Techniques for Text Classification and Text Generation: Enhanced Online Sexism Detection and Template driven Wikipedia Article Generation

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computational Linguistics by Research

by

Jayant Panwar
2019114013
jayant.panwar@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, India

April 2024

International Institute of Information Technology Hyderabad
Hyderabad, India

CERTIFICATE

This is to certify that work presented in this thesis titled ***Techniques for Text Classification and Text Generation: Enhanced Online Sexism Detection and Template driven Wikipedia Article Generation*** by **Jayant Panwar** has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Advisor: Dr. Radhika Mamidi

Dedicated to Mummy, Papa, and NaniMa

Acknowledgments

I am immensely grateful to my advisor, Dr. Radhika Mamidi, for her unwavering guidance and resolute support throughout my academic journey. Her expertise, encouragement, and commitment have been instrumental in shaping my academic and personal growth. I am truly fortunate to have had such a dedicated mentor who not only shared her knowledge but also inspired confidence in me. Her insightful feedback and constructive criticism have been invaluable, motivating me to strive for excellence. I am thankful for the freedom she granted me when discovering new problems and deeply grateful for the countless hours she has invested in mentoring me, providing invaluable insights that have significantly contributed to my development. I truly appreciate her dedication, patience, and belief in my abilities. This academic endeavour would not have been the same without her guidance, and I am sincerely grateful for her mentorship.

I would also like to thank Prof. Vasudeva Verma, Vibha ma'am, and Kasyap sir for letting me be a part of the IndicWiki project. This project helped me to develop important skills both as a researcher and as a mentor. The work done in this thesis would not be the same without the IndicWiki project. I would also like to extend my thanks to all the CLD professors who have taught me through the years: Prof. Dipti Misra, Prof. Aditi Mukherjee, Dr. Manish Shrivastava, Dr. Anil Vuppala, and Dr. Soma Paul. I was able to build a strong foundation of linguistic knowledge thanks to the concepts they have taught me in great detail over the years.

I am deeply indebted to my friends Dayitva, Debayan, Chirag, Akshett, Prajneya, Konda, Shaurya, Suyash, Harsh, Aaradhya, and Akhilesh Anna. Without your help, motivation, and support all along the way, I would not have developed into the person I am today. All those evening tea walks to JC with you all were the highlight of any day. I would like to extend a special thanks to all my wonderful seniors as well, Pranav, Arnav, Mehtab, and Anubhav, who were always there to guide me and help me out whenever I needed it.

I am extremely grateful for my amazing family. Without my parents' support and belief in me, I would not be the person I am today. Special thanks to NaniMa, cousins, and relatives as well for taking care of me throughout my childhood, especially during the COVID lockdown when times were very bad. I owe a great debt of gratitude to all of them.

Lastly, I would like to thank IIIT-H for being such an important presence in my life. It has given me priceless things that I will take from here for life, whether it be my career, my belief in myself, or friends and connections to cherish for a lifetime.

Abstract

Text classification and generation are linchpins in the explosive growth of the internet, seamlessly organizing and generating vast volumes of textual information. These pivotal techniques not only enhance Natural Language Processing but contribute indispensably to the dynamic evolution of online communication and knowledge dissemination in the modern world. This is why we have selected Sexism Detection and automated Wikipedia article generation in Indian languages as our topics for researching techniques in text classification and text generation, respectively. Detecting sexism in text promotes inclusivity and gender equality online, whereas generating Wikipedia articles in Indian languages enhances the accessibility of knowledge to diverse communities.

With the rapid growth of online communication on social media platforms, there has also been an increase in the amount of hate speech, especially in terms of sexist language online. The proliferation of such sexist language has an impact on the mental health and well-being of the users and, hence, underscores the need for automated systems to detect and flag such pieces of text online. In this thesis, we explore the effectiveness of both conventional machine-learning techniques and different modern techniques applied to BERT-based classifiers for detecting sexist text in the EDOS shared task. The results show that the performance of the baseline conventional classifiers and BERT-based classifiers is greatly improved when techniques like Data Resampling, Data Augmentation, Domain Adaptive Pre-training, and Learning Rate Scheduling are implemented.

The surge of data on the internet has also seen the rise of a knowledge giant, Wikipedia. In today's world, it is seen as one of the primary online sources for the distribution and consumption of free knowledge. Wikipedia users collaborate in a structured and organized manner to publish and update articles on numerous topics. English Wikipedia ranks at the top when it comes to the amount of information available for any language ($>$ 6.7 million articles). However, the amount of information available in Indian languages is lagging far behind. The same article in Hindi may be vastly different from its English version and generally contains less information. This poses a problem for native Indian language speakers who are not proficient in English. Therefore, having the same amount of information in Indian languages will help promote knowledge among such speakers. Publishing the articles manually, which has been the status quo for decades, is a time-consuming process. To get the amount of information in native Indian languages up-to-speed with the amount of information in English, automating the whole article generation process is the best option. In this thesis, we present a stage-wise approach ranging

from Data Collection to Summarization and Translation, and finally culminating with Template Creation. This approach ensures the efficient generation of a large amount of content in Hindi Wikipedia in less time.

All in all, this work aims to develop automated systems to foster a safe digital space for users of social media platforms and propose a stage-wise approach for the efficient generation of Wiki articles, hence promoting the dissemination of knowledge for native Indian language speakers.

Contents

List of Figures

List of Tables

Abbreviations

BERT	Bidirectional Encoder Representations from Transformers
CNN	Convolutional Neural Network
DAP	Domain Adaptive Pretraining
DAug	Data Augmentation
EDA	Easy Data Augmentation
EDOS	Explainable Detection of Online Sexism
FRES	Flesch Reading-Ease Score
LAMBADA	Language-Model-Based Data Augmentation
LDA	Latent Dirichlet Allocation
LeR	Learning Rate Scheduling
MLM	Masked Language Modelling
NER	Named Entity Recognition
NEs	Named Entities
NHS	National Health Service
NLP	Natural Language Processing
NPOV	Neutral Point of View
NSP	Next Sentence Prediction
POS	Part-of-speech
RD	Random Deletion
RDF	Resource Description Framework
RI	Random Insertion
RNN	Recurrent Neural Network
RS	Random Swapping
SPARQL	Simple Protocol and RDF Query Language
SR	Synonym Replacement
SVMs	Support Vector Machines
TF-IDF	Term Frequency-Inverse Document Frequency
XML	eXtensible Markup Language

Chapter 1

Introduction

In today's digitally driven world, the realms of text classification and generation play pivotal roles in shaping how we access and comprehend information. The sheer volume of textual data available necessitates advanced techniques to sift through and organize this information effectively. Text classification empowers automated systems to discern and categorize content, while text generation streamlines the creation of coherent and informative narratives. These techniques not only enhance efficiency but also contribute significantly to the creation and dissemination of knowledge.

With the increase in the amount of data, this digital age has also witnessed an alarming surge in hate speech, notably in the form of sexism, prevalent in online platforms. As instances of targeted attacks and victimization rise, there is an urgent need for robust automated applications to identify and curb such harmful content. Developing accurate and efficient text classification models can act as a potent tool in combating online harassment and fostering a safer digital environment. This thesis addresses this pressing need by exploring innovative techniques for text classification, specifically focusing on the detection and mitigation of sexism in online social platforms.

Amidst this staggering influx of data on the internet, Wikipedia has emerged as the quintessential global encyclopedia, shaping the way individuals seek and acquire knowledge. However, this wealth of information is not uniformly distributed across languages. While English-language Wikipedia thrives, articles in Indian languages remain comparatively sparse and outnumbered. This discrepancy is especially noteworthy given Wikipedia's status as a primary information source for a multitude of individuals and the fact that search engines often leverage it to provide quick answers. This thesis thus tackles the challenge of template-driven Wikipedia article generation in Indian languages, aspiring to bridge the existing gap and foster a more balanced representation of knowledge across linguistic domains.

1.1 Classifying Sexism

Building models for classifying sexism is imperative in combating the escalating wave of online harassment. As instances of gender-based discrimination permeate digital spaces, automated models play a pivotal role in swiftly identifying and addressing such harmful content. These models not only

contribute to a safer online environment but also empower platforms to proactively curb sexist language and promote inclusivity.

In this thesis, we take a look at the EDOS shared task at SemEval'23 [1]. The task involves building classifiers trained on the EDOS task dataset, which can correctly detect sexism in a piece of text and also which category and sub-category of sexism is present in the text. We study two different approaches to tackle this shared task: conventional machine learning and transformer-based deep learning. By conventional techniques, we refer to simple feature extraction and statistical classifiers, which have been the standard of text classification before the advent of the more recent and advanced deep learning-based techniques. Transformer-based deep learning approach refers to pre-trained language models that make use of the Transformer [2], essentially BERT [3] in this thesis' context. Even though we study different approaches, the crux of the work is to experiment with techniques apart from the architecture alone, like Data Resampling, Data Augmentation, Domain Adaptation, and Learning Rate scheduling.

1.2 Generating Indian language Wiki articles

As underscored already, the amount of information available in Indian languages on Wikipedia is lagging far behind the extent of information available on English Wikipedia. This is counter-intuitive, considering India's growth as a digital nation. A large populace of Indian people surfing the web and consuming knowledge from Wikipedia still are not as adept at understanding English as one might expect. For the promotion of knowledge among such individuals, it is imperative to make their native language's Wikipedia rich in information, and since generating articles manually is a costly approach because of the time spent doing so, there is a need for automated techniques to generate Wiki articles. Additionally, Wikipedia also serves as a cornerstone in the development and advancement of large language models, providing both data and knowledge essential for their training, fine-tuning, and evaluation processes [3], [4], [5]. Wikipedia provides vast amounts of text data in multiple languages, which serves as valuable training material for large language models. This diverse corpus helps in training models to understand and generate coherent text across various topics and languages.

Therefore, in order to bridge the vast knowledge gap for Indian languages in a timely manner and to aid in the development of better language models, we present a template-driven approach to generate Wikipedia articles in Hindi. The study involves choosing a target domain and collecting data for the same. The scraped data is usually unstructured because even though it is generated by humans, it is meant for the consumption of humans only and not automated systems. Before applying any automated technique, it is important to structure this data appropriately. Once the data has been structured, it needs to be translated and transliterated adequately to the target Indian language. A template can then be drafted wherein the data would fill the empty spaces of the template's sections and sub-sections. This approach ensures the prompt generation of thousands of Wiki articles, which, if generated manually, would have taken weeks, if not months.

1.3 Terminology

Throughout this thesis, we use certain terms according to their specific definitions. Although they are discussed in detail in their respective related chapters, it would be wise to still identify and define them beforehand:

- **Natural Language Processing (NLP)**: a field of artificial intelligence that focuses on the interaction between computers and human language, enabling machines to understand, interpret, and generate human-like text.
- **Text Classification**: the process of assigning predefined categories or labels to textual data based on its content, facilitating automated organization and analysis of large volumes of text.
- **Text Generation**: refers to the creation of human-like text by machines, often leveraging advanced language models and techniques to produce coherent and contextually relevant narratives.
- **Sexism**: refers to discriminatory or prejudiced attitudes, behaviours, or language based on a person's gender, typically targeting one gender in a way that perpetuates stereotypes or unequal treatment.
- **Conventional Machine Learning**: encompasses traditional approaches to machine learning, utilizing algorithms that do not involve Deep Neural Networks, such as Decision Trees, Support Vector Machines, or Naive Bayes.
- **Deep Learning**: a subset of machine learning that employs neural networks with multiple layers (deep neural networks) to learn and make predictions from large volumes of data, particularly effective in complex tasks of NLP like text generation, text classification, and speech recognition.
- **Transformer**: a type of deep learning model architecture that has significantly advanced natural language processing tasks, introducing attention mechanisms for efficient processing of sequential data [2].
- **BERT**: an encoder-only bidirectional representation of transformers developed by Google, excelling in understanding context and semantics, widely used for various NLP applications [3].
- **Class Imbalance**: refers to the unequal distribution of instances among different classes in a dataset, which can impact the performance of machine learning models, especially in tasks where one class is underrepresented.
- **Data Resampling**: involves adjusting the distribution of instances in a dataset, often by oversampling the minority class or undersampling the majority class, to mitigate class imbalance and improve model performance.

- **Data Augmentation**: involves artificially increasing the size of a dataset by applying various transformations or modifications to the existing data, enhancing the model's ability to generalize.
- **Pre-training**: involves training a model on a large dataset before fine-tuning it for a specific task, allowing the model to learn generic features and representations that can be beneficial for various applications and downstream tasks.
- **Domain Adaptation**: the process of adjusting a model trained on one domain to perform effectively on a different but related domain, improving its generalization to new data.
- **Learning Rate**: a critical hyperparameter in machine learning models, determining the size of the step taken during optimization. Properly setting the learning rate is crucial for achieving optimal model performance.
- **Fine-tuning**: the process of adjusting a pre-trained model on a specific task or dataset to improve its performance for a particular application, often done with a smaller, task-specific dataset.

1.4 Major Contributions

The following are the major technical contributions of the thesis:

- Enhance the performance of conventional machine learning techniques in sexism detection tasks by utilizing naive data resampling techniques.
- Introduce an ensemble of domain adaptive pre-training, data augmentation, and learning rate scheduling and demonstrate its effectiveness when applied to BERT-based models in multi-class sexism detection tasks. Furthermore, study the effect of each of the techniques individually in these multi-classification tasks.
- Propose a new template-driven approach to generate a large amount of Indian Language Wikipedia articles in a timely fashion. The articles will consequently also aid in the development of better language models for Indian Languages as Wikipedia serves as the basis for multiple state-of-the-art NLP datasets.
- Contribute automatically generated 1154 Hindi articles on the *Diseases and Disorders* category to the Hindi Wikipedia platform.

1.5 Organization of the Thesis

This thesis is organized into 7 chapters:

1. The **introductory chapter** introduces the main ideas of the work and the motivation behind them. We further highlight some important terminologies related to the work in this thesis and discuss the major technical contributions of this thesis.

2. **Chapter-2** revolves around the related academic literature of the main tasks we look at in this thesis. We discuss all the important and relevant work that has been done in the past by researchers and underscore how our work addresses issues that have not been looked at.

3. **Chapter-3** takes a detailed look at the EDOS task dataset. We analyze the important statistics of the dataset and highlight the significant class imbalance problem of this dataset. We conclude the chapter by underlining how sexism can sometimes be subjective and the differentiating line can get blurred which poses a major challenge which cannot be really countered by any technique as such.

4. **Chapter-4** highlights how some of the conventional machine learning techniques still perform competitively on simple binary classification tasks in the age of deep learning. Furthermore, we demonstrate how naive data resampling techniques can be applied to the dataset and give these conventional machine learning techniques a significant boost in their performance.

5. **Chapter-5** depicts how domain adaptive pre-training, data augmentation, learning rate scheduling, and their ensemble give a competitive edge in the performance of BERT-based classifiers. We analyze how each technique individually improves the performance of the classifier depending on the subtask.

6. **Chapter-6** proposes a template-driven approach to automatically generate Wikipedia articles in Hindi. We dive into the deep details of the approach, ranging from collecting unstructured online data to structuring and translating it and finally generating thousands of Wikipedia articles.

7. The **final chapter** of the thesis summarizes the important conclusions and contributions of this thesis. We also discuss some possible future directions for this work, which may either include working on fixing its limitations or taking upon something entirely new but related.

Chapter 2

Related Work

In this chapter, we delve into the existing body of knowledge that forms the foundation for our thesis, exploring a spectrum of related work that is based on Text Classification and Text Generation.

The chapter meticulously reviews literature related to conventional classifiers, assessing their performance in various classification tasks. Moving forward, we focus on the specific literature surrounding more recent and advanced deep learning techniques like BERT, a state-of-the-art transformer-based language model, and evaluate the effectiveness of BERT-based classifiers in diverse classification tasks. The discussion then extends to the critical domain of hate speech and sexism detection, where we explore pertinent literature addressing the challenges and advancements in identifying and mitigating offensive language in text.

Additionally, we investigate the landscape of summarization techniques for unstructured data, uncovering existing approaches and methodologies that contribute to the extraction of meaningful and concise summaries from extensive textual information in the context of Wikipedia. The chapter concludes by examining literature related to the automated generation of Wikipedia articles, shedding light on advancements and methodologies employed in automatically creating informative and coherent pieces of text. Together, this comprehensive review sets the stage for our research, identifying gaps, challenges, and opportunities in the existing literature that our thesis aims to address and contribute to.

2.1 Text Classification

As discussed in the previous chapter, Text Classification plays a vital role in the field of NLP. It has seen varied forms of developments and breakthroughs over the years. However, the general outline of each research study regarding text classification remains the same. The stages are mostly identical, i.e., data pre-processing, feature extraction, model training, and evaluation. Feature extraction and model training are the two stages where most of the studies differ. Efforts have been made over the years to develop models which excel in these stages more and more. Conventional or traditional machine learning models involve having a feature extraction stage followed by a statistical classifier. Recent advancements have replaced the role of the statistical classifier with large language models that are capable of

providing a better understanding of the language and the classification task.

One of the most important advancements in terms of techniques that improve the classifier's performance is Domain Adaptation. In our study, in Chapter-5, we try to adapt BERT-based models (for BERT, see Devlin et al. [3]) to hate speech, sexism to be specific. Gururangan et al. [6] have shown how models can improve performance by adapting a certain domain during their pre-training stage. For this, generally, a model is trained on a large unlabelled dataset and then fine-tuned on the smaller labelled dataset, which fits in line with our work. Furthermore, techniques that are not based on linguistic concepts and rely more on pure machine learning perspective, like learning rate scheduling, have also been known to improve classifier performance. Zhao et al. [7] showcased how important it is for the learning rates to adapt to the task so as to achieve the best performance in classification tasks. Techniques like learning rate scheduling help in faster convergence while training, which ultimately leads to better results.

2.1.1 Sexism Detection

There has been plenty of work in the field of Text Classification with respect to online sexism or hate speech in general. We derive inspiration for our work from some of them.

For example, one of the seminal works in this area was conducted by Waseem et al. [8], wherein they used a lexicon-based approach to detect hate speech and offensive language in social media. The authors used a publicly available dataset and reported high accuracy and F1 scores. Detecting sexism in a ternary setting can also be challenging, and in their work, Jha and Mamidi [9] showed how SVMs, Seq2Seq, and Fasttext classifiers can be used in determining the class of sexism. Finally, another work that looks at the problem in a way somewhat similar to ours is a study by Davidson et al. [10], in which they proposed a deep learning model for detecting hate speech in online social media. This work made use of a large dataset and showed that deep learning models outperformed traditional machine learning algorithms for detecting hate speech. We wish to explore this area a little deeper with our study. We compare the best-known conventional classifiers for the three subtasks to showcase their strengths and weaknesses.

Work has also been done to show how to make use of both conventional and deep learning approaches to identify various forms of sexism. For example, Rodriguez-Sanchez et al. [11] showcased how both of these approaches can be used to detect sexism in a multi-lingual setting. There have also been studies where the researchers have gone on to develop their own datasets to examine different forms of sexist content prevalent in today's world (Parikh et al. [12] and Samory et al. [13]). In our study, however, we stick to the EDOS task dataset [1], as it offers a foundation to compare the performance of the systems trained on our techniques against the performance of the best baseline models and top-ranked systems.

2.1.2 Data Augmentation

As we have discussed, the work related to Text Classification in this thesis will be based on the EDOS task dataset. The dataset, as we will analyze in the next chapter, is highly imbalanced. Too large of an imbalance can cause the trained models to be biased in their applications, as showcased by Johnson and Khoshgoftaar [14]. In order to balance datasets, naive resampling techniques like Undersampling and Oversampling exist, but to significantly improve the performance of the classifier, other data augmentation techniques must also be considered. For instance, the EDA framework proposed by Wei et al. [15], where simple updates like synonym replacement, random insertion, random swap, and random deletion improved classification performance to a considerable extent. Likewise, there are other data augmentation approaches, such as the stochastic replacement of words in the sentences with other relevant words using bidirectional recurrent neural networks, which was proposed by Kobayashi [16]. In more recent studies, pre-trained language models have been utilized to get data instances that are more diverse and linguistically correct. Anaby-Tavor et al. [17] have shown how GPT-2 can be applied to generate synthetic data for a given class in text classification tasks.

2.2 Text Generation

The automatic generation of articles in Wikipedia has been a topic of interest to researchers for the past few years. Some impressive approaches have been researched upon and proven effective over the years. Pochampally et al. [18] showcased how semi-supervised approaches can be utilized for the automatic generation of articles on named entities, especially actors. For more neutral and factual domains like Science and Technology, Minguillon et al. [19] showed how an entire corpus of Wikipedia articles on Science can be uncovered by studying the underlying graph structure of the strongly linked topics and categories. Liu et al. [20] have shown how treating the automatic generation of Wikipedia articles as a summarization task of long and detailed source documents can also yield good-quality results. More recently, Agarwal and Mamidi [21] have shown how WikiData[1] can be utilized as a knowledge base to generate Hindi Wikipedia articles automatically.

Our contribution to text generation, in the context of Wikipedia articles, involves scraping a lot of unstructured content from online resources and structuring and summarizing it into a presentable format. We take inspiration of summarizing and simplifying unstructured content information from Woodsend and Lapata [22], wherein they showcased how to select the most relevant information from textual data and rewrite it in a manner understandable to even the non-native speakers of the language.

[1] https://www.wikidata.org/

Chapter 3

Sexism on Social Media Platforms: EDOS Task Dataset

As we have already discussed, Sexism is one such form of hateful content which has been growing over the years. It has increased to such levels that human moderation alone is inadequate and inefficient to manage the volume of content, necessitating efficient technological solutions. The EDOS shared task [1] targets this problem of online detection of sexism in English and provides the right opportunity to test the performance of automated systems in doing so effectively.

In this chapter, we intend to dive deeply into the dataset of the EDOS shared task at SemEval '23. The motive is to explore and analyse the dataset thoroughly so as to gain a complete understanding of the same. Analysing the EDOS dataset is very important as this dataset forms the basis of the research studies carried out in the next two chapters. Furthermore, the analysis would provide a comprehensive understanding of the dataset's structure, patterns, and characteristics. It will allow us to identify and interpret trends, detect outliers, handle missing data, and assess overall data quality. The insights gained would guide decisions on feature selection, engineering, model choice, and preprocessing steps, ensuring data is appropriately prepared for subsequent analyses or studies. This chapter plays a crucial role in this work as it will contribute to the reliability, interpretability, and quality of the results obtained from this research work.

3.1 The Task

The EDOS shared task had three subtasks. The dataset itself consisted of 14,000 entries taken from posts and comments on two social networking platforms: Gab and Reddit. The entries also had their respective labels for each subtask. The task organizers had also provided a large amount of unlabelled data of about 2 million posts from Gab and Reddit. We do not review that dataset in detail in this chapter as it is only utilized for domain adaptive pertaining in Chapter-5. The default dataset for our research work is the labelled one with 14,000 entries, which we will analyse in detail. The input for any trained classifier would be the text of the post or comment, including the emojis, hashtags, hyperlinks, etc. and the classifier would have to predict the appropriate class of sexism. The output of the classifier varied according to the different subtasks:

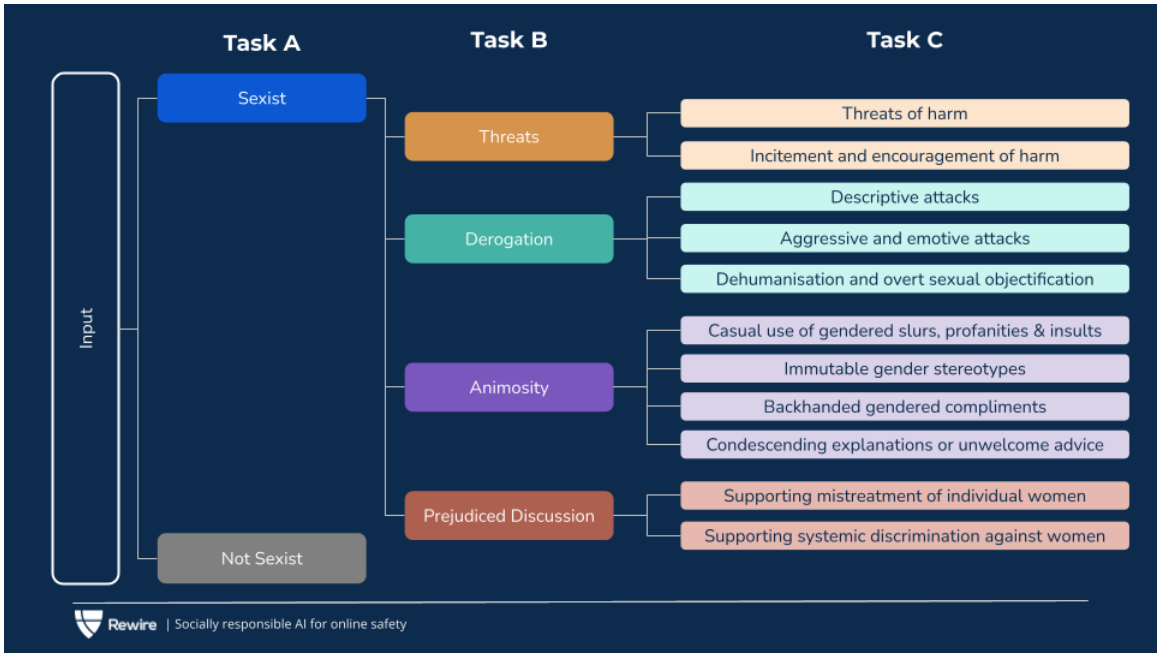


Figure 3.1 The EDOS shared task categories. Image adopted from the organizers of the task [1]

- For subtask A, the output is either the text is *sexist* or *not sexist*
- For subtask B, the output can take any of the four following values:
  - *1. threats, plans to harm and incitement*
  - *2. derogation*
  - *3. animosity*
  - *4. prejudiced discussions*
- For subtask C, the output is a more fine-grained version of subtask B's output. It is essentially a further categorization or, rather, an explanation of the form of sexism present in the text. The relation can be understood by the same numbering of the category in subtask B and the sub-category in subtask C.
  - *1.1 threats of harm*
  - *1.2 incitement and encouragement of harm*
  - *2.1 descriptive attacks*
  - *2.2 aggressive and emotive attacks*
  - *2.3 dehumanising attacks & overt sexual objectification*
  - *3.1 casual use of gendered slurs, profanities, and insults*
  - *3.2 immutable gender differences and gender stereotypes*

- *3.3 backhanded gendered compliments*
- *3.4 condescending explanations or unwelcome advice*
- *4.1 supporting mistreatment of individual women*
- *4.2 supporting systematic discrimination against women as a group*

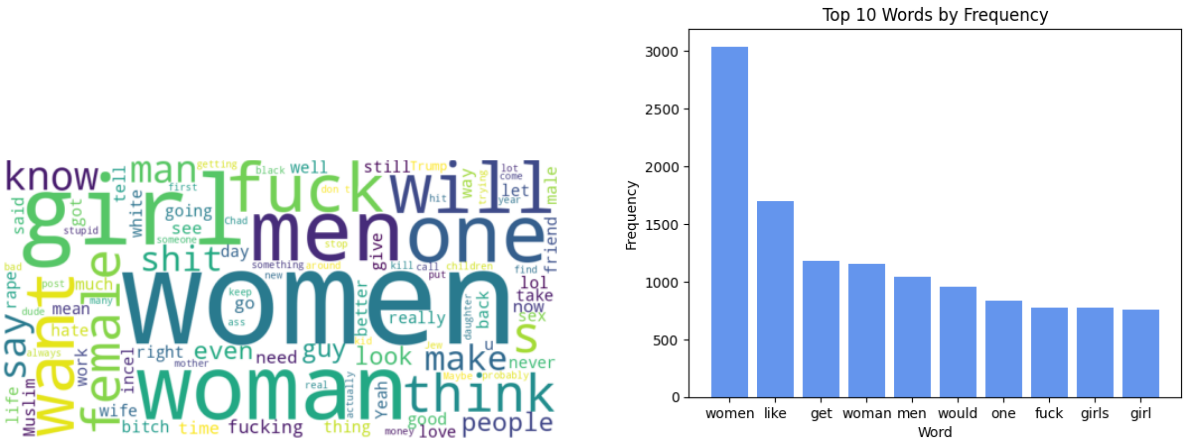
For an easier visual understanding, Figure-3.1 can be referred to. As visible from the figure and as we have explained above, in this shared task, we go from dealing with a simple binary classification problem to an 11-class multi-classification problem.

3.2 Statistics

In this section, we take a closer look at the important statistics and characteristics of the dataset. Gaining a better understanding of the dataset will facilitate us in making informed decisions about the machine learning models and deep learning techniques to utilize when building automated systems to detect sexism in online platforms.

3.2.1 Textual Insights: Word Frequencies, N-grams, NER, & POS

One of the best ways to understand any textual dataset is to gain a statistical understanding of its textual data. For the same, we can look at the most frequently occurring words, n-grams, Named Entity Recognition (NER) terms, and Part-of-speech (POS) tags.



(a) (b)

Figure 3.2 Analysing word frequencies using: (a) WordCloud (b) Frequency plot of top-10 non-stopwords

We can see the top word frequencies of the dataset in Figure-3.2. As expected from a sexism dataset, the most common word is ”women”. The presence of this word is almost two times of the next term,

”like”. Other terms that contribute more semantically are ”woman”, ”men”, ”girls”, and ”girl”. Almost a thousand text instances also contain the profane word ”fuck”. It can have a different meaning depending upon the context, but largely, it is used in a derogatory way towards women. For example, this sentence from the dataset: ”*Fuck her, do not date her.*”

Another important observation is that the dataset also contains hate towards other groups of people, and this can be noticed in the WordCloud in Figure-3.2(a), where terms like ”Muslim”, ”Jew”, ”black”, and ”incel” also have a prominent presence. Incel is a term that stands for involuntary celibate, and one which has been recently popularized on online social platforms and is used for anyone who cannot find a romantic partner instead of desiring one. This observation is confirmed by the presence of the following sentences in the dataset:

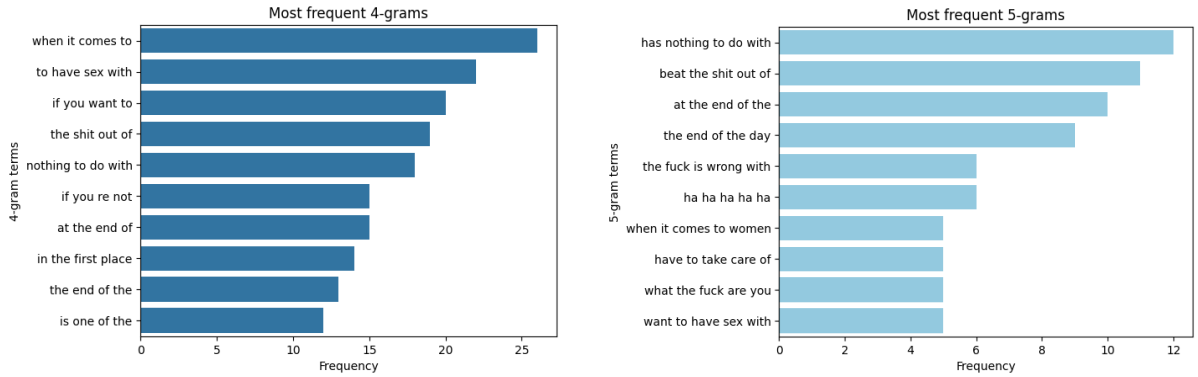
- *”Are you also a filthy muslim or are you just stupid?”*
- *”The incel mentality is based on low self-esteem and also the need to feel superior.”*
- *”Blacks are a tool, someone doesn’t want a lot of them around, starting with Sanger and the Robber Barons.”*
- *”Only a jew would gloat over the murder of a little White girl by an illegal alien.”*

Nevertheless, this study’s main focus is on sexism detection, and we stick to analyzing trends regarding the same. However, the presence of hatred towards other groups in sexist communities online must be studied and makes for a crucial research problem which ultimately would work towards the same goal, i.e., making digital platforms a safe space for everyone.

Words alone can only give us so much idea about the dataset. The next important step is to understand the context in which they are being used. For that, we shift our focus to the most frequent n-grams in the textual data. We analyze only the top 4-grams and 5-grams as bigrams and trigrams are often helping phrases like ”this is so”, ”we go on”, ”they are”, ”I am” etc. Expanding the scope to 6-grams or more ended up giving only a certain possibly-spam phrase, which was being used many times. This is why we limit our analysis to 4-grams and 5-grams.

In Figure-3.3, we can notice that phrases like ”nothing to do with”, ”beat the shit out of”, ”want to have sex with”, and ”when it comes to women” are the ones that contribute semantically more to any context over the others present in the graph. Almost all of them have a derogatory tone towards women, which goes on to show that the dataset is mostly filled with phrases targeting women in a negative sense. This analysis of the top 4-grams and 5-grams has further added to the credibility of the dataset by showing the different nuances present in the text, which has different forms of gender bias.

After discussing the most common words and phrases, it is also important to look at the most common Named Entities (NEs) and POS tags. This analysis is significant as these terms will add a layer of linguistic context and semantic understanding, which ultimately provide a more comprehensive

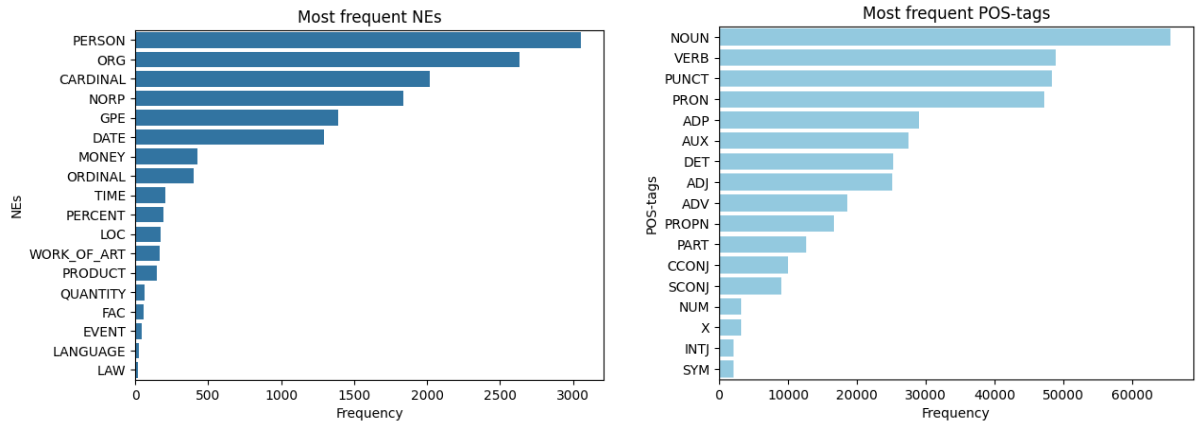


(a) (b)

Figure 3.3 Analysing most frequent (a) 4-grams (b) 5-grams

overview of the text. NER allows for the identification and classification of entities such as names, locations, organizations, and more. This information provides insights into the specific entities mentioned in the dataset, shedding light on the contextual relevance and potential biases associated with different named entities. POS analysis helps to understand the grammatical structure and syntactic nuances of the text. It allows for the differentiation between, for example, verbs, adjectives, and nouns, providing valuable information about the tone, sentiment, and intent behind the language used in the dataset.

While most common words and phrases offer a general overview, NER and POS analysis contribute to a more nuanced understanding of how these words are used in specific contexts. It helps discern whether certain entities or parts of speech are consistently associated with biased or discriminatory language.



(a) (b)

Figure 3.4 Analysing most frequent (a) NEs (b) POS-tags

As visible in Figure-3.4(a), six NEs have a frequency of more than 500 sentences: PERSON, ORG, CARDINAL, NORP, GPE, and DATE. Of these, CARDINAL and DATE do not offer much semantic context to the sentence, as CARDINAL refers to numerals in descriptive form while DATE just refers to date and time periods. PERSON refers to people in the sentence, ORG refers to companies and organizations, NORP refers to political or religious groups of people, and finally, GPE refers to countries, states, and cities.

The prevalence of PERSON entities suggests a focus on individuals, potentially providing information about key figures or emphasizing the importance of personal experiences in the dataset. ORG signals the presence of organizational landscape in the text, while NORP highlights the presence of diverse cultural and political contexts being discussed in the dataset. Finally, GPE suggests a significant emphasis on locations and places in the dataset. Individually, these NEs do not provide enough information for us to act upon when building machine learning solutions, but understanding their overall distribution gives us an important insight into what type of text to expect and how we can plan on dealing with it when building automated solutions.

POS-tagging is very essential to our understanding of the text. Not only does it offer information about the syntactic and grammatical structure of the text, but it also helps in semantic and sentiment analysis. POS tags contribute to semantic analysis by providing insights into the meaning of words based on their grammatical roles. Understanding whether a word is used as a noun or a verb, for example, aids in determining its semantic context and potential meanings within a given sentence. They also contribute to sentiment analysis by enabling the identification of sentiment-bearing words and their roles in a sentence. Understanding the grammatical structure aids in capturing the sentiment expressed by various parts of the text.

From Figure-3.4(b), we can notice that nouns, verbs, and pronouns, except punctuations, of course, are the most frequent POS-tags. Their frequency is almost double the frequency of the rest of the POS-tags. The prevalence of nouns, verbs, and pronouns suggests that the sexist content in the dataset is likely to be action-oriented and directly targeted towards individuals or groups. This may include explicit references, descriptions, or actions that contribute to a discriminatory narrative. Moreover, the frequent occurrence of nouns and pronouns may highlight a focus on gendered entities, such as individuals or groups based on their gender. This can include the use of gender-specific pronouns or nouns that contribute to reinforcing stereotypes and biases. Lastly, the prevalence of clear syntactic structures involving nouns, verbs, and pronouns provides a foundation for developing automated tools and models for the detection of sexist language. Understanding the linguistic characteristics helps in creating targeted solutions for identifying and addressing discriminatory content. It's important to note that while POS analysis provides valuable insights, the context, semantics, and intent behind the language should be further explored to fully comprehend the nature of sexism within the dataset. Additionally, qualitative analysis is crucial for a deeper understanding of the social and cultural implications embedded in discriminatory language.

3.2.2 Text Complexity

Analyzing the dataset's text complexity is crucial for multiple reasons. First of all, sexist language can manifest in various forms, from overt and explicit expressions to subtle and sophisticated language. Text complexity analysis helps identify instances where discriminatory language may be veiled in more complex structures, making it essential for a comprehensive assessment. Furthermore, it aids in differentiating the contextual sensitivity of sexist expressions. Certain linguistic structures may convey different meanings based on the context, and a thorough analysis ensures that the complexity of language is considered in interpreting the dataset accurately.

For the same, we choose the Flesch Reading-Ease Score (FRES) [23]:

$$FRES = 206.835 - 1.015\left(\frac{total\ words}{total\ sentences}\right) - 84.6\left(\frac{total\ syllables}{total\ words}\right) \tag{3.1}$$

The Flesch Reading-Ease score is designed to produce a higher score for easier-to-read text and a lower score for more complex and difficult-to-read text. The resulting score typically falls on a scale from 0 to 122, with higher scores indicating easier readability.

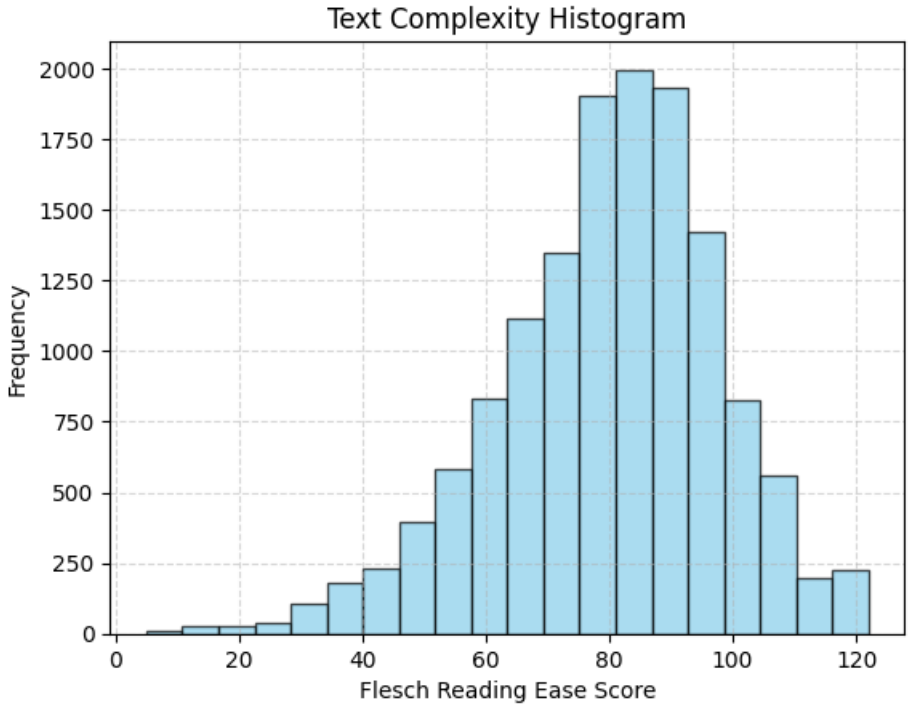


Figure 3.5 Frequency of FRE scores

As visible from Figure-3.5, the majority of the post-texts in the dataset have a FRES between 60 and 100. This indicates that the texts are written in a way that is easily understandable. This range is associ-

ated with plain and straightforward language, making the content comprehensible to a broad audience. Furthermore, texts within this FRES range are likely to be present on different social platforms, online websites, educational materials, and public communication. They can be easily adapted for different communication contexts. All in all, the text in the dataset is pretty straightforward, with a few complex outliers and can be easily understood by an audience of varying ages and educational backgrounds. This goes on to show that a large number of people are vulnerable to harmful content online and highlights the need for automated solutions to flag and hide such content even further.

3.2.3 Topic Modelling

Topic modelling plays a pivotal role in the analysis of textual datasets by offering a systematic and data-driven approach to uncover latent themes and patterns within the textual content. In the context of the EDOS dataset, where identifying and understanding discriminatory language is crucial, topic modelling provides a means to unveil the underlying topics or themes that may be implicit in the data. This analysis will aid us in categorizing and interpreting various manifestations of gender bias, ranging from overt expressions to subtler forms, and allows for a nuanced understanding of the diverse dimensions of sexism present in the dataset. Although topic modelling does not guarantee absolute verification of what the dataset is related to, it certainly does provide valuable insights as to whether the dataset aligns with the topic which the organizers of the dataset claim it is for. Therefore, it is essential and beneficial for us to carry out topic modelling for the EDOS dataset.

For performing topic modelling, we will implement the Latent Dirichlet Allocation (LDA) modelling technique [24]. This technique assumes that there are underlying topics within the text and that each document (post) is a mixture of topics and each topic is a mixture of words. The LDA process involves iterating through all topics and each word within the corpus. For each word, a random assignment to a specific topic is made, and then the frequency of the word's occurrence in that topic, along with its associations with other words, is evaluated. The important feature of LDA is that it also incorporates the saliency of words instead of just the frequency. We have already looked at the most frequent words in Section-3.2.1 but we could only infer so much from that. Saliency, in the context of NLP, refers to the quality of being noticeable or significant. It is a measure of the importance or prominence of a particular element, such as a word, within a given context. Saliency helps in extracting more meaningful and specific information from the corpus, leading to a more refined understanding of the underlying themes. While frequent words provide a general overview, salient words contribute to the depth and precision of topic representation.

We perform Topic Modelling using the LDA technique on our EDOS task dataset to gain a better understanding of the underlying topics. As visible in Figure-3.6, we limit our topic modelling analysis to four topics. Four is chosen as the number of topics because it should be enough to guide us on whether the dataset has sexism-related topics. It is important to remember that the LDA technique is not always accurate, and that is why we need to have certain 'n' topics modelled, and if sexism-related terms are in its top-3 or top-4 topics, then the dataset aligns with its identity, i.e., a dataset which can

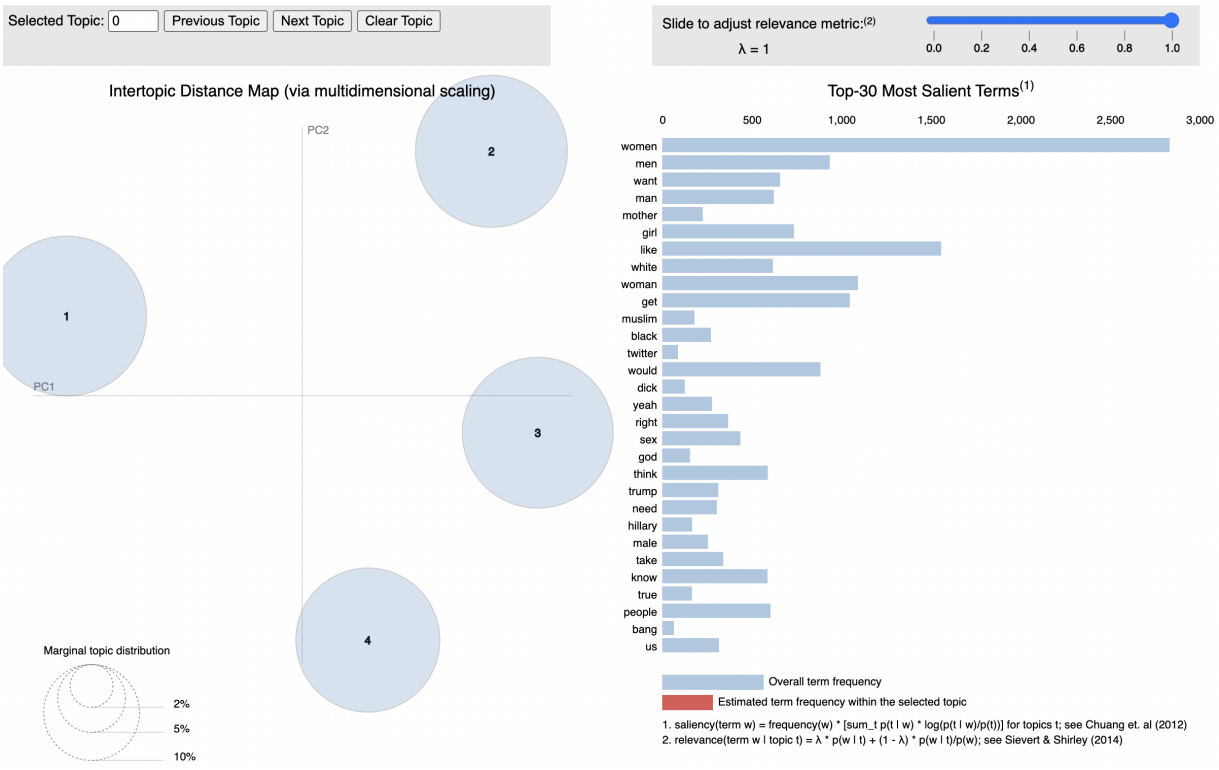


Figure 3.6 Topic modelling using LDA. Visualized using LDAvis [25]

be used to train models for sexism detection. The top-most salient term of the dataset happens to be ”women”, and other salient terms of the dataset also refer to genders like ”girl”, ”mother”, ”woman”, ”man”, ”men”, etc. This indicates that the dataset revolves around text including topics based on these terms, as expected in a sexist dataset. There is also a hint of racial profiling in the dataset, as hinted at in Section-3.2.1, as terms like ”black” and ”white” are also included in the top salient terms. All of the topics except 3 and 4 are in different quadrants, indicating a difference in the underlying themes of the topics. For further analysis, we will have to consider the topics and their most relevant terms individually. From Figure-3.7, we note the following:

Topic-1

Keywords: like, women, get, girl, one, girls, know, men, go, would

This topic suggests that the dataset revolves around discussions involving terms like ”girl,” ”girls,” and ”men,” suggesting a variety of gender-related narratives. The presence of terms like ”know” and ”go” indicates personal opinions or experiences and dynamic discussions about gender.

Topic-2

Keywords: would, like, woman, fuck, women, think, white, people, get, never

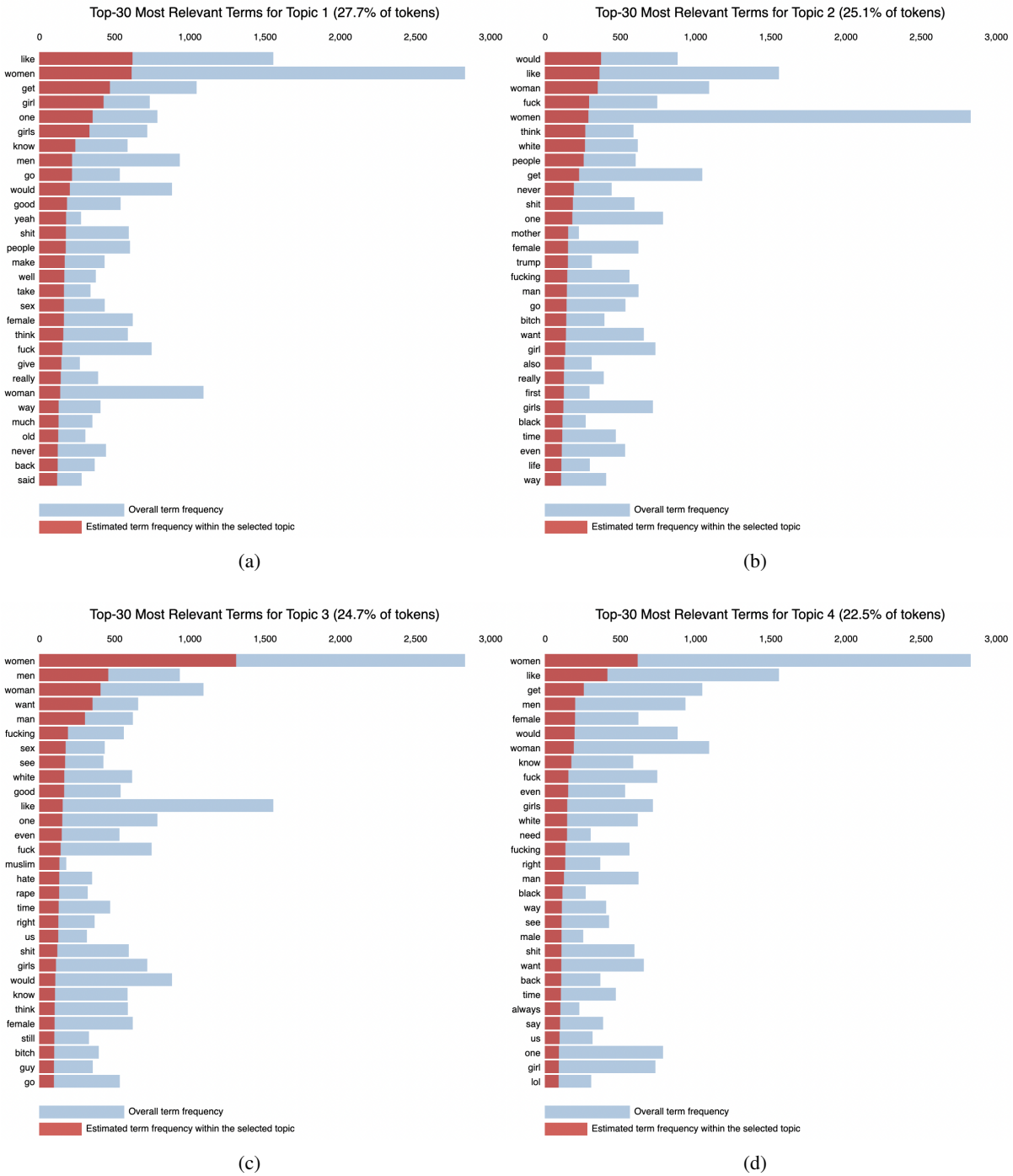


Figure 3.7 Analysing most relevant terms of (a) Topic-1 (b) Topic-2 (c) Topic-3 (d) Topic-4

This topic signals that the dataset possibly covers a wide range of gender-related discussions, including hypothetical scenarios (”would”), attitudes towards women, explicit language (”fuck”), racial

considerations (”white”), and general reflections (”think,” ”people”). It represents a diverse and complex set of gender-related narratives.

Topic-3

Keywords: women, men, woman, want, man, fucking, sex, see, white, good

This topic represents that the dataset is prominently centred around desires (”want”), relationships between women and men, explicit language (”fucking”), and possibly sexual references (”sex”). The inclusion of terms like ”white” and ”good” suggests potential racial or moral considerations within the context of gender discussions.

Topic-4

Keywords: women, like, get, men, female, would, woman, know, fuck, even

This topic hints that the dataset involves discussions about women and men, with an emphasis on potential gender differences or characteristics. Terms like ”know” and ”even” suggest nuanced explorations of gender-related topics. The inclusion of ”fuck” may indicate the presence of explicit or potentially confrontational language.

The four topics collectively highlight a dataset steeped in discussions related to gender, unveiling a spectrum from nuanced considerations to explicit language use. The recurring themes across topics, such as explicit terms like ”fucking”, references to desires and relationships, and potential racial considerations, strongly suggest that the dataset is significantly populated with instances that may indicate possible sexism. The varied and complex nature of these discussions underscores the need for a thorough analysis to discern the explicit and implicit biases present. A more detailed and comprehensive understanding will be crucial for addressing and mitigating instances of sexism.

3.3 High Imbalance

The most stand-out challenge with the EDOS task dataset is the fact that it has a highly imbalanced class distribution. Using an imbalanced dataset to train classifiers can result in a dangerous bias towards the majority class during prediction [14]. The extent of imbalance for each subtask can be noticed in Figure-3.8.

As visible in Figure-3.8(a), for Subtask A, the number of *non-sexist* entries is almost three times the number of *sexist* entries. For Subtask-B (Figure-3.8(b)), classes 1 and 4 have equal distribution but don’t even have half the number of instances classes 2 and 3 have. Similarly, in Figure-3.8(c), we see that for Subtask-C, the distribution is very skewed. 4 categories do not even have 100 instances in the dataset, while the majority class has over 700 instances.

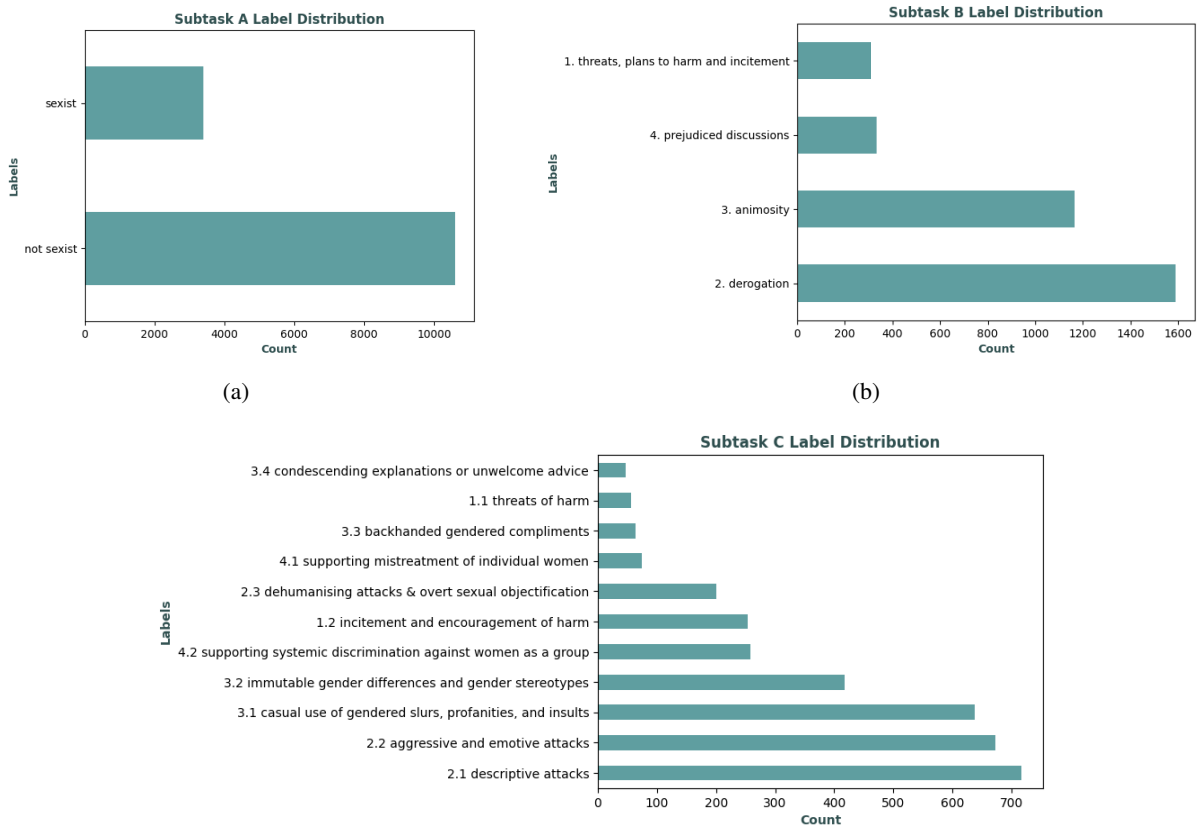


Figure 3.8 Class label distribution for (a) Task-A (b) Task-B (c) Task-C

This amount of imbalance in the class distribution will impact even the best of deep learning based classifiers. The impact of imbalanced class distribution cannot be overlooked. For starters, in the context of sexism detection, this extent of class imbalance means the trained model might be good at identifying non-sexist instances but perform poorly at detecting instances of sexism. This is because the model will learn patterns specific to the majority class and fail to capture the complexities associated with the minority class. As a result, its performance in detecting instances of sexism may be compromised. Linguistically speaking, the model will fail to learn the subtle features or nuances associated with sexism, especially if instances of sexism are underrepresented. Consequently, the model may struggle to correctly identify instances of sexism in real-world scenarios.

To counter this imbalance, data resampling techniques and data augmentation approaches need to be implemented. Depending upon their approach, they change the data in different ways. For instance, we can try to increase the instances in the minority class by simply duplicating them, or we can simply remove the extra instances of the majority class to strike a balance between the classes. For more complex approaches, we can pre-train a model with the dataset and fine-tune it to create similar examples. Since

the output does not guarantee a sure-shot example that will be true to its label, the outputs will have to be checked manually. All in all, countering this dataset imbalance is crucial for the development of smart and accurate automated solutions. We will look at some of the important resampling and augmentation techniques in the coming chapters.

Finally, high class imbalance means that care must also be taken when selecting the appropriate metric for evaluating the trained models and classifiers. It is possible that even after applying resampling and augmentation techniques, we are not able to achieve a balance between the classes. Moreover, while training, we must also represent the real-world truth in the class distribution, i.e., it is obvious that, more often than not, on social platforms, a piece of text is more likely to be non-sexist than sexist. In essence, the classes will always be somewhat imbalanced and choosing the correct metric is important. Evaluating the model solely based on accuracy might not provide an accurate representation of its performance. Metrics like precision, recall, F1-score, or area under the precision-recall curve become more informative, especially in understanding how well the model performs on the minority class. In conclusion, we also need to take into account the performance of the model on minority classes and not only the majority class when evaluating it.

3.4 Other Challenges

Apart from class imbalance, there are other challenges associated with a sexist dataset. It is important to note that even though we have a clearly defined definition of sexism, sometimes sexism can be subjective, and the lines get blurred on what does and what does not constitute sexism. In times like this, we are only left hoping that the labels given by the organizers for the task will be absolutely true and need not be challenged.

One of the main challenges is the usage of slang in the dataset. For example, the sentence: "*Landwhales have the pick of normal guys because Chad gets all the Top Stacies.*" is sexist, but to a model trained in basic English and even a corpora of hate-speech and latest trends might miss out on such sentences. The model might fail to correctly detect this sentence as it does not contain the common gender terms like women, girl, etc. and rather uses the term "*Landwhales*", which is the core of the sentence and represents a negative term referring to an overweight woman in this context.

Another noticeable challenge is the excessive use of offensive terms and derogatory remarks in the dataset. If the trained model is over-reliant on insults or negative remarks, it can misclassify non-sexist phrases as sexist and vice-versa. For example, "*Because they are afraid of confrontation, a woman will always escape*" is sexist, but as it does not contain offensive and explicit terms we have seen like "fuck", "shit", "bitch", etc., it is possible that a model might tag it as non-sexist. Similarly, consider: "*If my daughter got pregnant from this shit, That judge has a lawsuit coming!*". The sentence is non-sexist, but

as it contains explicit terms, the model might mistake it for a sexist sentence.

Lastly, there is also the challenge of models having an uninformed and traditional vocabulary. The use of slang terms, abusive words, derogatory remarks, and racial slurs we have seen are generally not seen in the type of corpora all language models are trained on. Therefore, care must also be taken to expand the models' vocabulary to account for such terms and remarks. Otherwise, the model will perform poorly for ambiguous statements, as we just saw.

3.5 Conclusion

To summarize, in this chapter, we analyzed the EDOS task dataset in detail. The different types of analyses we did: most frequent words, n-grams, NEs, and POS-tags, all confirmed the task organizers' claim that the dataset indeed contains data relevant for training models to detect sexism. Text complexity further confirmed that the dataset was definitely scraped from public forum websites according to the FRES score obtained. Topic modelling unveiled the major underlying topics are related to gender narratives, racial profiling, and bias in general.

We also looked at the different challenges the dataset possesses and introduces in the task of sexism detection, with the most significant of them being the imbalanced class distribution in the dataset for each subtask. In conclusion, the EDOS task dataset is more than fit for building models to detect online sexism, but techniques and approaches should be considered to counter its shortcomings and to ensure the development of an accurate and unbiased model. In the coming chapters, we will resample and augment this dataset to build automated solutions for the detection of online sexism using both conventional machine learning and recently advanced deep learning techniques.

Chapter 4

Conventional Machine Learning Techniques for Sexism Detection

In this chapter[1], we intend to explore the effectiveness of several conventional machine learning classifiers like Random Forests, Decision Trees, Support Vector Machines (SVMs), etc. The task remains the same as the EDOS task, i.e., to detect sexism or the appropriate category of sexism depending upon the subtask being considered. By conventional classifiers, we refer to those classifiers that have been developed before the emergence of the more recent techniques like deep learning and transformer-based approaches. They are simpler in design as well as easier to interpret and understand than their deep-learning counterparts. Moreover, they require fewer computational resources, which means that the training time for these classifiers is greatly reduced. Despite being "conventional", these classifiers are still widely used in different practical applications and have been effective in resolving many real-world problems [26]. The aim of this chapter will be to judge the effectiveness of these classifiers of these classifiers in detecting sexist text and also its appropriate category (as seen in subtasks B and C).

4.1 Conventional Classifiers

Before diving deep into the System Architecture and the Results, it would be wise to discuss the classifiers individually. By discussing each classifier separately, we can unravel the nuanced details of their functioning, allowing us to discern the strengths and weaknesses inherent in each method. This detailed examination not only provides us with a profound insight into the inner workings of these classifiers but also allows us to assess their applicability to our specific task.

4.1.1 Logistic Regression

Logistic Regression as a classifying algorithm has been used in machine learning for binary classification tasks. It implements a logistic function, which models the probability of an input data point belonging to one of the two possible classes. An in-depth application of Logistic Regression for classification was first introduced by Hosmer Jr. et al. [27]. It makes use of the Logistic or the Sigmoid function, which outputs a value between 0 and 1 and is defined as follows:

[1]This chapter is an extended version of the published work: `https://aclanthology.org/2023.semeval-1.211/`

$$\sigma(t) = \frac{1}{1 + e^{-t}} \tag{4.1}$$

4.1.2 Decision Tree

Decision Tree Classifier is a supervised learning algorithm which creates a tree-like model of decisions and their possible consequences. It utilizes a set of input features and classifies an input data point into one of the several pre-defined classes based on a sequence of decisions made on those input features. Fisher [28] first introduced the concept of Decision Tree based classification, and it has been improved upon over the years. The Decision Tree algorithm constructs a tree-like structure resembling a flowchart, where internal nodes represent attribute tests, branches indicate test outcomes and leaf nodes hold class labels. Throughout the training process, the algorithm selects the optimal attribute for data splitting, employing metrics like Entropy or Gini impurity to gauge impurity or randomness in subsets. The objective is to identify the attribute that maximizes information gain or minimizes impurity after a split.

Entropy and Information Gain are one of the two popular ways of splitting the tree to arrive at a root node or class label. We stick to this criterion instead of the Gini Impurity for our study. Entropy is a part of Information Theory, and it is formulated as below:

$$E = -\sum_{i=1}^{N} p_i \log_2(p_i) \tag{4.2}$$

where p_i refers to the probability of randomly selecting an example in class i. Information Gain represents how much information a feature provides for the target variable, and it is nothing but the difference in entropy between the parent node and the child node, depending upon the potential split from each variable. It can be formulated as below:

$$Information\ Gain = E_{parent} - E_{child} \tag{4.3}$$

4.1.3 Random Forest

Random Forest is an ensemble learning algorithm used in machine learning for classification tasks. It constructs multiple decision trees on different subsets of the input data and features and aggregates their outputs to make the final prediction. Breiman [29] originally introduced the concept of Random Forests. Random forest is less prone to overfitting and more accurate than a single decision tree as it is an ensemble of multiple decision trees.

4.1.4 XGBoost

XGBoost stands for Extreme Gradient Boosting, which is a decision tree ensemble learning algorithm just like Random Forest. The difference lies in how the individual decision trees are built and

combined. Random Forests use the "bagging" technique to build complete decision trees in parallel from random bootstraps of dataset samples. The final outcome is an average of all the decision tree predictions. XGBoost, on the other hand, makes use of the Boosting technique combined with a Gradient Descent objective. Boosting refers to combining multiple weak models additively to generate a single strong model. XGBoost extends this technique by formalizing it as a gradient descent algorithm. It sets targeted outcomes for the next model in an effort to minimize errors, and these targeted outcomes for each case are based on the gradient of the error with respect to the prediction. XGBoost iteratively trains an ensemble of shallow decision trees, with each iteration using the error residuals of the previous model to fit the next model. The final prediction is a weighted sum of all of the tree predictions.

Random forest "bagging" minimizes the variance and overfitting, while XGBoost "boosting" minimizes the bias and underfitting. XGBoost, as a classification and regression technique, has gained a lot of popularity over the years due to its model performance and computational speed. The modern-day implementation was first explained by Chen and Guestrin [30].

4.1.5 Support Vector Machine

The formulation of SVMs as a classification and regression method was introduced by Cortes and Vapnik [31]. The objective of a support vector machine is to optimally separate the input data points into different classes in a high-dimensional space by maximizing the margin between them. The objective can be formulated as an optimization problem in the following manner:

$$\min f : \frac{1}{2}||w||^2, \tag{4.4}$$

$$s.t.\ \ y^{(i)}(w^T x^{(i)} + b) \geq 1,\ \ i = 1, ..., m \tag{4.5}$$

where w is the weight vector, x is the input vector, and b represents the bias.

4.2 Resampling Techniques

As discussed in Chapter-3, the EDOS task dataset is highly imbalanced for every subtask. For instance, for Task-A, the number of *non-sexist* entries is almost three times the number of *sexist* entries. The most important concern that dataset imbalance raises is that training on it can result in the models being biased and ultimately performing poorly on the minority classes. In order to try to balance the dataset, we experiment with two major resampling techniques: Undersampling and Oversampling, which we will consider in detail in the coming subsections. These techniques assume nothing about the data, and no heuristics are used. They are simple to implement and fast to execute, which are desirable traits when dealing with large and complex datasets.

Another important factor behind choosing these resampling techniques was the classifiers being used. There exist various complicated data augmentation techniques like Generative sampling and Random swapping, which were sure to increase the model's accuracy and improve its performance, but using

them would require us to involve deep-learning and transformer-based models to generate meaningful samples. This would beat the whole point of this chapter's study, which is to stick to conventional machine learning techniques and outperform the conventional baselines mentioned in [1].

We also use the dataset as is, i.e., highly imbalanced, to showcase the effects of biased data. We discuss the performance of both the techniques and the original dataset in detail in section-4.4.

4.2.1 Undersampling Technique

The concept behind the Undersampling technique is a very naive yet effective one. If we cannot increase the samples in the minority class, we can decrease the amount of instances in the majority class. For example, we can consider the Task-A example again. We structure our new dataset in such a way that the number of non-sexist instances is only 1000 more than the number of sexist instances, compared to the 8000 posts gap that existed earlier. This removal is done randomly, and we have dropped more than half the non-sexist instances that originally existed in the dataset. We follow the same procedure for the other subtask datasets, too.

4.2.2 Oversampling Technique

Over the years, researchers have developed different forms of Oversampling. One of the earlier works by Chawla et al. [32] introduced SMOTE: Synthetic Minority Oversampling Technique wherein they considered the k-nearest neighbours of a minority instance, randomly selected one of those and then drew a line segment from the synthetic point to the actual point in the feature space. This line segment can be used to generate multiple new points for the minority class by a convex combination of the two endpoints. For our dataset, the idea of Oversampling remains the same. We increase the number of instances in the minority classes while keeping the number of majority class instances the same. We do so by simply duplicating the minority class samples in our dataset. For example, in Task A, if the number of samples in *sexist* class is close to 3000, we duplicate them, and the strength comes close to 6000, which is almost half of the majority class, i.e., the *non-sexist* class. We do not intend to achieve a 50-50 distribution as in real life, too, the content on social media is not always an equal split between sexist and non-sexist.

4.3 System Architecture

The general outline of the architecture can be noticed in Figure-4.1. The architecture mainly consists of five parts: Data Cleaning, Dataset Resampling & Splitting, Feature Extraction, Training, and Evaluation. We discuss the first four stages in detail in the following subsections. The evaluation stage will be discussed in the next section 4.4.

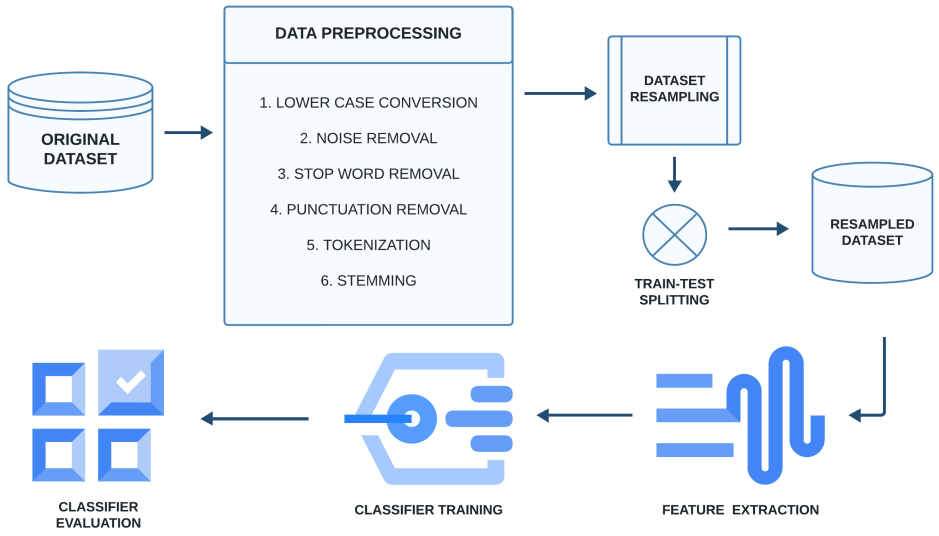


Figure 4.1 System Overview for Sexism Detection using Conventional Machine Learning Techniques

4.3.1 Data Preprocessing

The first and foremost stage is also one of the most crucial stages of the system. Data Preprocessing is where we ensure that the data being carried forward is fit for training our classifiers. This stage can be broken down into six different subparts:

1. Firstly, we convert the entire raw data entry into lowercase.
2. Then, we remove noise from the raw entry. In our context, we define noise as hyperlinks, emojis, hashtags, etc.
3. Further, we remove all the stopwords (words like ”the”, ”a”, ”an”, etc.) as their contribution to the semantics of the sentence is negligible in sexist settings.
4. We clean the entry by removing the punctuation marks.
5. After the cleaning stage, we now tokenize the text.
6. Lastly, we do stemming and lemmatization to reduce the words to their base forms before proceeding to the next stage. For example, ”shouting” gets truncated to ”shout”.

4.3.2 Dataset Resampling & Splitting

As discussed multiple times earlier, the EDOS task dataset is very skewed, and in order to tackle this, we employ very fundamental resampling techniques like undersampling and oversampling. The instances for resampling are chosen randomly. For the undersampling approach, we remove a large

number of instances from the majority class(es) in order to balance out the distribution of every class. For the oversampling approach, we duplicate the instances in the minority class(es) to have a better proportion of instances.

After resampling, we randomly split the dataset to generate our train-test split. The organizers of the EDOS task provided both the development and test datasets. The correct labels of the test set were not made accessible until the end of the task, so the classifiers were trained only using the development set. For the train-test split, we experimented with different splits like 65-35%, 70-30%, 75-25%, etc., but noticed a positive jump in performance only when the split was changed to 85-15%.

4.3.3 Feature Engineering

Now, we move on to the third stage of our architecture, which is Feature Engineering. This stage is mainly concerned with extracting the features from text by converting the text into vectors, which the classifier will be able to understand when modelling the data. For this stage, we make use of a simple Term Frequency-Inverse Document Frequency (TF-IDF) Vectorizer [33]. It converts the words or tokens in the text into vectors or embeddings by quantifying the relevance of a term in a sentence or a corpus by weighing how often it appears in a sentence against how often it appears in the entire corpus. It is calculated as follows for a term i in a document j:

$$tfidf_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right) \tag{4.6}$$

where $tf_{i,j}$ is the total number of times i appears in j, N is the total number of documents in the corpus, and df_i is the total number of documents containing i.

4.3.4 Model Training

Once we have extracted the features, the next stage is to train the classifiers individually. For implementing the conventional classifiers, we utilize the in-built classifiers of the scikit-learn[2] library. The majority of the classifiers were implemented with default parameters, as there were no significant differences in performance when changing them. However, for classifiers like the SVMs, we did observe a jump in performance with hyperparameter tuning. Therefore, to ensure that we get the best performance out of the SVM classifier, we choose the *linear* kernel, the penalty parameter (C) set to 1.0, degree set to 3, and gamma set to *auto*.

The training data is used to train the classifiers to understand the task and predict correct labels. Finally, the trained classifiers are fed with the testing data in order to evaluate their performance and gauge how accurately and precisely the classifiers can correctly classify the category of sexism.

[2] `https://scikit-learn.org/stable/`

4.4 Results

For evaluation, like the organizers of [1], we choose macro average F-1 scores as our metric. We use the in-built implementation of the scikit-learn library for the same. Choosing the same metric as [1] allows us to compare our performance with other teams and evaluate our model. Another reason why the macro-average F-1 score stands out as a choice for the metric is that the EDOS dataset is highly imbalanced, and the macro average technique treats each class of the data adequately. It can do so by first calculating the F-1 score for each class separately, and then it takes the arithmetic mean of all those scores. As a result, each class is given equal importance in the final score, regardless of its size or prevalence in the dataset.

4.4.1 Subtask A: Binary Classification

Model	*Development Set*	*Test Set*
Decision Tree	60.15	57.87
Logistic Regression	71.78	73.73
Random Forest	**73.91**	**75.83**
SVM Classifier	71.87	72.79
XGBoost Classifier	71.27	72.72

Table 4.1 Macro Avg. F-1 Scores of Classifiers on Subtask-A: Binary Classification

As expected, out of all the subtasks, subtask A was the one in which our classifiers performed the best, thanks to the low complexity of the subtask and the binary nature of the classification problem. Even though these statistical classifiers do not employ deep learning techniques like large pre-trained language models or deep neural networks, they could still output decent F1- scores for the subtask as visible in Table-4.1. Random Forest performed the best with a score of 75+. Other classifiers came close, but Random Forest was able to outperform them marginally.

4.4.2 Subtask B: 4-class Classification

Model	*Development Set*	*Test Set*
Decision Tree	32.66	37.58
Random Forest	15.91	15.94
SVM Classifier	**40.26**	**38.19**

Table 4.2 Macro Avg. F-1 Scores of Classifiers on Subtask-B: 4-class classification

As visible in Table-4.2, we report the F-1 scores of only three classifiers. This is because Logistic Regression was not suited for multi-class classification problems, and the XGBoost classifier had very poor performance, with its F-1 score failing to reach even 10. The SVM classifier outperformed both the

Decision Tree and Random Forest classifier by quite some margin. This can be owed to the SVM's ability to handle unbalanced data quite seamlessly by adjusting the penalty parameter (C) and by assigning higher weights to the minority classes during training. This makes the SVM classifier quite sensitive to the minority classes. In contrast, Random Forests and Decision Trees struggle with unbalanced data, as they tend to favour the majority class.

4.4.3 Subtask C: 11-class Classification

Model	*Development Set*	*Test Set*
Decision Tree	**21.84**	**21.85**
Random Forest	12.14	12.22
SVM Classifier	19.94	20.42

Table 4.3 Macro Avg. F-1 Scores of Classifiers on Subtask-C: 11-class classification

The performance of the classifiers for subtask C is visible in Table-4.3. As expected, all three classifiers performed quite underwhelmingly. The best F-1 score barely managed to cross the 20 mark. The performance, however, follows the same trend as in subtask B; both the Decision Tree and SVM classifier are closer together, and they outshine Random Forest by quite some margin.

4.4.4 Performance in Shared Task

For Task A, our best classifier achieved the rank of 73^{rd} among all participants, while for tasks B and C, our best-performing classifiers were ranked 65^{th} and 55^{th} respectively, among all the participating teams. Such rankings are only natural when our approach is limited to conventional machine learning techniques alone. The important observation to make here is how we fare against the baselines set for conventional machine-learning techniques by the organizers of the EDOS task.

Subtask	**B2 Baseline model**	**Our best model**	**B6 Baseline model**	**Top ranked model**
A: 2 class	49.33	75.83	82.35	87.46
B: 4 class	22.97	38.19	59.26	73.26
C: 11 class	8.81	21.85	31.71	56.06

Table 4.4 Comparison of the best F-1 scores attained for each subtask by our best model, top-ranked model and baseline models from task paper: B2 and B6. B2 was the best baseline for conventional classifying approaches, while B6 was the best baseline for transformer-based approaches.

Our best classifiers were able to outperform the baseline models B0, B1, and B2 of [1]. These baseline models were based on conventional classifiers like Uniform and XGBoost. For comparison, in Table-4.4, we only report the B2 model among conventional classifiers as it was the best-performing one between them for each subtask. On the other hand, our classifiers failed to reach the performance of the B6 baseline model, which was the best among B3, B4, B5, and B6 models, all of which were based

on advanced machine learning techniques like transformers and neural networks and utilized variations of DistilBERT and DeBERTa.

4.4.5 Impact of Dataset Resampling & Splitting

Optimization	*Macro Avg F-1 score*
Undersampling	**73.91**
Oversampling	67.50
No-sampling	68.69
75-25% data split	73.10
85-15% data split	**73.91**

Table 4.5 Macro Avg. F-1 Scores of Random Forest Classifier on Development Set of Subtask-A: Binary Classification with different optimizations

In table-4.5, we can notice how the tweaks and optimizations we had done to our dataset in terms of resampling and splitting provide a boost in performance for the Random Forest classifier in Task A. Increasing our training split from 75% to 85% increases the F-1 score by approximately 1 point. This is a very slight increase but nevertheless a very important one as every decimal point boost in performance counts in Shared Tasks. The Undersampling technique also had a great effect on the F-1 score attained, improving the score by more than 5 points when compared with the model trained on the original dataset without any resampling. The application of the Oversampling technique proved ineffective, leading to such a performance reduction that the system performed worse than its original state without employing any resampling technique.

4.4.6 Discussion

It is worth noting that a different conventional classifier has performed the best for every subtask in our study in this chapter. Subtask A was a straightforward binary classification problem where most of the classifiers eased past the F-1 score of 70. Random forest classifier performed the best in this subtask as it is not prone to overfitting. There were only three classifiers in the running for subtasks B and C. SVM was the best for subtask B, while Decision Tree was the best for subtask C.

In general, the performance of classifiers can vary for different subtasks due to various factors, like dataset characteristics, feature representation, hyperparameter tuning, and randomness. The performance of classifiers is directly influenced by the hyperparameters chosen for them. If they are not optimised or tuned specifically for each subtask, it can impact their performance. Moreover, the architecture of the classifier itself is one of the causes of different performances across the subtasks. For instance, SVM can outperform Decision Tree in subtask B because it treats the 4-class classification problem as four binary classification problems and is well suited for it, but when the data becomes highly non-linear and complex, like in subtask C, Decision Tree outperforms SVM.

4.5 Conclusion

In conclusion, in this chapter, we explored the effectiveness of some of the conventional machine learning classifiers in the task of online sexism detection. Although they performed commendably in the binary classification task, it is clear that they are not cut out for multi-class classification tasks. Random Forests seem to be the best choice, out of conventional machine learning classifiers, for simple binary classification to test the presence of sexism in a piece of text.

For future work directions, we explore deep learning neural networks and large pre-trained language models in the next chapter 5. Over the years, these transformer-based approaches, in shared tasks specifically, have outperformed conventional machine learning techniques with ease when it comes to text classification problems. This is why we intend to focus on them in the next chapter and expect them to improve upon the score achieved by our conventional systems in this chapter.

4.5.1 Limitations

The very nature of our approach has limitations associated with it. We delved only into conventional classifiers, which have been performing decently in text classification tasks but not as well as transformer-based approaches. Table-4.4 showcased this aptly. We were able to outperform the conventional baselines in [1], but our best-performing system fell short of the transformer-based baselines, let alone beating the top-ranked system. Our models come close in performance in Task A, but Tasks B and C are where the true difference in performance comes to light.

Another form of limitation is how we handle the data preprocessing or data cleaning stage. In our study, we completely remove any forms of hashtags and emojis from the raw content of the post during data preprocessing. This is limiting because studies like those done by Eisner et al. [34] have showcased how techniques like hashtag segmentation and emoji2vec can boost performance in text classification tasks. They directly facilitate the classifiers to learn the difference between sexist and non-sexist posts better and perform better in all subtasks.

Chapter 5

DAP-LeR-DAug Techniques for enhanced Sexism Detection

In this chapter[1], we intend to explore the performance of different modern techniques on BERT-based models in the EDOS shared task. We explore four such different techniques, namely, Domain Adaptive Pretraining (DAP), Learning Rate Scheduling (LeR), Data Augmentation (DAug), and an ensemble of all three. By "BERT-based" models, we refer to those classifiers which have Bidirectional Encoder Representations from Transformers (BERT) as their base and improve upon it with their own methods and techniques. For example, HateBERT is BERT with a lot of pretraining with Hate Speech data. We select other such variations of BERT which suit our problem and have proven effective performance in text classification tasks.

The results of the study show that each of these techniques improves performance differently for each subtask due to its unique approach, which may be suited to a certain problem more. The ensemble model performs the best in all three subtasks. All in all, this chapter explores the potential of DAP-LeR-DAug techniques in detecting sexist content. The results of this study will highlight the strengths and weaknesses of the three different techniques with respect to each subtask.

5.1 The Transformer & BERT

Before discussing the individual BERT-based classifiers in detail, it is imperative to understand what BERT is. BERT stands for Bidirectional Encoder Representations from Transformers and was introduced by Devlin et al. [3]. As the name suggests, BERT is largely based on Transformers [2]. Transformers are a deep learning model in which every output element is connected to every input element, and the weightings between them are dynamically calculated based on their connection. This is what attention stands for. Before the introduction of Transformers, research in NLP tasks heavily relied on the use of Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN) [35]. Although both these types of neural networks perform competitively, the Transformer is considered a significant improvement because it doesn't require sequences of data to be processed in any fixed order, whereas RNNs and CNNs do. Since transformers can process data in any order, they allow for training on a larger amount of data, something that was not feasible before. This, in turn, facilitated the creation

[1]This chapter is an extended version of the published work: `https://aclanthology.org/2023.icnlsp-1.5/`

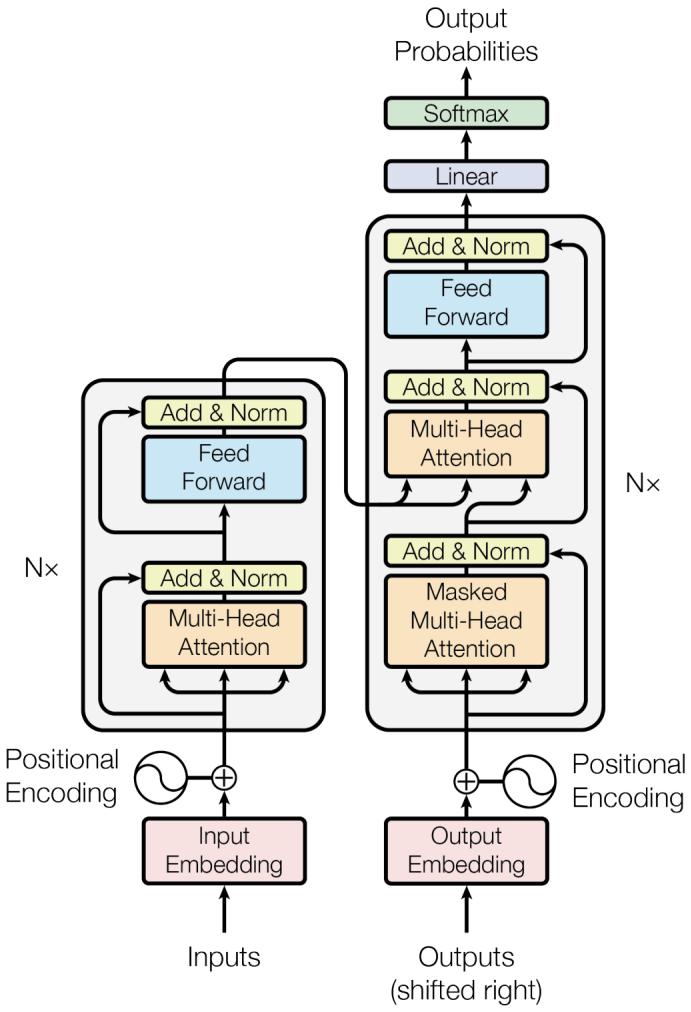


Figure 5.1 Transformer Architecture. Image adopted from Vaswani et al. [2]

of pre-trained models like BERT, which was trained on massive amounts of language data prior to its release.

As visible in Figure-5.1, Transformer models work by encoding the input text into a sequence of vectors using an encoder. This encoding is carried out using a self-attention process, which enables the model to learn the dependencies between the words in the phrase. After the input sentence has been encoded, the model decodes it into a series of output tokens. This decoding is also using a self-attention process. The attention mechanism is what enables transformer models to learn long-range dependencies between words in a phrase. When decoding the output tokens, the attention mechanism prioritises the most relevant words from the input text. Each layer in the Transformer architecture comprises two sub-layers: multi-head self-attention and a feed-forward network. The multi-head self-attention layer allows the model to attend to different parts of the input sequence, while the feed-forward network performs non-linear transformations on the output of the self-attention layer.

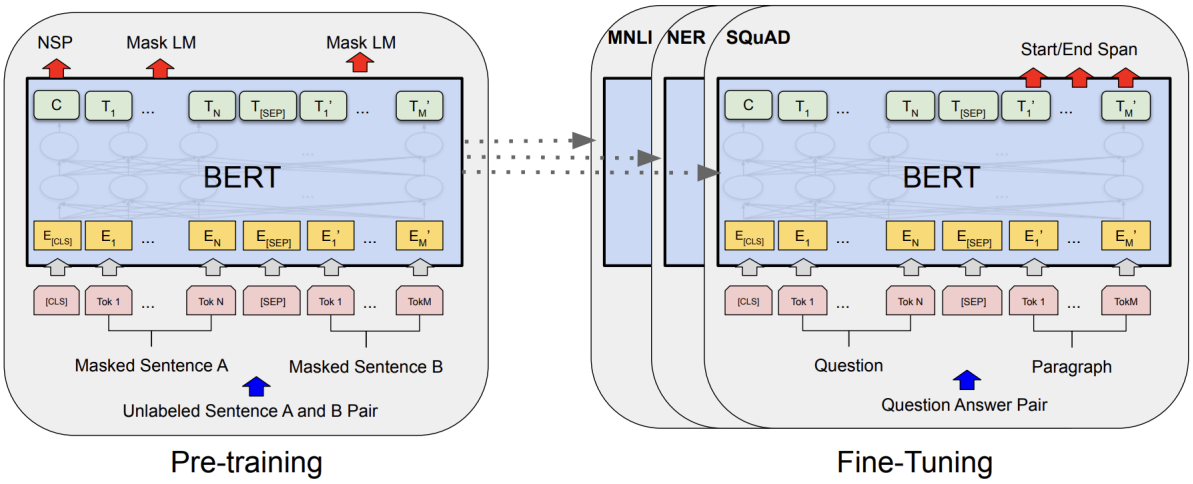


Figure 5.2 BERT pretraining and fine-tuning procedures. Image adopted from Devlin et al. [3]

The Transformer architecture has reshaped the field of NLP. It has shown state-of-the-art performance on multiple tasks, and its attention mechanism has become a fundamental component of the majority of the neural network architectures that have followed since. Consequently, it also lays the foundation for BERT. BERT, which is a pre-trained language model, makes use of the Transformer architecture to generate high-quality text representations. It is pre-trained on a large corpus of unlabeled text data, using two objectives: Masked Language Modelling (MLM) and Next Sentence Prediction (NSP) as part of its training process, as visible in Figure-5.2.

MLM is a technique where certain tokens in a sentence are randomly masked, i.e., replaced with a special [MASK] token. The model then has to predict the original token from the context of the other tokens in the sentence. BERT uses this technique during pre-training to build a deep understanding of how different words are related to each other in context. On the other hand, NSP is a learning task where the model is given two sentences and has to predict whether they are contiguous in a text or not. BERT uses this technique during pre-training to understand the relationship between different sentences and how they fit together in a larger piece of text. Together, MLM and NSP facilitate BERT in gaining a deep understanding of the meaning as well as the structure of natural language text. This pre-training process is what allows BERT to be fine-tuned for a variety of downstream tasks like Machine Translation, Question Answering, Text Classification, etc.

As BERT is an encoder-only model, its architecture can be imagined as a stack of encoders lying on top of each other. However, the number of encoders can vary according to the needs of the task. For example, the base version of BERT has 12 stacked encoders, while the large version of BERT has 24 encoders stacked on each other. Figure-5.3 paints the picture of BERT architecture aptly. Another one of the key features of the BERT architecture is its bidirectionality. Traditionally, language models have

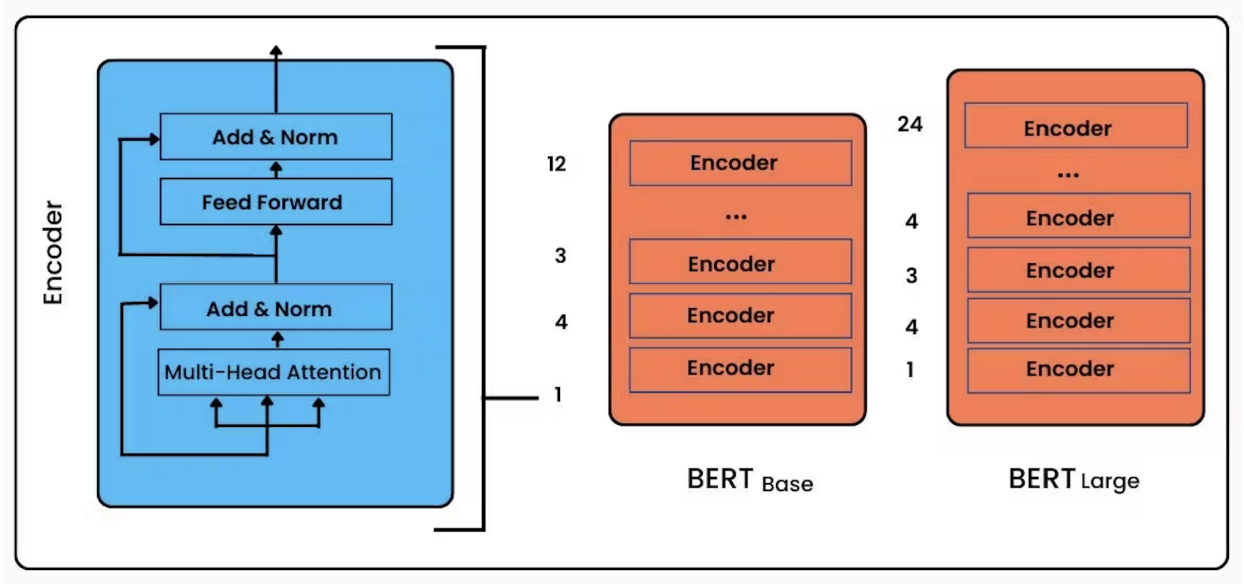


Figure 5.3 BERT Architecture

always processed text in a unidirectional manner, i.e., from right to left or left to right. BERT processes text in both directions, and this allows for a better and deeper understanding of the meaning and context of words. As already mentioned, its transformer base allows BERT to capture long-range dependencies between words, which helps in a critical understanding of the text. Furthermore, BERT generates contextual word embeddings. This implies that the meaning of the word is directly influenced by the words that precede or succeed it in the sentence, and capturing this information helps in a better understanding of the nuances and complexities of the language.

Due to its various positives and state-of-the-art performance on multiple NLP tasks, researchers over the years have come up with various ”flavors” of BERT. Each of them caters to a different sub-problem. For example, DistilBERT [36] is one such version of BERT which requires less computational resources for pre-training and fine-tuning and still gives a competitive performance in many NLP tasks. Similarly, there are other versions of BERT which are developed to counter specific problems like hate speech detection, legal document classification, etc. We select a few of them which we consider apt for our study in this chapter and discuss them in detail in the coming section.

5.2 BERT-based classifiers

As hinted at earlier in the chapter, we select certain BERT-based classifiers to experiment with our techniques. In this section, we look at them individually in detail.

5.2.1 RoBERTa

RoBERTa [4] stands for Robustly optimized BERT-pretraining approach. It is based on an enhanced BERT-pretraining approach, which improves performance on various natural language understanding tasks. It is done through extensive training with larger mini-batches and more data, which results in improved language representations.

The key catch in the pretraining approach is that the NSP training objective is removed, and it is only focused on the MLM objective. This allows RoBERTa to improve upon the masked language modelling objective compared with BERT and leads to better performance in some of the downstream tasks, which makes it a very competitive classifier for our task.

5.2.2 HateBERT

HateBERT [37] is a version of BERT which has been pre-trained on a large English dataset of comments ($>$ 1 million posts) from banned Reddit communities. These communities were banned on account of being offensive, abusive, or hateful in general. A detailed comparison between HateBERT and BERT, when it comes to detecting hate speech or offensive and abusive language, shows that the former outperforms the latter. This makes it very suitable for our task, i.e., detection of sexism, which is a form of hate speech prevalent in online forums.

5.2.3 BERTweet

BERTweet [38] is a mixture of the two flavours that we have described above. It is trained on a very large corpus of English tweets, and it was trained with the RoBERTa procedure, i.e., pretraining with MLM objective only and dropping the NSP objective. Experiments have shown that BERTweet outperforms strong baselines like RoBERTa and other bert-based state-of-the-art models on NLP tasks based on Twitter datasets like NER, POS tagging, and Text Classification. All of this makes BERTweet a more than suitable choice of classifier for our task.

5.3 DAP-LeR-DAug Techniques

Before moving forward with the system architecture and experiment details, it is imperative to understand the core of the study, i.e., the techniques which will be used to outperform the baselines of the BERT-based models.

5.3.1 DAP

DAP refers to Domain Adaptive Pretraining. The organizers of the EDOS task [1] had also provided an extra dataset of 2 million unlabeled posts from Gab and Reddit. The organizers claimed that these posts were scraped from related communities that had often been flagged for sexist remarks. We utilize this extra dataset with the MLM objective. This is because pretraining with this dataset would hold

the most promise for enhancing the performance of our BERT-based classifiers. The first and foremost step is to use the correct tokenizers and pre-process tokens that may not contribute a lot semantically to the sentence. For example, tokens that represent [USERNAME], [HASHTAGS], [MENTIONS/TAGS], [URLs], etc., can be removed to improve efficiency and accuracy. Then, we move on to create the masked sentences by randomly masking a certain percentage of the sentence. The model then learns by predicting the masked tokens based on the surrounding context. The goal is to minimize the loss between the actual masked tokens and the masked tokens predicted by the model.

By being subjected to extensive and diverse linguistic contexts from the unlabeled dataset during MLM pretraining, the individual classifiers would gain a robust and deep understanding of the language patterns and nuances that generally exist in a sexist context. This enriched linguistic foundation would form the cornerstone for improved comprehension of text, ultimately enabling the models to capture the subtle linguistic cues and contextual variations which are inherent in sexist content. During the fine-tuning stage, the model's already adept language representations would seamlessly adapt to the specific domain of sexism detection. This dual-stage process would harmonize the model's universal language understanding with domain-specific features, resulting in heightened discriminatory power to accurately identify and classify sexist text instances. The fusion of pretraining's broad language expertise and fine-tuning's task-specific tailoring would equip the model with a well-rounded ability to identify and categorize nuanced and varied forms of sexist content across the different classes of sexist content. All in all, by gaining a better idea of the contextual relationships from posts on sexist forums, the model should ideally perform better than without DAP.

5.3.2 LeR

LeR refers to Learning Rate Scheduling. Learning Rate is one of the most crucial hyperparameters when it comes to training neural networks. They directly impact the duration and effectiveness of the training process. A learning rate that is too low can cause the model to train very slowly or even stall. This is because the updates to the model's weights are very small, which means guiding the optimizer to the minima happens very gradually. There is also a high risk of the optimizer taking too long to converge or getting stuck in a local minima or plateau, which is not the optimal behaviour. On the contrary, if the learning rate of the model is large, the training process will be very swift, but it increases the risk of the optimizer overshooting on the minima because of the drastic weight updates. Moreover, as the weight updates are drastic, there is also a risk of the weights overflowing beyond their optimal values. Therefore, reaching the optimal learning rate is a tradeoff between the convergence rate and overshooting. We do not want it to be too small so our model can learn reasonably quicker, and at the same time, we do not want it to be so high that the model overshoots the optimal minima.

All in all, the aim of Learning Rate Scheduling is to enhance model performance by dynamically adjusting the step size during training. This technique accelerates convergence by initially allowing larger parameter updates, ensuring quicker progress towards the optimal solution. As the training advances, the

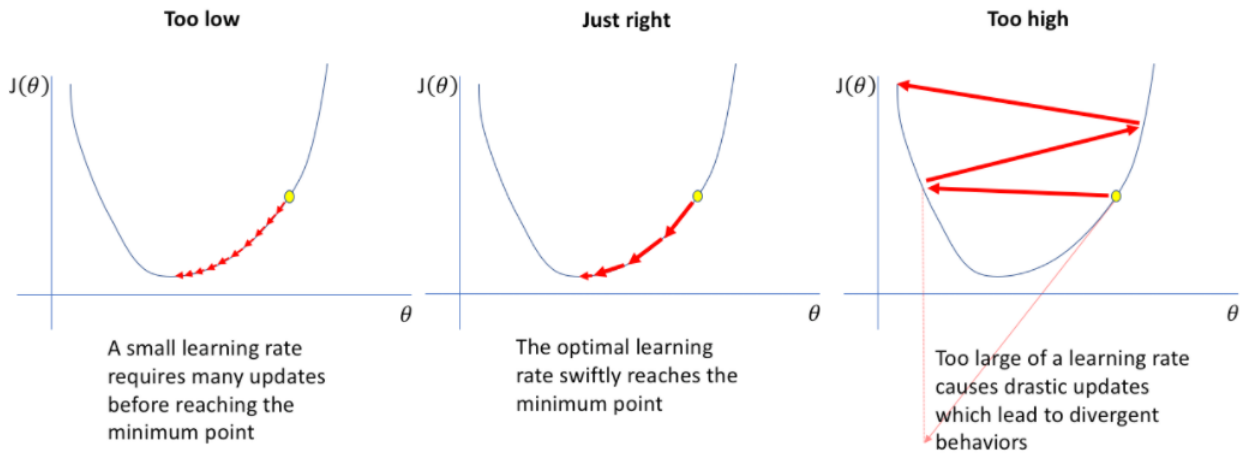


Figure 5.4 Importance of optimal Learning Rate. Adopted image*

* https://www.jeremyjordan.me/nn-learning-rate/

learning rate is reduced, which stabilizes optimization and prevents overshooting. By navigating the loss landscape more effectively, learning rate scheduling helps in evading the local minima and in improving generalization by mitigating noise fitting. Even though this technique does not contribute linguistically in terms of word embeddings or contextual understanding of the domain, it still proves to be a crucial step for model training and can enhance model performance. This technique is particularly valuable for stabilizing training with large batch sizes, adapting to data characteristics, and achieving fine-tuned results in transfer learning scenarios. In essence, learning rate scheduling fine-tunes the learning process itself, fostering quicker convergence, robustness, and overall improved model performance.

In our study, we experiment with four different types of LeR techniques.

5.3.2.1 Step Decay

It is the simplest scheduling policy. It reduces the learning rate by a constant factor every few epochs. It can be formulated as follows:

$$lr = lr_0 \cdot d^{(floor(1+n_i)/s)} \tag{5.1}$$

where lr is the new learning rate, lr_0 is the initial learning rate, d represents decay rate, s stands for step size, and n_i refers to the index of the epoch.

5.3.2.2 Exponential Decay

This policy reduces the learning rate using an exponential function. It is formulated as:

$$lr = lr_0 \cdot e^{-d \cdot n_i} \tag{5.2}$$

where d is the decay rate and n_i refers to the index of the epoch.

5.3.2.3 Cosine Annealing

This policy reduces the learning rate using a cosine-based schedule. It is defined as:

$$lr = lr_{min} + 0.5 \cdot (lr_{max} - lr_{min}) \cdot \left(1 + \cos\left(\frac{n_i}{n} \cdot \pi\right)\right) \tag{5.3}$$

where lr_{min} is the minimum learning rate, lr_{max} is the maximum learning rate, n_i is the current epoch, and n stands for the maximum number of epochs.

5.3.2.4 One Cycle LR

This technique schedules the learning rate using a cyclic policy. It anneals the learning rate from an initial learning rate to some maximum learning rate and then from that maximum learning rate to some minimum learning rate much lower than the initial learning rate. It changes the learning rate after every batch. This policy has been shown to train models with very fast convergence rates, ultimately earning the term "super-convergence" [39].

5.3.3 DAug

DAug stands for Data Augmentation technique. As discussed multiple times earlier, across chapters, the EDOS dataset is highly imbalanced for each subtask. For example, the majority class in tasks A and B has more than 3 times the number of data instances as compared to the minority class. For task C, the case is even worse, where there are minority classes with not even 100 instances, while some majority classes have more than 700 instances. Such an imbalance in the dataset can make the best of classifiers biased towards the majority class. There are various different techniques to counter that. We have discussed naive dataset resampling techniques in Chapter-4 already; in this chapter, we focus on Data Augmentation techniques. Data Augmentation concerns itself with creating new data for classes with a limited number of data instances. It can significantly enhance a dataset with limited sexist posts by generating diverse variations of existing examples. Techniques like synonym replacement, paraphrasing, and introducing minor textual perturbations help create additional instances of sexist content. By simulating different linguistic expressions and contexts, data augmentation enriches the dataset, which can, in turn, improve the model generalization and performance, even when the original sexist instances are sparse. This augmented data enriches the training dataset, improving the model's generalization and performance. EDA is demonstrated to be remarkably effective across various text classification tasks, showcasing its ability to alleviate the challenges posed by dataset imbalance and contributing to more robust and accurate text classification models. This is why the EDA approach will be beneficial for our study for all three subtasks.

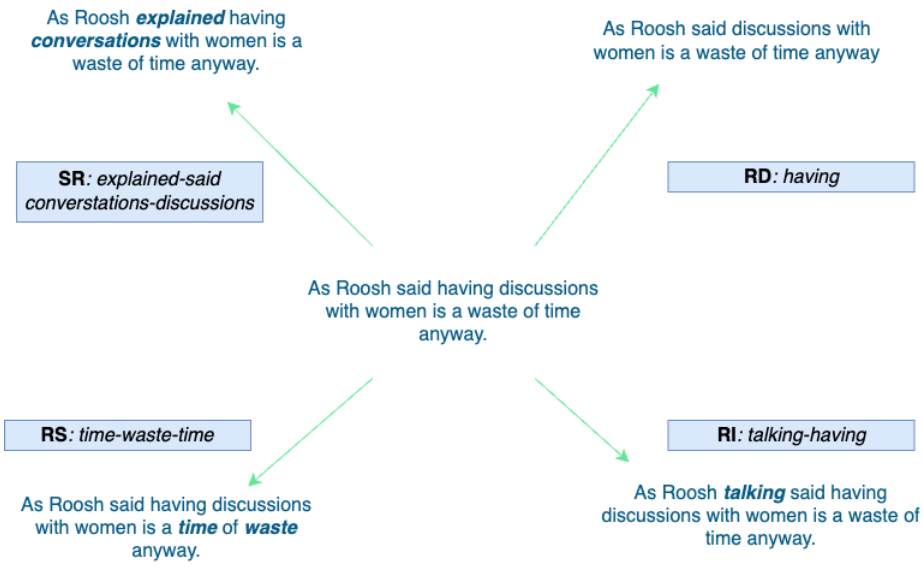


Figure 5.5 DAug example: How EDA framework generates 4 new instances from one

In our experiment, we make use of a similar approach as taken by the authors of Easy Data Augmentation (EDA) [15]. By introducing Synonym Replacement (SR), Random Insertion (RI), Random Swapping (RS), and Random Deletion (RD) to the text, the EDA framework generates diverse instances of the original text. All these techniques are applied to words that are not stopwords (like *the, a, an, it, was, is,*, etc.) so that the new data instance is semantically similar but diverse as well in the true sense. SR involves selecting random word(s) in the sentence and replacing them with their synonyms. RI involves selecting a word at random in the sentence and then inserting its synonym at a random place in the sentence. RD involves removing a randomly selected word from the sentence, while RS involves selecting two or more words in the sentence and swapping their positions in the sentence. Figure-5.5 shows how our DAug approach works on an example of sexist text taken from the EDOS task dataset.

For generating data, we decided to choose RoBERTa as it gave semantically closer data to the actual data when compared with the data generated by HateBERT and BERTweet. We limited the augmentation probability (α) to 0.3 as above this threshold, the system generates very noisy data, which can lead to a loss of semantics and an overall reduction in the performance of the classifiers. For SR, replacing too many words in the sentences of a post can cause the identity of the post to change and, therefore, cause label changes in more fine-grained classification subtasks like B and C. The value of α does not matter a lot when it comes to RI because the relative order and context of the sentence remain the same. For RS and RD, the value chosen for α becomes very sensitive, as swapping too many words can cause loss of semantics and context while deleting too many words may cause the entire meaning of the sentence to change.

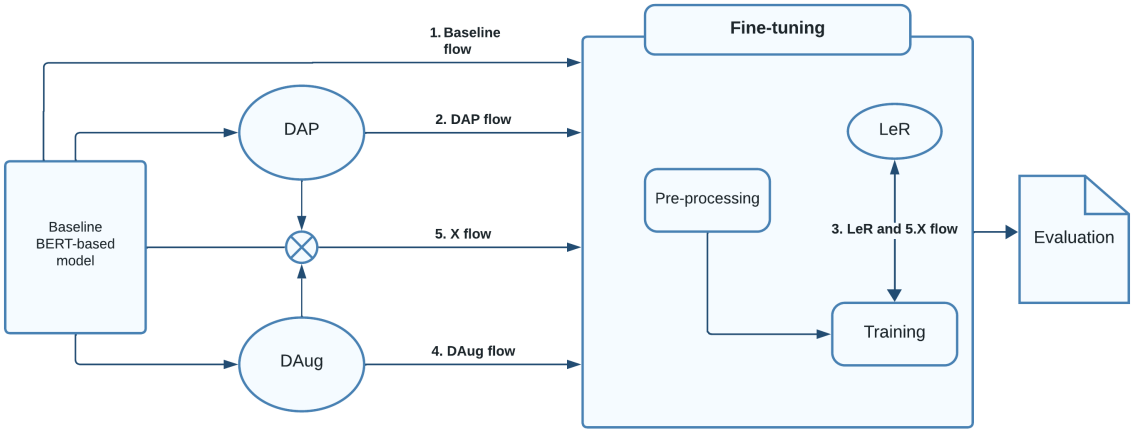


Figure 5.6 System Architecture for enhanced sexism detection using DAP-LeR-DAug techniques

5.4 System Architecture

The system for our study can be broken into five different parts. Firstly, we have the baselines, then the three individual DAP-LeR-DAug models, and finally, an ensemble model of all three techniques. For a quick overview, Figure-5.6 can be referred to.

5.4.1 Baselines

As discussed earlier, the three BERT-based classifiers: RoBERTa, HateBERT, and BERTweet will serve as our model baselines in this study. As can be seen in figure-5.6, the baseline flow is quite intuitive. We use the classifiers as they are, without any modification or technique, and fine-tune them on the EDOS task dataset.

5.4.2 DAP-LeR-DAug models

We experiment with each technique as its own different model. DAP model involves domain-based pertaining before the actual fine-tuning while DAug flow is concerned with balancing the dataset by having an increased number of data instances in the minority classes. LeR model only comes into the picture during the model training stage, as it is concerned with optimizing the training process and not with increasing the language understanding of the model directly.

5.4.3 X model

The final or the X model of our system is basically a combination or an ensemble of all the three unique techniques we have discussed so far. The ensemble capitalizes on the complementary strengths of each technique, effectively navigating linguistic complexities through pre-trained domain understand-

ing, fine-tuning with task-specific context, and enriched data diversity. This holistic approach promotes greater robustness to nuances in sexist content and addresses challenges posed by limited labelled data. Ideally, this should outperform the individual technique models and ultimately lead to the best performance when it comes to classifying sexist content.

5.4.4 Model Training

As discussed multiple times already, one of the major challenges with the EDOS task dataset is that it is imbalanced. This is taken care of through the DAug technique, but in order to do justice to other techniques, so as not to make their classifiers biased toward the majority class, we had to consider the resampling approaches discussed in Section-4.2. To briefly recapitulate the important points, in Undersampling, we remove a certain number of data instances from the majority class to make sure the classes are more or less balanced. On the opposite hand, in Oversampling, we replicate data instances of the minority class until we have achieved balance among all the classes in the dataset. We have seen in Section-4.4.5 that Oversampling not only performs worse than Undersampling, but its performance is even inferior to using the dataset as it is, i.e., imbalanced. Therefore, for the model variations that do not include the DAug technique, i.e., Baseline flow, DAP flow, and LeR flow, we use Undersampling to balance the dataset. This helps in training all the different flow classifiers with as little bias to any particular class as possible.

For the fine-tuning setup, we again refer to Section-4.4.5, wherein we noticed that an 85-15% training-validation split works best for training the classifiers. The organizers of the EDOS task [1] had provided separate data for testing, and thus, it would be better to test our classifiers using the same instead of creating our own test dataset. This would not only help compare our baselines against the baseline scores set by the task paper but also compare them with other top-performing teams to gauge just how effective the techniques we intend to explore are. During the training phase, we do simple pre-processing. Most of the pre-processing is handled comfortably with the appropriate tokenizers of the different BERT-based models we have considered. However, we take extra care on our own end to remove tokens that do not contribute semantically to the system. For example, hashtags, emojis, noisy tokens like ”heyyyyyy”, ”yooooooooo”, ”brrrooooooo”, etc. For training our classifiers, we set epochs to 10 and batch size as 16. Once training is completed, we proceed to evaluate the classifiers.

5.5 Results

For evaluation, we make use of the macro-average F-1 scores. This will help in comparing our models' performance with that of the baselines in the task paper and other top-ranked teams in the task. A major reason for adopting the macro average F-1 score could be how it treats each class of the dataset appropriately during evaluation. This is extremely beneficial in cases similar to our study, where the dataset is imbalanced.

Model	*Task-A: 2 class*	*Task-B: 4 class*	*Task-C: 11 class*
RoBERTa-base	83.22	59.77	34.01
RoBERTa-DAP	84.67	62.23	36.44
RoBERTa-LeR	82.45	63.97	38.21
RoBERTa-DAug	83.59	63.87	39.89
RoBERTa-X	**85.09**	**65.89**	**40.23**
HateBERT-base	82.13	60.56	33.84
HateBERT-DAP	82.55	62.23	35.19
HateBERT-LeR	82.14	64.12	37.77
HateBERT-DAug	82.34	64.01	38.09
HateBERT-X	82.78	64.53	38.34
BERTweet-base	84.01	61.12	30.01
BERTweet-DAP	84.33	62.01	32.71
BERTweet-LeR	84.03	62.89	34.66
BERTweet-DAug	84.12	62.88	34.81
BERTweet-X	84.39	63.45	35.22

Table 5.1 Macro Avg. F-1 Scores of DAP-LeR-DAug-X Classifiers on all subtasks

From the results in Table-5.1, it can be noticed that the ensemble model with RoBERTa as baseline performs the best on the evaluation dataset. There can be multiple reasons for that, but the primary reason has to be the architecture of RoBERTa and the fact that we have utilized RoBERTa-base for our Data Augmentation technique. Even though models like HateBERT and BERTweet have a good understanding of hate speech beforehand, thanks to their architecture and pre-training, it is possible that techniques like DAP and DAug did not help them as much as they helped RoBERTa. This could be down to the fact that they have already been exposed to a wide variety of hate speech data and our techniques did not increase their contextual understanding or vocabulary a whole lot.

5.5.1 X-factor

Another important observation to note is that the X model performs the best for each baseline. All three techniques, when employed together, can cause the model to perform best. It is in line with previous studies like Ruta & Gabrys 2005 [40], Zhang et al. 2014 [41], which prove that the ensembling of multiple models can potentially prevent dire mistakes made by a single model alone. It is also intuitive to expect the X model to be the best performer as it has been pre-trained heavily on 2 million posts for adapting to sexist content domain, has got augmented data with minority classes also being represented adequately, and finally, can train optimally thanks to the learning rate scheduling technique. All in all, the three techniques complement each other and bring out the best when used together.

5.5.2 DAP vs. LeR vs. DAug

We really notice the individual impact of the techniques when we look at the results task-wise.

For task A, we can see that the DAP technique improves the score the most on the baselines. This is intuitive, too, as for a simple binary classification problem, having more embeddings and wider vocabulary to work with makes it even easier for the model to figure out if the content is plain sexist or not.

The LeR approach works best with the increasing complexities of the task. It works better for Tasks B and C than it does for Task A. The effect of optimal convergence is noticed more easily when there are more classes involved in the task. It performs the best for Task B and also performs well for Task C.

It is not that LeR's performance drops in Task C but that DAug works too well for Task C and easily outshines the LeR technique. We have established multiple times in this study that the dataset is imbalanced, and this imbalance increases with the increasing complexity of the task. Undersampling can only work so well when we have to deal with 11 classes in Task C, where the majority of the classes are very under-represented. This is where Data Augmentation comes in handy. By creating more data instances for the minority classes, we are able to give the model more data to work with and thus increase its performance in classification.

5.5.3 LeR policies

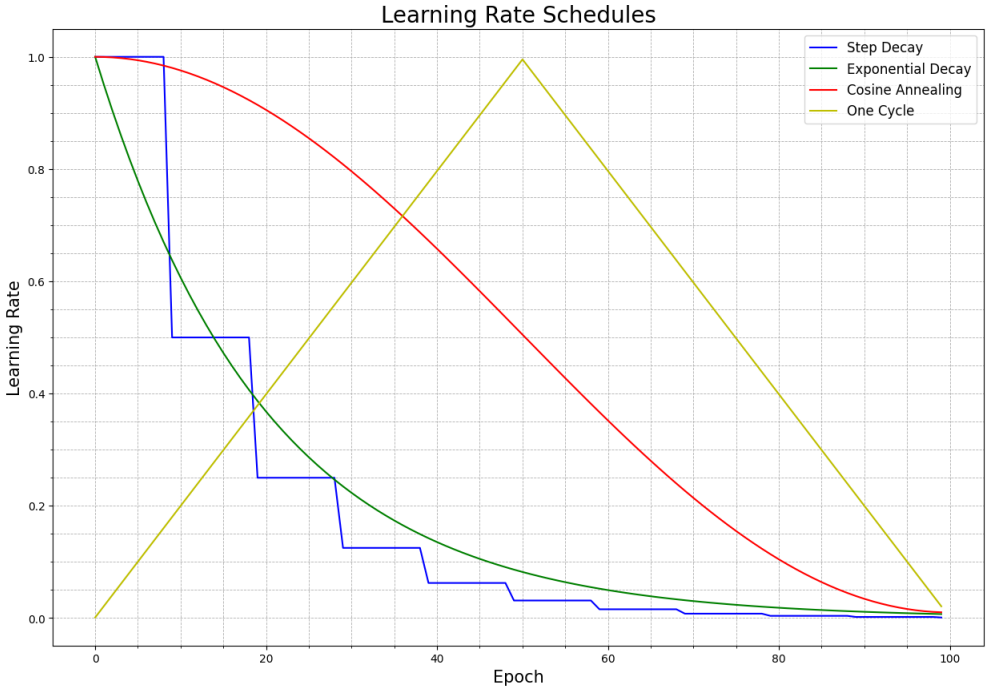


Figure 5.7 Performance of different LeR policies. The values are taken randomly to visualize the difference and do not represent the actual values observed during model training.

As discussed in Section-5.3.2, we experiment with four different LeR techniques in our study. To recap, they are Step Decay, Exponential Decay, Cosine Annealing, and One Cycle scheduling. They

performed more or less similarly (as visible in Figure-5.7), with the only difference when it came to the X or the ensemble model. In that case, Cosine Annealing edges out other approaches, and this may be due to the fact that the X model has a lot going on underneath the layers. Not only does it have more contextual embeddings thanks to DAP, but it also has more data to work with because of DAug. These rising complexities require complex learning rate scheduling policies like that of Cosine Annealing.

5.5.4 Performance in Shared Task

Model	***Task A***	***Task B***	***Task C***
Best Baseline	82.35	59.26	31.71
RoBERTa-X	85.09	65.89	40.23
Top-ranked	87.46	73.26	56.06

Table 5.2 Comparison of the performances of the Best Baseline model in task paper, the top-ranked systems for each subtask, and our best performing model: RoBERTa-X

Lastly, we compare our best-performing model, i.e., RoBERTa-X, with the best baseline model of [1] and the top-ranked systems for each task. Thanks to the ensemble of our effective techniques, we are able to comfortably outperform the best task paper baseline model in each of the subtasks. We were not able to beat the top-ranked system in any subtask, even though we came close. However, it must be noted that for [1], no single approach was the top-ranked among all the three subtasks. The top-ranked system score for each subtask in Table-5.2 is from a different participating team. On the other hand, through our study, we were able to create a single ensemble approach that at least beat the best baseline if it did not perform the best. Comparing our scores with the task leaderboard [2], our RoBERTa-X model would stand in the top 30% submissions in Task A, top 25% submissions in Task B, and top 40% submissions in Task C.

5.6 Conclusion

Through this study, we were able to explore the effectiveness of the DAP-LeR-DAug techniques when it comes to classifying hate speech in the form of sexism. We were able to demonstrate that each technique works well with a specific task, and when employed together in the form of an ensemble, we are able to get the best performance out of them, irrespective of the BERT-based model being used as the baseline. This goes on to show that the scores achieved were not coincidental, and the techniques indeed complement each other in a good way.

Although the DAP-LeR-DAug techniques do not perform the best for any specific subtask when compared with top-ranked systems, it should be pointed out that they do surpass the scores achieved by the best baseline model in [1] quite comfortably. Nevertheless, there are a lot of ways to improve performance, and we discuss them below.

[2] `https://github.com/rewire-online/edos/tree/main/leaderboard`

5.6.1 Limitations

Like any research study, this, too, has its own share of limitations. Overcoming some of these would directly result in better scores for each subtask, while some others may increase the training time but nonetheless will improve the performance of the models.

First of all, we have used only the base versions of the BERT-based models. If not for the restraint on computational resources, we could have used the large, extra-large versions of the baseline models. The larger vocabulary and increased number of parameters would directly help to achieve better scores in all three subtasks.

Another way to improve performance is using more data for DAP. The suggestion is indeed greedy but will improve the performance nonetheless. Similarly, we could experiment with other forms of hyperparameter tuning apart from LeR alone. Some of them could be optimizing the dropout rate, loss functions, weight decay, and action variation functions. The impact of tuning such hyperparameters may not be very noticeable, but it will optimize our model training nonetheless.

We can also experiment with different Data Augmentation techniques. In this study, we have only used the EDA framework, but there are more complex ways to augment the data. For example, the Language-Model-Based Data Augmentation (LAMBADA) framework [17], where a state-of-the-art model is fine-tuned for the task using the limited data and then generates synthetic labelled data which can be used to expand the original dataset. For a much simpler alternative, Back-translation can also be considered. For that, we first translate the English sentence into a certain language and then back to English. This is an easy and effective way to generate more samples for under-represented classes and ultimately balance the dataset.

Finally, we can try to improve upon the pre-processing stage as well. As discussed in Section-4.5.1, getting rid of all the emojis and hashtags is costly in the sense that they contribute to the sentence a lot semantically. Eisner et al. 2016 [34] have showcased how techniques like hashtag segmentation and emoji-2-vec can be used to enhance model performance. Emojis converted to vector embeddings and combined with word embeddings will facilitate the models to learn more about any given sentence. This improvement is definitely worth considering as emojis and hashtags are used a lot on social platforms nowadays and they contribute to the context and semantics of the text.

Chapter 6

Text-2-Wiki: Template driven Wikipedia Article Generation

Text generation has been one of the most vital points of research in NLP. It serves as a cornerstone for the development of advanced natural language understanding systems. Such systems can then be applied to populate information in domains and languages that have very limited presence in the public domain. In this study, we aim to populate our native language, i.e., Hindi's Wikipedia, on a certain domain by making use of a template-driven and summarization-based approach. We have selected Diseases and Medical Conditions as our domain for the study as they are a very important topic for readers of any language.

The gist of the study is to scrape and collect data from public online resources and then clean it accordingly. Then, we summarize the scraped data and organize it to give it structure. We follow it with a translation and transliteration stage to convert the data into Hindi. This stage involves both machine and manual translation and transliteration. Lastly, we end the study by drafting up a template in which the data can be filled as fill-in-the-blanks. This ensures the automatic generation of a large amount of articles in accordance with the Wiki structure.

6.1 Motivation

Wikipedia, the free online encyclopedia, stands as a monumental testament to the collaborative power of global knowledge sharing. Launched in 2001, it has evolved into an unparalleled digital repository, offering a vast compendium of information spanning subjects as diverse as science, history, and popular culture. Driven by a legion of volunteer editors, Wikipedia has become a symbol of the democratization of information, revolutionizing the way individuals access and contribute to the collective wisdom of humanity. Users on Wikipedia collaborate in a structured and organized manner to publish and update articles on numerous topics, which makes Wikipedia a very rich source of knowledge. English Wikipedia has the most amount of information available ($>$ 6.7 million articles) out of all the languages on the platform. On the other hand, Hindi Wikipedia has approximately only 160k articles. Moreover, the same article in Hindi can be vastly different from its English version and generally contains less information. This poses a problem for native Hindi speakers who are not proficient in English. Therefore,

having the same amount of information in Indian Languages will help promote knowledge among those who are not well-versed in English.

The usual process of publishing articles in Global Wikipedia is a very time-consuming process as it is entirely manual. To get the amount of information in native Indian Languages up-to-speed with the amount of information in English, automating the whole article generation process is the best option. In the coming sections, we will be exploring a stage-wise approach ranging from Data Collection to Summarization and Translation and finally ending with Template Creation and Article Generation as visible in Figure-6.1. This approach ensures the efficient and automatic generation of a large amount of content in Hindi Wikipedia in less time. With the help of this study, we will demonstrate how to successfully generate more than a thousand articles in Hindi Wikipedia with ease compared to the manual, time-consuming process.

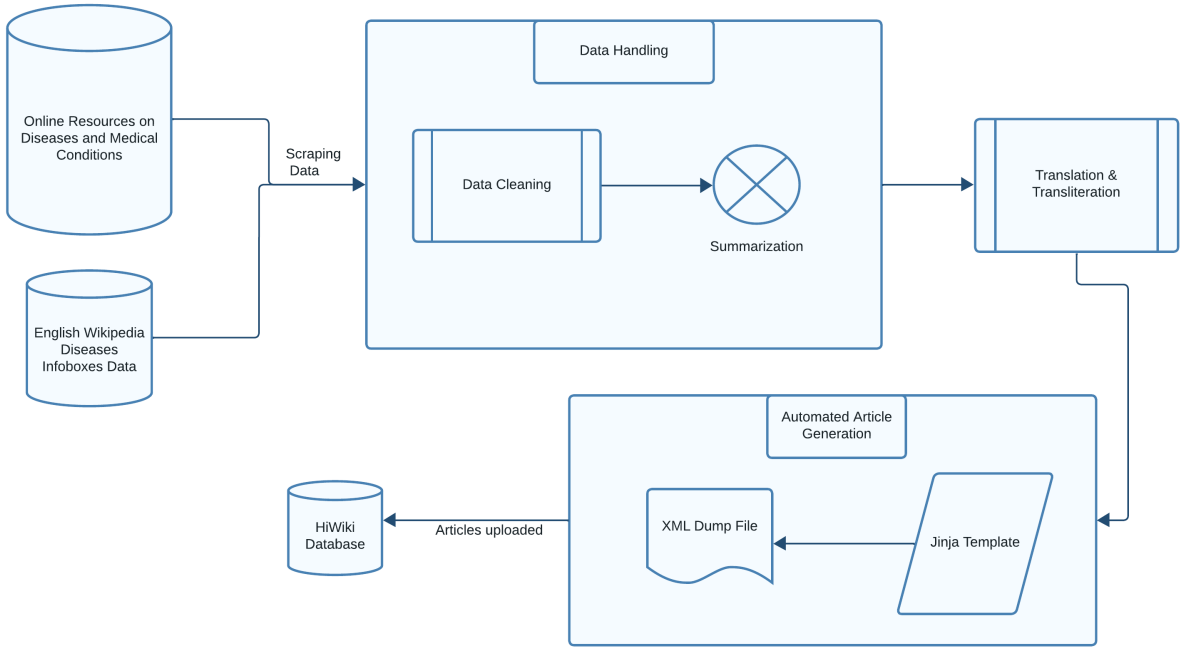


Figure 6.1 Text-2-Wiki: Presenting an alternate way to create Wiki Articles

6.2 Data Collection

Data is arguably the most important feature of any research study, and having a dedicated stage for collecting it has become even more necessary. As mentioned earlier, we had selected *Diseases and Medical Conditions* as the domain for our articles. There were multiple dedicated resources available

online for scraping data related to the domain.

Firstly, Simple Protocol and RDF Query Language (SPARQL)[1] queries were explored to learn about scraping data from Wikipedia specifically. SPARQL is a Resource Description Framework (RDF) language, which refers to a semantic query language that is able to retrieve and manipulate data stored in RDF format. The idea was to gather data about diseases from English Wikipedia. However, the diseases were not stored in specific categories and had varying sections. This made the data too unstructured, which would have caused a problem later on in the Template Creation stage. Having a different template for every other disease or condition would beat the entire point of the study, i.e., to automatically generate a large number of articles in a time-efficient manner.

Another option to explore in English Wikipedia itself was the translation of the articles on Diseases and Conditions into Hindi directly. However, this approach also had its fair share of issues. Firstly, even the English Wikipedia articles on the Diseases domain are not as rich in information as some of the articles on online medical websites, and the depth of information varied wildly among the different articles on Wikipedia. Moreover, the number of Diseases written about in Wikipedia is limited. It was visible that many diseases did not have a dedicated page in English Wikipedia. Therefore, choosing this approach would have limited our study to a very narrow use case, and the articles generated would have failed to make the desired impact.

As a result, other resources dedicated solely to the medical field were explored. Web scraping using Selenium was carried out to collect the data. During the scraping phase, websites like PharmEasy and Netmeds were also considered, but due to Copyright restrictions, they were dropped from the list of potential resources. The two major resources that were used for data collection for the study then are:

- Mayo Clinic[2]: Mayo Clinic is a nonprofit American academic medical centre focused on integrated health care, education, and research. The majority of the web scraping was done from this online medical resource.

- National Health Service (NHS)[3]: The National Health Service is the publicly funded healthcare system in England and one of the four National Health Service systems in the United Kingdom. This resource offered an additional way to gain information on more diseases and conditions and, at times, even additional information for the diseases we had already scraped from Mayo Clinic.

All the required data was sourced from the sources discussed above only. The scraped data was saved in a .CSV format file with the following 10 attributes for each disease and condition:

Name, Link, Overview, Symptoms, Causes, Risk Factors, Remedies, Diagnosis, Treatment, Medication

[1] `https://query.wikidata.org/`
[2] `https://www.mayoclinic.org/diseases-conditions`
[3] `https://www.nhs.uk/conditions/`

6.2.1 Infobox Scraping

Infoboxes form a very important part of any Wikipedia article as they help provide concise information in case the reader does not have the time to go through the entire article. Infoboxes for different categories have different formats on the Wikipedia platform. For example, articles in the Diseases categories will have infobox attributes like Causes, Remedies, and Risks, while articles in the Cricketers categories will have Total Runs, Average Runs, and ODI records as the attributes for their infobox format. Even though the Wikipedia articles themselves had limitations when being considered for data collection, their infoboxes were free from any such drawback. They offered a concise way to represent the most important information of the article and were mostly available for all diseases. Figure-6.2 showcases the structure of a typical infobox for an article on Medican Conditions.

As discussed earlier in this section, a considerable number of diseases did not have their own page on Wikipedia. This causes concern in the cases where we have collected data from the Mayo Clinic or NHS on a particular disease or condition, but we do not have any Wikipedia page on the disease. For such conditions, we need an alternate way of creating infoboxes. Firstly, we noticed that three attributes, namely, *Causes, Symptoms, Medication*, were the most important and common infobox attributes among all the infoboxes we had scraped from Wikipedia thus far. It is important to limit this detour to only a few variables because a condition that does not have a Wikipedia page is bound to be far less known than others. Moreover, the data for such diseases on Mayo Clinic and NHS is very limited, too. These three sections were the only ones that had data for every disease scraped from NHS and Mayo Clinic. From there, we went ahead and created a list of all the possible values for those three attributes using the infoboxes we had scraped. Next, for the diseases that did not have their page on Wikipedia, we traversed their data, which was scraped from the Mayo Clinic or NHS, and looked for the values we had just enlisted. If there is a value present in a sentence in the relevant section, we choose that value from the list and select it to create this condition's infobox. For instance, if Migraine did not have a Wikipedia page and we had to create its infobox, we would first traverse the Symptoms section from the data we had scraped from NHS/Mayo

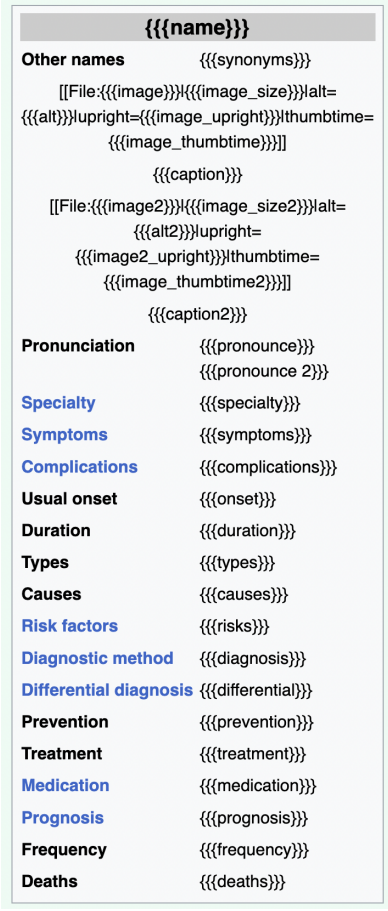


Figure 6.2 Structure of a Medical Condition Infobox

Clinic. Non-stopwords would be searched in the value list we had created, and we would have a match for headache and nausea as they are symptoms of other diseases, too. We would select them and move on to other sections. Finally, we will have collected enough data to create an infobox for a disease that did not even have a Wikipedia page to begin with.

The infoboxes on the English Wikipedia had images as well, so we scraped their URLs, too. Since the images on Wikipedia are affiliated with the WikiCommons[4] project, they can be used in the study without any copyright infringement issues. Ultimately, infobox scraping introduced 13 new variables in our dataset, namely: *Specialty, Symptoms, Usual onset, Duration, Causes, Risk Factors, Diagnostic method, Prevention, Treatment, Prognosis, Frequency, Medication, ImageURL*

These new infobox variables were merged with the scraped dataset and had the prefix *info-* attached to them in order to avoid confusion with variables of the same name (like *Causes, Symptoms*) in the scraped dataset. After merging the scraped and the infobox datasets, we had 23 attributes in total: *Name, Link, Overview, Symptoms, Causes, Risk Factors, Remedies, Diagnosis, Treatment, Medication, info-Specialty, info-Symptoms, info-Usual onset, info-Duration, info-Causes, info-Risk Factors, info-Diagnostic method, info-Prevention, info-Treatment, info-Prognosis, info-Frequency, info-Medication, ImageURL*

6.3 Structuring the Data

As we have scraped entire articles on diseases from the Mayo Clinic and NHS, the values are in the form of sentences and not words like the infobox attributes. For generating articles in the third-person language according to the Neutral Point of View (NPOV) guidelines[5] issued by Wikipedia, it is important to convert the language from its first-person and second-person tone. For this, the data needs to be cleaned and structured properly.

6.3.1 Data Cleaning

Some of the sentences in the scraped dataset were links to other related articles and sections, while others were sponsored content. This sort of noisy data is counterproductive to the final form we intend to present the data in, i.e., neutral third-language Wikipedia articles. In order to clean the data, we ensured to remove links to other sections and articles, footnotes, section and subsection headers, duplicate sentences, and language pertinent to the website, for example, ”Try our voice app”, ”Learn more about our missions here”, etc.

[4] `https://wikimediafoundation.org/our-work/commons/`
[5] `https://en.wikipedia.org/wiki/Wikipedia:Neutral_point_of_view`

6.3.2 Summarization

Many of the diseases, at least the common and widely known ones, have a plethora of information in the articles related to them on Mayo Clinic and NHS. As discussed in the earlier section, many of the sentences have an instructional tone, non-third-person narrative and informal language, which is not ideal for the type of articles we intend to generate in this study. The generated articles need to have structured and formal sentences on diseases and conditions like those found in English Wikipedia. Therefore, there is a need for a selective summarization technique that filters out the unnecessary sentences and retains only those sentences that are structured and contribute to the meaning of the paragraph in the final version of the dataset.

Our technique empahises upon the simple weighted word frequency and sentence scoring approach. It is pretty straight-forward and simple yet effective. An important note to keep in mind is that we obviously emit stop-words when calculating the scores.

$$WF = \frac{f_w}{f_{w^*}} \tag{6.1}$$

where WF is the weighted frequency of the word, f_w is the frequency of the current word in the entire text, and f_{w^*} refers to the frequency of the most occurring word in the text.

$$SS = \sum_{1}^{n} WF\,, \tag{6.2}$$

$$Avg.\,SS = \frac{\sum SS}{s} \tag{6.3}$$

where SS refers to the sentence score of a sentence, n is the number of weighted frequency non-stopwords in the sentence, and WF is the weighted frequency score of the word, s is the total number of sentences in the text.

The concept behind the technique is to set a threshold value of SS, and any sentence whose SS score is more than this value can be extracted to be kept in the final structured dataset. For coherency and context, we pick two to three sentences before and after that sentence, depending on its position in the paragraph. All in all, the dataset had the same attributes as earlier, but their values had been fine-tuned and polished according to the study's requirements. A considerable amount of content was lost, but the informativeness and coherence were retained.

6.4 Translation & Transliteration

Out of the 23 variables of the merged dataset, only 21 needed to be translated/transliterated to Hindi as the other two, namely: *Link* and *ImageURL*, were hyperlinks. For translation, the Google Translate[6]

[6] `https://translate.google.com/`

library was decided upon after also experimenting with Bing translator[7]. We observed that Bing translator provides better performance only when context is involved in sentences and they are of a longer length, but as our sentences were context-free, Google translator gave more than satisfactory performance.

For transliteration, the Indic transliteration library[8] was used. Generally, the transliteration was of a decent standard, but the names of approximately 30 diseases were not transliterated properly. For those, the transliteration was done manually. The transliteration was shown to native Hindi speakers, who were able to comprehend it without any issues. All in all, the transliteration was impressively clear, making the content of the article comprehensible for those with limited to moderate understanding of the disease or medical terminology.

6.5 Template-driven Generation

The next stage of the study involves coding the Jinja2 template. This template, along with Python scripting, would help to generate thousands of articles automatically. In the template, all the structured dataset variables like *Overview, Causes, Medication*, etc. function like the sections of our wiki articles. For the infobox template, the official Wikipedia medical condition infobox format[9] (Figure-6.2) was used. Using the official format gives the advantage of automatic translation when the Wiki Sandbox has other languages' extensions embedded. Wiki Sandbox is a very powerful tool which automatically renders anything with a proper Wikimedia link. This is exactly why we had two different attributes for storing the *Link* and *ImageURL*. The links of the sources will be automatically generated in the *References* section of the Wiki article as sources the article was inspired from, and the image will automatically render in the infobox thanks to its Wiki Commons Image URL.

The very last stage of the study involves the creation of an eXtensible Markup Language (XML) dump that will contain all the thousands of generated articles in a single XML format file. Extensible Markup Language is a markup language and file format for storing, transmitting, and reconstructing arbitrary data. It defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. For our study, this XML header file is created, which has all the XML meta-info of the TeWiki (our copy of the Wiki environment for testing and reviewing articles before publishing on the global Wikipedia) website. Whenever a single article is generated, it is appended to this same header file. This process repeats until the very last article is generated. This automation saves a huge amount of time for writing Wikipedia articles.

[7] `https://www.bing.com/translator`
[8] `https://github.com/indic-transliteration/indic_transliteration_py`
[9] `https://en.wikipedia.org/wiki/Template:Infobox_medical_condition`


माइग्रेन

From tewiki

Contents [hide]
1 अवलोकन
2 लक्षण
3 कारण
4 जोखिम कारक
5 निदान
6 इलाज
7 उपचार
8 दवाई
9 सन्दर्भ

अवलोकन [edit | edit source]

एक माइग्रेन गंभीर धड़कते दर्द या एक स्पंदन सनसनी पैदा कर सकता है, आमतौर पर सिर के एक तरफ। इसमें दृश्य गड़बड़ी भी शामिल हो सकती है, जैसे प्रकाश की चमक या अंधे धब्बे, या अन्य गड़बड़ी, जैसे चेहरे के एक तरफ या हाथ या पैर में झुनझुनी और बोलने में कठिनाई।

लक्षण [edit | edit source]

माइग्रेन, जो अक्सर बचपन, किशोरावस्था या शुरुआती वयस्कता में शुरू होता है, चार चरणों में आगे बढ़ सकता है: प्रोड्रोम, ऑरा, अटैक और पोस्ट-ड्रोम। माइग्रेन के हमले के बाद, एक व्यक्ति एक दिन तक थका हुआ, भ्रमित और धुला हुआ महसूस कर सकता है।

कारण [edit | edit source]

हालांकि माइग्रेन के कारणों को पूरी तरह से समझा नहीं गया है, आनुवंशिकी और पर्यावरणीय कारक एक भूमिका निभाते हैं। ब्रेनस्टेम में परिवर्तन और ट्राइजेमिनल तंत्रिका के साथ इसकी बातचीत, एक प्रमुख दर्द मार्ग, शामिल हो सकता है। कुछ न्यूरोट्रांसमीटर माइग्रेन के दर्द में भूमिका निभाते हैं, जिसमें कैल्सीटोनिन जीन-संबंधित पेप्टाइड (सीजीआरपी) शामिल है। महिलाओं में हार्मोनल परिवर्तन सहित कई माइग्रेन ट्रिगर हैं। हार्मोनल दवाएं, जैसे कि मौखिक गर्भ निरोधकों और हार्मोन रिप्लेसमेंट थेरेपी, भी माइग्रेन को खराब कर सकती हैं। हालांकि, कुछ महिलाएं इन दवाओं को लेते समय अपने माइग्रेन को कम बार पाती हैं। तेज़ रोशनी और सूरज की चकाचौंध से तेज़ आवाज़ के साथ-साथ माइग्रेन भी हो सकता है। परफ्यूम, पेंट थिनर, सेकेंड हैंड स्मोक और अन्य सहित तेज़ गंध कुछ लोगों में माइग्रेन को ट्रिगर करती है। नींद न आना, बहुत अधिक नींद लेना या जेट लैग कुछ लोगों में माइग्रेन को ट्रिगर कर सकता है। यौन गतिविधि सहित तीव्र शारीरिक परिश्रम, माइग्रेन को भड़का सकता है। मौखिक गर्भनिरोधक और वैसोडिलेटर, जैसे नाइट्रोग्लिसरीन, माइग्रेन को बढ़ा सकते हैं। कारणों में कई खाद्य पदार्थों में पाए जाने वाले स्वीटनर एस्पार्टेम और परिरक्षक मोनोसोडियम ग्लूटामेट (एमएसजी) भी शामिल हैं।

जोखिम कारक [edit | edit source]

पारिवारिक इतिहास सहित कई कारक एक व्यक्ति को माइग्रेन होने का अधिक खतरा बनाते हैं। माइग्रेन किसी भी उम्र में शुरू हो सकता है, हालांकि पहली बार किशोरावस्था के दौरान होता है। माइग्रेन 30 के दशक के दौरान चरम पर होता है और बाद के दशकों में धीरे-धीरे कम गंभीर और कम

माइग्रेन

माइग्रेन

Specialty	Lua error in package.lua at line 80: module 'strict' not found.
Symptoms	सिरदर्द, मतली, प्रकाश के प्रति संवेदनशीलता, ध्वनि के प्रति संवेदनशीलता, गंध के प्रति संवेदनशीलता
Usual onset	यौवन के आसपास
Duration	आवर्तक, दीर्घकालिक
Causes	पर्यावरण और आनुवंशिक
Risk factors	पारिवारिक इतिहास, महिला
Prevention	मेटोप्रोलोल, वैल्प्रोएट, टोपिरामेट
Medication	इबुप्रोफेन, पेरासिटामोल (एसिटामिनोफेन), ट्रिप्टान, एर्गोटामाइन
Frequency	12.6%

Figure 6.3 Snapshot of Hindi Wiki article generated on Migraine disorder

6.6 Results

Finally, the XML dump for all 1154 articles was generated and shared with the IndicWiki project[10] coordinators. The criteria for evaluation is entirely manual. The articles are reviewed by the editors manually and quite thoroughly. It is a lengthy process which takes time. The small errors relating to the manual aspect of the methodology, for example, transliteration, were corrected manually. Apart from those, no major concerns or issues were raised as such. The articles were imported onto the TeWiki website without any issues, and in due time, they will be uploaded to the global Hindi Wikipedia. Even if the reviewers may have missed some errors, it would not be a major concern as the Wikipedia platform is meant for the users to not only read the articles but also help improve them by pointing out mistakes and correcting them manually as and when needed. As discussed earlier, the Wiki Sandbox is a very powerful feature, and the XML meta-data of each disease or condition is rendered quite seamlessly as Wiki articles. Figure-6.3 shows the generated Hindi Wiki article for Migraine.

6.7 Contribution

In terms of the amount of information, the generated articles had more information than their global Hindi Wikipedia counterparts and, in some cases, even more than their global English Wikipedia pages.

[10] https://indicwiki.iiit.ac.in/

This is primarily due to the fact that our sources for collecting information on diseases were NHS and Mayo Clinic medical articles, and they often contain more information on a particular disease than any global English Wikipedia article. The reason for this is the fact that they are authored by certified medical professionals and other subject matter experts, unlike the Wikipedia articles.

We also consider the contribution of our study to the Wikipedia platform. As discussed earlier, Hindi Wikipedia has only a fraction of the amount of articles of that of English Wikipedia. The same holds true for the *Human Diseases and Disorders* category. English Wikipedia has approximately 1500 articles on the subject, while Hindi Wikipedia's count comes to around 500 articles. With our contribution, this number will rise to a little over a thousand and five hundred articles. Even if consider 500 articles to be repeated in both versions, our contribution still adds more than 600 new articles while also adding extra information to about 500 persisting articles in Hindi Wikipedia. This clearly highlights that the automated template-driven generation of Wikipedia articles can help close or at least shorten the vast gap that exists between information available in English and native Indian Languages.

6.8 Conclusion

Through this study, we were able to explore the effectiveness of a summarization-based and template-driven approach to generate a large amount of Wikipedia articles automatically. We demonstrated a step-by-step approach from collecting data from online medical resources and infoboxes of Wikipedia articles to cleaning and structuring it appropriately. Then, we followed it by translating and transliterating the data into Hindi and forming a Jinja2 template which when coupled with Python scripting, was able to generate an XML dump of more than a thousand Hindi Wiki articles.

This goes on to show that the stages indeed complement each other in a commendable manner. With this study, we are able to demonstrate a very important application of NLP techniques in the modern day, i.e., producing a large quantity of good quality content in the native language of the readers on a platform which follows strict guidelines when it comes to the neutral tone of the articles. This method may be used to generate Wiki articles in any number of languages, given the availability of required Data Resources, Machine Translation and Transliteration tools, and expert assistance for that language.

6.8.1 Limitations

Before closing the chapter, it is imperative to also discuss the limitations of this study. Overcoming some of these would directly result in a better quality of articles, while some others may increase the amount of information present in each article.

First and foremost, we can try to improve our summarization technique by making use of more efficient and accurate deep-learning-based approaches. This would help us to lengthen or shorten our sections appropriately. We can also try to use abstractive summarization rather than just extractive sum-

marization. Implementing abstractive summarization while keeping Wikipedia's strict neutral tone in articles in mind is a challenge worth taking on. The results of such a study could prove beneficial for anyone interested in the automated generation of Wikipedia articles.

Efforts can also be made to automate the evaluation stage. The current setup requires a long time for evaluation, given the manual nature of it. A spelling and grammar check bot can be developed with built-in Wikipedia's NPOV features. However, building such a bot is a research study on its own and would require proper time and resources for its development. Lastly, we can try to include more linguistic variation in our template to make sure the articles produced have neutral but varied tones, as reading monotonous articles will not prove ideal for native language readers.

Chapter 7

Conclusion

In this thesis, we have looked at some impressive techniques for text classification and text generation. We analyzed the EDOS shared task dataset in great detail and reached the conclusion that it is more than fit for use in the development of automated systems that aim to detect sexism online. For building these automated systems, we considered two different machine-learning approaches. The initial approach included conventional machine learning techniques using simple statistical classifiers and support vector machines. We demonstrated a considerable increase in the performance of this technique by utilizing different data resampling techniques. With these improvements, our models were able to outperform the baseline scores set by the organizers for the shared task. Moving on to more advanced approaches, we also experimented with BERT-based classifiers for the shared task. We illustrated how the ensemble of the three different techniques, DAP-LeR-DAug, was not only able to comfortably outperform the best baseline scores set by the organizers for the EDOS shared task but also reached quite close to the top-ranked systems in some of the subtasks. We also noted the importance of each technique for the boost in performance it provided to each subtask and also realized how non-linguistic purely mathematical techniques like Learning Rate scheduling can help better the performance of text classification models. Lastly, we also discussed the significance of these techniques in the context that no single approach was the top-ranked system for all the three subtasks in the EDOS shared task leaderboard. Although our techniques did not perform the best, the gain in performance they provided for each subtask was substantial, and they can be used to provide similar improvements in performance for other text classification tasks.

In terms of data availability in Indian languages, we highlighted the vast gap that exists between English Wikipedia and any other Indian language Wikipedia. In an effort to shorten this gap, we proposed a template-driven approach to generate a large number of Wikipedia articles automatically. We illustrated how relevant data for the target domain can be scraped from reliable sources online. Data in online platforms mostly exists in unstructured format, and we provided guidelines on how to structure and summarize it appropriately so as to create a meaningful textual dataset. We experimented with different translation and transliteration libraries available to ensure that the data was translated into Hindi in the best quality possible. We even had to resort to manual transliteration of some of the complex disease

names, given the libraries' inability to do the same. Finally, we demonstrated how a template can be drafted for a sample article of the target domain and how we can treat the activity of generating articles as filling in the blanks of the drafted template using the dataset attributes and corresponding values. We concluded this endeavour by generating an XML file containing data of 1154 articles on *Diseases and Disorders*, which is ready to be published on Hindi Wikipedia.

7.1 Future Work

We have already discussed in great detail the limitations of each of our techniques and studies at the end of each chapter. Now, we take a look at how we can meaningfully improve upon our work in the future and some other directions we can extend our work in:

- Firstly, techniques like **hashtag segmentation** and **emoji-2-vec** [34] can be utilized so the models do not miss out on important information for detecting sexism. The current setup for both conventional and BERT-based classifiers removes them during the pre-processing stage.
- The work in this thesis has only utilized the base versions of the BERT-based classifiers. The **XL and XXL versions of the classifiers** should also be experimented with as their extended vocabulary and increased number of parameters should hypothetically provide a boost in performance.
- In terms of the DAug approach, the impact of other techniques like **Back-translation and LAMBADA** [17] should also be tested. Only after having experimented with each of the well-known augmentation techniques can we gauge how impactful the DAug stage really is.
- Apart from the Learning rate alone, the **extreme tuning of other hyperparameters**, like optimizing the dropout rate, loss functions, weight decay and activation functions, should also be studied. Studying their impact on the model performance will shed light on to what extent non-linguistic techniques can improve the performance of NLP models.
- In terms of the summarization techniques used to extract only the required sentences from the dataset for the generation of Wiki articles, we can experiment with **abstractive summarization techniques** as well instead of only focusing on extractive summarization.
- For producing better quality Wiki articles, we can strive to include **more linguistic variation in our template**. This would help the articles have neutral but varied tones and save the readers from having to read monotonous literature.
- Finally, in an effort to blend our two text classification and text generation studies, work can also be directed towards building **a Wikibot trained for detecting harassment or vandalism on Wikipedia pages**. Since Wikipedia is a free and public platform, it suffers from vandalism frequently, and the development of such a Wikibot would ensure a safe digital space on the platform for promoters of knowledge.

List of Related Publications

[P1] Jayant Panwar and Radhika Mamidi, **"PanwarJayant at SemEval-2023 Task 10: Exploring the Effectiveness of Conventional Machine Learning Techniques for Online Sexism Detection"**, in proceedings of *the 17th International Workshop on Semantic Evaluation (SemEval-2023), pages 1531–1536, Toronto, Canada. Association for Computational Linguistics*, 2023.

[P2] Jayant Panwar and Radhika Mamidi, **"DAP-LeR-DAug: Techniques for enhanced Online Sexism Detection"**, in proceedings of *the 6th International Conference on Natural Language and Speech Processing (ICNLSP 2023), pages 51–58, Online. Association for Computational Linguistics*, 2023.

[P3] Jayant Panwar and Radhika Mamidi, **"Text-2-Wiki: Summarization and Template-driven Article Generation"**, in proceedings of *the 20th International Conference on Natural Language Processing (ICON 2023), Goa, India. Association for Computational Linguistics*, 2023.

Bibliography

[1] H. R. Kirk, W. Yin, B. Vidgen, and P. Röttger, "SemEval-2023 Task 10: Explainable Detection of Online Sexism," in *Proceedings of the 17th International Workshop on Semantic Evaluation*. Toronto, Canada: Association for Computational Linguistics, July 2023. [Online]. Available: http://arxiv.org/abs/2303.04222

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[4] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *Computing Research Repository*, vol. arXiv:1907.11692, 2019, version 1. [Online]. Available: http://arxiv.org/abs/1907.11692

[5] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of machine learning research*, vol. 21, no. 140, pp. 1–67, 2020.

[6] S. Gururangan, A. Marasovi'c, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith, "Don't stop pretraining: Adapt language models to domains and tasks," *Computing Research Repository*, vol. arXiv:2004.10964, 2020, version 3. [Online]. Available: http://arxiv.org/abs/2004.10964

[7] K. Zhao, X. Jin, S. Guan, J. Guo, and X. Cheng, "MetaSLRCL: A self-adaptive learning rate and curriculum learning based framework for few-shot text classification," in *Proceedings of the 29th International Conference on Computational Linguistics*. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, Oct. 2022, pp. 2065–2074. [Online]. Available: https://aclanthology.org/2022.coling-1.180

[8] Z. Waseem, T. Davidson, D. Warmsley, and I. Weber, "Understanding abuse: A typology of abusive language detection subtasks," in *Proceedings of the First Workshop on Abusive Language Online*. Vancouver, BC, Canada: Association for Computational Linguistics, Aug. 2017, pp. 78–84. [Online]. Available: https://aclanthology.org/W17-3012

[9] A. Jha and R. Mamidi, "When does a compliment become sexist? analysis and classification of ambivalent sexism using twitter data," in *Proceedings of the Second Workshop on NLP and Computational Social Science*. Vancouver, Canada: Association for Computational Linguistics, Aug. 2017, pp. 7–16. [Online]. Available: https://aclanthology.org/W17-2902

[10] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated hate speech detection and the problem of offensive language," *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 11, no. 1, pp. 512–515, May 2017. [Online]. Available: https://ojs.aaai.org/index.php/ICWSM/article/view/14955

[11] F. Rodríguez-Sánchez, J. Carrillo-de Albornoz, and L. Plaza, "Automatic classification of sexism in social networks: An empirical study on twitter data," *IEEE Access*, vol. 8, pp. 219 563–219 576, 2020.

[12] P. Parikh, H. Abburi, P. Badjatiya, R. Krishnan, N. Chhaya, M. Gupta, and V. Varma, "Multi-label categorization of accounts of sexism using a neural framework," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 1642–1652. [Online]. Available: https://aclanthology.org/D19-1174

[13] M. Samory, I. Sen, J. Kohne, F. Floeck, and C. Wagner, ""call me sexist, but...": Revisiting sexism detection using psychological scales and adversarial samples," *Computing Research Repository*, vol. arXiv:2004.12764, 2021, version 2. [Online]. Available: http://arxiv.org/abs/2004.12764

[14] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, vol. 6, no. 1, pp. 1–54, 2019.

[15] J. Wei, K. Zou, M. Chen, and L. Li, "Eda: Easy data augmentation techniques for boosting performance on text classification tasks," *Computing Research Repository*, vol. arXiv:1901.11196, 2019, version 2. [Online]. Available: http://arxiv.org/abs/1901.11196

[16] S. Kobayashi, "Contextual augmentation: Data augmentation by words with paradigmatic relations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 452–457. [Online]. Available: https://aclanthology.org/N18-2072

[17] A. Anaby-Tavor, B. Carmeli, E. Goldbraich, A. Kantor, G. Kour, S. Shlomov, N. Tepper, and N. Zwerdling, "Do not have enough data? deep learning to the rescue!" *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, pp. 7383–7390, Apr. 2020. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/6233

[18] Y. Pochampally, K. Karlapalem, and N. Yarrabelly, "Semi-supervised automatic generation of wikipedia articles for named entities," *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 10, no. 2, pp. 72–79, Aug. 2021. [Online]. Available: https://ojs.aaai.org/index.php/ICWSM/article/view/14834

[19] J. Minguillón, M. Lerga, E. Aibar, J. Lladós-Masllorens, and A. Meseguer-Artola, "Semi-automatic generation of a corpus of wikipedia articles on science and technology," *Profesional de la información*, vol. 26, no. 5, p. 995–1005, Sep. 2017. [Online]. Available: https://revista.profesionaldelainformacion.com/index.php/EPI/article/view/epi.2017.sep.20

[20] P. J. Liu, M. Saleh, E. Pot, B. Goodrich, R. Sepassi, L. Kaiser, and N. Shazeer, "Generating wikipedia by summarizing long sequences," 2018.

[21] A. Agarwal and R. Mamidi, "Automatically generating hindi wikipedia pages using wikidata as a knowledge graph: A domain-specific template sentences approach," in *Proceedings of the 14th International Conference on Recent Advances in Natural Language Processing*. Varna, Bulgaria: INCOMA Ltd., Shoumen, Bulgaria, September 2023, pp. 11–21. [Online]. Available: https://aclanthology.org/2023.ranlp-1.2

[22] K. Woodsend and M. Lapata, "Wikisimple: Automatic simplification of wikipedia articles," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 25, no. 1, pp. 927–932, Aug. 2011. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/7967

[23] R. F. Flesch, "How to test readability," *New York: Harper and Row*, 1951.

[24] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[25] C. Sievert and K. Shirley, "LDAvis: A method for visualizing and interpreting topics," in *Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces*, J. Chuang, S. Green, M. Hearst, J. Heer, and P. Koehn, Eds. Baltimore, Maryland,

USA: Association for Computational Linguistics, Jun. 2014, pp. 63–70. [Online]. Available: https://aclanthology.org/W14-3110

[26] N. Sideris, G. Bardis, A. Voulodimos, G. Miaoulis, and D. Ghazanfarpour, "Using random forests on real-world city data for urban planning in a visual semantic decision support system." *Sensors (Basel, Switzerland)*, 2019. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6567884/

[27] D. W. Hosmer Jr., S. Lemeshow, and R. X. Sturdivant, *Applied Logistic Regression*, 3rd ed. Hoboken, NJ: John Wiley & Sons, 2013.

[28] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annual Eugenics*, vol. 7, no. Part II, pp. 179–188, 1936.

[29] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[30] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 785–794. [Online]. Available: https://doi.org/10.1145/2939672.2939785

[31] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[32] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[33] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing Management*, vol. 24, no. 5, pp. 513–523, 1988. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0306457388900210

[34] B. Eisner, T. Rocktäschel, I. Augenstein, M. Bošnjak, and S. Riedel, "emoji2vec: Learning emoji representations from their description," in *Proceedings of the Fourth International Workshop on Natural Language Processing for Social Media*. Austin, TX, USA: Association for Computational Linguistics, Nov. 2016, pp. 48–54. [Online]. Available: https://aclanthology.org/W16-6208

[35] W. Yin, K. Kann, M. Yu, and H. Schütze, "Comparative study of CNN and RNN for natural language processing," *CoRR*, vol. abs/1702.01923, 2017. [Online]. Available: http://arxiv.org/abs/1702.01923

[36] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter," *CoRR*, vol. abs/1910.01108, 2019. [Online]. Available: http://arxiv.org/abs/1910.01108

[37] T. Caselli, V. Basile, J. Mitrović, and M. Granitzer, "HateBERT: Retraining BERT for abusive language detection in English," in *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 17–25. [Online]. Available: https://aclanthology.org/2021.woah-1.3

[38] D. Q. Nguyen, T. Vu, and A. Tuan Nguyen, "BERTweet: A pre-trained language model for English tweets," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 9–14. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.2

[39] L. N. Smith and N. Topin, "Super-convergence: Very fast training of residual networks using large learning rates," *CoRR*, vol. abs/1708.07120, 2017. [Online]. Available: http://arxiv.org/abs/1708.07120

[40] D. Ruta and B. Gabrys, "Classifier selection for majority voting," *Information fusion*, vol. 6, no. 1, pp. 63–81, 2005.

[41] Y. Zhang, H. Zhang, J. Cai, and B. Yang, "A weighted voting classifier based on differential evolution," in *Abstract and applied analysis*, vol. 2014. Hindawi Limited, 2014, pp. 1–6.