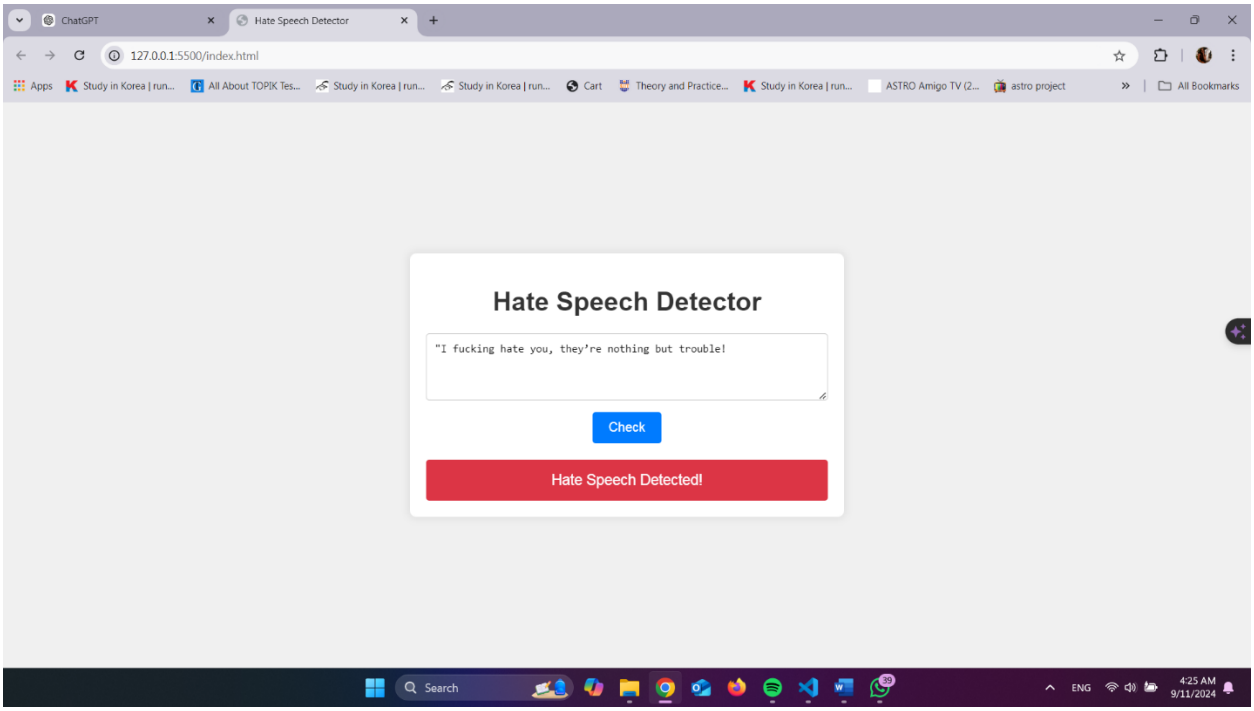

Figure 66 Testing 2

5.1.1 Unit Testing

We tested this through extensive unit testing. This approach allowed us to pinpoint any potential issues and improve the overall accuracy of the predictive models. We thoroughly examined the machine learning models and diagnostic reporting features to confirm their functionality and resolve any issues. By thoroughly testing. We ensured the system's precision and reliability in recognizing hate speech. For each unit, we discuss the testing strategies we employed and the outcomes we obtained.

Name of the Unit	Evaluation	Results
Predefined Models	Manual evaluation	Evaluation of model predictions. Evaluation of accuracy and performance
Custom-Built Models	Manual evaluation	Evaluation of model predictions. Evaluation of accuracy and performance
Data Preprocessing	Manual evaluation	Examination of data preprocessing steps Verification of data integrity and consistency.
Feature Engineering	Manual evaluation	Examination of feature engineering techniques. Verification of feature relevance and effectiveness.

Figure 67 Unit Testing

5.1.2 Performance Testing

This testing phase addressed several topics, 38 such as resource usage, stability, scalability, and responsiveness. To ensure the system complied with accepted industry standards, we assessed its capacity to handle various loads and scenarios. Performance testing confirmed our machine

learning-based system's capacity to function well in real-world educational settings by offering crucial insights into the system's reliability and effectiveness.

5.2 Implementation

After training lots of models I selected the model that have high accuracy. The selected model is the BiLSTM model. And the word embedding method that I selected is the customized word2vec model.

Customized word2vec model

It is necessary to employ a customized Word2Vec model with the purpose of augmenting the performance of hate speech detection systems. While generic, pre-trained embeddings are available, the customized model will be specifically trained on the domain-specific corpus regarding hate speech and related text. The customization would let the model grasp the contextually relevant word relationships and nuances specific to hate speech language, such as slang, coded language, or frequently used offensive terms. By customizing the word embeddings on the hate speech dataset, the model will more completely grasp subtle variations in word meanings, particularly when the context involves masked offensive language with sarcasm or implicit expressions. A tailor-made Word2Vec model also improves the representation of words that are rarely used or depend on context, which are important in distinguishing between hate speech and neutral language. This means not only an improvement of the feature extraction accuracy but also an increase in the robustness of the classification performance. The result will be a hate speech detection system that is more sensitive to linguistic subtleties and more effective in realistic applications.

Bi-LSTM model

1. Contextual Understanding:

Bi-directionality: The BiLSTM models process the sequences in both left-to-right and right-to-left directions, thus allowing them to capture the word with respect to both the previous as well as the subsequent context. This comprehensive contextual understanding is of utmost importance in hate speech, whose meaning often relies heavily on the surrounding words.

Nuance Recognition: The model has managed to pick up subtle variations in languages, such as sarcasm or indirect references, which are some of the more common stumbling blocks in the detection of hate speech.

2. Better Performance:

Capturing Long-Term Dependencies: BiLSTM learns the relationship of words that are placed far apart in a sentence, ensuring no contextual information gets lost. This leads to better performance in situations involving long statements or complicated ones.

Less Misclassification: As it keeps in consideration the entire context of every word, BiLSTM minimizes the chances of misclassifying any ambiguous or subtlety-marked text that mostly confuses the simpler models.

3. Handling Complex Language Structure:

Sarcasm and Implicit Meaning: The BiLSTM models become more efficient in handling sarcasm, coded language, and implicit modes of hate speech that demand deep linguistic contextual understanding.

Semantic Similarity: The model picks up semantic relations between words better, thus amplifying the dissimilarity in representation for the target variable of hate speech against neutral content.

4. Integration of Word Embeddings:

Deep Word Representations: Pre-trained word embeddings like Word2Vec, BiLSTM models become cognizant of the more profound word meaning and its relationship. In turn, this improves their performance by adding contextual features useful for classification.

Domain adaptation training: The fine-tuning of the embeddings based on hate speech data makes the model sensitive to domain-specific terms and patterns that may regularly appear in such language; hence, it improves performance.

5. Generalization and Robustness:

Flexibility: This model works well with explicit or implicit forms of hate speech, no matter what writing style or contextual framework it comes under. Thus, it becomes suitable for real-world applications.

Cross-Dataset Performance: Besides ensuring good general performance across datasets, it means that models will perform reliably on both diverse and unseen hate speech content.

Implementing API

The deployment of our model for hate speech detection based on a Restful API using Flask, a light and efficient Python web framework. With the help of Flask, it able to expose a simple, robust interface to our model. The API consumes the text data input, feeds the same into the trained model, and returns predictions classifying the text as hate speech, offensive, or neutral.

The implementation brings about good communication between the client and server through its fast and reliable responses. Further, Flask simplicity and flexibility go well with developing and scaling API when needed. Structuring our endpoints to handle various requests optimizes the API for real-time detection, making our hate speech detection system easily accessible and deployable across different platforms.

6. RESULT AND DISCUSSION

6.1 Result

In our hate speech detection system, we evaluated various classical machine learning models alongside deep learning architectures, including LSTM, BiLSTM, RNN, and BERT, to determine the most effective approach for accurately identifying hate speech in text. Initially, we explored classical models such as Logistic Regression, Support Vector Machines, and Random Forest, but they did not yield satisfactory results in terms of accuracy.

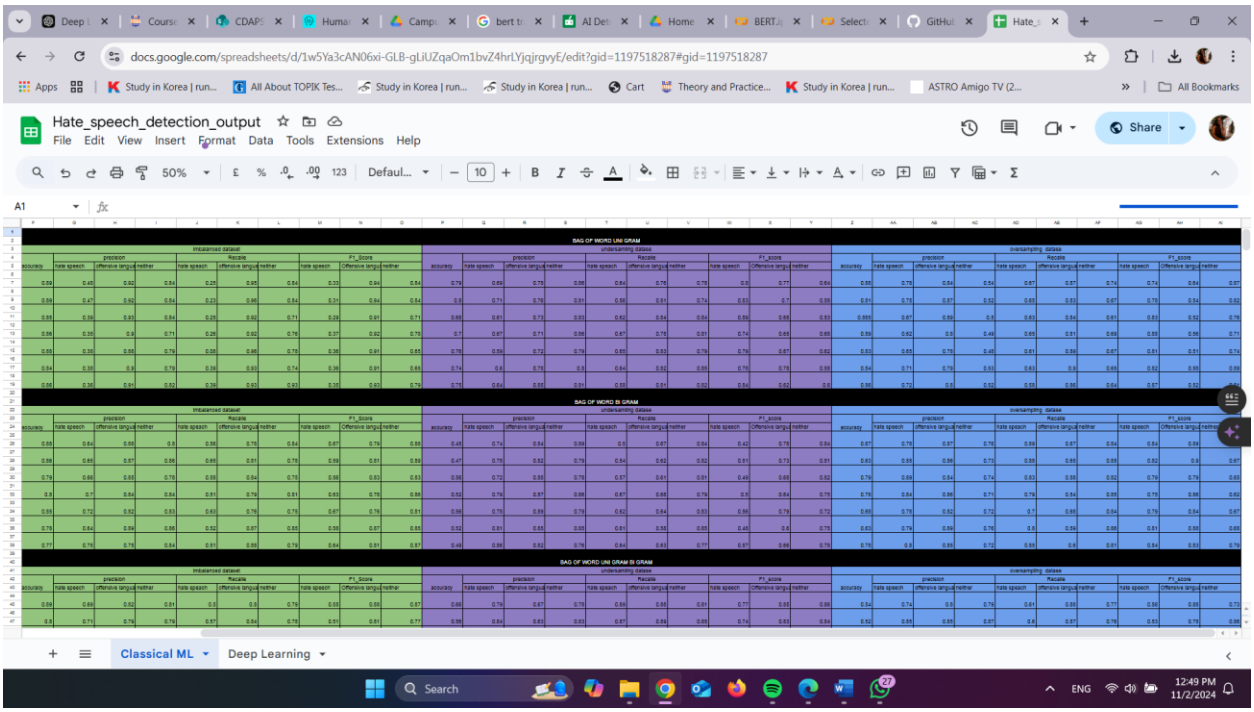

Figure 68 Hate Speech detection output

We then transitioned to deep learning models, experimenting with LSTM and RNN, which improved performance but still fell short of our expectations. Upon implementing BERT, we achieved significant advancements.

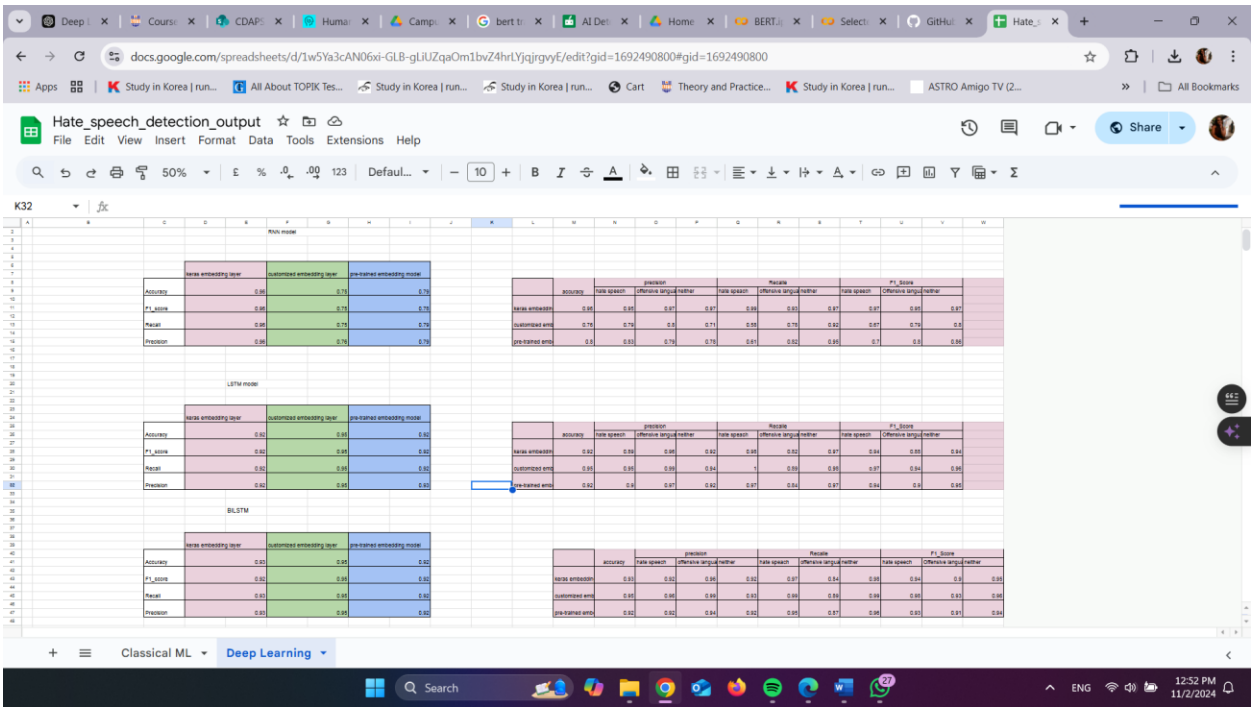

Figure 69 Hate Speech detection output 1

However, after conducting thorough fine-tuning, we found that the BiLSTM model delivered the highest accuracy, reaching an impressive 95%.

To enhance the model's performance further, we utilized a customized Word2Vec model tailored to our specific hate speech dataset. This integration allowed the BiLSTM model to leverage contextually relevant word embeddings, capturing nuanced language patterns and improving the model's ability to differentiate between hate speech and neutral text effectively.

The resulting system, powered by the optimized BiLSTM and customized Word2Vec embeddings, demonstrates robust performance and reliability in detecting hate speech, contributing to a safer and more respectful online environment. Our commitment to leveraging advanced deep learning technologies reflects our dedication to addressing this pressing issue with effective solutions.

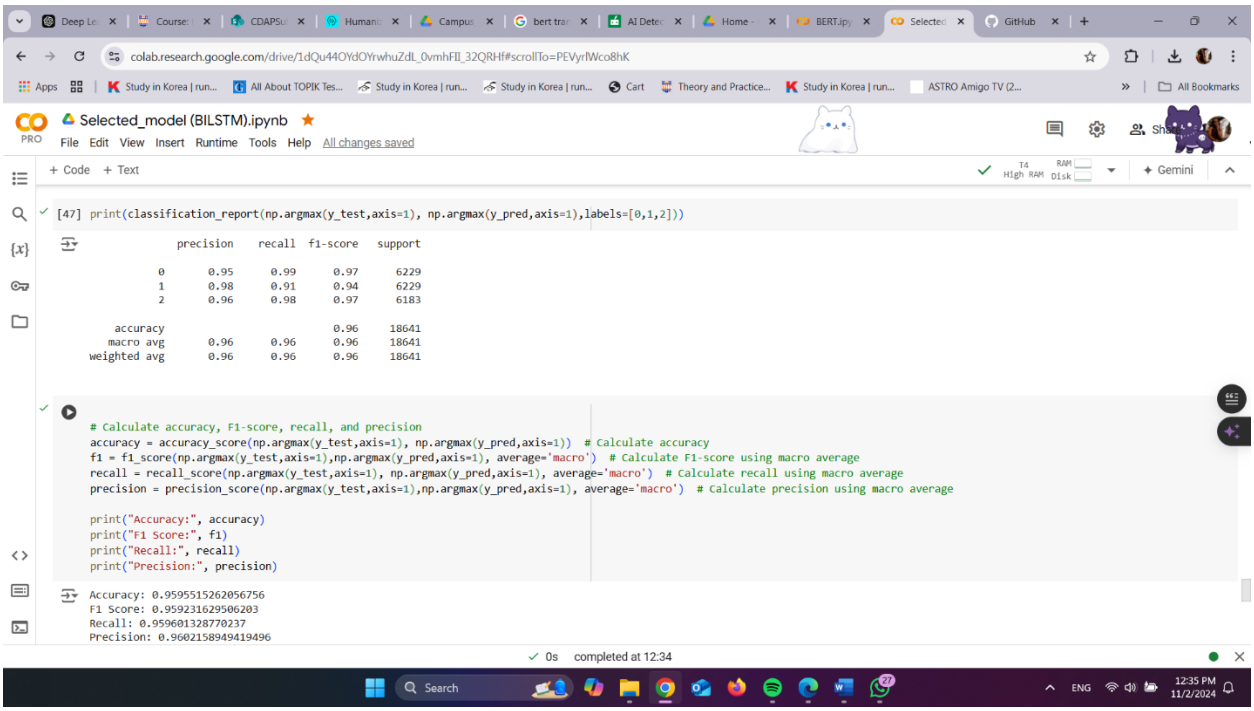

Figure 70 Optimized BiLSTM and customized Word2Vec embedding

6.2 Discussion

The results from our experiments highlight the effectiveness of Bidirectional Long Short-Term Memory (BiLSTM) networks for hate speech detection, with an emphasis on classifying offensive language. In this study, a BiLSTM model was applied to classify text data into three categories: "hate speech," "offensive," and "neither." The model demonstrated a high level of sensitivity in distinguishing offensive language, as evidenced by the predicted output and confidence levels for each label.

For instance, phrases containing explicit offensive language such as "you bitch" were accurately classified as "offensive," while more neutral statements like "you're very good" were correctly identified as "neither." This accuracy is attributed to the BiLSTM's ability to capture sequential patterns in language, allowing it to distinguish context and sentiment effectively.

The model's interpretability was further enhanced by the use of LIME (Local Interpretable Model-Agnostic Explanations), which provided insights into the features contributing to each classification. For example, when analyzing the phrase "You're not even human; you're a mistake of nature," LIME identified words such as "mistake" and "human" as strong indicators of offensive intent. This visualization helps in understanding the model's decision-making process and validates that key offensive terms significantly impact the classification.

Despite the high accuracy, the BiLSTM model occasionally misclassified phrases. For instance, positive phrases like "you are an amazing person" were mistakenly flagged as "hate speech" with a high probability (0.97). This misclassification highlights a limitation in the model's sensitivity to nuanced, positive language, which could be due to imbalances in the training dataset or over fitting to specific offensive terms without adequate context.

Overall, the BiLSTM model proves to be an effective tool for hate speech detection in real-world applications, especially in identifying harmful and offensive language. However, challenges remain in distinguishing false positives where innocuous phrases are misclassified. Future work could involve refining the model by augmenting the dataset with more diverse samples and applying techniques such as attention mechanisms to better capture contextual nuances, improving its interpretability and robustness.

In comparison to simpler and more complex models, BiLSTM strikes a balance in computational efficiency and accuracy. While more complex models like Transformer-based architectures may yield higher precision, BiLSTM is particularly well-suited for applications requiring real-time analysis on limited computational resources. Consequently, this model is a viable choice for practical implementations in systems that monitor social media or online forums for hate speech, offering an effective balance between performance and computational cost.

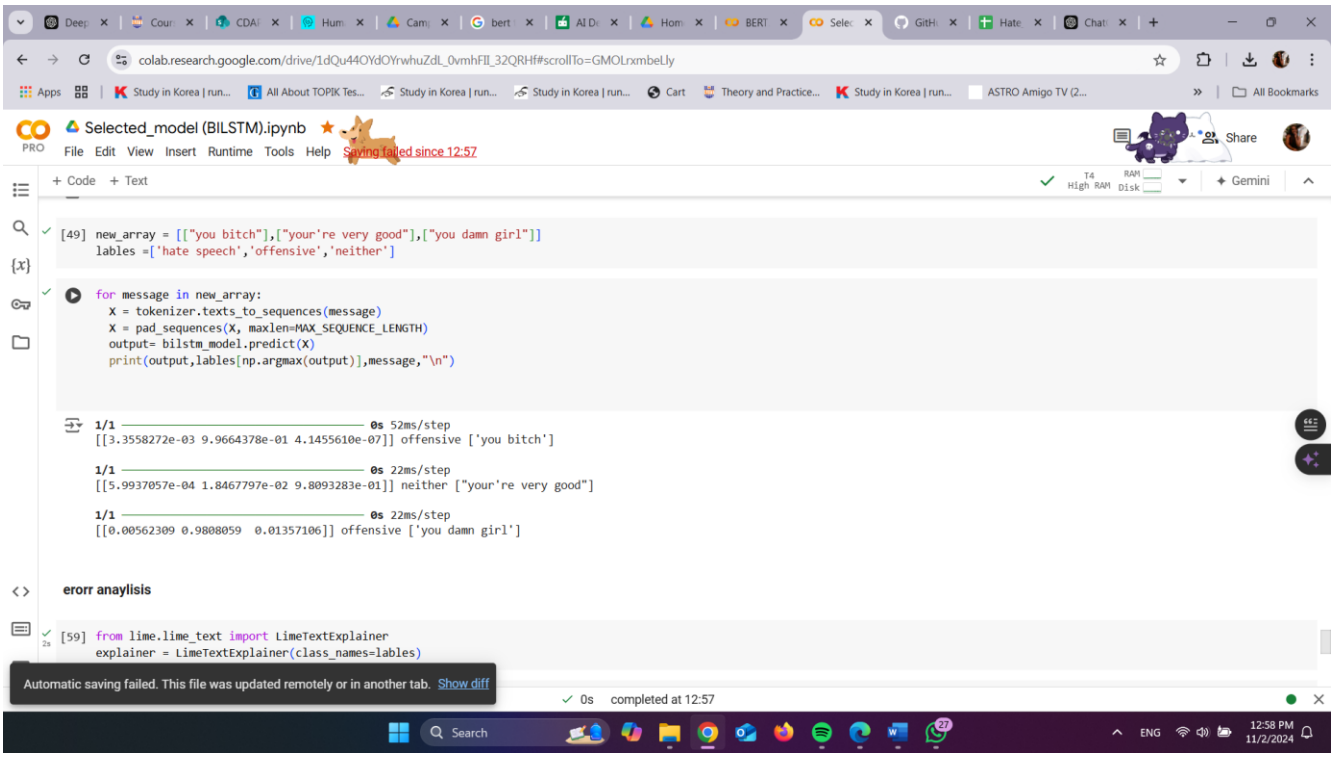

Figure 71 Effectiveness of BiLSTM model

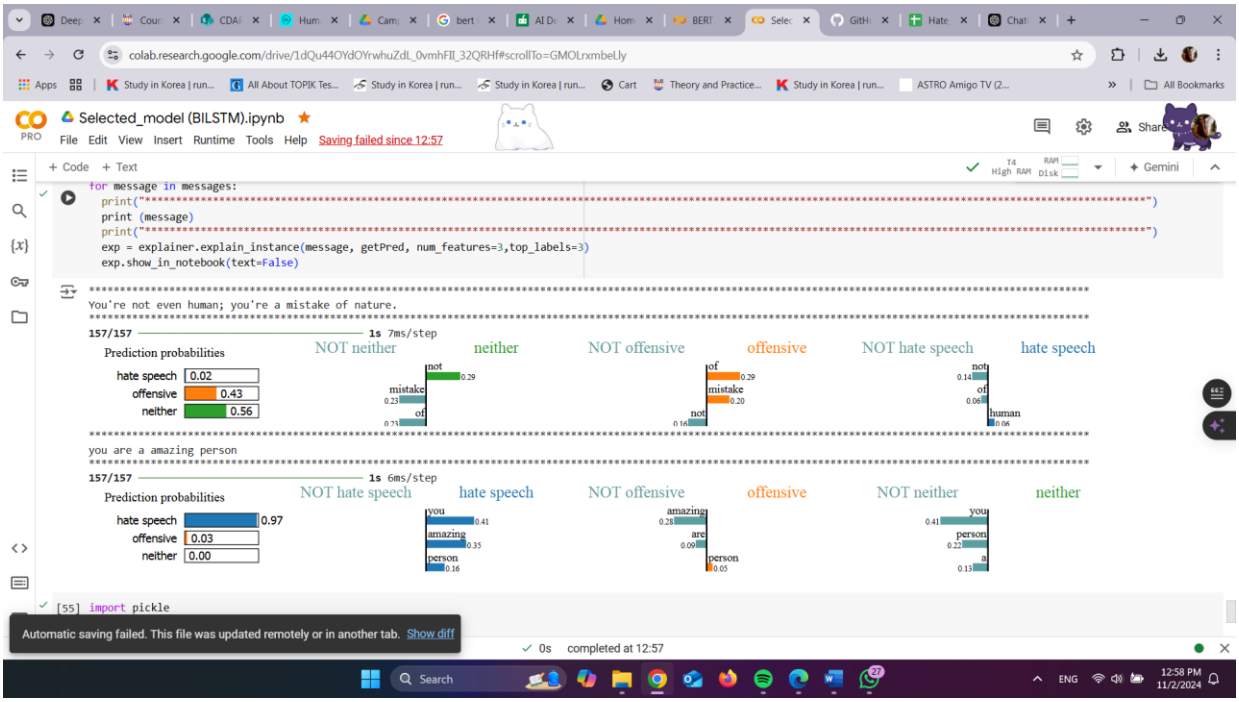

Figure 72 Effectiveness of BiLSTM model 1

7. LIMITATION SOULUTION AND FUTURE WOEK

7.1 Limitations

- Imbalanced Dataset: The dataset used in this work is imbalanced for classes like "hate speech," "offensive," and "neither," which may give bias to the classification performance.
- False Positives in Positive Sentences: Sometimes, the BiLSTM model fails to classify positive sentences as "offensive" or "hate speech," which could indicate some difficulties of treating nuanced, context-dependent language.
- Over fitting Certain Words: The model may over fit on some general offensive words without understanding context and misclassify. Lack of deeper context understanding, especially when it comes to highlighting subtle differences between offensive and non-offensive language, especially in longer sentences.
- Computational Cost: Deep learning models, such as BiLSTM, require a lot of computational resources for the training of the models. This can be a factor in scalability.

7.2 Solutions

- Data Augmentation: Leverage deep data augmentation techniques such as generating synthetic text data to balance out the class distribution rather than mere oversampling and under-sampling techniques.
- Hybrid Models: Combine BiLSTM with attention mechanisms or Transformer-based architecture to enhance the classifying to be more context sensitive in order to reduce false positives for positive statements.

7.3 Future Work

- Context-Aware Ensemble Model: To develop an ensemble model that combines different models-which can be BiLSTM, CNN, or classical ML models-in order to improve classification by leveraging complementary strengths in different classifiers.

- Fine-Tune Class-Specific Thresholds: Perform experimentation with class-specific decision thresholds in order to decrease misclassification rates, especially for "offensive" and "neither" classes.
- Implementing Explain ability Techniques: Do more research on Explainable AI techniques to get more insight from the model's decisions and hence allow the users to interpret the classification outputs.

CONCLUSION

This work presents a hate speech detection model suitable for sensitive real-world applications. Sectors of especial interest in this work pertain to military news and politics. Various machine learning and deep learning models, which ranged from classical approaches to state-of-the-art deep learning architectures such as RNN, CNN, LSTM, and BiLSTM, were evaluated. Amongst these, the best performance was obtained using the BILSTM model. The BiLSTM performed very well with an accuracy of 95%, somewhat balanced for the categories of hate speech, offensive, and neutral, which is further attested to by the superior F1 score of this model against the rest.

Our methodology used a variation of TF-IDF, Bag of Words, and Word2Vec embeddings for feature extraction. To counter the problem of an imbalanced dataset, we tried both oversampling and under-sampling. Despite this, there is an evident deficiency in the model, with major defects including the false positives from misclassified positive statements, over fitting into terms, and computation, which is customary in deep learning models, being extremely resource-intensive.

The outcome of this study has shown that BiLSTM can serve well for hate speech detection. However, there is still room for future improvements that could consider .Hybrid models with the incorporation of BiLSTM with attention mechanisms to show more sensitivity toward subtle language. Continuous model retraining with updated vocabulary and dynamic vocabulary expansion would also help adapt to evolving language trends in online discourse. Explanation techniques and a feedback loop in place will further help the deployment to be more explainable and user-oriented; this can enforce the model to be much more practical and interpretive.

In a nutshell, this work contributes a strong framework in hate speech detection with very promising performance to real-world applications. However, overcoming the present limitations and enhancing adaptiveness and scalability of the model should be emphasized in future efforts to extend the applications to a broader coverage and make it resilient against diverse linguistic or domain variations.

REFERENCES

[1] F. Tahosin, P. C. Shill, and M. G. R. Alam, "Multi-modal Hate Speech Detection using Machine Learning," in *2021 IEEE International Conference on Big Data (Big Data)*, Dec. 2021, pp. 1–10, doi: 10.1109/BigData52589.2021.9671955.

[2] S. S. Alaoui, Y. Farhaoui, and B. Aksasse, "Hate Speech Detection Using Text Mining and Machine Learning," *International Journal of Decision Support System Technology*, vol. 14, no. 1, pp. 330–348, 2022, doi: 10.4018/IJDSST.286680.

[3] N. B. Ocampo, E. Sviridova, E. Cabrio, and S. Villata, "An In-depth Analysis of Implicit and Subtle Hate Speech Messages," in *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, Dubrovnik, France, May 2023, pp. 1997–2013, doi: 10.18653/v1/2023.eacl-main.147.

[4] A. Natha, B. Behera, D. Mishra, S. R. Sahu, and S. Mohapatra, "A Comparative Analysis of Machine Learning Models Used for Hate Speech Detection of Odia Language," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 16s, pp. 464-476, 2024.

[5] A. Toktarova, D. Syrlybay, B. Myrzakhmetova, G. Anuarbekova, G. Rakhimbayeva, B. Zhylanbaeva, N. Suieuova, and M. Kerimbekov, "Hate Speech Detection in Social Networks using Machine Learning and Deep Learning Methods," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 14, no. 5, pp. 396-405, 2023.

[6] A. Abraham, A. J. Kolanchery, A. A. Kanjookaran, B. T. Jose, and D. PM, "Hate Speech Detection in Twitter Using Different Models," *ITM Web of Conferences*, vol. 56, no. 04007, pp. 1-6, 2023.

[7] S. Das, K. Bhattacharyya, and S. Sarkar, "Hate Speech Detection from Social Media Using One Dimensional CNN Coupled with Global Vector," *European Chemical Bulletin*, vol. 12, no. SI10, pp. 691-699, July 2023, doi: 10.48047/ecb/2023.12.si10.0081.