AugmenToxic: Leveraging Reinforcement Learning to Optimize LLM Instruction Fine-Tuning for Data Augmentation to Enhance Toxicity Detection

AREZO BODAGHI*, Concordia Institute for Information Systems Engineering (CIISE), Concordia University, Montreal, Canada
BENJAMIN C. M. FUNG, School of Information Studies, McGill University, Montreal, Canada
KETRA A. SCHMITT, Centre for Engineering in Society, Concordia University, Montreal, Canada

Addressing the challenge of toxic language in online discussions is crucial for the development of effective toxicity detection models. This pioneering work focuses on addressing imbalanced datasets in toxicity detection by introducing a novel approach to augment toxic language data. We create a balanced dataset by instructing fine-tuning of Large Language Models (LLMs) using Reinforcement Learning with Human Feedback (RLHF). Recognizing the challenges in collecting sufficient toxic samples from social media platforms for building a balanced dataset, our methodology involves sentence-level text data augmentation through paraphrasing existing samples using optimized generative LLMs. Leveraging generative LLM, we utilize the Proximal Policy Optimizer (PPO) as the RL algorithm to fine-tune the model further and align it with human feedback. In other words, we start by fine-tuning a LLM using an instruction dataset, specifically tailored for the task of paraphrasing while maintaining semantic consistency. Next, we apply PPO and a reward function, to further fine-tune (optimize) the instruction-tuned LLM. This RL process guides the model in generating toxic responses. We utilize the Google Perspective API as a toxicity evaluator to assess generated responses and assign rewards/penalties accordingly. This approach guides LLMs through PPO and the reward function, transforming minority class samples into augmented versions. The primary goal of our methodology is to create a balanced and diverse dataset to enhance the accuracy and performance of classifiers in identifying instances from the minority class. Utilizing two publicly available toxic datasets, we compared various techniques with our proposed method for generating toxic samples, demonstrating that our approach outperforms all others in producing a higher number of toxic samples. Starting with an initial 16,225 toxic prompts, our method successfully generated 122,951 toxic samples with a toxicity score exceeding 30%. Subsequently, we developed various classifiers using the generated balanced datasets and applied a cost-sensitive learning approach to the original imbalanced dataset. The findings highlight the superior performance of classifiers trained on data generated using our proposed method. These results highlight the importance of employing RL and a data-agnostic model as a reward mechanism for augmenting toxic data, thereby enhancing the robustness of toxicity detection models.

CCS Concepts: • **Computing methodologies** → **Natural language generation**; **Neural networks**; **Sequential decision making**; **Neural networks**; **Natural language processing**; • **Information systems** → **Social networking sites**; **Data mining**.

Additional Key Words and Phrases: Text Data Augmentation, Imbalanced Toxic Datasets, Large Language Models, Reinforcement Learning

Authors' Contact Information: Arezo Bodaghi, Concordia Institute for Information Systems Engineering (CIISE), Concordia University, Montreal, Quebec, Canada; e-mail: Bodaghi.arezo@gmail.com; Benjamin C. M. Fung, School of Information Studies, McGill University, Montreal, Quebec, Canada; e-mail: ben.fung@mcgill.ca; Ketra A. Schmitt, Centre for Engineering in Society, Concordia University, Montreal, Quebec, Canada; e-mail: ketra.schmitt@concordia.ca.

ACM 1559-114X/2024/10-ART
https://doi.org/10.1145/3700791

1 Introduction

The rapid development of communication technology and the internet has transformed virtual communities, making social media platforms more accessible and user-friendly, yet also presenting significant challenges [3]. Toxic language, a prevalent issue within online discussions, is frequently characterized by disrespectful responses that can deter participants from engaging in meaningful conversations [93, 116]. To ensure the safety of online civil discussions and mitigate the potential harm caused by toxic language, the vast amount of user-generated content (UGC) necessitates the implementation of data-driven techniques, including Machine Learning (ML) algorithms, for the automatic classification of UGC within modern content moderation systems [22, 74, 118]. Developing efficient detection systems for toxic content is heavily dependent on the availability of appropriate training datasets. This is in line with the fundamental principle in data-driven research, which states that the quality of outputs is directly influenced by the quality of inputs [140].

Imbalanced datasets and lack of annotated samples pose significant challenges in various classification tasks, such as toxicity detection [100, 118]. In these datasets, one class is often much more numerous than the others, typically known as the majority class, which can result in the development of biased models that yield unsatisfactory results when dealing with the underrepresented minority class [54, 100]. The problem is particularly acute in toxicity detection, as the frequency of toxic samples is often low compared to nontoxic ones. In other words, it is difficult to collect roughly the same number of samples for both toxic and nontoxic classes. For instance, Madukwe et al. [90] highlighted a pronounced class imbalance problem in hate speech detection, with the hate class constituting less than 12% in multi-class datasets and less than half in binary datasets. Additionally, a systematic review of datasets for automatic hate speech detection revealed that 41% of the datasets are small (0-5k posts), with 37% containing less than 20% offensive content [63]. This confirms the challenges associated with obtaining extensive labeled data and underscores the potential pitfalls of class imbalance in training datasets. The imbalance in data distribution can lead to overfitting and hinder generalizability, especially for Deep Learning (DL) models. This imbalance may result in models that excel in detecting nontoxic language but perform poorly on toxic content.

To address imbalanced classification challenges, a diverse set of techniques is utilized at both the data and algorithmic levels [21]. At the data level, these techniques encompass various re-sampling approaches, including oversampling of minority classes and undersampling of the majority class [100, 135]. Undersampling techniques consist of methods such as random undersampling [109], Inverse random undersampling [136], and directed undersampling (informed undersampling) [35]. On the contrary, oversampling techniques [45] include random oversampling with replacement, directed oversampling (which entails informed choices for replacing samples, rather than random selection), and oversampling methods with informed generation of new samples. Moreover, an alternative strategy involves a hybrid method [24] that combines both undersampling and oversampling techniques. In addition to data-level strategies, at the algorithmic level, methods such as cost-sensitive learning (which entails adjusting costs associated with different classes), asymmetric classification, dimension reduction, expert systems, and ensemble techniques such as bagging, boosting, and stacking all assume crucial roles [42, 50, 55, 115, 129, 132]. Although these techniques have shown promising results in improving the performance of models on imbalanced datasets, they also have limitations. Oversampling and undersampling can lead to overfitting and underfitting, respectively, and may not work well when the dataset is extremely imbalanced [49, 143]. Cost-sensitive learning requires an accurate estimation of the misclassification costs, which may be difficult in practice [156]. In the case of ensemble methods, as sampling techniques are used to balance the data in each iteration, they can potentially eliminate valuable information and be prone to overfitting [133]. To mitigate overfitting, an alternative method is automatic data augmentation (DA), involving the creation of synthetic data based on an existing dataset [104]. This can contribute to improving the generalizability of text classification models, making them more adept at performing well with unseen data [85]. The methods

for augmenting text data depend on the specific task and the type of text data under consideration [104]. Text data augmentations (TDA) can be classified into several categories, including the injection of textual noise or spelling errors, word replacement using a thesaurus, and the generation of paraphrases through syntactic tree transformations, back-translation, and pre-trained transformer networks [27, 128]. However, these techniques are not yet ideal, and their effectiveness is hindered because the structure and meaning of the text are closely connected, making it challenging to manipulate one aspect without affecting the other satisfactorily [85].

Therefore, to address the challenge of balancing datasets and enhancing the detection of toxic language through a data augmentation approach, there is a pressing need to develop effective techniques for generating toxic text. While some solutions have been proposed to generate toxic text [52], and general text generation techniques have been tested for toxic language detection [34, 68, 152], they all exhibit limitations. For instance, certain methods necessitate precise and well-written prompts, potentially performing inadequately with original samples. Others rely on zero-shot learning, only fine-tuning models for toxic/nontoxic samples, neglecting the importance of instruction fine-tuned language models for specific tasks. Another issue arises from assigning the same label to generated samples as the prompt sample, even if the generated content may not be toxic.

As we address these challenges, our objective is to introduce a novel technique for sentence-level TDA, specifically targeting toxic language. This approach aims to overcome the limitations observed in previous works by making a substantial contribution to the development of techniques for creating a balanced and diverse toxic dataset, ultimately enhancing classifier accuracy and performance.

We present a TDA framework guided by Reinforcement Learning from Human Feedback (RLHF) [39], specifically employing the Proximal Policy Optimization (PPO) algorithm [124]. This framework operates on an Instruction Fine-Tuned (IFT) Large Language Model (LLM) [163] and focuses on refining the model's ability to paraphrase text. The process begins by fine-tuning the LLM using an instruction dataset derived from the PAWS dataset (Paraphrase Adversaries from Word Scrambling) [155].PAWS is specifically chosen because it is human-labeled, enabling it to distinguish between paraphrases with equivalent semantic meaning and those with high lexical overlap but different meanings. This helps ensure that the paraphrasing task maintains semantic accuracy. Throughout our experiment, we exclusively concentrate on paraphrase pairs exhibiting identical semantic meaning. After this initial fine-tuning, the model is further optimized using PPO, in conjunction with a reward model that guides the paraphrasing and augmentation process. PPO operates within a hybrid architecture that combines value-based and policy-based methods, enhancing training stability by iteratively updating the model's policy in a proximal manner, preventing drastic changes that could disrupt training.

To encourage paraphrasing while maximizing toxicity, we utilize the Google Perspective API to evaluate the toxicity level of each generated text. This API assigns toxicity scores ranging from 0 to 1, indicating the likelihood of a text being toxic. This incentivizes the model to produce toxic samples as it seeks to maximize rewards. Additionally, to prevent the LLM from generating peculiar and non-human-like responses solely to maximize rewards, we employ Kullback-Leibler (KL) Divergence as a penalty. This divergence is calculated between the active policy, influenced by reinforcement signals for toxicity, and the reference policy derived from the initial instruction-tuned LLM. This measure ensures that when the model is hallucinated, it aligns closer to the reference model, striking a balance between maximizing rewards and maintaining human-like response characteristics. By imposing this penalty, the model is motivated to generate paraphrases that closely align with the reference model, thus achieving a balance between maximizing rewards and preserving human-like response characteristics.

It is important to clarify why we refer to this approach as RLHF. Although we use automated tools such as the Google Perspective API for toxicity scoring, the human-labeled PAWS dataset plays a central role in training PPO, and the Perspective API itself is built upon data that was labeled by humans. As a result, our approach is grounded in human feedback, with the PPO optimization process benefiting from this foundational human-labeled data. Therefore, despite the involvement of automated components, the essence of our framework lies in leveraging human-labeled datasets to guide the reinforcement learning process.

We evaluated the proposed method and compared it with other techniques using the Jigsaw toxic dataset [26] and the ToxiGen dataset [52]. In summary, our contributions can be outlined as:

- A novel method for enhancing toxic text data through Instruction Fine-tuning on the pretrained FLAN-T5 model, precisely crafted for paraphrasing with semantic equivalence using PAWS.
- Applying Proximal Policy Optimization (PPO) to further fine-tune (optimize) the instruction-tuned FLAN-T5; our approach incorporates a reward model within the PPO framework to ensure the generated responses maintain the specified level of toxicity.
- Utilizing the Google Perspective API to score toxicity and assign rewards accordingly, while implementing KL-Divergence as a penalty in the reward function to ensure the generated text maintains human-like responses.
- Expanded the imbalanced Jigsaw dataset, which originally included 143,346 nontoxic samples and 16,225 toxic samples, into a balanced dataset comprising over 278,000 samples.
- Outperforming other data augmentation techniques, such as zero-shot learning, back-translation, and instruction-tuned LLMs, which lack RLHF optimization.

This is the first work to employ an optimized instruction-fine-tuned language model (LLM) to paraphrase existing unstructured data, thereby augmenting toxic textual samples in the minority class. Furthermore, our dataset is one of the largest and most balanced available. Additionally, we applied zero-shot learning, and back-translation techniques to benchmark our developed model against other methods, resulting in the creation of the largest balanced dataset for toxicity detection, generated through back-translation from nine different languages into English. This dataset cannot be made publicly available as it is proprietary and owned by a third-party company. Although we have obtained permission to use the dataset for research purposes, we do not have the rights to share or distribute it publicly.

The paper is organized as follows: After the introduction, Section 2 presents an in-depth analysis of the techniques proposed to address the problem of class imbalance. Section 3 covers the preliminaries, followed by Section 4, which delineates our proposed method. Section 5 details the experimental setup, and the results are presented in Section 6. Section 7 concludes the paper with remarks and discussions.

2 Related Work

In the literature, various solutions have been proposed to the class imbalance problem. These techniques can be categorized into Data Level, Algorithmic Level, Ensemble Learning, and Data Augmentation, each of which is briefly discussed in this section. Furthermore, we examine methodologies employed in toxicity detection.

2.1 Data-level approaches

In managing the class imbalance at the data level, the goal is to adjust the distribution of classes by resampling the data space. This involves increasing instances of the underrepresented class through oversampling and reducing instances of the overrepresented class through undersampling, with the possibility of employing a combination of both techniques [20, 38, 75, 132]. Each of these techniques will be explored individually in the following subsections.

2.1.1 Undersampling. In the undersampling method, the primary focus is on the majority class within the dataset, from which instances are extracted either randomly or through specific techniques to achieve class balance [53, 125]. Undersampling, while incurring the main drawback of information loss through the deletion of examples from the training data, nonetheless, offers the benefit of reducing the time required to train models by diminishing the size of the training dataset [125]. The most straightforward method of undersampling involves randomly choosing a portion of samples from the majority class [100]. The random undersampling (RUS) strategy poses a significant risk of eliminating potentially valuable data from the majority class [82, 133]. In response

to this limitation, specific methodologies, such as informative undersampling techniques, selectively eliminate insignificant patterns from the majority class, thereby aiming to maintain performance levels and overcome this drawback [139]. One suggested informative undersampling method, the Condensed Nearest Neighbour (CNN) discussed in [51], serves as a data reduction approach to create a representative subset of the original training set, proficient in accurately classifying all instances [139]. Similarly, the Edited Nearest Neighbors (ENN) uses a K-nearest neighbors (K-NN) approach to identify atypical examples within their neighborhood and subsequently removes them [148]. Furthermore, one-sided selection [76] serves as an alternative approach utilizing Tomek links [138] to detect and eliminate such atypical instances. These approaches are not very useful for text data; they are more applicable to numerical data. In the case of toxic language detection, this technique proves ineffective because we still need a sufficient number of samples from the nontoxic class to successfully train classifiers for distinguishing toxic from nontoxic content. Therefore, our proposed method is specifically developed for textual data and can effortlessly handle unstructured sentences while increasing toxicity scores. There is no need to remove samples from the nontoxic class.

An alternate approach proposed to tackle the RUS limitation involves replacing the strategy with a clustering technique as discussed by [82]. Employing cluster-based techniques aims to group similar objects, or data samples, into the same clusters, with objects in distinct clusters differing in their feature representations. In 2017, imbalanced-learn, an open-source Python toolbox, aimed to address imbalanced dataset challenges in ML and pattern recognition by incorporating state-of-the-art techniques grouped into four categories: (i) under-sampling, (ii) over-sampling, (iii) combined over- and under-sampling, and (iv) ensemble learning methods [80]. Imbalanced-learn [1] offers tools such as ClusterCentroids and RandomUnderSampler. ClusterCentroids reduces the majority class by substituting a cluster with the centroid from a KMeans algorithm. RandomUnderSampler swiftly balances data by selecting a subset randomly. Mediratta and Oswal [94] employed RandomUnderSampler to tackle the imbalance issue in a toxic content classification model. In their comparison of various machine learning models (Support Vector Machine (SVM), Naïve Bayes, Gated Recurrent Unit (GRU), Long Short-term Memory (LSTM), they observed that SVM and Naïve Bayes achieved high accuracy even without addressing the imbalance, effectively learning from imbalanced datasets. The most promising outcomes occurred with GRU when handling imbalance with a random sampler and employing GloVe (Global Vectors for Word Representation) word embedding. Rupapara et al., [119] introduced an ensemble technique for toxic comment detection, comparing its performance with other ML classifiers on both imbalanced and balanced datasets. They employed various resampling methods, including RUS and oversampling, highlighting that machine learning models achieved superior performance when using oversampling for dataset balance.

2.1.2 Oversampling. Undersampling techniques work well for datasets with a lower class imbalance ratio, while oversampling methods effectively manage high-class imbalance [31]. Yet, oversampling tends to expand the training set size by replicating patterns, leading to extended learning times and potential overfitting [21, 82, 132]. Similar to random undersampling, oversampling can occur randomly but involves replicating instances from the minority class to achieve dataset balance [41]. An alternative oversampling approach, referred to as informative oversampling, focuses on amplifying the smaller class. In contrast to generating new samples, this method selectively chooses samples from the minority class for resampling instead of employing a random approach [132]. Another technique for oversampling is synthetic oversampling, where artificial samples are generated for the minority class [127]. These additional samples supplement vital information to the minority class, preventing misclassification of its instances. The Synthetic Minority Oversampling Technique (SMOTE), introduced by [20], operates as an oversampling approach that aims to generate additional instances for the minority class by interpolating between various neighboring instances within that class [41]. Continuing the exploration of oversampling techniques for the minority class, additional methods include the Modified Synthetic Minority

[1] https://imbalanced-learn.org/stable/

Oversampling Technique (MSMOTE) [60], and the Selective Preprocessing of Imbalanced Data (SPIDER) [131]. MSMOTE modifies SMOTE by categorizing minority class instances into safe, border, and latent noise groups, adjusting the neighbor selection strategy accordingly, and SPIDER combines local minority class oversampling with identifying noisy majority class instances, implementing different preprocessing methods to enhance the minority class and eliminate remaining noisy majority class examples [41]. In the case of toxic language, where the number of samples in the nontoxic class significantly surpasses the number of samples in the toxic class, oversampling can be a beneficial strategy. However, it must be executed with caution. The nature of language, particularly sentences, implies that the combination of various words together can be quite intricate. Some words may not be toxic when considered individually, but they can become toxic when used in combination. In our proposed method, we leverage pretrained large language models that have been trained on extensive datasets and exposed to numerous sample sentences. This approach helps us in generating new samples in the minority class, thereby increasing the diversity of samples.

2.1.3 Hybrid Sampling. Both undersampling and oversampling techniques present challenges by respectively risking the removal of vital majority class examples, potentially causing underfitting, and inducing overfitting through an increased number of specific but potentially misleading minority class samples affecting the model's decision boundaries [20, 99]. A sought-after approach involves integrating the benefits of both techniques to manage imbalanced medical diagnostic data.

2.2 Algorithmic-level

This approach involves developing new algorithms or adapting existing ones to be more responsive to class imbalance issues [11, 89]. In this approach, the focus is on addressing the minority class, preventing the learner from exhibiting bias toward the majority class to mitigate the overall cost associated with misclassification [7, 65]. Usually, these methods include the utilization of cost-sensitive and ensemble approaches [36, 40, 137]. Resampling techniques and algorithmic methods alone may prove insufficient in addressing class imbalance challenges in high-dimensional scenarios [88, 144].

2.2.1 Cost-sensitive Learning. The cost-sensitive learning framework integrates strategies at both the data and algorithmic levels, considering the increased costs associated with misclassifying samples from the positive class compared to the negative ones [10, 11, 41]. This approach has been applied to address imbalanced labels in toxic content detection by incorporating it into machine learning and deep learning models to enhance overall performance. However, it is crucial to note that accurate estimation of misclassification costs is necessary, and it can be challenging to achieve in practical applications [156].

2.3 Ensemble Learning

The limitation of traditional approaches (sampling, algorithm level, and cost-sensitive) lies in the requirement to define misclassification costs, often unavailable in datasets, leading to the introduction of ensemble-based methods that combine ensemble learning algorithms with data-level and cost-sensitive techniques to address class imbalance, although the challenge of defining costs persists [41]. Boosting, bagging, and stacking stand out as the most frequently employed techniques within this category [129]. According to a survey on the application of ensemble learning methods for class imbalance problems, Random Forest (RF) and XGBoost have emerged as the most commonly utilized methods in the literature, with both demonstrating reliable performance [96]. Another review paper focusing on ensemble learning and data augmentation models for class imbalance issues demonstrated that various combinations of ensemble learning and oversampling techniques, including SMOTE-LightGBM and random oversampling-LightGBM (ROS-LightGBM), are effective approaches for addressing this challenge [70].

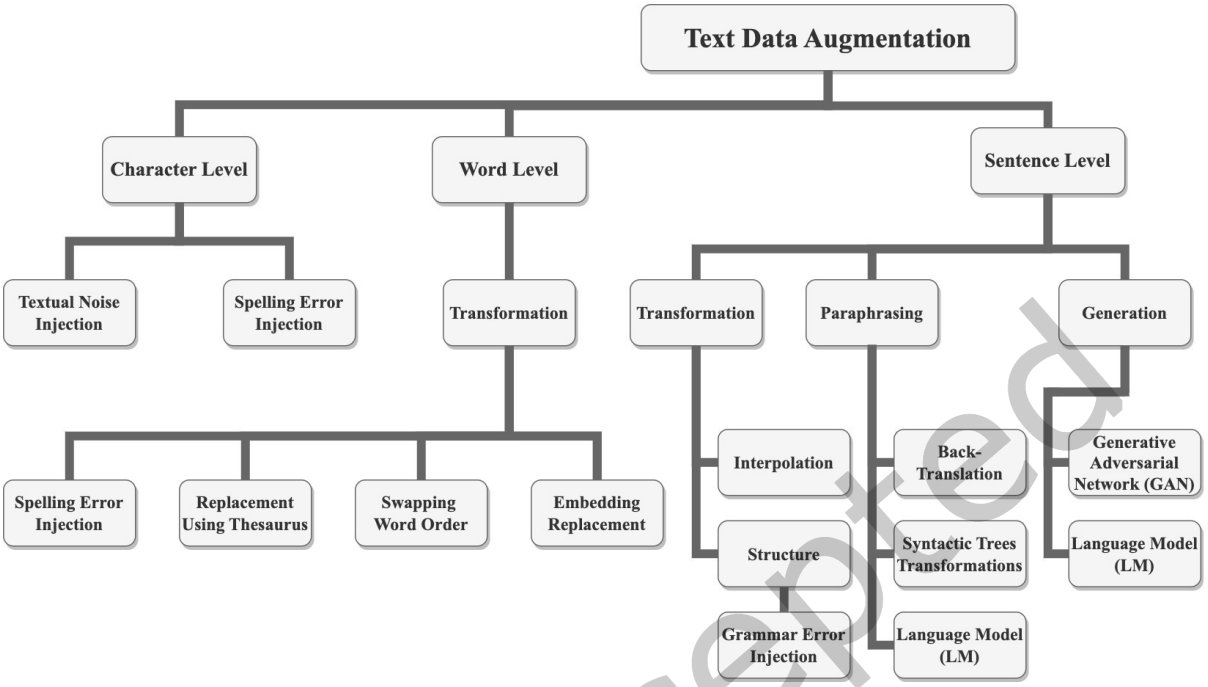

Fig. 1. Taxonomy of Text Data Augmentation Techniques

Addressing data imbalance is a crucial aspect in ML, and various techniques have been proposed to tackle this issue across different domains. However, when it comes to textual data, especially in the context of toxic language detection, existing methods face challenges. Traditional data imbalance techniques, such as oversampling or undersampling, may not be directly applicable to textual data due to its unique characteristics. Moreover, the nature of toxic language data often involves intricate linguistic nuances, making it difficult to apply standard resampling methods effectively. In light of these challenges, there is a growing need for specialized techniques in data augmentation tailored for textual data, which can help alleviate data imbalances and enhance the performance of toxic language detection models. Hence, we further explore text data augmentation techniques, particularly those designed for balancing datasets to improve online toxicity detection.

2.4 Text Data Augmentation

Text data augmentation (TDA) involves generating additional training data from existing data, thereby expanding the dataset available for training classifiers or classification models [104]. Unlike image data augmentation, where simple transformations such as rotation and translation easily preserve the original label, these methods for TDA present a greater challenge in maintaining the original label after perturbations [2]. In recent years, researchers have proposed both unsupervised and supervised TDA methods, generating synthetic data through advanced techniques, where unsupervised methods do not rely on labeled data and supervised methods utilize labeled data for augmentation [104]. The taxonomy of TDA techniques is illustrated in Figure 1, accompanied by a summary of citations included in Table 1. Comprehensive reviews of TDA methodologies can be found in survey papers, such as [2, 14, 27]. These surveys provide an in-depth synthesis of the field. TDA encompasses diverse strategies applied at different levels, including character, word, sentence, and document levels. At the

Table 1. Summary of Text Data Augmentation Techniques with Citations

TDA Level	Method	Reference
Character Level	Textual Noise Injection	[27]
	Spelling Error Injection	[27]
Word Level	Spelling Error Injection	[111]
	Replacement Using Thesaurus	[67]
	Swapping Word Order	[103]
	Embedding Replacement	[162]
Sentence Level	Interpolation	[162]
	Structure	[165]
	Grammar Error Injection	[71]
	Back-Translation	[2]
	Syntactic Trees Transformations	[27]
	Generation- Language Model (LM)	[77]
	Generative Adversarial Network (GAN)	[48]
	Paraphrasing - Language Model (LM)	[149]

character level, techniques involve Textual Noise Injection and Spelling Error Injection. Moving to the word level, augmentation includes the introduction of spelling errors, random deletion, replacement using the thesaurus, swapping word order, and embedding replacement. In the realm of toxic language, a specific instance involved replacing words with their synonyms [117] using word embeddings such as Word2Vec [97], GloVe [108], Fattest [17]. Word replacement can also be performed using features obtained from ConceptNet relations and their descriptions extracted from Wikidata [126].

Expanding to larger units, such as phrases or sentences, TDA incorporates transformation, paraphrasing, and sentence generation. These techniques can be further segmented into specialized methods. For instance, back-translation involves translating a sentence from one language to another and then translating it back into the original language. The back-translation method was employed for data augmentation in the context of hate speech and cyberbullying, involving the initial translation of English text to German and then translating it back to English [15]. Syntactic tree transformations [164] are additional techniques applied at this level. Moreover, the generation of new sentences is achieved through advanced approaches, including Generative Adversarial Networks (GANs) and generative language models. These models contribute to the creation of diverse and contextually relevant text during the augmentation process. As an illustration, generative language models such as Generative Pretrained Transformer 2 (GPT-2) [112] were employed to generate extra-textual samples for the minority class by fine-tuning on existing minority class samples [34, 152]. TOXIGEN, a large-scale dataset of toxic and benign statements about minority groups, generated using a demonstration-based prompting framework and an adversarial classifier-in-the-loop decoding method [52]. They demonstrate that TOXIGEN improves the performance of toxicity classifiers on human-written data and can also help fight machine-generated toxicity. Leveraging the extensive ToxiGen dataset, ConPrompt introduces an innovative pre-training strategy tailored for implicit hate speech detection. Through contrastive learning and prompt-based positive sampling, ConPrompt, embodied by ToxiGen-ConPrompt, emerges as a leading solution. Experimental findings undeniably demonstrate ToxiGen-ConPrompt's superiority over established models like HateBERT and fBERT, showcasing its exceptional generalization and bias mitigation [73].

In certain studies, researchers assigned the minority class label to each newly generated sample [68]. Alternatively, in other studies, the classifiers were fine-tuned for toxicity detection, and only the samples identified as toxic by the classifier were retained after analysis [151]. To ensure that augmentation samples capture target class features, using off-the-shelf language models is limited due to their undirected and random generation, as noted by Liu et al. [86]. In response, they presented Data Boost, a text data augmentation framework guided by reinforcement learning and based on an off-the-shelf language model (GPT-2). The approach involves computing a Salience Score for each word and selecting the top-N highest-scoring words as the salient lexicon for the target class label. Lee et al. [79] explore alignment algorithms, particularly Direct Preference Optimization (DPO), and their role in reducing toxicity in pre-trained language models like GPT2-medium. Their research delves into toxicity representation and elicitation in these models, showcasing how DPO can curb toxic outputs while preserving learned abilities. Furthermore, they present a method to revert models to toxic behavior, underscoring the significance of understanding alignment algorithms in natural language processing.

Comprehensive utilization of these techniques not only aids in overcoming overfitting but also enriches the input feature range, enhancing the overall robustness of classification models [123, 150]. However, current methodologies for data augmentation exhibit significant limitations and remain imperfect. For example, labeling all generated samples with the minority class label [68] is flawed because it does not guarantee the preservation of the intended target class label or ensure that the generated samples adhere to the characteristics of their supposed label. Furthermore, analyzing the toxicity of generated samples using a classifier trained on the same data used for fine-tuning generative language models may introduce bias, given its lack of data agnosticism and potential inefficacy on diverse datasets [95]. In addition, there is a common deficiency in instruction fine-tuning of large language models (LLMs) for specific tasks. In the context of conditional generation, the proposed reward function, which employs Salience Gain [86], tends to prioritize token similarity over toxicity level and may exhibit limited improvement in tasks involving challenging class modeling. Moreover, it faces challenges in extracting explicit lexical features for metaphor, sarcasm, and formality. These techniques may struggle to effectively implement data augmentation, particularly on a large scale, especially for unstructured sample sentences from social media. Additionally, generated samples may deviate from the intended scope of the work, possibly leading to hallucinatory outcomes [61, 83, 110]. To address these limitations, we propose a sentence-level data augmentation technique based on paraphrasing. In this approach, we employ fine-tuning a pretrained Large language model (LLM) through instruction specifically for the task of text paraphrasing. We optimize the model by using reinforcement learning to generate toxic samples. Additionally, data-agnostic models are used to assess the toxicity of generated samples, enabling the assignment of toxic rewards accordingly. This approach is suitable for generating samples at a large scale and has demonstrated superior performance compared to other data augmentation techniques, such as back-translation.

2.5 Toxicity Detection Approaches

In recent years, there has been a notable adoption of machine learning (ML) approaches, encompassing both classical ML and deep learning (DL) algorithms for identifying toxicity in online conversations [8]. Various classical ML techniques, such as logistic regression (LR), decision trees, random forest (RF), and support vector machine (SVM), have been effectively utilized in numerous studies to address online toxicity by identifying toxic content [92, 114, 121]. Commonly employed in feature extraction are methods such as bag-of-words (BOW), Term-frequency-inverse document frequency (TF-IDF), and word embeddings [6]. Moreover, word embeddings have emerged as a widely adopted approach for word representation, proving highly effective in capturing semantic relationships and contextual information. Word2Vec [97], GloVe (Global Vectors) [108], and FastText [66] are among the commonly employed word embeddings extensively utilized in toxicity detection tasks. Extensively utilized for detecting toxic content are various models based on Deep Neural Networks (DNN),

including Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN), Long Short-term Memory (LSTM), Bidirectional LSTM (BiLSTM), and Bidirectional GRU (BiGRU) [1, 62, 93]. DL models autonomously extract complex features from raw data without the requirement for manually crafted features, showcasing their ability to learn representations directly from the input datasets [30]. Word embeddings are utilized as input features for DNN classifiers, as demonstrated in Mohammed et al.'s study [98], which compared various models in two scenarios: one with a standard embedding layer and the other integrating pre-trained embedding corpora like GloVe, Word2Vec, and FastText. Recent advancements in text classification tasks have witnessed a shift towards utilizing pre-trained word embeddings and language models trained on extensive unlabeled corpora. Examples include Bidirectional Encoder Representations from Transformers (BERT) [32] and its variants such as DistilBERT [122], RoBERTa [87], and AlBERT [78]. Ashwin Geet D'Sa et al. explored binary and multi-class classification on a Twitter corpus. They compared two techniques: one involved extracting word embeddings followed by a DNN classifier, while the other fine-tuned a pre-trained BERT model, ultimately demonstrating the superior performance of fine-tuning BERT [33]. In another study, the performance of various pre-trained language models was evaluated across three distinct architectures for toxic language classification: BERT, RoBERTa, XLM, a bi-LSTM + BERT/RoBERTa/XLM, and a CNN + BERT/RoBERTa/XLM [161].

Existing methods predominantly rely on supervised learning approaches, which are heavily dependent on labeled datasets. This becomes particularly challenging due to the inherent imbalance in toxic language datasets [18]. In most online conversations, the vast majority of content is nontoxic, while only a small fraction contains toxic language [160]. Madukwe et al. [90] highlighted this significant class imbalance in hate speech detection, noting that the hate class accounted for less than 12% of multi-class datasets and less than half of the data in binary classification tasks. Similarly, a systematic review of hate speech detection datasets revealed that 41% were small (containing 0-5k posts), and 37% included less than 20% offensive content [63]. This imbalance results in skewed label distributions, where the nontoxic class dominates, causing standard classifiers to become biased toward predicting the majority class (nontoxic) [28]. Consequently, these models often struggle to accurately detect the minority class (toxic). To address this issue, our proposed technique seeks to mitigate the imbalance in toxic datasets and improve the performance of toxicity detection classifiers.

3 Preliminaries

3.1 Text Generation

Imagine a language model, denoted as M, which responds to a consistent prompt P by generating a response y. The process involves the model sampling from its distribution $M(P)$ through decoding, represented as $y \sim M(P)$. In typical text generation scenarios, M calculates the probability distribution for the next token tk based on the prior context $C_{<tk}$, expressed as $p_\omega(tk|C_{<tk})$ [84]. The model learns parameters ω in training by maximizing the likelihood of observed data. This learned probability distribution ($p_\omega(tk|C_{<tk})$) is crucial in guiding the model as it decodes the next token, shaping the coherence of the generated text. By leveraging its learned parameters ω, the model captures relationships between tokens, enabling the creation of coherent and contextually relevant sequences. Various decoding algorithms, including greedy decoding, beam search, temperature sampling, and top-p sampling [56], play a pivotal role in how the model selects and arranges tokens, contributing to the overall coherence and relevance of the generated output.

In the context of text paraphrasing, for two sentences l and l' that serve as paraphrases of each other, we express this relationship as $l \equiv l'$, indicating their equivalent meanings (Semantic Meaning(l) = Semantic Meaning(l') where $l \in L$ and $l' \in L'$). Consequently, when using a prompt to generate a paraphrase for l, we denote the paraphrased version as l', where $M(l|P) = l'$. Please note that a prompt is composed of an instruction I and an input $x \in X$ so that $P = I(x)$. Therefore, $(M(P) = y) \rightarrow (I(x)) = y)$. In the case of paraphrasing, where the instruction is a request for paraphrasing and the input is $l \in L$, then $M(I(l)) = l'$ where $l \equiv l'$.

Response generation employs various methodologies, including zero-shot learning, one-shot learning, few-shot learning, and fine-tuning language models on datasets with instructional annotations [47, 153]. Zero-shot learning enables a model to generate responses for categories or prompts it hasn't been explicitly trained on by leveraging its understanding of underlying concepts or patterns from the training data [130]. Conversely, few-shot learning involves training the model with minimal examples of a specific category or prompt, yet it still learns to generalize and produce responses for similar, previously unseen categories or prompts, showcasing its adaptability and generalization prowess [113]. Research has shown that instruction tuning substantially enhances zero-shot performance on unseen tasks [145]. In the following section, we will delve deeper into the details of Instruction Fine-tuning.

3.2 Instruction Fine-tuning (ITune)

Generative LLMs are initially pre-trained on an extremely large and diverse public dataset, and their weights can be fine-tuned for each task of interest using a much smaller task-specific dataset. Instruction fine-tuning (ITune) is a process in which a pre-trained model, represented by parameters ω, undergoes refinement based on a specialized instruction dataset D_{I}. This dataset includes input-output pairs that serve as explicit instructions or prompts for the model. Therefore, $D_{\text{I}} = \{(I_i, x_i, y_i)\}_{i=1}^{n}$ represents an instruction dataset with each sample consisting of an instruction (I_i), an input sequence ($x_i \in X$), and its corresponding response ($y_i \in Y$), with n total samples in the dataset.

The objective is to adapt a M to better understand and generate responses aligned with the provided instructions. The updated parameters after fine-tuning are denoted as ω', and the fine-tuning process can be expressed as $\omega' = \text{ITune}(\omega, D_{\text{I}})$. This notation captures the transformation of the model's parameters to enhance its performance in generating contextually relevant outputs in response to specific instructions provided in the training dataset.

3.3 Parameter-efficient Fine-tuning (PEFT)

In the conventional fine-tuning process, model weights are usually copied from a pre-trained language model and adapted for a specific downstream task, requiring the generation of new weights for each task. However, full fine-tuning of parameters becomes impractical due to the rapidly growing size of models, making it infeasible to fine-tune the entire model and store separate copies of parameters for numerous downstream tasks [134]. Parameter-efficient techniques have been introduced to address concerns related to storage and computational costs associated with full fine-tuning [57, 91, 154]. As a noteworthy contribution to parameter-efficient fine-tuning (PEFT) techniques, one approach is Low-Rank Adaptation (LoRA) [58]. In the LoRA methodology, the pre-trained model weights remain frozen, and trainable rank decomposition matrices are introduced into each layer of the Transformer architecture. For each M, the hyperparameters r (representing the rank of the update matrices) and α_{LoRA} (a scaling factor crucial for stabilizing training) are fine-tuned [147]. This innovative technique effectively reduces the number of trainable parameters for downstream tasks, lowering GPU memory requirements and demonstrating a commitment to parameter efficiency in the adaptation process.

3.4 Reinforcement Learning (RL)

Reinforcement Learning (RL) is a ML paradigm where an agent learns to make sequential decisions by interacting with an environment. The agent takes actions ($a \in A$), receives a reward ($r \in R$) or penalties in return, and adjusts its strategy to maximize the cumulative reward over time. Trajectories in RL, denoted as τ, refer to sequences of states ($s \in S$), actions ($a \in A$), and rewards ($r \in R$) that an agent experiences during its interactions with the environment. In RL, the agent's goal is to learn a policy π_θ, parameterized by θ, that maps states to actions in a

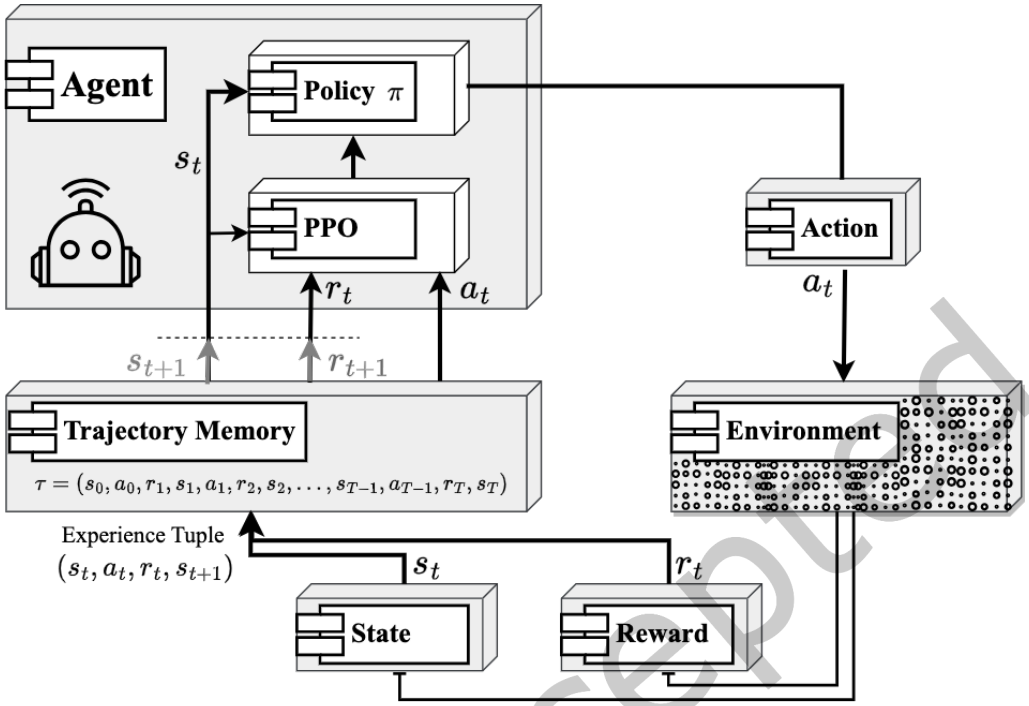

Fig. 2. PPO Model Architecture Overview

way that maximizes the expected cumulative reward over time. The performance of RL algorithms is evaluated based on their ability to learn effective policies in diverse and complex environments.

The pre-trained LM can be customized according to the user's preferences through Reinforcement Learning from Human Feedback (RLHF). It is achieved by defining a reward model and then fine-tuning the LM using RL. Incorporating human feedback aims to capture sentiments in the LLM's responses, with the reward model mapping the model's output to a scalar reward that represents human preferences [16].

3.5 Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO), introduced by Schulman et al. [124], stands out as a robust and widely used deep reinforcement learning algorithm. PPO is known for its stability and efficiency in optimizing policies against specified reward functions. It consistently achieves state-of-the-art performance across a wide range of challenging tasks.

The core idea behind PPO involves iteratively updating the policy π_θ to improve its performance while ensuring stability during training. It aims to balance exploration and exploitation effectively. In PPO, exploration refers to the agent's ability to try out different actions to learn about the environment, while exploitation involves exploiting the knowledge gained to select actions that maximize expected rewards. However, achieving a balance between exploration and exploitation presents a challenge [12]. Figure 2 provides an overview of the architecture of a PPO model. PPO employs a reward model R that considers step-level rewards (r_t) and handles the terminal state (s_{T+1}) with special attention. The step-level reward is a combination of the step-level reward of the last step

(r_T) and a penalization term incorporating the Kullback–Leibler (KL) divergence between the current policy and a reference policy π_{θ_0}. The parameter θ_0 typically represents the initialized policy of PPO.

The step-level reward (r_t) is defined as follows [84]:

$$r_t = \begin{cases} -\beta \log \frac{p_\theta(a_t|s_t)}{p_{\theta_0}(a_t|s_t)} + r(s_{T+1}), & \text{if } t = T \\ -\beta \log \frac{p_\theta(a_t|s_t)}{p_{\theta_0}(a_t|s_t)}, & \text{if } 1 \leq t < T \end{cases}$$

Here, $p_\theta(a_t|s_t)$ denotes the probability of taking action a_t in state s_t under the policy π_θ, and β is a hyperparameter controlling the impact of the penalization term. The formulation balances the desire to increase expected rewards with the need to stay close to the reference policy.

This comprehensive framework allows PPO to effectively learn policies in complex environments, making it a popular choice in the field of deep reinforcement learning.

4 Methodology

In this section, we outline the structure of our proposed approach. Initially, we conduct fine-tuning on the generative LM by utilizing an instruction dataset to paraphrase samples while preserving their semantic meaning (subsection 4.1). Subsequently, to optimize the process, we employ PPO and reward function to guide the model toward toxic paraphrasing, aiming to generate more toxic responses (subsection 4.2).

4.1 Supervised Instruction Fine-tuning

Our primary goal is to increase the number of minority class samples through paraphrasing techniques using generative Language Models (LMs). While models like zero-shot learning may offer simplicity, they prove less effective in paraphrasing toxic samples from online conversations. The subtle and context-dependent nature of toxicity in online discussions poses challenges beyond the capabilities of general language understanding methods, including zero-shot, one-shot, or few-shot learning [101, 106]. Limited exposure to samples can hinder the model's paraphrasing accuracy, making it challenging to maintain meaningful output [25]. To address this, instruction fine-tuning emerges as a promising alternative. This approach involves training an LM on a specific task with explicit instructions, enhancing its ability to paraphrase toxic content while preserving semantic meaning. Instruction fine-tuning offers a focused and tailored training process, enabling the model to adapt more effectively to the nuances of paraphrasing toxic language in unstructured online comments, ultimately improving the quality of generated samples.

4.1.1 Instruction Dataset. An instruction dataset D_{I} typically refers to a specific dataset designed to provide explicit instructions for training a model on a particular task. It contains examples paired with clear instructions on how the model should interpret or respond to those examples. The purpose of an instruction dataset is to guide the model's learning process and help it acquire specific skills or behaviors.

To construct D_{I}, we utilize a structured format containing examples, making it more intuitive for the generative model M to learn in accordance with our requirements. The dataset includes pairs $s \in S$ and $s' \in S'$ representing paraphrases of each other ($s \sim s'$) while maintaining semantic equivalence.

To convert the dataset into an instruction format, each sample s undergoes a wrapping process with an instruction, as illustrated in Figure 3. The provided instruction is formulated as follows: "Paraphrase the following text while maintaining its semantic meaning: {text}." This instruction serves as a directive for the model, guiding it to generate paraphrases that retain the same semantic meaning as the original text. Additionally, all samples, both input prompt (instruction + s) and output (s'), undergo tokenization using the LM's tokenizer. Padding is also applied to the tokenized sequences, ensuring they have the same maximum length.

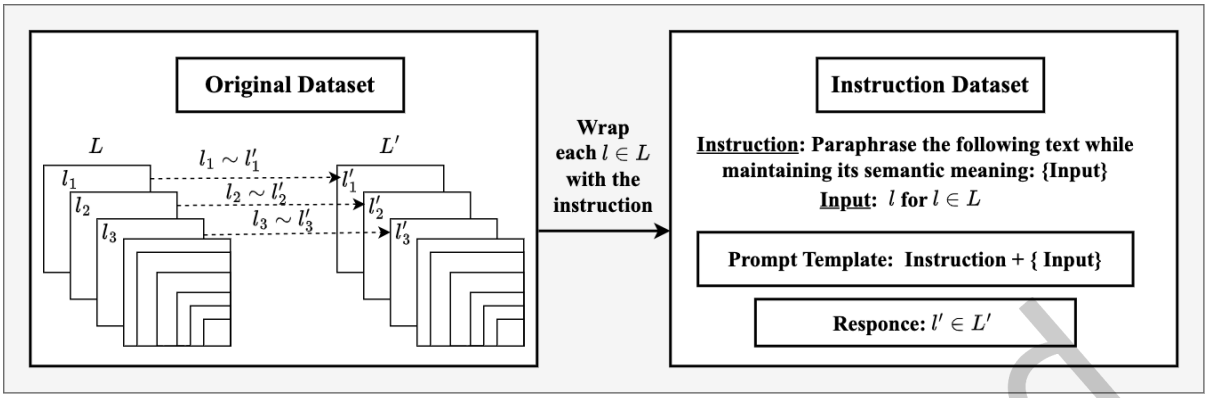

Fig. 3. Illustration of the process to convert the dataset into an instruction format.

4.1.2 Instruction Fine-tuning. With the constructed instruction dataset D_{I} (refer to subsubsection 4.1.1), the pre-trained model M undergoes fine-tuning in a fully supervised manner. This process entails training M to predict each token in the output sequentially, guided by the instruction dataset [158]. The fine-tuning adapts M's parameters based on the task-specific information embedded in the instruction dataset, thereby enhancing its performance in generating paraphrases while preserving semantic meaning.

Recognizing the advantages of Parameter-Efficient Fine-Tuning (PEFT) techniques, such as reducing computational costs, minimizing memory usage during training, streamlining the storage and deployment of task-specific fine-tuned parameters [72], directed us to utilize PEFT as a substitute for the full fine-tuning process in our instruction fine-tuning approach. Furthermore, its demonstrated superiority over full fine-tuning across a diverse array of tasks [147] further supported our decision.

PEFT methods (refer to subsection 3.3) involve freezing the majority of parameters in pre-trained models while still demonstrating comparable capabilities in downstream tasks [157]. Specifically, we considered fine-tuning through Low-Rank Adaptation (LoRA) [58] as an additive fine-tuning scheme, as defined in subsection 3.3. The instruction fine-tuning of the LM for the specific task of paraphrasing is illustrated in Figure 4.

While the fine-tuned model M_{LoRa} excels at paraphrasing existing samples in the minority class and generating new samples, it may encounter challenges when faced with unstructured input from online conversations. Particularly, its performance might diminish when dealing with less well-written or unstructured prompts, even after pre-processing. This challenge is exacerbated when the model is tasked with generating multiple responses for each prompt. Moreover, we anticipate scenarios where the model may generate toxic samples, and there is a need for the LM to rephrase the input while retaining or even increasing its toxicity. Therefore, following the fine-tuning process via LoRA (M_{LoRa}), an additional optimization step using Reinforcement Learning from Human Feedback (RLHF) becomes essential. RLHF aims to guide the LM in rephrasing while preserving toxicity or potentially intensifying it, based on human feedback expressed through a reward function. The subsequent section will delve into the specifics of the optimization process using RLHF.

4.2 Optimization using Reward Function

In this study, we employ Reinforcement Learning from Human Feedback (RLHF) subsection 3.4, specifically utilizing Proximal Policy Optimization (PPO) as described in subsection 3.5. Our objective is to optimize the fine-tuned model M_{LoRa} to generate toxic responses, acknowledging that not all augmented sentences may exhibit toxicity. The development of efficient optimization algorithms is crucial across various scientific disciplines, as

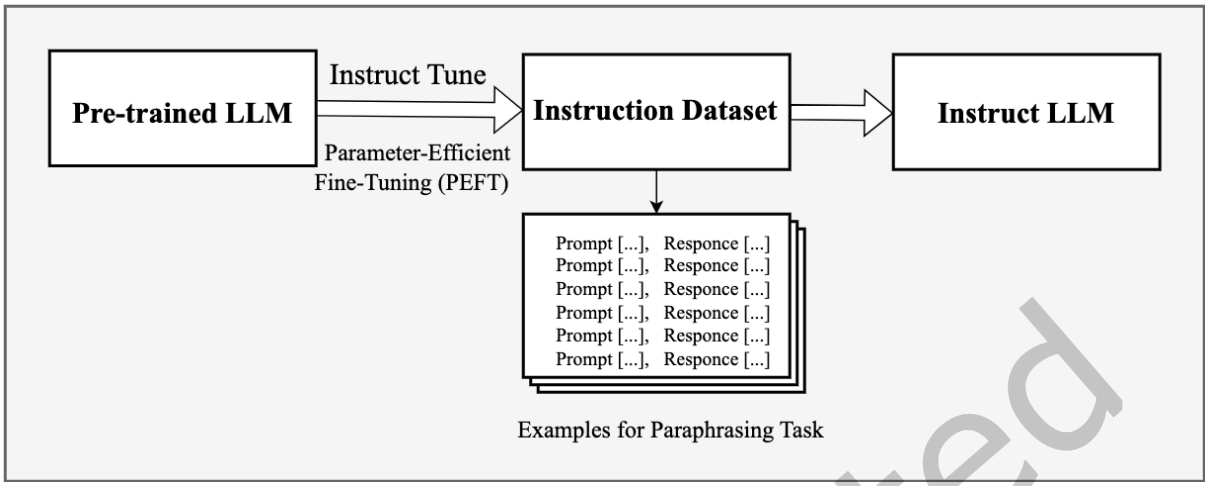

Fig. 4. Instruction fine-tuning of pre-trained Large Language Models (LLMs) for paraphrasing

researchers seek faster and stronger algorithms capable of optimizing a wide range of functions [4, 5, 43, 59]. After the initial fine-tuning with LoRA, where only a portion of parameters was trained, we seek further refinement by updating these trainable parameters to obtain an optimized fine-tuned model. To achieve this, we incorporate a data-agnostic classifier to assess the toxicity of generated responses and assign rewards or penalties accordingly. It is important to note that D_{I} used for instruction fine-tuning is nontoxic, but our aim is to increase toxicity using the reward model, directing M_{LoRa} to generate toxic tokens.

As a reward model, we leverage the Google Perspective API (API) [2], a machine learning-based tool designed to detect abusive comments. This API furnishes toxicity scores ranging from 0 to 1, serving as a probability indicator without delineating severity. Higher scores indicate a greater likelihood of resembling patterns observed in toxic comments. We employ PPO to fine-tune M_{LoRa} with respect to the reward model, resulting in $M_{\text{PPO-API}}$. While M_{LoRa} has initially been trained using the instruction dataset, our aim is to optimize its performance leveraging the reward model.

In the proposed PPO framework, M_{LoRa} serves as an active model (M_{act}) during training and as a reference model (M_{ref}) when not trainable. The generative model functions as an agent, selecting tokens during language generation. The agent initializes its policy with M_{LoRa} and, at each time step t, observes the current state s_t (previously generated tokens), takes action a_t according to the policy ϕ, and transitions to the next state. The agent receives a reward r from the reward model, aiming to maximize the expected reward during PPO training. The framework is visually depicted in Figure 5.

A prompt x_t from D_{I} is simultaneously inputted into both the active model M_{act} and the reference model M_{ref}. Active policy $\pi_{\theta_{\text{act}}}$ and reference policy $\pi_{\theta_{\text{ref}}}$ are initialized. The active model M_{act} generates a response (paraphrased input), such that $\pi_{\theta_{\text{act}}}(a_t \mid s_t) \longrightarrow M_{\text{act}}(x_t) = y_{\text{act}_t}$. Similarly, the reference model M_{ref} generates a response $\pi_{\theta_{\text{ref}}}(a_t \mid s_t) \longrightarrow M_{\text{ref}}(x_t) = y_{\text{ref}_t}$). Subsequently, the response generated by the active model y_{act_t} is decoded and passed to the Google Perspective API, which assigns a toxicity score. A reward of 1 is given to samples with toxicity scores exceeding 0.7, while those below this threshold receive a penalty of -10. The Google Perspective API assigns scores on a scale from 0 to 1, with higher values indicating greater toxicity. To ensure that the generated responses are sufficiently toxic, we set the threshold at 0.7. While a lower threshold, such as

[2]https://perspectiveapi.com/

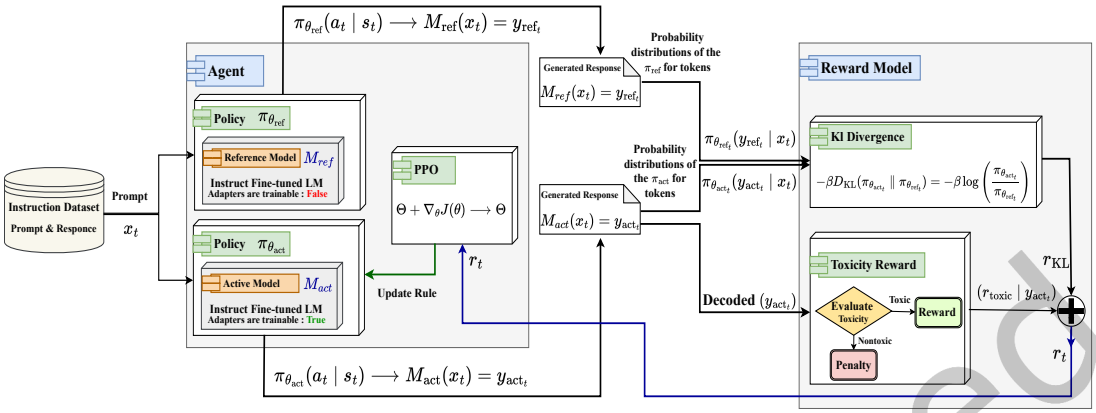

Fig. 5. The proposed solution for paraphrasing toxic samples

0.5, could be considered, setting it at 0.7 ensures that only responses with a higher degree of toxicity are classified as toxic, thereby encouraging the generation of distinctly toxic samples. This reward scheme is formalized as follows:

$$r_{\text{toxic}} = \begin{cases} +1 & \text{if API}(y_{\text{act}_t}) \geq 0.7 \\ -10 & \text{if API}(y_{\text{act}_t}) < 0.7 \end{cases}$$

We assigned greater value to punishment compared to reward because if both were given equal values, such as +1 and -1, the agent might learn to generate only nontoxic responses. In such a scenario, the penalty for producing a nontoxic response would be only 1, which would not sufficiently differentiate between the reward and the penalty. Additionally, to discourage M_{act} from producing unnatural responses solely for increased rewards, a reference model with frozen weights serves as a fixed point of reference. The Kullback-Leibler (KL) Divergence is calculated between the two policies $\pi_{\theta_{\text{ref}_t}}$ and $\pi_{\theta_{\text{act}_t}}$ as follows:

$$r_{\text{kl}} = \left\{ \beta D_{\text{KL}}(\pi_{\theta_{\text{act}_t}} \parallel \pi_{\theta_{\text{ref}_t}}) = -\beta \log \left(\frac{\pi_{\theta_{\text{act}_t}}}{\pi_{\theta_{\text{ref}_t}}} \right) \right.$$

r_{kl} serves as a penalty, ensuring that when M_{act} generates hallucinations, it aligns closer to M_{ref}. This penalty is added to the toxicity reward, constraining the update within a trust region defined by the distance between the two policies. The total reward is computed as:

$$r_t = r_{toxic} + r_{kl} \tag{1}$$

This cumulative reward guides PPO through multiple prompt-response experiments, facilitating ranking averages and employing backpropagation to optimize the response of M_{act}. The proposed framework is illustrated in Figure 5.

5 Experimental Setup

5.1 Instruction Dataset

To fine-tune the LLM for text paraphrasing, we employed the Paraphrase Adversaries from Word Scrambling (PAWS) dataset [155], introduced by Google AI Language in 2019. This openly accessible dataset comprises 108,463 thoughtfully crafted pairs, encompassing both paraphrases and non-paraphrases with significant lexical overlap. Specifically, we utilized the PAWS-Wiki Labeled "Final" version, which includes 65,401 pairs generated through both word swapping and back translation methods. All pairs have undergone human assessments for both paraphrasing fidelity and fluency. The dataset is divided into Train, Validation, and Test sets containing 49,401, 8,000, and 8,000 respectively with no overlap of source sentences across sets. Maintaining high quality, the dataset is structured with three columns—sentence 1 (L_1), sentence 2 (L_2), and a label (0 or 1). In this organization, sentence 1 ($l_1 \in L_1$) represents the primary text, and sentence 2 ($l_2 \in L_2$) serves as its counterpart. A label of 1 signifies that sentence 2 is a paraphrase of sentence 1, while a label of 0 indicates a distinct semantic meaning between the two. Combining all sets (train, validation, and test) into one dataset, comprising "65,401" samples, we exclusively consider the "28,904" samples with label (1), denoting paraphrasing, and discarding samples with label (0). Subsequently, an 80-20 split was performed, allocating "23,123" samples to the training set and "5,781" samples to the test set. Within the training set, a further subdivision was implemented for hyperparameter tuning, with "2,312" samples, constituting 10% of the training data, reserved for validation purposes.

5.2 Toxic Data sets

5.2.1 ***Jigsaw-dataset***. We employed a publicly available dataset provided by Google Jigsaw and Kaggle [26], which comprises 159,571 Wikipedia comments human-rated for toxicity across six categories: toxic (15,294), severe toxic (1,595), insult (7,877), obscene (8,449), threat (478), and identity hate (1,405). The remaining data (143,346), which is not included in any type of toxicity, is considered nontoxic. Notably, this dataset exhibits a significant imbalance, with the majority class consisting of nontoxic samples. Within this dataset, aside from nontoxic samples, others may bear multiple labels. For instance, a sample could be labeled as both insult and obscene without carrying the toxic label, while another might solely be labeled as insult, and yet another could have labels for both insult and toxic. Unfortunately, precise descriptions or definitions for the various types are not published. Consequently, we classify all samples with at least one form of toxicity as toxic and convert the dataset into binary labels (toxic, nontoxic). It is crucial to emphasize that, in this paper, we use the term 'toxic' to encompass any type of toxicity. As a result, the final dataset includes 143,346 nontoxic samples and 16,225 toxic samples. This implies that, for every toxic sample, approximately eight nontoxic examples exist in the dataset. To ensure an adequately balanced dataset, we decided to generate nine samples per toxic sample, resulting in a total of 146,025 samples. Since not all generated samples are necessarily toxic, we opted for generating additional samples to maintain a sufficient number even if nontoxic ones are removed, thereby achieving an equal balance of toxic and nontoxic samples in the final dataset.f

5.2.2 ***ToxiGen-dataset***. ToxiGen is a machine-generated dataset comprising 274,000 statements, encompassing both toxic and benign content associated with 13 distinct minority groups [52]. They gathered human-written sentences showcasing implicit toxicity directed at the 13 minority groups, resulting in 26 sets of prompts. Each set includes two variations (benign and toxic) for every target group. From these sets, we specifically chose 2000 prompts displaying hateful content. To achieve our objective, we aim to rephrase toxic prompts and generate additional toxic samples. Subsequently, to ensure a balanced dataset, we randomly select benign samples from the dataset equal to the number of augmented toxic samples.

5.3 Instruction Fine-tuning

In our study, we employed FLAN-T5 [25] for the data augmentation task by paraphrasing existing samples in the minority class. FLAN-T5 is a Fine-tuned Language Network (FLAN) built on the T5 (Text-To-Text Transfer Transformer) architecture [69] and pre-trained on an extensive text corpus. It demonstrates robust generalization across multiple tasks [9]. We specifically employed **flan-t5-base**, which features 250 million parameters, an encoder-decoder architecture, and span corruption [25].

To fine-tune the flan-t5-base model through instruction, we employed the PAWS dataset subsection 5.1 for the targeted paraphrasing task. Following the methodology explained in subsubsection 4.1.1, we first constructed the instruction dataset. The PAWS dataset was adapted into the instruction dataset, where the prompt is generated by wrapping inputs ($\forall s_1 \in S_1$) with paraphrasing instructions, and the response is provided by S_2. The minimum and maximum lengths of the input data are set to 10 and 300, respectively. As mentioned earlier in subsection 3.3, we utilized LoRA to efficiently fine-tune the flan-t5-base model with limited computational resources. The training consisted of 15 epochs with a learning rate of 1×10^{-5}, where lora-rank (r) was set to 70, α to 70, and dropout to 0.05. The total number of trainable parameters for the original model is 255,319,296. With the use of PEFT, this figure is reduced to 7,741,440, signifying that only 3.03% of the model parameters are now trainable.

To assess the performance of the model for paraphrase generation before and after instruction fine-tuning, there is a lack of consensus on the metrics appropriate for these task-specific models, resulting in variations in measurements across different studies. One commonly utilized metric in the assessment of summarization tasks is **ROUGE** which is Recall-Oriented Understudy for Gisting Evaluation [81]. This metric, which predominantly emphasizes recall, is widely employed and extends its applicability to paraphrase evaluation [23, 107]. ROUGE captures the n-gram overlap between responses generated by LM and reference responses provided by humans. The reference responses come from the PAWS dataset, which has been manually generated by humans. It is diversified into several types, including ROUGE-1, ROUGE-2, ROUGE-N, and ROUGE-L, ROUGE-W, ROUGE-S, each tailored to specific features. As an example, ROUGE-N emphasizes gram count and calculates recall by examining matching unigrams in the context of unigram analysis (ROUGE-1). Conversely, ROUGE-L evaluates the Longest Common Subsequence (LCS), ROUGE-W focuses on Weighted LCS, ROUGE-S delves into skip-bigram co-occurrence statistics, and ROUGE-LSUM shows the Length of LCS normalized by the total words in the reference. Additionally, we incorporated other metrics to assess the quality of generated responses, including **METEOR** (Metric for Evaluation of Translation with Explicit Ordering) [13], and **BERTScore**[159]. METEOR integrates semantic understanding into its translation evaluation process by assessing matches in terms of exactness, stemming, or synonymy [120]. BERTScore leverages pre-trained contextual embeddings from BERT-based models to compare words between the source and generated texts, employing cosine similarity for matching [44]. We utilized METEOR 1.5 [29] and the "bert-base-uncased" model for BERTScore.

Table 2 presents the assessment outcomes using ROUGE, METEOR, and BERTScore metrics, comparing the initial model pre-finetuning using Zero-shot learning ($flant5_{\text{Zero-shot}}$) with the instruct-tuned flan-t5-base ($flant5_{LoRA}$).

The findings indicate that the $flant5_{LoRA}$ exhibits absolute percentage enhancements compared to the $flant5_{\text{Zero-sho}}$), with improvements in all metrics. Please note that PAWS does not include any toxic samples, and, so far, flan-t5-base is only instruct-tuned for the paraphrasing task while preserving semantic meaning. It may not perform well in countering toxic content or may unintentionally remove toxic words to avoid generating harmful samples. Therefore, further optimization is needed using a reward function to perform paraphrasing while preserving semantic meaning and addressing or potentially increasing the toxicity level.

In the following section, we explain our experimental setup for optimizing instruction fine-tuning of flan-t5-base using a reward function.

Table 2. Comparing Model Performance in Paraphrasing Tasks Pre and Post Instruct-Finetuning

Model	Metric	Value
$flant5_{\text{Zero-shot}}$	ROUGE-1	0.324
	ROUGE-2	0.289
	ROUGE-L	0.318
	ROUGE-LSUM	0.321
	METEOR	0.317
	BERTScore-Precision	0.332
	BERTScore-Recall	0.323
	BERTScore-F1	0.326
$flant5_{LoRA}$	ROUGE-1	**0.416**
	ROUGE-2	**0.392**
	ROUGE-L	**0.412**
	ROUGE-LSUM	**0.413**
	METEOR	**0.526**
	BERTScore-Precision	**0.547**
	BERTScore-Recall	**0.531**
	BERTScore-F1	**0.539**

5.4 Optimization

While the instruct-tuned flan-t5-base ($flant5_{LoRA}$) has demonstrated improvements over the flan-t5-base using zero-shot learning ($flant5_{\text{initial}}$), further optimization is possible through the integration of a reward function and PPO. Due to the unstructured nature of toxic samples, characterized by online comments deviating from ordinary grammar and vocabulary, the task of paraphrasing becomes notably challenging. To address this challenge, optimization involves leveraging a reward function capable of evaluating text toxicity. By rewarding or penalizing the model accordingly, the objective is to encourage the generation of toxic samples compared to nontoxic ones.

For optimization, we follow the method explained by subsection 4.2 and use $flant5_{\text{LoRA}}$ as a reference model ($flant5_{\text{RF}}$), where its adaptors are not trainable, and all weights are frozen, and an active model ($flant5_{\text{ACT}}$) with trainable adaptors.

We also employ the Google Perspective API [3] as the toxicity detector in the reward model to evaluate the toxicity of generated samples. The Perspective API utilizes machine learning to identify abusive comments, providing toxicity scores between 0 and 1 as a probability indicator, not a severity measure. Higher scores indicate a greater likelihood of resembling patterns in toxic comments, and developers can set thresholds based on these scores without quantifying the degree of toxicity [64].

Additionally, we utilize another toxicity detector to score the toxicity of generated paraphrased samples and gain a toxicity reward. The **facebook/roberta-hate-speech**[4], which we refer to as *HateRoBERTa* in this paper, is a RoBERTa model fine-tuned on a hate/toxic speech dataset [141], available on Hugging Face[5], specifically designed for detecting hate/toxic speech.

[3] https://perspectiveapi.com/
[4] https://huggingface.co/facebook/roberta-hate-speech-dynabench-r4-target
[5] https://huggingface.co/

Table 3. Examining Paraphrasing Model Performance Enhanced by PPO Using Diverse Toxicity Reward Mechanisms

Model	Metric	Value
$flant5_{\text{PPO-API}}$	ROUGE-1	**0.798**
	ROUGE-2	**0.793**
	ROUGE-L	**0.797**
	ROUGE-LSUM	**0.798**
	METEOR	**0.831**
	BERTScore-Precision	**0.924**
	BERTScore-Recall	**0.916**
	BERTScore-F1	**0.920**
$flant5_{\text{PPO-RoBERTa}}$	ROUGE-1	0.788
	ROUGE-2	0.784
	ROUGE-L	0.789
	ROUGE-LSUM	0.789
	METEOR	0.803
	BERTScore-Precision	0.893
	BERTScore-Recall	0.889
	BERTScore-F1	0.892

The goal is to compare the performance of $flant5_{\text{LoRA}}$ when optimized by PPO, and the toxicity detector in the reward function is Perspective API ($flant5_{\text{PPO-API}}$), and when the optimization is done using *HateRoBERTa* as the toxicity detector ($flant5_{\text{PPO-RoBERTa}}$). Perspective API is more data-agnostic than *HateRoBERTa*, and we expect to see superior performance.

The toxicity reward is then added to the penalty. The reference model ($flant5_{\text{RF}}$) also acts as a fixed point of reference, ensuring that when the active model hallucinates, it aligns closer to the reference model, providing positive responses that are not bizarre. KL-Divergence is then calculated and used as a penalty. This penalty is added to the toxicity reward, and the total reward is passed to the value function (PPO) to update the policy accordingly.

To optimize both optimization techniques utilizing various toxicity reward functions, we utilized the trl package [142]. This involved setting generation parameters to Top-k=0.0, Top-p=1.0, output-min-length=50, output-max-length=512, and implementing a maximum of 20 PPO steps.It is important to note that the Top-k sampler restricts sampling to the k most probable tokens, while the Top-p (nucleus) sampler constrains sampling to the smallest set of tokens [37, 56, 146]. Additionally, we conducted iterative tests using a validation set and experimented with random values for hyperparameters to determine the optimal settings.

To assess the quality of paraphrased responses generated by $flant5_{\text{PPO-API}}$ and $flant5_{\text{PPO-RoBERTa}}$, we utilized ROUGE, METEOR, and BERTScore metrics to analyze the impact of PPO optimization on paraphrasing quality. Table 3 presents the results for these two models across various metrics. The findings indicate that their performance is quite comparable, with optimization via PPO leading to noticeable improvements in paraphrasing quality. Furthermore, a comparison between the results presented in Table 3 and those in Table 2, which includes all four models (including $flant5_{\text{Zero-shot}}$, $flant5_{\text{LoRa}}$, $flant5_{\text{PPO-API}}$, and $flant5_{\text{PPO-RoBERTa}}$), underscores the significant enhancement in generated response quality achieved through PPO optimization compared to Zero-shot learning or instruct-finetuning.

Table 4. Percentage Enhancement in Toxicity Scores for Paraphrasing Post-Optimization with PPO

Model	Improvement in Toxicity Score (%)	
	Average	Standard Deviation
$flant5_{\text{PPO-RoBERTa}}$	12.35	27.44
$flant5_{\text{PPO-API}}$	**28.42**	25.18

Furthermore, to assess the ability of the developed models to generate toxic samples, we employed a test set for sentence paraphrasing tasks. We assessed the toxicity of the test set, calculated the average and standard deviation of toxic scores for paraphrasing samples generated by the reference model ($flant5_{\text{RF}}$), and compared it with the toxicity of samples generated by $flant5_{\text{PPO-API}}$ and $flant5_{\text{PPO-RoBERTa}}$. Table 4 outlines the results, demonstrating the enhancement in the toxicity of generated samples through the optimization task. Specifically, $flant5_{\text{PPO-API}}$ could generate more toxic samples compared to $flant5_{\text{PPO-RoBERTa}}$ on average by 21.68% and 7.57%, respectively.

Please note that the toxicity of generated responces for samples in the test set was evaluated by HateBERT [19]. In other words, for samples generated by $flant5_{\text{PPO-API}}$, and $flant5_{\text{PPO-RoBERTa}}$, *HateBERT* was employed to score toxicity and facilitate comparison between the reference model and the optimized model.

After constructing the models for sentence-level augmentation through paraphrasing, we utilized the toxic datasets detailed in subsection 5.2 to evaluate our proposed model.

5.4.1 ***Jigsaw***. As previously mentioned, the jigsaw toxic dataset comprises 143,346 nontoxic samples and 16,225 toxic samples. The objective is to generate additional toxic samples to achieve a balanced dataset with an almost equal number of toxic and nontoxic samples. To accomplish this, we aimed to augment toxic samples approximately $9 \times 16,225 = 146,025$, considering that not all augmented samples are necessarily toxic, and some may be removed later.

To meet this objective, we chose to generate nine different paraphrases per toxic sample using all instruct-tuned models $flant5_{LoRA}$, $flant5_{\text{PPO-RoBERTa}}$, $flant5_{\text{PPO-API}}$. Subsequently, we transformed the toxic dataset into an instruction dataset by encapsulating all 16,255 toxic samples with instructions to form prompts. These prompts were then fed into the models with the following generation parameters: minimum length, top-k, top-p, and the number of returned sequences set to $(5, 0.0, 1.0, 9)$. The results will be presented in the results section subsection 6.1.

5.4.2 ***ToxiGen***. We followed a similar approach as with the ToxiGen dataset, starting with 2,000 toxic samples. Utilizing all models including $flant5_{LoRA}$, $flant5_{\text{PPO-RoBERTa}}$, and $flant5_{\text{PPO-API}}$, we generated an additional 2,000 samples. In other words, we paraphrased every sample in the ToxiGen dataset, resulting in 2,000 augmented samples. The outcomes are detailed in the following section.

5.5 Baselines

We evaluate our proposed method for augmenting toxic samples through paraphrasing by comparing it against four baselines. The first baseline is zero-shot learning technique ($flant5_{\text{Zero-shot}}$) , the second one is the instruct-tuned FLAN-T5 model, represented as $flant5_{\text{LoRA}}$. These baseline allows us to observe how the incorporation of RLHF can enhance model performance, especially in the context of toxic paraphrasing.

The third baseline is the optimized model, $flant5_{\text{PPO-RoBERTa}}$, utilizing $HateRoBERTa$ as the toxicity detector in the reward model. A comparison between our proposed technique and this baseline enables us to assess the impact of different reward models on performance.

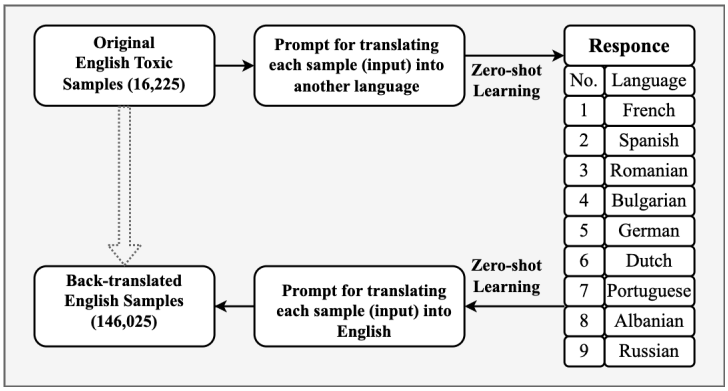

Fig. 6. Illustration of the Back-Translation Technique, where English toxic samples are translated into multiple languages and then back into English for data augmentation.

Additionally, we aim to broaden our comparison to include other data augmentation techniques, specifically, back-translation. We have already detailed the setup for the first three baselines. In the following subsection, we will guide you through the setup for data augmentation using back-translation.

5.5.1 Zero-shot Learning. In our approach to generating toxic text via paraphrasing, we leveraged zero-shot learning, which involves training a model without explicit examples of the task it is meant to perform. We employed "flan-t5-base" or this purpose.

First, we transformed our toxic dataset (subsection 5.2) into an instruction dataset. This involved encoding each toxic sample along with a prompt requesting its paraphrasing while maintaining the same semantic meaning. This method allowed us to generate toxic text without explicitly providing examples of such text. Instead, the model learned to generate toxic text by understanding the underlying semantics of the provided prompts and applying paraphrasing techniques accordingly.

5.5.2 Back Translation. To evaluate our proposed approach against existing text augmentation techniques, we utilized back-translation to generate additional toxic samples. For Jigsaw dataset, our objective was to expand the dataset by creating nine additional samples for each original toxic sample while maintaining a balanced distribution. We employed the flan-t5-base model, known for its multilingual capabilities, for this task. However, we encountered a challenge when translating English toxic samples into nine different versions of a single language, such as French. To overcome this challenge, we opted to translate each English toxic sample into nine different languages and subsequently back into English. Notably, we employed the Zero-shot learning technique for this process. In zero-shot learning, the model is trained to perform a task without explicit examples or training data. In our case, we provided prompts to the flan-t5-base model, instructing it to translate each English toxic sample into nine different languages. Crucially, we did not provide any explicit examples for the model to learn from; instead, it generalized its translation capabilities based on the prompt and input provided. This zero-shot learning technique allowed us to effectively generate diverse translations for each English toxic sample without the need for specific training data in each target language. By back-translating these multilingual translations into English, we were able to augment our dataset with additional diverse toxic samples, enhancing the robustness of our evaluation.

We translate toxic samples into French, Spanish, Romanian, Bulgarian, German, Dutch, Portuguese, Albanian, and Russian, followed by translating all samples back into English. The selection of these languages was determined

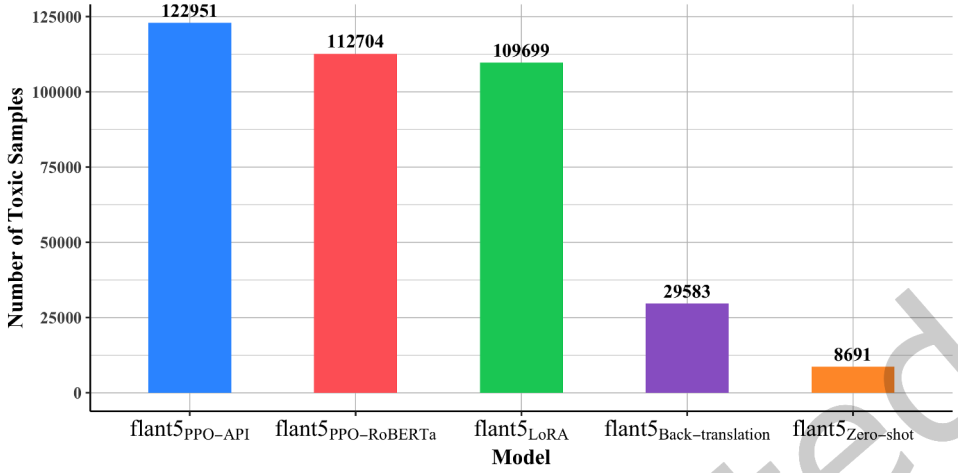

Fig. 7. Total Toxic Samples Generated by Each Model: Toxicity Scores ≥ 0.3 (Jigsaw Dataset)

through trial and error, as we experimented with various languages. Flan-t5-base did not consistently respond appropriately in some instances, generating responses with unintelligible characters, leading us to settle on the aforementioned languages.

We used all toxic samples as input for the prompt requesting translation from English into different languages and vice versa. The method is illustrated in Figure 6.

For the ToxiGen dataset, we followed a similar process, but with a slight modification. Instead of utilizing all nine languages, we restricted our focus to English and French, and their respective translations, due to the limited availability of toxic prompts (2k).

5.6 Computational Resources

We utilized the cloud computing instance 'Paperspace P6000' equipped with NVIDIA P6000 GPUs. The P6000 features a GPU with 24 GB memory, 30 GB RAM, 8 vCPUs, and supports multi-GPU instances of 2X and 4X. The cost per hour for each task on the P6000 is $1.10.

The Google Perspective API is free and processes each query per second. To expedite the process, we employed five different API keys simultaneously and divided the Test set into five parts, with each part assigned to a separate API key.

6 Experimental Results

6.1 Generating Toxic Samples based on Jigsaw dataset

As detailed in section (section 5), the proposed technique ($flant5_{\text{PPO-API}}$) and four baselines, including $flant5_{Zero-shot}$, $flant5_{LoRA}$, $flant5_{\text{PPO-RoBERTa}}$, and $flant5_{\text{Back-translation}}$, were employed to generate toxic samples.

A total of '146,025' samples were generated by each of the models $flant5_{\text{PPO-API}}$, $flant5_{\text{PPO-RoBERTa}}$, $flant5_{LoRA}$, $flant5_{Zero-shot}$ and $flant5_{\text{Back-translation}}$. However, not all samples are deemed acceptable, as some may be repetitive, nonsensical, or nontoxic, lacking coherence or logical consistency. Consequently, we conducted an analysis of the generated samples, selecting only those that are readable for both humans and machines.

The $flant5_{\text{Zero-shot}}$ model struggled to produce multiple responses per request, leading us to limit each request to one response. Ultimately, it generated 16,225 responses, each corresponding to a single prompt. However, the

Table 5. Composition of Balanced Datasets: Model-Generated Toxic Samples and Random Nontoxic Samples-Jigsaw

Model	Generated Toxic Samples > 0.3	Total Toxic Samples	Nontoxic	Toxic + Nontoxic
$flant5_{\text{Zero-shot}}$	8,691	24,916	24,916	49,832
$flant5_{\text{Back-translation}}$	29,583	45,808	45,808	91,616
$flant5_{LoRA}$	109,699	125,924	125,924	251,848
$flant5_{\text{PPO-RoBERTa}}$	112,704	128,929	128,929	257,858
$flant5_{\text{PPO-API}}$	122,951	139,176	139,176	**278,352**

model encountered difficulty in generating distinct responses. Specifically, for 458 toxic samples, the paraphrased responses were identical. To address this issue, we filtered out the redundant responses, resulting in a final count of '15,763' unique responses.

We leveraged Google Perspective API to evaluate the toxicity of our selected samples. Notably, we also incorporated *HateRoBERTa* into our analysis to mitigate potential bias, ensuring a comprehensive assessment of toxicity. Interestingly, the results from both evaluators were highly consistent. As such, for the sake of clarity and simplicity in reporting, we focus solely on the findings obtained through Perspective API.

Following evaluation with Perspective API, all samples received toxicity scores ranging from 0 to 1. Subsequent manual inspection revealed an apparent threshold: samples scoring below 0.3 typically exhibited nontoxic characteristics, while those surpassing 0.3 were deemed potentially toxic. As a result, we made the decision to discard samples with scores below 0.3 and focus solely on those with toxicity scores above this threshold. This ensured that only samples exhibiting a significant level of toxicity were included in our analysis. It is worth noting that during the training phase, we incentivized the model to generate responses with higher toxicity by rewarding samples with scores above 0.7. After experimenting with various thresholds, we determined that a threshold of 0.7 or higher yielded the most satisfactory results during hyperparameter tuning. Consequently, this threshold was chosen as the desired level of toxicity for our study.

Figure 7 below displays the final number of toxic samples generated by each model. The analysis reveals that the proposed model ($flant5_{\text{PPO-API}}$) generates a higher number of toxic samples (122,951) compared to other models. In contrast, $flant5_{\text{Zero-shot}}$ exhibits the poorest performance, generating only 8,691 toxic samples out of 15,763 responses generated. Following this, $flant5_{\text{Back-translation}}$ generates 29,583 toxic samples out of the total 146,025 generated samples. Overall, the instruction-tuned models demonstrate exceptional performance in the task of text data augmentation, with potential for further enhancement through Reinforcement Learning with Human Feedback (RLHF). Back-translation technique did not work well, because toxic language often contains subtle nuances, sarcasm, or contextual references that may not translate accurately or be preserved through this process. As a result, paraphrased versions may fail to capture the original toxicity or convey the intended tone. Moreover, the complexity of toxic language and the need for contextual understanding pose challenges for back-translation models, which may struggle to accurately reproduce the nuanced toxicity present in the original text. Additionally, the effectiveness of back-translation relies on the quality and capabilities of the underlying translation model, which may further limit its suitability for toxic text paraphrasing tasks.

Now, we need to understand how the generated toxic samples (GTS) can impact the performance of classifiers for toxic language detection. Therefore, we first build balanced datasets and then employ some classifiers to test the quality and effectiveness of the balanced dataset. Since we have ensured that the original toxic samples (16,225 toxic samples) are not included in the augmented dataset, we add them to all augmented samples. Subsequently, we randomly select an equal number of nontoxic samples from the toxic dataset in subsection 5.2 to create a balanced dataset. Note that different models generated different numbers of toxic samples, as illustrated in

Table 6. Classification Results-CNN-Jigsaw

Dataset	Accuracy	Precision	Recall	F1-Score
$Unbalanced_{\text{Jigsaw}}$	94.78%	71.39%	81.25%	76%
$Balanced_{\text{Zero-shot}}$	89.20%	88.42%	90.85%	89.50%
$Balanced_{\text{Back-translation}}$	90.20%	88.85%	91.95%	90.37%
$Balanced_{LoRA}$	93.75%	92.03%	95.79%	93.88%
$Balanced_{\text{PPO-RoBERTa}}$	94.01%	92.22%	96.12%	94.13%
$Balanced_{\text{PPO-API}}$	**95.28%**	**94.23%**	**96.46%**	**95.33%**

Table 7. Classification Results-CNN-FastText-Jigsaw

Dataset	Accuracy	Precision	Recall	F1-Score
$Unbalanced_{\text{Jigsaw}}$	89.98%	50.40%	91.52%	65%
$Balanced_{\text{Zero-shot}}$	90.49%	89.12%	91.63%	90.62%
$Balanced_{\text{Back-translation}}$	91.32%	92.68%	89.72%	91.18%
$Balanced_{LoRA}$	93.35%	90.69%	96.61%	93.56%
$Balanced_{\text{PPO-RoBERTa}}$	94.00%	91.60%	96.31%	94.17%
$Balanced_{\text{PPO-API}}$	**94.23%**	**92.47%**	**96.89%**	**94.35%**

Figure 7. Therefore, we will have balanced datasets with varying numbers of toxic and nontoxic samples, as shown in Table 5.

6.2 Classification of Balanced Datasets Generated from Jigsaw Prompts

After preparing all balanced datasets, our focus shifted to evaluating the quality of the generated samples and determining which dataset could enhance the accuracy of classifiers for toxicity detection. In this phase, we selected four different classifiers; two of them are CNN-based, and the other two are transformer-based, to be trained/fine-tuned using the balanced datasets and the original unbalanced toxic dataset. The classifiers include Convolutional Neural Networks (CNN) [105], CNN with FastText embeddings, where FastText embeddings are a vector representation technique created and released by Facebook AI research [46]. Furthermore, our selection of transformer-based models encompassed Bidirectional Encoder Representations from Transformers (BERT) [32], HateBERT [19], a specialized variant of BERT tailored for detecting abusive language in English, and RoBERTa [87], a robust BERT model that employs dynamic masking during pre-training to enhance its performance. Additionally, we incorporated BERTweet [102], a pre-trained language model designed specifically for English Tweets, to further enrich our analysis.

As discussed, we incorporated the primary Jigsaw toxic dataset including 143,346 nontoxic samples, and 16,225 toxic samples to examine the impact of an imbalanced training set on classifier performance. To mitigate dataset imbalance, we implemented a weighted loss function during training, prioritizing the minority class. This strategy enhances the model's ability to learn from underrepresented data, effectively addressing challenges posed by imbalanced distributions. Traditional methods like oversampling and undersampling, as tested on the Jigsaw dataset, often fall short compared to more sophisticated approaches such as ensemble learning [119]. Undersampling risks losing valuable data and features, potentially degrading model performance, while oversampling may introduce redundancy and overfitting, particularly if not carefully applied. Additionally, both

Table 8. Classification Results-BERT-Jigsaw

Dataset	Accuracy	Precision	Recall	F1-Score
$Unbalanced_{\text{Jigsaw}}$	96.04%	75.75%	89.83%	82.19%
$Balanced_{\text{Zero-shot}}$	94.31%	92.74%	96.14%	94.41%
$Balanced_{\text{Back-translation}}$	95.15%	95.61%	94.66%	95.13%
$Balanced_{LoRA}$	96.70%	96.11%	97.34%	96.72%
$Balanced_{\text{PPO-RoBERTa}}$	96.82%	96.28%	97.41%	96.84%
$Balanced_{\text{PPO-API}}$	**97.27%**	**96.91%**	**97.65%**	**97.28%**

Table 9. Classification Results-RoBERTa-Jigsaw

Dataset	Accuracy	Precision	Recall	F1-Score
$Unbalanced_{\text{Jigsaw}}$	96.68%	82.11%	86.19%	84.10%
$Balanced_{\text{Zero-shot}}$	94.27%	93.27%	95.42%	94.33%
$Balanced_{\text{Back-translation}}$	95.68%	95.88%	95.45%	95.67%
$Balanced_{LoRA}$	96.65%	95.53%	97.69%	96.59%
$Balanced_{\text{PPO-RoBERTa}}$	96.65%	95.89%	97.47%	96.67%
$Balanced_{\text{PPO-API}}$	**97.01%**	**96.13%**	**97.97%**	**97.04%**

methods may fail to accurately capture the underlying data distribution, leading to biased models and suboptimal performance. Details about the experimental setup of classifiers are mentioned in Appendix A. The performance of classifiers is then evaluated based on Accuracy, Precision, Recall, and F1-score. All results per classifier are demonstrated in Table 6, Table 7, Table 8, Table 9, Table 10 and Table 11.

The results indicate that the dataset generated by $flant5_{\text{PPO-API}}$ significantly enhanced the performance of all classifiers, outperforming alternative versions trained or fine-tuned with different datasets. Classifiers developed using $balanced_{flant5_{\text{PPO-API}}}$ achieved the highest metrics in accuracy, precision, recall, and F1-score. Despite addressing the imbalance issue with a weighted loss function, prioritizing F1-score for comparison due to the dataset's imbalance revealed BERT developed by $balanced_{flant5_{\text{PPO-API}}}$ as the top-performing classifier with an outstanding F1-score of **97.28%**. Following closely is HateBERT, fine-tuned by $balanced_{flant5_{\text{PPO-API}}}$, which achieved a notable F1-score of **97.12%**. Conversely, classifiers developed using the $Unbalanced_{\text{Jigsaw}}$ dataset did not yield satisfactory results. Furthermore, classifiers developed with the $Balanced_{\text{PPO-RoBERTa}}$ dataset exhibited good performance, closely trailing $balanced_{flant5_{\text{PPO-API}}}$, but fell short of surpassing our proposed technique.

6.3 Generating Toxic Samples and Classification with ToxicGen Dataset

We selected 2,000 toxic prompts and utilized various models, including $flant5_{\text{Zero-shot}}$, $flant5_{\text{LoRA}}$, $flant5_{\text{PPO-RoBERTa}}$, $flant5_{\text{PPO-API}}$, and $flant5_{\text{back-translation}}$, to generate toxic samples. It is important to note that we aimed to generate a maximum of 2,000 samples, as only one response was requested per prompt. Subsequently, all generated samples underwent toxicity evaluation using the Perspective API, and only those with a toxicity score of 0.30 or higher were retained. Figure 8 illustrates the distribution of generated toxic samples across different models.

Then, all augmented samples were added to the initially selected toxic samples used as prompts to create a balanced dataset. To achieve balance, an equal number of nontoxic (benign) samples were randomly selected from the ToxiGen dataset. The final results are shown in Table 12. The results indicate that applying reinforcement

Table 10. Classification Results-HateBERT-Jigsaw

Dataset	Accuracy	Precision	Recall	F1-Score
$Unbalanced_{\text{Jigsaw}}$	96.52%	82.9%	86.11%	84.55%
$Balanced_{\text{Zero-shot}}$	94.32%	93.24%	95.56%	94.39%
$Balanced_{\text{Back-translation}}$	95.35%	95.51%	95.16%	95.34%
$Balanced_{LoRA}$	96.72%	95.88%	97.63%	96.74%
$Balanced_{\text{PPO-RoBERTa}}$	96.75%	96.07%	97.48%	96.77%
$Balanced_{\text{PPO-API}}$	**97.03%**	**96.20%**	**97.98%**	**97.12%**

Table 11. Classification Results-BERTweet-Jigsaw

Dataset	Accuracy	Precision	Recall	F1-Score
$Unbalanced_{\text{Jigsaw}}$	96.41%	82.45%	86.03%	84.21%
$Balanced_{\text{Zero-shot}}$	94.22%	93.19%	95.44%	94.24%
$Balanced_{\text{Back-translation}}$	95.48%	95.49%	95.12%	95.30%
$Balanced_{LoRA}$	96.68%	95.83%	97.60%	96.64%
$Balanced_{\text{PPO-RoBERTa}}$	96.71%	96.67%	97.41%	96.96%
$Balanced_{\text{PPO-API}}$	**97.02%**	**96.26%**	**97.95%**	**97.00%**

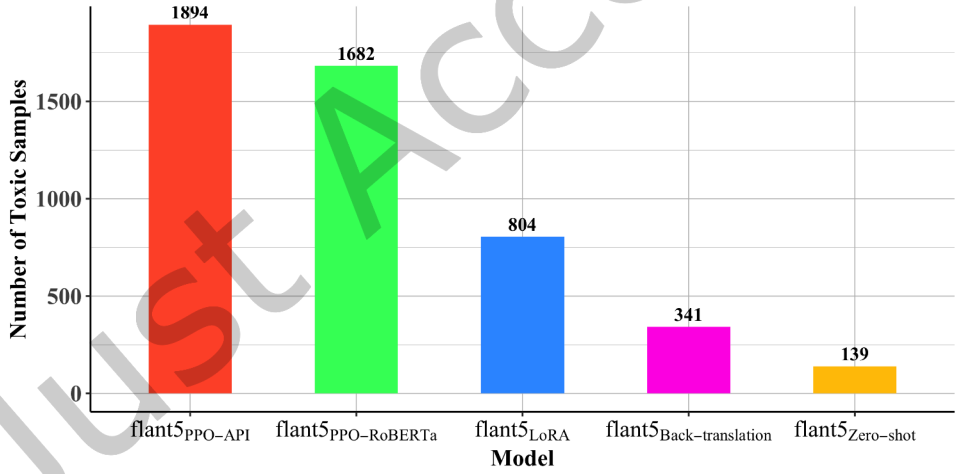

Fig. 8. Total Toxic Samples Generated by Each Model: Toxicity Scores ≥ 0.3 (ToxiGen Dataset)

learning for instruction tuning surpasses other techniques such as zero-shot learning, back-translation, or even simple instruction fine-tuning. Moreover, utilizing the Perspective API as a toxicity evaluator for toxicity reward yields better performance compared to HateRoBERTa. Finally, 1,894 toxic samples were generated by $flant5_{\text{PPO-API}}$, which, combined with the initial 2,000 toxic prompts, resulted in a total of 3,894 toxic samples. Subsequently, a balanced dataset comprising 7,788 samples, both toxic and nontoxic, was created.

Table 12. Composition of Balanced Datasets: Model-Generated Toxic Samples and Random Nontoxic Samples-ToxiGen

Model	Generated Toxic Samples > 0.3	Total Toxic Samples	Nontoxic	Toxic + Nontoxic
$flant5_{\text{Zero-shot}}$	139	2,139	2,139	4,278
$flant5_{\text{Back-translation}}$	341	2,341	2,341	4,682
$flant5_{LoRA}$	804	2,804	2,804	5,608
$flant5_{\text{PPO-RoBERTa}}$	1,682	3,682	3,682	7,364
$flant5_{\text{PPO-API}}$	**1,894**	3,894	3,894	**7,788**

Table 13. CNN-Based Classification Performance-ToxiGen

Model	Dataset	Accuracy	Precision	Recall	F1-Score
CNN	$Balanced_{\text{Zero-shot}}$	90.18%	89.31%	91.67%	90.24%
	$Balanced_{\text{Back-translation}}$	91.34%	89.97%	93.78%	91.42%
	$Balanced_{LoRA}$	92.40%	91.73%	94.15%	92.61%
	$Balanced_{\text{PPO-RoBERTa}}$	93.36%	91.58%	94.28%	93.71%
	$Balanced_{\text{PPO-API}}$	**94.30%**	**93.86%**	**95.75%**	**94.62%**
CNN-FastText	$Balanced_{\text{Zero-shot}}$	91.20%	90.42%	92.85%	91.50%
	$Balanced_{\text{Back-translation}}$	92.31%	91.27%	93.84%	92.69%
	$Balanced_{LoRA}$	93.75%	91.03%	94.79%	93.88%
	$Balanced_{\text{PPO-RoBERTa}}$	94.01%	94.22%	95.12%	94.24%
	$Balanced_{\text{PPO-API}}$	**95.28%**	**94.10%**	**96.73%**	**95.47%**

After creating balanced datasets from ToxiGen, various classifiers, such as CNN, CNN-FastText, BERT, RoBERTa, and HateBERT, and BERTweet were trained on the complete datasets. Each dataset was split into training and testing sets, with an 80% and 20% ratio, respectively. The classification results can be found in Table 13, and Table 14.

The classification results for datasets developed using ToxiGen prompts show that classifiers trained on the $Balanced_{\text{PPO-API}}$ dataset outperform other models. Overall, transformer-based models outperform CNN-based models. Specifically, BERT fine-tuned with the $Balanced_{\text{PPO-API}}$ dataset achieves the best performance with an accuracy of 97.98% and an F1-score of 98.05%. In contrast, the CNN classifier trained on the $Balanced_{\text{Zero-shot}}$ dataset performs the worst, with an accuracy of 90.18% and an F1-score of 90.24%.

To conclude, our proposed approach to text data augmentation for constructing a balanced dataset has not only demonstrated its effectiveness in generating a larger quantity of high-quality toxic samples but also led to enhanced performance across a diverse range of evaluation metrics. This augmentation technique, leveraging state-of-the-art language models and reinforcement learning, provides a straightforward and effective strategy for addressing imbalances in toxic datasets, contributing to superior classifier performance in toxicity detection tasks.

However, an important question arises regarding the significance of our improvements. Despite our model consistently outperforming the baseline methods, the progress might seem modest, especially considering that the baselines were already quite effective. It is crucial to emphasize that even small improvements in accuracy and performance have significant implications, particularly in areas such as detecting toxic content in online

Table 14. Transformer-Based Classification Performance-ToxiGen

Model	Dataset	Accuracy	Precision	Recall	F1-Score
BERT	$Balanced_{\text{Zero-shot}}$	92.46%	91.25%	92.84%	92.39%
	$Balanced_{\text{Back-translation}}$	92.71%	91.44%	93.06%	92.68%
	$Balanced_{LoRA}$	96.88%	95.03%	96.87%	96.75%
	$Balanced_{\text{PPO-RoBERTa}}$	97.52%	97.12%	96.22%	97.43%
	$Balanced_{\text{PPO-API}}$	**97.98%**	**96.23%**	**98.26%**	**98.05%**
RoBERTa	$Balanced_{\text{Zero-shot}}$	92.01%	90.34%	93.59%	92.27%
	$Balanced_{\text{Back-translation}}$	92.68%	91.38%	92.91%	92.46%
	$Balanced_{LoRA}$	95.88%	94.39%	95.97%	95.76%
	$Balanced_{\text{PPO-RoBERTa}}$	96.83%	96.22%	97.12%	96.43%
	$Balanced_{\text{PPO-API}}$	**97.31%**	**96.04%**	**97.59%**	**97.29%**
BERTweet	$Balanced_{\text{Zero-shot}}$	92.27%	93.59%	90.34%	92.16%
	$Balanced_{\text{Back-translation}}$	92.23%	93.83%	91.48%	92.57%
	$Balanced_{LoRA}$	96.05%	94.83%	96.11%	95.98%
	$Balanced_{\text{PPO-RoBERTa}}$	96.94%	96.17%	97.29%	96.88%
	$Balanced_{\text{PPO-API}}$	**97.36%**	**96.54%**	**97.62%**	**97.34%**
HateBERT	$Balanced_{\text{Zero-shot}}$	92.35%	91.22%	92.76%	92.27%
	$Balanced_{\text{Back-translation}}$	92.65%	91.39%	93.01%	92.59%
	$Balanced_{LoRA}$	96.81%	94.98%	96.79%	96.67%
	$Balanced_{\text{PPO-RoBERTa}}$	97.47%	97.02%	96.15%	97.25%
	$Balanced_{\text{PPO-API}}$	**97.85%**	**97.19%**	**98.13%**	**97.92%**

conversations. Additionally, the ability of our method to create more toxic samples is extremely useful, especially when there are not many toxic examples available.

In terms of the computational aspect, we understand the importance of balancing the complexity of our model with the available resources. Our approach includes methods such as PEFT to reduce computational overhead and make better use of memory during training. Furthermore, we use RLHF to improve our model's performance while keeping computational demands low.

In summary, our proposed model makes a significant contribution to the field of toxicity detection by addressing data imbalances and strengthening our model's resilience.

7 Conclusion

In conclusion, our paper presents a novel approach to sentence-level text data augmentation, employing reinforcement learning guided by human feedback to enhance the performance of the fine-tuned FLAN-T5 model. By prioritizing paraphrasing while maintaining semantic coherence and generating toxic responses, our method effectively tackles the challenges posed by imbalanced datasets in toxic language detection. Central to our approach is the utilization of Proximal Policy Optimization as a reinforcement learning technique, coupled with the Google Perspective API as a toxicity evaluator and the integration of Kullback-Leibler Divergence. These components synergistically produce high-quality toxic responses, yielding a balanced and diverse dataset that outperforms existing data augmentation methods. Through a comprehensive exploration of various methodologies for toxic text generation, including Zero-shot learning, Back-translation, and instruct-tuned FLAN-T5,

optimized with Proximal Policy Optimization (PPO) and evaluated using different toxicity evaluators such as HateRoBERTa and the Google Perspective API, we have demonstrated the superiority of our proposed framework. Our findings, validated across two distinct toxic datasets, Jigsaw and ToxiGen, consistently point to the efficacy of instruct-tuned FLAN-T5 with PPO and Perspective API as a superior approach. Our approach has yielded impressive results, generating 122,951 toxic samples with a toxicity score exceeding 30%, highlighting its potential to significantly advance toxicity detection models in online conversations. Notably, over 20,000 of the generated samples exhibit toxicity levels exceeding 90%, further emphasizing the impact of our method on enhancing toxicity detection models. However, it is essential to acknowledge the limitations of our approach, which we aim to address in future research endeavors. Specifically, the simplification of the Jigsaw toxic dataset into two broad categories limits the granularity of our analysis. Future work will focus on generating samples for individual toxic labels to create a more detailed and balanced dataset, enabling a deeper understanding and modeling of various forms of toxic language. Additionally, exploring the performance of other FLAN-T5 variants or larger models could provide additional insights and improvements. Moreover, addressing the need for human processing of generated responses remains a challenge that will be a focus of future iterations of our research. In conclusion, our study represents a significant advancement in the development of robust toxicity detection models for online conversations. By acknowledging current limitations and outlining future research directions, we aim to continue making impactful contributions to the field of natural language processing and online moderation.

Acknowledgments

This research is supported by NSERC Discovery Grants (RGPIN-2024-04087) and Canada Research Chairs Program (CRC-2019-00041).

References

[1] Ahmed Abbasi, Abdul Rehman Javed, Farkhund Iqbal, Natalia Kryvinska, and Zunera Jalil. 2022. Deep learning for religious and continent-based toxic content detection and classification. *Scientific Reports* 12, 1 (Oct. 2022), 17478. https://doi.org/10.1038/s41598-022-22523-3 Publisher: Nature Publishing Group.

[2] Hugo Queiroz Abonizio, Emerson Cabrera Paraiso, and Sylvio Barbon. 2022. Toward Text Data Augmentation for Sentiment Analysis. *IEEE Transactions on Artificial Intelligence* 3, 5 (Oct. 2022), 657–668. https://doi.org/10.1109/TAI.2021.3114390

[3] Laith Abualigah, Yazan Yehia Al-Ajlouni, Mohammad Sh. Daoud, Maryam Altalhi, and Hazem Migdady. 2024. Fake news detection using recurrent neural network based on bidirectional LSTM and GloVe. *Social Network Analysis and Mining* 14, 1 (Feb. 2024), 40. https://doi.org/10.1007/s13278-024-01198-w

[4] Jeffrey O. Agushaka, Absalom E. Ezugwu, and Laith Abualigah. 2022. Dwarf Mongoose Optimization Algorithm. *Computer Methods in Applied Mechanics and Engineering* 391 (March 2022), 114570. https://doi.org/10.1016/j.cma.2022.114570

[5] Jeffrey O. Agushaka, Absalom E. Ezugwu, and Laith Abualigah. 2023. Gazelle optimization algorithm: a novel nature-inspired metaheuristic optimizer. *Neural Computing and Applications* 35, 5 (Feb. 2023), 4099–4131. https://doi.org/10.1007/s00521-022-07854-6

[6] Stephen Akuma, Tyosar Lubem, and Isaac Terngu Adom. 2022. Comparing Bag of Words and TF-IDF with different models for hate speech detection from live tweets. *International Journal of Information Technology* 14, 7 (Dec. 2022), 3629–3635. https://doi.org/10.1007/s41870-022-01096-4

[7] Haseeb Ali, MN Mohd Salleh, Rohmat Saedudin, Kashif Hussain, and Muhammad Faheem Mushtaq. 2019. Imbalance class problems in data mining: A review. *Indonesian Journal of Electrical Engineering and Computer Science* 14, 3 (2019), 1560–1571. https://www.academia.edu/download/99950055/12240.pdf

[8] Darko Androcec. 2020. Machine learning methods for toxic comment classification: a systematic review. *Acta Universitatis Sapientiae, Informatica* 12 (Dec. 2020), 205–216. https://doi.org/10.2478/ausi-2020-0012

[9] Kaveri Anuranjana. 2023. DiscoFlan: Instruction Fine-tuning and Refined Text Generation for Discourse Relation Label Classification. In *Proceedings of the 3rd Shared Task on Discourse Relation Parsing and Treebanking (DISRPT 2023)*, Chloé Braud, Yang Janet Liu, Eleni Metheniti, Philippe Muller, Laura Rivière, Attapol Rutherford, and Amir Zeldes (Eds.). The Association for Computational Linguistics, Toronto, Canada, 22–28. https://doi.org/10.18653/v1/2023.disrpt-1.2

[10] Małgorzata Bach and Aleksandra Werner. 2018. Cost-Sensitive Feature Selection for Class Imbalance Problem. In *Information Systems Architecture and Technology: Proceedings of 38th International Conference on Information Systems Architecture and Technology – ISAT 2017 (Advances in Intelligent Systems and Computing)*, Leszek Borzemski, Jerzy Świątek, and Zofia Wilimowska (Eds.). Springer International

Publishing, Cham, 182–194. https://doi.org/10.1007/978-3-319-67220-5_17

[11] Małgorzata Bach, Aleksandra Werner, and Mateusz Palt. 2019. The Proposal of Undersampling Method for Learning from Imbalanced Datasets. *Procedia Computer Science* 159 (Jan. 2019), 125–134. https://doi.org/10.1016/j.procs.2019.09.167

[12] Jianfu Bai, Yifei Li, Mingpo Zheng, Samir Khatir, Brahim Benaissa, Laith Abualigah, and Magd Abdel Wahab. 2023. A Sinh Cosh optimizer. *Knowledge-Based Systems* 282 (Dec. 2023), 111081. https://doi.org/10.1016/j.knosys.2023.111081

[13] Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, Jade Goldstein, Alon Lavie, Chin-Yew Lin, and Clare Voss (Eds.). Association for Computational Linguistics, Ann Arbor, Michigan, 65–72. https://aclanthology.org/W05-0909

[14] Markus Bayer, Marc-André Kaufhold, and Christian Reuter. 2022. A Survey on Data Augmentation for Text Classification. *Comput. Surveys* 55, 7 (Dec. 2022), 146:1–146:39. https://doi.org/10.1145/3544558

[15] Djamila Romaissa Beddiar, Md Saroar Jahan, and Mourad Oussalah. 2021. Data expansion using back translation and paraphrasing for hate speech detection. *Online Social Networks and Media* 24 (July 2021), 100153. https://doi.org/10.1016/j.osnem.2021.100153

[16] Desirée Bill and Theodor Eriksson. 2023. *Fine-tuning a LLM using Reinforcement Learning from Human Feedback for a Therapy Chatbot Application*. ROYAL INSTITUTE OF TECHNOLOGY. https://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-331920

[17] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics* 5 (2017), 135–146. https://doi.org/10.1162/tacl_a_00051 Place: Cambridge, MA Publisher: MIT Press.

[18] Rui Cao and Roy Ka-Wei Lee. 2020. HateGAN: Adversarial Generative-Based Data Augmentation for Hate Speech Detection. In *Proceedings of the 28th International Conference on Computational Linguistics*, Donia Scott, Nuria Bel, and Chengqing Zong (Eds.). International Committee on Computational Linguistics, Barcelona, Spain (Online), 6327–6338. https://doi.org/10.18653/v1/2020.coling-main.557

[19] Tommaso Caselli, Valerio Basile, Jelena Mitrović, and Michael Granitzer. 2021. HateBERT: Retraining BERT for Abusive Language Detection in English. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, Aida Mostafazadeh Davani, Douwe Kiela, Mathias Lambert, Bertie Vidgen, Vinodkumar Prabhakaran, and Zeerak Waseem (Eds.). Association for Computational Linguistics, Online, 17–25. https://doi.org/10.18653/v1/2021.woah-1.3

[20] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research* 16 (June 2002), 321–357. https://doi.org/10.1613/jair.953 arXiv:1106.1813 [cs].

[21] Nitesh V. Chawla, Nathalie Japkowicz, and Aleksander Kotcz. 2004. Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter* 6, 1 (June 2004), 1–6. https://doi.org/10.1145/1007730.1007733

[22] Charalampos Chelmis and Daphney-Stavroula Zois. 2021. Dynamic, Incremental, and Continuous Detection of Cyberbullying in Online Social Media. *ACM Transactions on the Web* 15, 3 (May 2021), 14:1–14:33. https://doi.org/10.1145/3448014

[23] Xiaoqiang Chi and Yang Xiang. 2021. Augmenting Paraphrase Generation with Syntax Information Using Graph Convolutional Networks. *Entropy* 23, 5 (May 2021), 566. https://doi.org/10.3390/e23050566

[24] Shabrina Choirunnisa and Joko Lianto. 2018. Hybrid Method of Undersampling and Oversampling for Handling Imbalanced Data. In *2018 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*. IEEE, 276–280. https://doi.org/10.1109/ISRITI.2018.8864335

[25] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling Instruction-Finetuned Language Models. http://arxiv.org/abs/2210.11416 arXiv:2210.11416 [cs].

[26] cjadams, Jeffrey Sorensen, Julia Elliott, Lucas Dixon, Mark McDonald, nithum, and Will Cukierski. 2017. Toxic Comment Classification Challenge. https://kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge

[27] Claude Coulombe. 2018. Text Data Augmentation Made Simple By Leveraging NLP Cloud APIs. https://doi.org/10.48550/arXiv.1812.04718 arXiv:1812.04718 [cs].

[28] Mithun Das, Somnath Banerjee, and Punyajoy Saha. 2021. Abusive and Threatening Language Detection in Urdu using Boosting based and BERT based models: A Comparative Approach. http://arxiv.org/abs/2111.14830 arXiv:2111.14830 [cs].

[29] Michael Denkowski and Alon Lavie. 2014. Meteor Universal: Language Specific Translation Evaluation for Any Target Language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, and Lucia Specia (Eds.). Association for Computational Linguistics, Baltimore, Maryland, USA, 376–380. https://doi.org/10.3115/v1/W14-3348

[30] Danilo Dessì, Diego Reforgiato Recupero, and Harald Sack. 2021. An Assessment of Deep Learning Models and Word Embeddings for Toxicity Detection within Online Textual Comments. *Electronics* 10, 7 (Jan. 2021), 779. https://doi.org/10.3390/electronics10070779 Number: 7 Publisher: Multidisciplinary Digital Publishing Institute.

[31] Debashree Devi, Saroj K. Biswas, and Biswajit Purkayastha. 2020. A Review on Solution to Class Imbalance Problem: Undersampling Approaches. In *2020 International Conference on Computational Performance Evaluation (ComPE)*. IEEE, Shillong, India, 626–631. https://doi.org/10.1109/ComPE49325.2020.9200087

[32] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]* (May 2019). http://arxiv.org/abs/1810.04805 arXiv: 1810.04805.

[33] Ashwin Geet D'Sa, Irina Illina, and Dominique Fohr. 2020. BERT and fastText Embeddings for Automatic Detection of Toxic Speech. In *2020 International Multi-Conference on: "Organization of Knowledge and Advanced Technologies" (OCTA)*. IEEE, 1–5. https://doi.org/10.1109/OCTA49274.2020.9151853

[34] Ashwin Geet D'Sa, Irina Illina, Dominique Fohr, Dietrich Klakow, and Dana Ruiter. 2021. Exploring Conditional Language Model Based Data Augmentation Approaches for Hate Speech Classification. In *Text, Speech, and Dialogue*, Kamil Ekštein, František Pártl, and Miloslav Konopík (Eds.). Vol. 12848. Springer International Publishing, Cham, 135–146. https://doi.org/10.1007/978-3-030-83527-9_12 Series Title: Lecture Notes in Computer Science.

[35] Hussein El Saadi, Ahmed Farouk Al-Sadek, and Mohamed Waleed Fakhr. 2012. Informed under-sampling for enhancing patient specific epileptic seizure detection. *International Journal of Computer Applications* 57, 16 (2012), 41–46. https://www.academia.edu/download/66547918/pxc3883733.pdf Publisher: Foundation of Computer Science.

[36] Charles Elkan. 2001. The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence*, Vol. 17. Lawrence Erlbaum Associates Ltd, 973–978. http://cseweb.ucsd.edu/~elkan/rescale.pdf Issue: 1.

[37] Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical Neural Story Generation. https://doi.org/10.48550/arXiv.1805.04833 arXiv:1805.04833 [cs].

[38] Alberto Fernández, Salvador García, Mikel Galar, Ronaldo C. Prati, Bartosz Krawczyk, and Francisco Herrera. 2018. *Learning from Imbalanced Data Sets*. Springer International Publishing, Cham. https://doi.org/10.1007/978-3-319-98074-4

[39] Jerzy Filar and Koos Vrieze. 2012. *Competitive Markov Decision Processes*. Springer Science & Business Media. Google-Books-ID: uXDjBwAAQBAJ.

[40] Yoav Freund and Robert E. Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences* 55, 1 (1997), 119–139. https://www.sciencedirect.com/science/article/pii/S002200009791504X Publisher: Elsevier.

[41] Mikel Galar, Alberto Fernandez, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. 2012. A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42, 4 (July 2012), 463–484. https://doi.org/10.1109/TSMCC.2011.2161285 Conference Name: IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews).

[42] Vaishali Ganganwar. 2012. An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering* 2 (Jan. 2012), 42–47.

[43] Mojtaba Ghasemi, Mohsen Zare, Amir Zahedi, Mohammad-Amin Akbari, Seyedali Mirjalili, and Laith Abualigah. 2024. Geyser Inspired Algorithm: A New Geological-inspired Meta-heuristic for Real-parameter and Constrained Engineering Optimization. *Journal of Bionic Engineering* 21, 1 (Jan. 2024), 374–408. https://doi.org/10.1007/s42235-023-00437-8

[44] A. V. Glazkova and D. A. Morozov. 2023. Applying Transformer-Based Text Summarization for Keyphrase Generation. *Lobachevskii Journal of Mathematics* 44, 1 (Jan. 2023), 123–136. https://doi.org/10.1134/S1995080223010134

[45] Anjana Gosain and Saanchi Sardana. 2017. Handling class imbalance problem using oversampling techniques: A review. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, 79–85. https://doi.org/10.1109/ICACCI.2017.8125820

[46] Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning Word Vectors for 157 Languages. https://doi.org/10.48550/arXiv.1802.06893 arXiv:1802.06893 [cs].

[47] Prakhar Gupta, Cathy Jiao, Yi-Ting Yeh, Shikib Mehri, Maxine Eskenazi, and Jeffrey P. Bigham. 2022. InstructDial: Improving Zero and Few-shot Generalization in Dialogue through Instruction Tuning. http://arxiv.org/abs/2205.12673 arXiv:2205.12673 [cs].

[48] Rahul Gupta. 2019. Data augmentation for low resource sentiment analysis using generative adversarial networks. https://doi.org/10.48550/arXiv.1902.06818 arXiv:1902.06818 [cs, stat].

[49] Haibo He and E.A. Garcia. 2009. Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering* 21, 9 (Sept. 2009), 1263–1284. https://doi.org/10.1109/TKDE.2008.239

[50] Guo Haixiang, Li Yijing, Jennifer Shang, Gu Mingyun, Huang Yuanyue, and Gong Bing. 2017. Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications* 73 (May 2017), 220–239. https://doi.org/10.1016/j.eswa.2016.12.035

[51] P. Hart. 1968. The condensed nearest neighbor rule (Corresp.). *IEEE Transactions on Information Theory* 14, 3 (May 1968), 515–516. https://doi.org/10.1109/TIT.1968.1054155 Conference Name: IEEE Transactions on Information Theory.

[52] Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. ToxiGen: A Large-Scale Machine-Generated Dataset for Adversarial and Implicit Hate Speech Detection. https://doi.org/10.48550/arXiv.2203.09509 arXiv:2203.09509 [cs].

[53] Khan Md Hasib, Md Sadiq Iqbal, Faisal Muhammad Shah, Jubayer Al Mahmud, Mahmudul Hasan Popel, Md Imran Hossain Showrov, Shakil Ahmed, and Obaidur Rahman. 2020. A Survey of Methods for Managing the Classification and Solution of Data Imbalance Problem. *Journal of Computer Science* 16, 11 (Nov. 2020), 1546–1557. https://doi.org/10.3844/jcssp.2020.1546.1557 arXiv:2012.11870 [cs].

[54] Haibo He and Edwardo A. Garcia. 2009. Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering* 21, 9 (Sept. 2009), 1263–1284. https://doi.org/10.1109/TKDE.2008.239 Conference Name: IEEE Transactions on Knowledge and Data Engineering.

[55] Shohei Hido, Hisashi Kashima, and Yutaka Takahashi. 2009. Roughly balanced bagging for imbalanced data. *Statistical Analysis and Data Mining: The ASA Data Science Journal* 2, 5-6 (2009), 412–426. https://doi.org/10.1002/sam.10061 _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/sam.10061.

[56] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The Curious Case of Neural Text Degeneration. https://doi.org/10.48550/arXiv.1904.09751 arXiv:1904.09751 [cs].

[57] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-Efficient Transfer Learning for NLP. https://doi.org/10.48550/arXiv.1902.00751 arXiv:1902.00751 [cs, stat].

[58] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-Rank Adaptation of Large Language Models. https://doi.org/10.48550/arXiv.2106.09685 arXiv:2106.09685 [cs].

[59] Gang Hu, Yuxuan Guo, Guo Wei, and Laith Abualigah. 2023. Genghis Khan shark optimizer: A novel nature-inspired algorithm for engineering optimization. *Advanced Engineering Informatics* 58 (Oct. 2023), 102210. https://doi.org/10.1016/j.aei.2023.102210

[60] Shengguo Hu, Yanfeng Liang, Lintao Ma, and Ying He. 2009. MSMOTE: Improving Classification Performance When Training Data is Imbalanced. In *2009 Second International Workshop on Computer Science and Engineering*, Vol. 2. IEEE, 13–17. https://doi.org/10.1109/WCSE.2009.756

[61] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions. http://arxiv.org/abs/2311.05232 arXiv:2311.05232 [cs].

[62] Mai Ibrahim, Marwan Torki, and Nagwa El-Makky. 2018. Imbalanced Toxic Comments Classification Using Data Augmentation and Deep Learning. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 875–878. https://doi.org/10.1109/ICMLA.2018.00141

[63] Md Saroar Jahan and Mourad Oussalah. 2021. A systematic review of Hate Speech automatic detection using Natural Language Processing. *arXiv:2106.00742 [cs]* (May 2021). http://arxiv.org/abs/2106.00742 arXiv: 2106.00742.

[64] Jigsaw. 2021. What do Perspective's scores mean? https://medium.com/jigsaw/what-do-perspectives-scores-mean-113b37788a5d

[65] M.V. Joshi, V. Kumar, and R.C. Agarwal. 2001. Evaluating boosting algorithms to classify rare classes: comparison and improvements. In *Proceedings 2001 IEEE International Conference on Data Mining*. 257–264. https://doi.org/10.1109/ICDM.2001.989527

[66] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of Tricks for Efficient Text Classification. https://doi.org/10.48550/arXiv.1607.01759 arXiv:1607.01759 [cs].

[67] Michał Jungiewicz and Aleksander Smywiński-Pohl. 2019. Towards textual data augmentation for neural networks: synonyms and maximum loss. *Computer Science* Vol. 20 (1) (2019), 57–83. https://doi.org/10.7494/csci.2019.20.1.3023

[68] Mika Juuti, Tommi Gröndahl, Adrian Flanagan, and N. Asokan. 2020. A little goes a long way: Improving toxic language classification despite data scarcity. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 2991–3009. https://doi.org/10.18653/v1/2020.findings-emnlp.269

[69] Mihir Kale and Abhinav Rastogi. 2020. Text-to-Text Pre-Training for Data-to-Text Tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, Brian Davis, Yvette Graham, John Kelleher, and Yaji Sripada (Eds.). Association for Computational Linguistics, Dublin, Ireland, 97–102. https://aclanthology.org/2020.inlg-1.14

[70] Azal Ahmad Khan, Omkar Chaudhari, and Rohitash Chandra. 2024. A review of ensemble learning and data augmentation models for class imbalanced problems: Combination, implementation and evaluation. *Expert Systems with Applications* 244 (June 2024), 122778. https://doi.org/10.1016/j.eswa.2023.122778

[71] Hazel Kim, Daecheol Woo, Seong Joon Oh, Jeong-Won Cha, and Yo-Sub Han. 2021. ALP: Data Augmentation using Lexicalized PCFGs for Few-Shot Text Classification. https://doi.org/10.48550/arXiv.2112.11916 arXiv:2112.11916 [cs].

[72] Jeonghoon Kim, Jung Hyun Lee, Sungdong Kim, Joonsuk Park, Kang Min Yoo, Se Jung Kwon, and Dongsoo Lee. 2023. Memory-Efficient Fine-Tuning of Compressed Large Language Models via sub-4-bit Integer Quantization. https://doi.org/10.48550/arXiv.2305.14152 arXiv:2305.14152 [cs].

[73] Youngwook Kim, Shinwoo Park, Youngsoo Namgoong, and Yo-Sub Han. 2023. ConPrompt: Pre-training a Language Model with Machine-Generated Data for Implicit Hate Speech Detection. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 10964–10980. https://doi.org/10.18653/v1/2023.findings-emnlp.731

[74] Ian D. Kivlichan, Zi Lin, Jeremiah Liu, and Lucy Vasserman. 2021. Measuring and Improving Model-Moderator Collaboration using Uncertainty Estimation. https://doi.org/10.48550/arXiv.2107.04212 arXiv:2107.04212 [cs].

[75] Michał Koziarski. 2021. CSMOUTE: Combined Synthetic Oversampling and Undersampling Technique for Imbalanced Data Classification. In *2021 International Joint Conference on Neural Networks (IJCNN)*. 1–8. https://doi.org/10.1109/IJCNN52387.2021.9533415 ISSN: 2161-4407.

[76] Miroslav Kubat and Stan Matwin. 1997. Addressing the curse of imbalanced training sets: one-sided selection. In *Icml*, Vol. 97. Citeseer, 179. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=ebc3914181d76c817f0e35f788b7c4c0f80abb07 Issue: 1.

[77] Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2021. Data Augmentation using Pre-trained Transformer Models. https://doi.org/10.48550/arXiv.2003.02245 arXiv:2003.02245 [cs].

[78] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. https://doi.org/10.48550/arXiv.1909.11942 arXiv:1909.11942 [cs].

[79] Andrew Lee, Xiaoyan Bai, Itamar Pres, Martin Wattenberg, Jonathan K. Kummerfeld, and Rada Mihalcea. 2024. A Mechanistic Understanding of Alignment Algorithms: A Case Study on DPO and Toxicity. http://arxiv.org/abs/2401.01967 arXiv:2401.01967 [cs].

[80] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. 2017. Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research* 18, 17 (2017), 1–5. http://jmlr.org/papers/v18/16-365.html

[81] Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*. Association for Computational Linguistics, Barcelona, Spain, 74–81. https://aclanthology.org/W04-1013

[82] Wei-Chao Lin, Chih-Fong Tsai, Ya-Han Hu, and Jing-Shang Jhang. 2017. Clustering-based undersampling in class-imbalanced data. *Information Sciences* 409-410 (Oct. 2017), 17–26. https://doi.org/10.1016/j.ins.2017.05.008

[83] Fang Liu, Yang Liu, Lin Shi, Houkun Huang, Ruifeng Wang, Zhen Yang, and Li Zhang. 2024. Exploring and Evaluating Hallucinations in LLM-Powered Code Generation. http://arxiv.org/abs/2404.00971 arXiv:2404.00971 [cs].

[84] Jiacheng Liu, Andrew Cohen, Ramakanth Pasunuru, Yejin Choi, Hannaneh Hajishirzi, and Asli Celikyilmaz. 2023. Don't throw away your value model! Making PPO even better via Value-Guided Monte-Carlo Tree Search decoding. https://doi.org/10.48550/arXiv.2309.15028 arXiv:2309.15028 [cs].

[85] Pei Liu, Xuemin Wang, Chao Xiang, and Weiye Meng. 2020. A survey of text data augmentation. In *2020 International Conference on Computer Communication and Network Security (CCNS)*. IEEE, 191–195. https://ieeexplore.ieee.org/abstract/document/9240734/

[86] Ruibo Liu, Guangxuan Xu, Chenyan Jia, Weicheng Ma, Lili Wang, and Soroush Vosoughi. 2020. Data Boost: Text Data Augmentation Through Reinforcement Learning Guided Conditional Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 9031–9041. https://doi.org/10.18653/v1/2020.emnlp-main.726 arXiv:2012.02952 [cs].

[87] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. https://doi.org/10.48550/arXiv.1907.11692 arXiv:1907.11692 [cs].

[88] Rushi Longadge and Snehalata Dongre. 2013. Class Imbalance Problem in Data Mining Review. http://arxiv.org/abs/1305.1707 arXiv:1305.1707 [cs].

[89] Victoria López, Alberto Fernández, Salvador García, Vasile Palade, and Francisco Herrera. 2013. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences* 250 (Nov. 2013), 113–141. https://doi.org/10.1016/j.ins.2013.07.007

[90] Kosisochukwu Madukwe, Xiaoying Gao, and Bing Xue. 2020. In Data We Trust: A Critical Analysis of Hate Speech Detection Datasets. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*. Association for Computational Linguistics, Online, 150–161. https://doi.org/10.18653/v1/2020.alw-1.18

[91] Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient Low-Rank Hypercomplex Adapter Layers. https://doi.org/10.48550/arXiv.2106.04647 arXiv:2106.04647 [cs].

[92] Raghvendra Mall, Mridul Nagpal, Joni Salminen, Hind Almerekhi, Soon-Gyo Jung, and Bernard J. Jansen. 2020. Four Types of Toxic People: Characterizing Online Users' Toxicity over Time. In *Proceedings of the 11th Nordic Conference on Human-Computer Interaction: Shaping Experiences, Shaping Society*. ACM, Tallinn Estonia, 1–11. https://doi.org/10.1145/3419249.3420142

[93] Viera Maslej-Krešňáková, Martin Sarnovský, Peter Butka, and Kristína Machová. 2020. Comparison of Deep Learning Models and Various Text Pre-Processing Techniques for the Toxic Comments Classification. *Applied Sciences* 10, 23 (Jan. 2020), 8631. https://doi.org/10.3390/app10238631 Number: 23 Publisher: Multidisciplinary Digital Publishing Institute.

[94] Deepshi Mediratta and Nikhil Oswal. 2019. Detect Toxic Content to Improve Online Conversations. http://arxiv.org/abs/1911.01217 arXiv:1911.01217 [cs].

[95] Yu Meng, Jiaxin Huang, Yu Zhang, and Jiawei Han. 2022. Generating Training Data with Language Models: Towards Zero-Shot Language Understanding. https://doi.org/10.48550/arXiv.2202.04538 arXiv:2202.04538 [cs].

[96] Ibomoiye Domor Mienye and Yanxia Sun. 2022. A survey of ensemble learning: Concepts, algorithms, applications, and prospects. *IEEE Access* 10 (2022), 99129–99149. https://ieeexplore.ieee.org/abstract/document/9893798/ Publisher: IEEE.

[97] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*, Vol. 26. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2013/hash/9aa42b31882ec039965f3c4923ce901b-Abstract.html

[98] Hamza Haruna MOHAMMED, Erdogan DOGDU, Abdül Kadir GÖRÜR, and Roya CHOUPANI. 2020. Multi-Label Classification of Text Documents Using Deep Learning. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 4681–4689. https://doi.org/10.1109/BigData50022.2020.9378266

[99] M. Molinara, M.T. Ricamato, and F. Tortorella. 2007. Facing Imbalanced Classes through Aggregation of Classifiers. In *14th International Conference on Image Analysis and Processing (ICIAP 2007)*. 43–48. https://doi.org/10.1109/ICIAP.2007.4362755

[100] Loris Nanni, Carlo Fantozzi, and Nicola Lazzarini. 2015. Coupling different methods for overcoming the class imbalance problem. *Neurocomputing* 158 (June 2015), 48–61. https://doi.org/10.1016/j.neucom.2015.01.068

[101] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2024. A Comprehensive Overview of Large Language Models. https://doi.org/10.48550/arXiv.2307.06435 arXiv:2307.06435 [cs].

[102] Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English Tweets. https://doi.org/10.48550/arXiv.2005.10200 arXiv:2005.10200 [cs].

[103] Tong Niu and Mohit Bansal. 2019. Automatically Learning Data Augmentation Policies for Dialogue Tasks. https://doi.org/10.48550/arXiv.1909.12868 arXiv:1909.12868 [cs].

[104] Aytuğ Onan. 2023. GTR-GA: Harnessing the power of graph-based neural networks and genetic algorithms for text augmentation. *Expert Systems with Applications* 232 (Dec. 2023), 120908. https://doi.org/10.1016/j.eswa.2023.120908

[105] Keiron O'Shea and Ryan Nash. 2015. An Introduction to Convolutional Neural Networks. https://doi.org/10.48550/arXiv.1511.08458 arXiv:1511.08458 [cs].

[106] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. https://doi.org/10.48550/arXiv.2203.02155 arXiv:2203.02155 [cs].

[107] Hemant Palivela. 2021. Optimization of paraphrase generation and identification using language models in natural language processing. *International Journal of Information Management Data Insights* 1, 2 (Nov. 2021), 100025. https://doi.org/10.1016/j.jjimei.2021.100025

[108] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). Association for Computational Linguistics, Doha, Qatar, 1532–1543. https://doi.org/10.3115/v1/D14-1162

[109] Joseph Prusa, Taghi M. Khoshgoftaar, David J. Dittman, and Amri Napolitano. 2015. Using Random Undersampling to Alleviate Class Imbalance on Tweet Sentiment Data. In *2015 IEEE International Conference on Information Reuse and Integration*. 197–202. https://doi.org/10.1109/IRI.2015.39

[110] Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. 2023. Fine-tuning Aligned Language Models Compromises Safety, Even When Users Do Not Intend To! http://arxiv.org/abs/2310.03693 arXiv:2310.03693 [cs].

[111] J. Quijas. 2017. Analysing the effects of data augmentation and free parameters for text classification with recurrent convolutional neural networks. https://www.semanticscholar.org/paper/Analysing-the-effects-of-data-augmentation-and-free-Quijas/7dd727962dfe8a303a191aed177a3d6c89b6ab5b

[112] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9. https://insightcivic.s3.us-east-1.amazonaws.com/language-models.pdf

[113] Shafin Rahman, Salman Khan, and Fatih Porikli. 2018. A Unified Approach for Conventional Zero-Shot, Generalized Zero-Shot, and Few-Shot Learning. *IEEE Transactions on Image Processing* 27, 11 (Nov. 2018), 5652–5667. https://doi.org/10.1109/TIP.2018.2861573 Conference Name: IEEE Transactions on Image Processing.

[114] Rahul, Harsh Kajla, Jatin Hooda, and Gajanand Saini. 2020. Classification of Online Toxic Comments Using Machine Learning Algorithms. In *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE, 1119–1123. https://doi.org/10.1109/ICICCS48265.2020.9120939

[115] Satyendra Singh Rawat and Amit Kumar Mishra. 2022. Review of Methods for Handling Class-Imbalanced in Classification Problems. https://doi.org/10.48550/arXiv.2211.05456 arXiv:2211.05456 [cs].

[116] Julian Risch and Ralf Krestel. 2020. Toxic Comment Detection in Online Discussions. In *Deep Learning-Based Approaches for Sentiment Analysis*, Basant Agarwal, Richi Nayak, Namita Mittal, and Srikanta Patnaik (Eds.). Springer, Singapore, 85–109. https://doi.org/10.1007/978-981-15-1216-2_4

[117] Georgios Rizos, Konstantin Hemker, and Björn Schuller. 2019. Augment to Prevent: Short-Text Data Augmentation in Deep Learning for Hate-Speech Classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM '19)*. Association for Computing Machinery, New York, NY, USA, 991–1000. https://doi.org/10.1145/3357384.3358040

[118] Sergio Rojas-Galeano. 2017. On Obstructing Obscenity Obfuscation. *ACM Transactions on the Web* 11, 2 (May 2017), 1–24. https://doi.org/10.1145/3032963

[119] Vaibhav Rupapara, Furqan Rustam, Hina Fatima Shahzad, Arif Mehmood, Imran Ashraf, and Gyu Sang Choi. 2021. Impact of SMOTE on Imbalanced Text Features for Toxic Comments Classification Using RVVC Model. *IEEE Access* 9 (2021), 78621–78634. https://doi.org/10.1109/ACCESS.2021.3083638

[120] Hadeel Saadany and Constantin Orasan. 2021. BLEU, METEOR, BERTScore: Evaluation of Metrics Performance in Assessing Critical Translation Errors in Sentiment-oriented Text. https://doi.org/10.26615/978-954-452-071-7_006

[121] Mujahed A. Saif, Alexander N. Medvedev, Maxim A. Medvedev, and Todorka Atanasova. 2018. Classification of online toxic comments using the logistic regression and neural networks models. *AIP Conference Proceedings* 2048, 1 (Dec. 2018), 060011. https://doi.org/10.1063/1.5082126

[122] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. https://doi.org/10.48550/arXiv.1910.01108 arXiv:1910.01108 [cs].

[123] Claudio Filipi Gonçalves Dos Santos and João Paulo Papa. 2022. Avoiding Overfitting: A Survey on Regularization Methods for Convolutional Neural Networks. *Comput. Surveys* 54, 10s (Jan. 2022), 1–25. https://doi.org/10.1145/3510413

[124] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. https://doi.org/10.48550/arXiv.1707.06347 arXiv:1707.06347 [cs].

[125] Chris Seiffert, Taghi M. Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. 2010. RUSBoost: A Hybrid Approach to Alleviating Class Imbalance. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 40, 1 (Jan. 2010), 185–197. https://doi.org/10.1109/TSMCA.2009.2029559 Conference Name: IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans.

[126] Sima Sharifirad, Borna Jafarpour, and Stan Matwin. 2018. Boosting Text Classification Performance on Sexist Tweets by Text Augmentation and Text Generation Using a Combination of Knowledge Graphs. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, Darja Fišer, Ruihong Huang, Vinodkumar Prabhakaran, Rob Voigt, Zeerak Waseem, and Jacqueline Wernimont (Eds.). Association for Computational Linguistics, Brussels, Belgium, 107–114. https://doi.org/10.18653/v1/W18-5114

[127] Mayuri S. Shelke, Prashant R. Deshmukh, and Vijaya K. Shandilya. 2017. A review on imbalanced data handling using undersampling and oversampling technique. *International Journal of Recent Trends in Engineering & Research (IJRTER)* 3, 4 (2017), 444–449. https://doi.org/10.23883/IJRTER.2017.3168.0UWXM

[128] Connor Shorten, Taghi M. Khoshgoftaar, and Borko Furht. 2021. Text Data Augmentation for Deep Learning. *Journal of Big Data* 8, 1 (July 2021), 101. https://doi.org/10.1186/s40537-021-00492-0

[129] Ajeet Singh and Anurag Jain. 2022. An efficient credit card fraud detection approach using cost-sensitive weak learner with imbalanced dataset. *Computational Intelligence* 38, 6 (2022), 2035–2055. https://doi.org/10.1111/coin.12555 _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/coin.12555.

[130] Yisheng Song, Ting Wang, Puyu Cai, Subrota K. Mondal, and Jyoti Prakash Sahoo. 2023. A Comprehensive Survey of Few-shot Learning: Evolution, Applications, Challenges, and Opportunities. *Comput. Surveys* 55, 13s (July 2023), 271:1–271:40. https://doi.org/10.1145/3582688

[131] Jerzy Stefanowski and Szymon Wilk. 2008. Selective Pre-processing of Imbalanced Data for Improving Classification Performance. In *Data Warehousing and Knowledge Discovery (Lecture Notes in Computer Science)*, Il-Yeol Song, Johann Eder, and Tho Manh Nguyen (Eds.). Springer, Berlin, Heidelberg, 283–292. https://doi.org/10.1007/978-3-540-85836-2_27

[132] Yanmin Sun, Mohamed S. Kamel, Andrew K.C. Wong, and Yang Wang. 2007. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition* 40, 12 (Dec. 2007), 3358–3378. https://doi.org/10.1016/j.patcog.2007.04.009

[133] Zhongbin Sun, Qinbao Song, Xiaoyan Zhu, Heli Sun, Baowen Xu, and Yuming Zhou. 2015. A novel ensemble method for classifying imbalanced data. *Pattern Recognition* 48, 5 (May 2015), 1623–1637. https://doi.org/10.1016/j.patcog.2014.11.014

[134] Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. 2022. VL-ADAPTER: Parameter-Efficient Transfer Learning for Vision-and-Language Tasks. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, New Orleans, LA, USA, 5217–5227. https://doi.org/10.1109/CVPR52688.2022.00516

[135] Seba Susan and Amitesh Kumar. 2021. The balancing trick: Optimized sampling of imbalanced datasets—A brief survey of the recent State of the Art. *Engineering Reports* 3, 4 (2021), e12298. https://doi.org/10.1002/eng2.12298 _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/eng2.12298.

[136] Muhammad Atif Tahir, Josef Kittler, and Fei Yan. 2012. Inverse random under sampling for class imbalance problem and its application to multi-label classification. *Pattern Recognition* 45, 10 (Oct. 2012), 3738–3750. https://doi.org/10.1016/j.patcog.2012.03.014

[137] Mohammed Temraz and Mark T. Keane. 2022. Solving the class imbalance problem using a counterfactual method for data augmentation. *Machine Learning with Applications* 9 (Sept. 2022), 100375. https://doi.org/10.1016/j.mlwa.2022.100375

[138] Tomek. 1976. Two Modifications of CNN. In *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-6. 769–772. https://doi.org/10.1109/TSMC.1976.4309452 ISSN: 0018-9472, 2168-2909 Issue: 11 Journal Abbreviation: IEEE Trans. Syst., Man, Cybern..

[139] Jason Van Hulse, Taghi M. Khoshgoftaar, and Amri Napolitano. 2009. An empirical comparison of repetitive undersampling techniques. In *2009 IEEE International Conference on Information Reuse & Integration*. 29–34. https://doi.org/10.1109/IRI.2009.5211614

[140] Bertie Vidgen and Leon Derczynski. 2020. Directions in abusive language training data, a systematic review: Garbage in, garbage out. *PLOS ONE* 15, 12 (Dec. 2020), e0243300. https://doi.org/10.1371/journal.pone.0243300 Publisher: Public Library of Science.

[141] Bertie Vidgen, Tristan Thrush, Zeerak Waseem, and Douwe Kiela. 2021. Learning from the Worst: Dynamically Generated Datasets to Improve Online Hate Detection. http://arxiv.org/abs/2012.15761 arXiv:2012.15761 [cs].

[142] Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, and Shengyi Huang. 2020. TRL: Transformer Reinforcement Learning. https://github.com/huggingface/trl

[143] Shoujin Wang, Wei Liu, Jia Wu, Longbing Cao, Qinxue Meng, and Paul J. Kennedy. 2016. Training deep neural networks on imbalanced data sets. In *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, Vancouver, BC, Canada, 4368–4374. https://doi.org/10.1109/IJCNN.2016.7727770

[144] Mike Wasikowski and Xue-wen Chen. 2010. Combating the Small Sample Class Imbalance Problem Using Feature Selection. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (Oct. 2010), 1388–1400. https://doi.org/10.1109/TKDE.2009.187 Conference Name: IEEE Transactions on Knowledge and Data Engineering.

[145] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2022. FINETUNED LANGUAGE MODELS ARE ZERO-SHOT LEARNERS. (2022).

[146] Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural Text Generation with Unlikelihood Training. http://arxiv.org/abs/1908.04319 arXiv:1908.04319 [cs, stat].

[147] Martin Weyssow, Xin Zhou, Kisub Kim, David Lo, and Houari Sahraoui. 2023. Exploring Parameter-Efficient Fine-Tuning Techniques for Code Generation with Large Language Models. http://arxiv.org/abs/2308.10462 arXiv:2308.10462 [cs].

[148] Dennis L. Wilson. 1972. Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. *IEEE Transactions on Systems, Man, and Cybernetics* SMC-2, 3 (July 1972), 408–421. https://doi.org/10.1109/TSMC.1972.4309137 Conference Name: IEEE Transactions on Systems, Man, and Cybernetics.

[149] Sam Witteveen and Martin Andrews. 2019. Paraphrasing with Large Language Models. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*. 215–220. https://doi.org/10.18653/v1/D19-5623 arXiv:1911.09661 [cs].

[150] Sebastien C. Wong, Adam Gatt, Victor Stamatescu, and Mark D. McDonnell. 2016. Understanding Data Augmentation for Classification: When to Warp?. In *2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. 1–6. https://doi.org/10.1109/DICTA.2016.7797091

[151] Tomer Wullach, Amir Adler, and Einat Minkov. 2020. Towards Hate Speech Detection at Large via Deep Generative Modeling. http://arxiv.org/abs/2005.06370 arXiv:2005.06370 [cs].

[152] Tomer Wullach, Amir Adler, and Einat Minkov. 2021. Fight Fire with Fire: Fine-tuning Hate Detectors using Large Samples of Generated Hate Speech. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (Eds.). Association for Computational Linguistics, Punta Cana, Dominican Republic, 4699–4705. https://doi.org/10.18653/v1/2021.findings-emnlp.402

[153] Hanzi Xu, Muhao Chen, Lifu Huang, Slobodan Vucetic, and Wenpeng Yin. 2024. X-Shot: A Unified System to Handle Frequent, Few-shot and Zero-shot Learning Simultaneously in Classification. http://arxiv.org/abs/2403.03863 arXiv:2403.03863 [cs].

[154] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A. Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, Sergey Yekhanin, and Huishuai Zhang. 2022. Differentially Private Fine-tuning of Language Models. http://arxiv.org/abs/2110.06500 arXiv:2110.06500 [cs, stat].

[155] Zhang, Yuan and Baldridge, Jason and He, and Luheng. 2019. PAWS: Paraphrase Adversaries from Word Scrambling. https://github.com/google-research-datasets/paws original-date: 2019-03-12T23:00:22Z.

[156] Chong Zhang, Kay Chen Tan, Haizhou Li, and Geok Soon Hong. 2019. A Cost-Sensitive Deep Belief Network for Imbalanced Classification. *IEEE Transactions on Neural Networks and Learning Systems* 30, 1 (Jan. 2019), 109–122. https://doi.org/10.1109/TNNLS.2018.2832648 Conference Name: IEEE Transactions on Neural Networks and Learning Systems.

[157] Renrui Zhang, Jiaming Han, Chris Liu, Peng Gao, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, and Yu Qiao. 2023. LLaMA-Adapter: Efficient Fine-tuning of Language Models with Zero-init Attention. http://arxiv.org/abs/2303.16199 arXiv:2303.16199 [cs].

[158] Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2023. Instruction Tuning for Large Language Models: A Survey. http://arxiv.org/abs/2308.10792 arXiv:2308.10792 [cs].

[159] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675* (2019). https://arxiv.org/abs/1904.09675

[160] Yaqi Zhang, Viktor Hangya, and Alexander Fraser. 2024. A Study of the Class Imbalance Problem in Abusive Language Detection. In *Proceedings of the 8th Workshop on Online Abuse and Harms (WOAH 2024)*, Yi-Ling Chung, Zeerak Talat, Debora Nozza, Flor Miriam Plaza-del Arco, Paul Röttger, Aida Mostafazadeh Davani, and Agostina Calabrese (Eds.). Association for Computational Linguistics, Mexico City, Mexico, 38–51. https://doi.org/10.18653/v1/2024.woah-1.4

[161] Zhixue Zhao, Ziqi Zhang, and Frank Hopfgartner. 2021. A Comparative Study of Using Pre-trained Language Models for Toxic Comment Classification. In *Companion Proceedings of the Web Conference 2021 (WWW '21)*. Association for Computing Machinery, New York, NY, USA, 500–507. https://doi.org/10.1145/3442442.3452313

[162] Jinhua Zhu, Fei Gao, Lijun Wu, Yingce Xia, Tao Qin, Wengang Zhou, Xueqi Cheng, and Tie-Yan Liu. 2019. Soft Contextual Data Augmentation for Neural Machine Translation. http://arxiv.org/abs/1905.10523 arXiv:1905.10523 [cs].

[163] Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-Tuning Language Models from Human Preferences. https://doi.org/10.48550/arXiv.1909.08593 arXiv:1909.08593 [cs, stat].

[164] Gözde Gül Şahin and Mark Steedman. 2018. Data Augmentation via Dependency Tree Morphing for Low-Resource Languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, Brussels, Belgium, 5004–5009. https://doi.org/10.18653/v1/D18-1545

[165] Gözde Gül Şahin and Mark Steedman. 2019. Data Augmentation via Dependency Tree Morphing for Low-Resource Languages. https://doi.org/10.48550/arXiv.1903.09460 arXiv:1903.09460 [cs].

A Appendix A

In this section, we outline the experimental configurations for the classifiers developed in this research. We divided all datasets, including five balanced datasets generated using TDA and the original imbalanced dataset, into training and testing sets. The training set consists of 70% of the data, with 10% allocated for the validation set, and the remaining 20% designated for the test set, applied to both the Jigsaw and ToxiGen datasets. To ensure consistency across classifiers and datasets, we used a random-state approach for the train-test split, resulting in identical training and testing sets for all classifiers. All six classifiers received the same training and testing sets to maintain uniformity in the experimental setup.

A.1 CNN-based: CNN and CNN-fastText for balanced datasets

We started the training process with a data cleaning and preprocessing phase as the initial step. This process includes the removal of URLs, punctuation, digits, stop-words, and the normalization of cases, acronyms, and abbreviations. Subsequently, the tokenization process takes place, with text comments either truncated or padded to achieve a fixed length before being transformed into vector words. The model architecture incorporates four 1D convolutional layers with multiple window sizes and output channels. A pooling layer is employed to reduce the dimensionality of the convolutions, followed by a global max pooling step to further diminish the dimension. ReLU activation functions are used in the hidden layers, while the output layer employs a Sigmoid activation function. During training, the binary cross-entropy loss function measures the disparity between predicted and actual labels. The Adam optimizer is applied to update the model's parameters, facilitating faster convergence and improved accuracy.

To expedite the training process, we integrated pre-trained fastText word embeddings into our CNN model, resulting in the CNN-fastText variant. Specifically, we utilized the "crawl-300d-2M" embeddings, consisting of 2 million word vectors trained on Common Crawl using a continuous bag of words (CBOW) with position weights, character n-grams of length 5, and a window of size 5 and 10 negatives. The CNN model convolves over these embedded vectors, which then undergo max pooling to reduce dimensionality. The final classification is executed using fully connected layers with a Sigmoid activation function. Incorporating pre-trained word embeddings significantly reduced the training time of our CNN model. We systematically tested hyperparameters to optimize results for each dataset. A detailed overview of the final hyperparameter settings is presented in Table 15 for datasets developed using prompts from the Jigsaw dataset and in Table 16 for those developed using the ToxiGen dataset.

Table 15. Experimental Configurations for CNN-based Classifiers based on Jigsaw Dataset

Model	Dataset	Epoch	Batch size	Learning rate	Dropout rate	Filters	Max Features	Max Length
CNN	$Unbalanced_{\text{Jigsaw}}$	35	128	5.00E-05	0.5	128	100000	400
	$Balanced_{\text{Zero-shot}}$	25	128	5.00E-03	0.3	128	100000	400
	$Balanced_{\text{Back-translation}}$	18	512	2.00E-05	0.5	128	100000	400
	$Balanced_{LoRA}$	28	512	2.00E-05	0.3	128	100000	400
	$Balanced_{\text{PPO-RoBERTa}}$	15	128	2.00E-04	0.5	300	100000	400
	$Balanced_{\text{PPO-API}}$	10	512	1.00E-05	0.5	300	100000	400
CNN-fasText	$Unbalanced_{\text{Jigsaw}}$	19	32	1.00E-03	0.4	128	100000	400
	$Balanced_{\text{Zero-shot}}$	17	256	5.00E-05	0.5	256	100000	400
	$Balanced_{\text{Back-translation}}$	33	128	3.00E-04	0.3	128	100000	400
	$Balanced_{LoRA}$	6	256	2.00E-04	0.4	300	100000	400
	$Balanced_{\text{PPO-RoBERTa}}$	22	256	5.00E-04	0.5	128	100000	400
	$Balanced_{\text{PPO-API}}$	25	512	5.00E-05	0.5	128	100000	400

Table 16. Experimental Configurations for CNN-based Classifiers based on ToxiGen Dataset

Model	Dataset	Epoch	Batch size	Learning rate	Dropout rate	Filters	Max Features	Max Length
CNN	$Unbalanced_{\text{ToxiGen}}$	31	512	5.00E-05	0.5	300	100000	400
	$Balanced_{\text{Zero-shot}}$	26	256	5.00E-03	0.5	300	100000	400
	$Balanced_{\text{Back-translation}}$	20	128	5.00E-05	0.5	300	100000	400
	$Balanced_{LoRA}$	22	512	5.00E-05	0.4	128	100000	400
	$Balanced_{\text{PPO-RoBERTa}}$	27	256	5.00E-05	0.4	128	100000	400
	$Balanced_{\text{PPO-API}}$	17	512	5.00E-05	0.4	128	100000	400
CNN-fasText	$Unbalanced_{\text{Jigsaw}}$	12	256	2.00E-03	0.5	128	100000	400
	$Balanced_{\text{Zero-shot}}$	15	256	5.00E-05	0.5	300	100000	400
	$Balanced_{\text{Back-translation}}$	24	128	4.00E-04	0.5	300	100000	400
	$Balanced_{LoRA}$	18	64	4.00E-04	0.5	300	100000	400
	$Balanced_{\text{PPO-RoBERTa}}$	15	64	5.00E-05	0.5	300	100000	400
	$Balanced_{\text{PPO-API}}$	23	64	5.00E-05	0.5	300	100000	400

A.2 Transformer-based: BERT, RoBERTa, HateBERT, and BERTweet

We used the Huggingface transformer library, compatible with Tensorflow 2.14.0, for our work. To identify toxic content, we employed the bert-base-uncased, roberta-base, GroNLP/hateBERT, and vinai/bertweet-base variants. Text data preparation involved the use of tokenizers, which convert raw text into tokens compatible with BERT/RoBERTa. For each model, we explored a distinct set of hyperparameters randomly to identify the configuration yielding the most accurate results. The optimal hyperparameters for each model developed via Jigsaw-based datasets are detailed via Table 17 and Table 18 for ToxiGent-based datasets.

Received 11 January 2024; revised 9 September 2024; accepted 8 October 2024

Table 17. Experimental Configurations for Transformer-based Classifiers: Jigsaw-based Datasets

Model	Dataset	Epoch	Batch size	Learning rate	Dropout rate	Max Length
BERT	$Unbalanced_{\text{Jigsaw}}$	2	16	2e-5	0.5	400
	$Balanced_{\text{Zero-shot}}$	2	8	3e-5	0.4	400
	$Balanced_{\text{Back-translation}}$	2	8	3e-5	0.4	400
	$Balanced_{LoRA}$	1	8	4e-5	0.5	400
	$Balanced_{\text{PPO-RoBERTa}}$	1	2	2e-5	0.4	400
	$Balanced_{\text{PPO-API}}$	1	4	2e-5	0.5	400
RoBERTa	$Unbalanced_{\text{Jigsaw}}$	2	16	4e-5	0.3	400
	$Balanced_{\text{Zero-shot}}$	2	8	3e-5	0.4	400
	$Balanced_{\text{Back-translation}}$	3	8	3e-5	0.3	400
	$Balanced_{LoRA}$	1	16	3e-5	0.4	400
	$Balanced_{\text{PPO-RoBERTa}}$	1	8	3e-5	0.5	400
	$Balanced_{\text{PPO-API}}$	1	2	3e-5	0.4	400
HateBERT	$Unbalanced_{\text{Jigsaw}}$	2	16	4e-5	0.3	400
	$Balanced_{\text{Zero-shot}}$	2	128	3e-5	0.4	400
	$Balanced_{\text{Back-translation}}$	2	64	2e-5	0.4	400
	$Balanced_{LoRA}$	1	16	2e-5	0.4	400
	$Balanced_{\text{PPO-RoBERTa}}$	1	4	2e-5	0.5	400
	$Balanced_{\text{PPO-API}}$	1	4	2e-5	0.5	400
BERTweet	$Unbalanced_{\text{Jigsaw}}$	1	16	4e-5	0.3	400
	$Balanced_{\text{Zero-shot}}$	1	256	3e-5	0.5	400
	$Balanced_{\text{Back-translation}}$	1	64	2e-5	0.5	400
	$Balanced_{LoRA}$	1	8	2e-5	0.5	400
	$Balanced_{\text{PPO-RoBERTa}}$	1	4	2e-5	0.5	400
	$Balanced_{\text{PPO-API}}$	1	4	2e-5	0.5	400

Table 18. Experimental Configurations for Transformer-based Classifiers: ToxiGen-based Datasets

Model	Dataset	Epoch	Batch size	Learning rate	Dropout rate	Max Length
BERT	$Balanced_{\text{Zero-shot}}$	2	128	2e-5	0.5	400
	$Balanced_{\text{Back-translation}}$	2	64	2e-5	0.5	400
	$Balanced_{LoRA}$	1	16	2e-5	0.5	400
	$Balanced_{\text{PPO-RoBERTa}}$	1	2	2e-5	0.4	400
	$Balanced_{\text{PPO-API}}$	1	2	2e-5	0.5	400
RoBERTa	$Balanced_{\text{Zero-shot}}$	2	128	2e-5	0.4	400
	$Balanced_{\text{Back-translation}}$	1	64	2e-5	0.3	400
	$Balanced_{LoRA}$	1	16	2e-5	0.5	400
	$Balanced_{\text{PPO-RoBERTa}}$	1	4	2e-5	0.5	400
	$Balanced_{\text{PPO-API}}$	1	4	2e-5	0.4	400
HateBERT	$Balanced_{\text{Zero-shot}}$	2	256	2e-5	0.4	400
	$Balanced_{\text{Back-translation}}$	2	128	2e-5	0.3	400
	$Balanced_{LoRA}$	1	16	2e-5	0.5	400
	$Balanced_{\text{PPO-RoBERTa}}$	1	2	2e-5	0.5	400
	$Balanced_{\text{PPO-API}}$	1	2	2e-5	0.5	400
BERTweet	$Unbalanced_{\text{Zero-shot}}$	2	128	2e-5	0.5	400
	$Balanced_{\text{Back-translation}}$	2	64	2e-5	0.5	400
	$Balanced_{LoRA}$	1	64	2e-5	0.5	400
	$Balanced_{\text{PPO-RoBERTa}}$	1	2	2e-5	0.5	400
	$Balanced_{\text{PPO-API}}$	1	2	2e-5	0.5	400