Generative LLM oversampling for the class imbalance problem in hate speech detection

Nicolas Antonio Cloutier[1] and Nathalie Japkowicz[2]

[1] Jackson-Reed High School, Washington, D.C., USA
nicocloutier1@gmail.com
[2] American University, Washington, D.C., USA
japkowicz@american.edu

Abstract. Online hate speech has become increasingly prevalent with the rise of social media. As such, methods for automatically detecting and classifying hate speech have become the subject of much research. A common challenge in this and other domains is the class imbalance problem, where one class in a dataset is far more common than another. We propose the use of a generative large language model (LLM), specifically OpenAI's GPT-2, as a method of oversampling text data in order to account for this imbalance, comparing it to other resampling methods on three tasks: binary classification of tweets as antisemitic or not, multiclass classification of antisemitic tweets into subtypes, and a combination of the two. We find that generative LLM resampling does not produce better results for binary classification than other resampling methods, but does improve performance on the other two tasks.

1 Introduction

In recent years, hate speech has become increasingly mainstream and common within social media sites [29]. This rise has not only caused online spaces to become less hospitable, but has also had several offline effects. On top of having severe psychological effects on the recipient [29], online hate speech also played a role in disseminating extremist anti-Rohingya voices leading to violence in Myanmar [13], and has provided motivation for several perpetrators of offline violent hate crimes [29]. These events have motivated responses from numerous parties, including the social media sites themselves, that are increasingly looking to stop the spread of these messages [31], and governmental bodies, that are seeking to regulate or prevent the spread of hate speech [2]. At the same time as hate speech has been becoming more common, social media sites have been generating more and more content, with popular social media site Twitter generating an average of 500 million tweets per day in 2019 [24].

With these developments and the large amount of content on social media sites, these sites have been increasingly looking to automatic detection methods for hate speech [31]. These methods use Machine Learning (ML) to automatically detect and classify hateful speech, removing some of the work done by moderators, whose primary job is to respond reactively to user reports of hate

speech [31]. [4] used a combination of image and text processing and classification algorithms to classify images and text on social media sites Twitter and Gab as antisemitic or not [4]. In this paper, we seek a lower cost method that uses only the text component of their dataset to further analyze the presence of hate speech on Twitter and investigate new algorithms for classification.[3]

One difficulty with this dataset is that it is imbalanced, meaning one classification is far more common in the dataset than another. Imbalanced data can negatively impact the performance of ML classification algorithms [30], affecting numerous domains that use ML classification. Many ML algorithms are inadequately prepared to handle the class imbalance problem [30], leading many to look to other solutions, including resampling methods, that in some way change the training data by adding or removing samples in order to allay the effects of the class imbalance problem [16].

With this in mind, along with the recent advances in generative LLMs, we ask whether such methods can allay the class imbalance problem in text data better than traditional approaches, and to what extent that may improve hate speech detection on social media.

2 Previous work

Antisemitism detection, as well as hate speech detection generally, has been the subject of much research. [23] used ML models to analyze and classify antisemitism in posts on social media sites Gab and 4chan, and [12] used large language models (LLMs) to do the same. [4] introduced a new dataset of labeled antisemitic posts, including text and images. These posts were labeled not only for whether or not they are antisemitic, but also their type of antisemitism, with the researchers grouping antisemitic posts into political, economic, religious, and racial antisemitism, and trained models to classify both antisemitic status and type of antisemitism. This Twitter dataset is imbalanced, with the non-antisemitic class far outnumbering the antisemitic class, and with political and racial antisemitism being more common than religious and economic antisemitism.

Imbalance is a common issue in ML classification problems. The class imbalance problem can severely negatively affect model predictive power, with the less common class (termed the "minority class") often being misclassified due to its low prevalence in the dataset relative to the larger class (the "majority class") [1]. In multiclass classification, the classes with fewer samples will similarly perform worse to those with more samples. This problem has affected fields as distinct as medicine, fraudulent call detection, and risk management [30]. Due to the severity with which this problem can limit model performance and its widespread nature, being seen in numerous distinct fields, it has become a focus of researchers as an area of improvement [1], with numerous techniques being created to allay the effects of the problem.

[3] Our dataset contains fewer samples than that of [4] because Twitter removed some of the hateful posts before we collected the data.

One such group of techniques is resampling, altering the training data by adding new samples or taking existing samples away in order to improve model performance. Generally, there are two types of resampling: oversampling, which describes the process of adding samples to a dataset, and undersampling, which describes the process of removing existing samples from a dataset [28]. Resampling techniques can improve the performance of ML models when trained on imbalanced data [21] [18].

One method for oversampling is the use of generative models, including generative adversarial networks (GANs) and autoencoders to generate synthetic data. These methods seek to match the distribution of the original dataset and create synthetic samples in accordance with that distribution [14], with the goal of creating synthetic examples for model training that match the original distribution closely. They have achieved success in their applications, but primarily in computer vision and tabular data [14] [8] [3] [5]. Applications of these generative models to natural language processing (NLP) do occur [25], but they are generally less common than applications to other areas. NLP tends to use more traditional statistical methods for oversampling such as SMOTE and random oversampling [33] [11].

While these advances in resampling have been occurring, similar advances have been made in generative LLMs, such as the GPT series of models from OpenAI, which have the ability to generate human-like text [10] and have been applied to fields such as patent claim generation [20] and healthcare education [26]. These LLMs have been investigated as a method for oversampling text data to account for the class imbalance problem, with relatively consistent success over not using resampling [7] [32] [27]. That being said, these studies do not compare the method with other, simpler or more traditional resampling methods. [6] argues that complex methods of resampling are difficult to justify without marked improvements in performance over simpler methods. This provides motivation, now that the method has shown success in accounting for the class imbalance, for investigating how it compares to existing methods of resampling and under which circumstances it improves relative to those methods.

3 Methodology

The purpose of our paper is to study the effects of various types of resampling on hate speech detection and classification tasks. Our models were trained on three different tasks. The first was a simple binary classification task, where the models attempted to classify the text of a tweet as either antisemitic or not antisemitic. The second was the 4-class type classification, where the dataset was limited to only antisemitic samples, that was then grouped into four classes: political, economic, religious, and racial antisemitism. Finally, the models were trained on a 5-class type classification task, where the dataset included both antisemitic and non-antisemitic samples, and the model attempted to classify the samples into one of the four groups of antisemitism or classify them as not antisemitic, creating five classes total. Every model was trained and tested separately for all three tasks.

In order to best represent a variety of ML architectures, different algorithms and methods of text representation were used. For algorithms, we trained classifiers utilizing the Naïve Bayes, Extreme Gradient Boosting, Decision Tree, and Support Vector Machine algorithms. We also used three methods of representing text: term frequency-inverse document frequency (TF-IDF), raw frequency, and Bag of Words (BoW).[4] Each model was trained with each method of text representation, creating a total of twelve models that were trained for each task. In order to reduce the dimensionality of the dataset, representations for training and testing data were limited to words that had a frequency score of at least 0.5%, meaning that the word had to have a frequency of at least 0.5% of the original, full length of the text in order to be considered by the models.

The dataset is imbalanced. Figure 1 shows the distribution of different classes in the dataset. The first and second charts are for the entire dataset, and the third, containing classifications on types of antisemitism, only contains samples that are antisemitic.

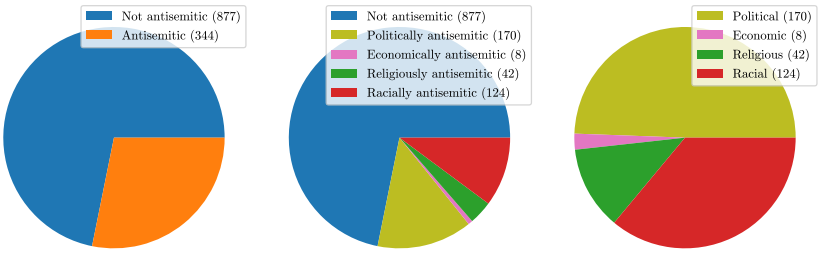

Figure 1. Classification distributions

Several methods of resampling were used to reduce the impact of the class imbalance, which were chosen to represent a variety of resampling techniques and algorithms. One set of models was trained with no resampling, with additional sets being trained using random undersampling [28], random oversampling [28], SMOTE with Tomek Links[5] [9], AdaSyn [15], and a final set being trained on the augmented dataset generated using the LLM. GPT-2 was used to generate the samples because it is easily available and callable programmatically with the HuggingFace API, unlike more recently released GPT models. The augmented dataset was created by generating 20 samples from each sample in the original dataset, creating a dataset that was larger, but had the same percentages of class imbalance. These models then used random undersampling on this larger dataset to account for this. This was done in order to avoid having to regenerate samples for each model, due to the large number of models being trained.

[4] We initially included a BERT-based representation method, but the models trained with this method produced results no better than chance.

[5] SMOTE with Tomek links was selected over only SMOTE in order to represent combined oversampling and undersampling methods, as there are two other methods already (AdaSyn and random oversampling) that are pure oversampling.

In regards to testing, each model was tested using 10-fold cross-validation on the original dataset. It was ensured that no samples generated from oversampling methods were used during testing, and additionally that, when testing the augmented model, if a sample that was used to generate more samples appeared in the testing split, the samples generated with it would not appear in the training split. This ensures that the models were not unfairly advantaged, and that each was tested on a large amount of genuine, unseen data.

For model evaluation, the three main metrics used were the macro-averages of the recall, precision, and F_1 scores across each of the classes the model had to classify for each task. These were used in lieu of accuracy in order to account for the class imbalance in the data. Once the models were evaluated, their answers to the testing samples were converted to a binary matrix with each column representing a model and each row representing a sample. We then used the Cochran's Q-test, an extension of McNemar's test appropriate for the case of comparing multiple algorithms on a single domain, to test for significant difference in the models, then the Dunn test for post-hoc analysis.[6] When testing the resampling methods against each other, the data were turned into another matrix with each column representing a resampling method and each row representing a model trained with that method, with the value in the cell being the mean recall, precision, or F_1 score of that model. A Friedman's χ^2 test was then performed with a post-hoc Nemenyi test to analyze the results.

4 Results

Overall, the models trained on the LLM-augmented data showed no improvement on the binary task, but they did show improvement on the 5-class classification and especially the 4-class classification tasks. Tables 1 contains information for the binary classification task with each resampling method, as well as the mean model performance for each resampling method out of the twelve combinations of each algorithm (Naïve Bayes, SVM, Decision Tree, and Extreme Gradient Boosting) and representation method (BoW, TF-IDF, and frequency), with "performance" meaning the mean recall, precision, or F_1 score the model achieved. The data used to create the F_1 column of Table 1 received a Friedman χ^2 test p-value of 2.2×10^{-2}, allowing a rejection of H_0 that each method comes from the same distribution with $\alpha = 0.05$. After each table, an array is provided with the results of the post-hoc Nemenyi test on the resampling methods for the F_1 score, with p-values displayed in the array. A higher p-value and lighter color means we cannot reject the null hypothesis, while a lower p-value and darker color means we can. In these arrays, "RU" stands for random undersampling, "ST" stands for SMOTE with Tomek links, "RO" stands for random oversampling, "AD" stands for AdaSyn, "NO" stands for no resampling method, and "LA" stands for LLM-augmented.

[6] See [17]

Table 1. Binary task resampling method performance

Resampling Method	Recall (% over none)	Precision (% over none)	F_1 (% over none)
None (NO)	0.613 (0%)	**0.644 (0%)**	0.627 (0%)
RandomUnder (RU)	**0.638 (4.1%)**	0.621 (-3.6%)	**0.630 (0.5%)**
SMOTE-Tomek (ST)	0.614 (0.1%)	0.612 (-5.0%)	0.613 (-2.2%)
Random Over (RO)	0.623 (1.6%)	0.617 (-4.2%)	0.620 (-1.1%)
AdaSyn (AD)	0.608 (-0.8%)	0.600 (-6.8%)	0.604 (-3.7%)
LLM-Augmented (LA)	0.617 (0.6%)	0.613 (-4.8%)	0.614 (-2.1%)

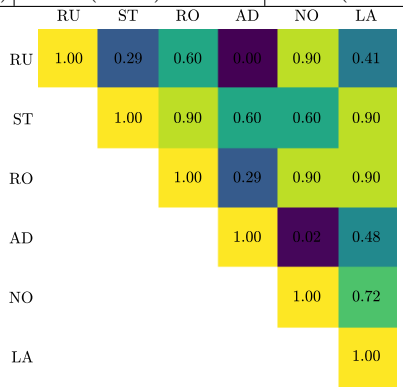

Figure 2. Binary F_1 Nemenyi test results

In the binary task, the LLM-augmented models show no performance improvement over the other models. Generally, the resampling methods tend to do better than the models with no resampling on recall, but have worse precision performance, with this pattern being sustained by the models trained on LLM-augmented data.

The results of the 5-class classification task are shown below. The data used to create the F_1 column of Table 2 received a Friedman χ^2 test p-value of 1.6×10^{-3}.

Table 2. 5-class task resampling method performance

Resampling Method	Recall (% over none)	Precision (% over none)	F_1 (% over none)
None (NO)	0.265 (0%)	**0.325 (0%)**	0.286 (0%)
RandomUnder (RU)	0.265 (0%)	0.249 (-23.4%)	0.256 (-10.4%)
SMOTE-Tomek (ST)	0.271 (2.3%)	0.264 (-18.8%)	0.267 (-6.6%)
Random Over (RO)	0.298 (12.4%)	0.271 (-16.6%)	0.283 (-1.0%)
AdaSyn (AD)	0.251 (-5.3%)	0.286 (-12.0%)	0.264 (-7.7%)
LLM-Augmented (LA)	**0.347 (30.9%)**	0.270 (-16.9%)	**0.303 (5.9%)**

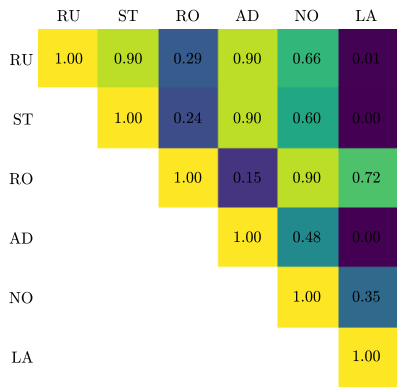

Figure 3. 5-class F_1 Nemenyi test results

The results on the 5-type task begin to show the benefits of LLM augmentation. The pattern of no resampling outperforming resampling on precision while the inverse happens on recall remains, but now LLM augmentation improves over no resampling by a far wider margin than any other resampling method, making its gain in mean recall account for its loss in precision, giving it a higher F_1 score than the other methods.

The results of the 4-class classification task are shown below. The data used to create the F_1 column of Table 3 received a Friedman χ^2 test p-value of 7.1×10^{-5}.

Table 3. 4-class task resampling method performance

Resampling Method	Recall (% over none)	Precision (% over none)	F_1 (% over none)
None (NO)	0.328 (0%)	0.372 (0%)	0.346 (0%)
RandomUnder (RU)	0.339 (3.3%)	0.323 (-13.2%)	0.330 (-4.6%)
SMOTE-Tomek (ST)	0.366 (11.6%)	0.347 (-6.7%)	0.356 (2.9%)
Random Over (RO)	0.358 (9.1%)	0.351 (-5.6%)	0.354 (2.3%)
AdaSyn (AD)	0.322 (-1.8%)	0.344 (-7.5%)	0.331 (-4.3%)
LLM-Augmented (LA)	**0.433 (32.0%)**	**0.409 (9.9%)**	**0.419 (21.1%)**

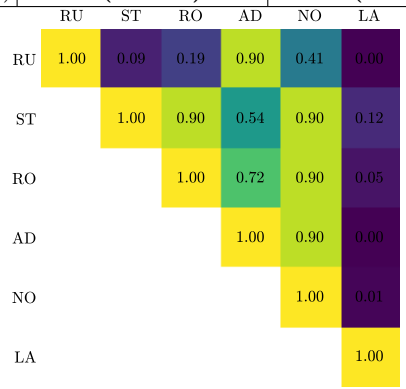

Figure 4. 4-class F_1 Nemenyi test results

Now, the pattern of gains on recall and losses on precision continue for all methods except LLM augmentation. LLM-augmented models not only increase the recall by a greater margin than any other method, but also increase the precision

of the model on the 4-class task. This makes it the only resampling method in any of the three tasks to receive a higher precision score either on the best model or on mean score than no resampling, on top of widening the margin by which it beats no resampling compared to the 5-class task.

One initial possible explanation for the high performance of the models with LLM-augmented data relative to other models on the 5-class and especially 4-class tasks is that the LLM augmentation provides performance improvements with only one particular class that other resampling methods did poorly on, artificially raising the mean recall even though there was only one class affected significantly. Looking further at the data, however, while some classes were more affected than others, overall, the models trained on LLM-augmented data show improvements in multiple classes. On the 4-type classification task, Table 4 shows the mean F_1 of each method on each particular class, respectively.

Table 4. 4-class task class-wise resampling method F_1 score

Resampling Method	Political	Economic	Religious	Racial
None (NO)	**0.597**	0.059	0.233	0.315
Random Undersampling (RU)	0.463	0.050	0.276	0.253
SMOTE-Tomek (ST)	0.568	0.056	0.304	0.288
Random Oversampling (RO)	0.558	0.038	0.294	0.280
AdaSyn (AD)	0.581	0.046	0.228	0.301
LLM-Augmented (LA)	0.526	**0.161**	**0.403**	**0.415**

In F_1 score, the model is outperformed in political antisemitism (the largest class), but it outperforms the other models in all other classes.

5 Discussion

The most striking result is the difference in relative performance of the resampling methods between the three tasks. The binary task sees small improvements in recall with overall larger losses in precision, the 5-class task sees large losses in precision with large gains in recall, particularly by the LLM, and the 4-class task sees large gains in recall and precision for the LLM, with other methods having smaller recall gains and performing worse than no resampling on precision. This can primarily be explained through two factors: the degree of imbalance of the data and the degree of appropriateness of an oversampling technique to the task at hand.

First, the degree of imbalance can affect the relative performance of resampling methods. The binary task saw the least benefit from resampling methods, as its losses in precision generally outweighed its gains in recall. This can be explained by the fact that the binary task had relatively large amounts of data for both classes, making resampling less useful compared to other tasks, hence why the observed improvement in recall is less than that of other tasks. For the 5-class task, on the other hand, the classes are imbalanced to a very high degree, with sample sizes ranging from 8 to 877. In order to make these classes equal in size, drastic measures must be taken, either by removing large portions of the dataset or adding a quantity of synthetic or duplicated samples that outnumber the original size of that class within the dataset. This can, in the case of under-

sampling, remove critical pieces of information that could help the model [22] [19], and, in the case of oversampling, introduce excessive amounts of synthetic data or make the dataset prone to overfitting [22] [19]. This could explain the large decreases in precision in this task. On the 4-class task, the imbalance is far more even, with sample sizes ranging from 8 to 170, certainly still a large difference, but a far cry from the 5-class task. This could explain why the losses in precision for the 4-class task are not as high as those in the 5-class task.

This factor, however, does not explain all of the difference. Why, for example, do the models trained on LLM-generated data outperform other models by such a wide margin in the 4-class task, not only in recall but also precision? Here, it is necessary to look at the methods themselves and the algorithms they use. The generative LLM, being trained on large amounts of text data, has likely been trained partially on samples similar in topic, purpose, or register to the ones present in the dataset. Its familiarity with this type of text allows it to generate more appropriate synthetic samples than those generated purely geometrically by such algorithms as SMOTE and AdaSyn, while avoiding the overfitting that can plague random oversampling as a result of simply duplicating data [22] [19].

These two factors, when considered together, can explain a large portion of the observed differences in performance of the resampling methods across the three tasks. The binary case saw very little improvement from any method because the degree of imbalance was not very high, and there was ample data present from both classes. The 5-class task saw the best performance from the models trained on LLM-generated data because this method is the best-suited to the task, but the very large class imbalance negatively affected precision to a high degree. The 4-class task also saw the best performance from these models for the same reason, but the lower degree of class imbalance made the losses in precision lower, making all methods, but especially the generative LLM due to its high appropriateness to the task, perform better on average precision on this task than the previous one.

6 Conclusions and Future Work

In this paper, we propose a method of oversampling for text data involving the use of generative LLMs to generate text samples and account for the class imbalance problem. We apply this method to an imbalanced antisemitism classification dataset on three tasks, one binary classification and two multiclass classification. We find that this method does not provide benefits over other methods on the binary classification task, but provides improvements over other methods in the 5-class and especially 4-class tasks. Our findings suggest that generative LLM oversampling, like other resampling methods, does not provide large benefits when there is plentiful data for both classes, and will not perform as well when there is a very large degree of imbalance, but has merit over other methods in certain tasks. We hypothesize that this is because the generative LLMs, by basing the generation on existing text knowledge rather than geometry like other synthetic oversampling methods, are more appropriate to the task of oversampling text data.

For future work in the area, we believe it is important to verify the results and conclusions we have reached. This could be achieved through testing the methods on a wide variety of tasks, with different types of text data (different registers, topics, etc.), different numbers of classes, and different degrees of imbalance. These studies could help test our analyses of the results and pinpoint the situations where generative LLM resampling is most appropriate.

References

1. Abd Elrahman, S. M., Abraham, A: A review of class imbalance problem. Journal of Network and Innovative Computing **1**, 332–340 (2013)
2. Banks, J.: Regulating hate speech online. International Review of Law, Computers and Technology **24**(3), 233–239 (2010)
3. Bellinger, C., Japkowicz, N., Drummond, C.: Synthetic oversampling for advanced radioactive threat detection. IEEE Conference on Machine Learning and Applications 2015
4. Chandra, M., Pailla, D., Bhatia, H., Sanchawala, A., Gupta, M., Shrivastava, M., Kumaraguru, P.: "Subverting the Jewtocracy": Online Antisemitism Detection Using Multimodal Deep Learning. ACM Web Science Conference 2021, 148–157
5. Dai, W., Ng, K., Severson, K., Huan, W., Anderson, F., Stultz, C.: Generative oversampling with a contrastive variational autoencoder. IEEE International Conference on Data Mining 2019, 101–109
6. Drummond, C., Holte, R. C.: C4.5, Class Imbalance, and Cost Sensitivity: Why Under-Sampling beats Over-Sampling. Workshop on Learning from Imbalanced Datasets 2003
7. Edwards, A., Ushio, A., Camacho-Collados, J., De Ribaupierre, H., Preece, A.: Guiding Generative Language Models for Data Augmentation in Few-Shot Text Classification. *arXiv preprint arXiv:2111.09064*, (2021)
8. Engelmann, J., Lessmann, S.: Conditional Wasserstein GAN-based oversampling of tabular data for imbalanced learning. Expert Systems with Applications **174**, (2021)
9. Fernández, A., García, S., Herrera, F., Chawla, N. V.: SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary. Journal of Artificial Intelligence Research **61**, 863–905 (2018)
10. Floridi, L., Chiriatti, M.: GPT-3: Its nature, scope, limits, and consequences. Minds and Machines **30**, 681–694 (2020)
11. Glazkova, A.: A comparison of synthetic oversampling methods for multi-class text classification. *arXiv preprint arXiv:2008.04636t*, (2020)
12. González-Pizarro, F., Zannettou, S.: Understanding and detecting hateful content using contrastive learning. *arXiv preprint arXiv:2201.08387*, (2022)
13. Green, P., MacManus, T., De la Cour Venning, A.: Countdown to Annihilation: Genocide in Myanmar. International State of Crime Initiative, London (2015)
14. Hao, J., Wang, C., Zhang, H., Yang, G.: Annealing genetic GAN for minority oversampling. *arXiv preprint*, (2020)
15. He, H., Bai, Y., Garcia, E. A., Li, S.: ADASYN: Adaptive Synthetic Approach for Imbalanced Learning. IEEE International Joint Conference on Neural Networks 2008, 1322–1328
16. Japkowicz, N., Shaju, S.: The class imbalance problem: A systematic study. Intelligent Data Analysis **6**(5), 429–449 (2002)

17. Japkowicz, N., Boukouvalas, Z., Shah, M.: Machine Learning Evaluation (in preparation)
18. Khushi, M., Shaukat, K., Alam, T. M., Hammed, I. A., Uddin, S., Luo, S., Yang, X., Consuelo Reyes, M.: A Comparative Performance Analysis of Data Resampling Methods on Imbalance Medical Data. IEEE Access **9**, 109960–109975 (2021)
19. Kraiem, M. S., Sánchez-Hernández F., Moreno-García M. N.: Selecting the suitable resampling strategy for imbalanced data classification regarding dataset properties. an approach based on association models. Applied Sciences **11**(18), 8546 (2021)
20. Lee, J. S., Hsiang, J.: Patent Claim Generation by Fine-Tuning OpenAI GPT-2. World Patent Information **62**, (2020)
21. Lee, P. H.: Resampling methods improve the predictive power of modeling in class-imbalanced datasets. International Journal of Environmental Research and Public Health **11**(9), 9776–9789.
22. Marqués, A. I., García, V., Salvador Sánchez, J.: On the suitability of resampling techniques for the class imbalance problem for credit scoring. Journal of the Operational Research Society **64**(7), 1060–1070 (2013)
23. Martins, R., Gomes, M., Almeida, J. J., Novais, P., Henriques, P.: Hate speech classification in social media using emotional analysis. Brazilian Conference on Intelligent Systems 2018, 61–66
24. Pereira-Kohatsu, J. C., Quijano-Sánchez, L., Liberatore, F., Camacho-Collados, M.: Detecting and Monitoring Hate Speech in Twitter. Sensors **19**(21), 4654–4691 (2019)
25. Phung, N. M., Mimura, M.: Evaluation of a cGAN Model and Random Seed Oversampling on Imbalanced JavaScript Datasets. Journal of Information Processing **30**, 591–600 (2022)
26. Sallam, M.: The utility of ChatGPT as an example of large language models in healthcare education, research and practice: Systematic review on the future perspectives and potential limitations. *medRxiv preprint*, (2023)
27. Shaikh, S., Daudpota, S. M., Imran, A. I., Kastrati, Z.: Towards Improved Classification Accuracy on Highly Imbalanced Text Dataset Using Deep Neural Language Models. Applied Sciences **11**(2), 869 (2021)
28. Shelke, M. S., Deshmukh, P. R., Shandilya, V. K.: A review on imbalanced data handling using undersampling and oversampling technique. International Journal of Recent Trends in Engineering Research **3**(4), 444–449 (2017)
29. Siegel, A.: Online Hate Speech. Social media and democracy: The state of the field, prospects for reform, 56–88 (2020)
30. Sun, Y., Wong, A., Kamel, M.: Classification of Imbalanced Data: A Review. International Journal of Pattern Recognition and Artificial Intelligence **23**(4), 687–719 (2009)
31. Ullmann, S., Tomalin, M.: Quarantining online hate speech: technical and ethical perspectives. Ethics and Information Technology **22**, 69–80 (2020)
32. Usuga-Cadavid, J. P., Grabot, B., Lamouri, S., Fortin, A.: Artificial Data Generation with Language Models for Imbalanced Classification in Maintenance. International Workshop on Service Orientation in Holonic and Multi-Agent Manufacturing 2021, 57–68
33. Wijaya, I. D., Putrada, A. G., Oktaria, D.: Overcoming Data Imbalance Problems in Sexual Harassment Classification with SMOTE. International Journal on Information and Communication Technology **8**(1), 20–29 (2022)

A Algorithm-specific data

In order to get a more precise understanding of what happens for each combination of text representation and classification algorithm, we report the F_1 results obtained in each of the 12 cases considered. Space limitation prevented us from reporting the precision and recall results, but they are available upon request. The results show that for all but one combination models resampled with LLM augmentation fared better than all the other models in the 4-class case. The results are mixed in the 5-class case with XGBoost, decision trees, and Naïve Bayes sometimes benefitting from LLM-Augmented resampling. Overall, the best method in the 4-class classification is XGBoost with TF-IDF and LLM resampling, while in the other two tasks it is SVM with BoW and no resampling.

A.1 Extreme Gradient Boosting

Table 5. XGBoost-Bag of Words F_1 scores

Resampling Method	Binary	5-class	4-class
None (NO)	0.656	0.294	0.338
Random Undersampling (RU)	**0.666**	0.238	0.333
SMOTE-Tomek (ST)	0.660	0.286	0.372
Random Oversampling (RO)	0.663	**0.319**	0.386
AdaSyn (AD)	0.629	0.251	0.342
LLM-Augmented (LA)	0.660	0.300	**0.432**

Table 6. XGBoost-TF-IDF F_1 scores

Resampling Method	Binary	5-class	4-class
None (NO)	0.630	0.293	0.320
Random Undersampling (RU)	0.629	0.296	0.306
SMOTE-Tomek (ST)	0.642	0.273	0.327
Random Oversampling (RO)	0.627	0.274	0.318
AdaSyn (AD)	**0.644**	0.254	0.307
LLM-Augmented (LA)	0.636	**0.313**	**0.520**

Table 7. XGBoost-Frequency F_1 scores

Resampling Method	Binary	5-class	4-class
None (NO)	0.635	0.272	0.342
Random Undersampling (RU)	0.630	0.269	0.309
SMOTE-Tomek (ST)	**0.644**	0.287	0.357
Random Oversampling (RO)	0.636	0.272	0.363
AdaSyn (AD)	0.630	0.255	0.309
LLM-Augmented (LA)	0.642	**0.329**	**0.460**

A.2 Support Vector Machine

Table 8. SVM-Bag of Words F_1 scores

Resampling Method	Binary	5-class	4-class
None (NO)	**0.699**	**0.363**	0.362
Random Undersampling (RU)	0.674	0.249	0.356
SMOTE-Tomek (ST)	0.665	0.281	0.400
Random Oversampling (RO)	0.682	0.298	0.380
AdaSyn (AD)	0.646	0.302	0.338
LLM-Augmented (LA)	0.673	0.332	**0.413**

Table 9. SVM-TF-IDF F_1 scores

Resampling Method	Binary	5-class	4-class
None (NO)	**0.679**	**0.340**	0.332
Random Undersampling (RU)	0.649	0.254	0.379
SMOTE-Tomek (ST)	0.667	0.287	0.377
Random Oversampling (RO)	0.659	0.329	0.378
AdaSyn (AD)	0.648	0.310	0.314
LLM-Augmented (LA)	0.658	0.295	**0.403**

Table 10. SVM-Frequency F_1 scores

Resampling Method	Binary	5-class	4-class
None (NO)	0.663	0.307	0.395
Random Undersampling (RU)	**0.668**	0.236	0.327
SMOTE-Tomek (ST)	0.654	0.298	0.340
Random Oversampling (RO)	0.656	**0.316**	0.370
AdaSyn (AD)	0.653	0.289	0.298
LLM-Augmented (LA)	0.634	0.312	**0.402**

A.3 Decision Tree

Table 11. Decision Tree-Bag of Words F_1 scores

Resampling Method	Binary	5-class	4-class
None (NO)	0.619	0.283	0.392
Random Undersampling (RU)	**0.650**	0.257	0.343
SMOTE-Tomek (ST)	0.638	0.267	0.384
Random Oversampling (RO)	0.635	0.292	0.384
AdaSyn (AD)	0.602	0.234	0.323
LLM-Augmented (LA)	0.577	**0.313**	**0.453**

Table 12. Decision Tree-TF-IDF F_1 scores

Resampling Method	Binary	5-class	4-class
None (NO)	0.603	0.250	0.316
Random Undersampling (RU)	**0.613**	0.274	0.353
SMOTE-Tomek (ST)	0.590	0.246	0.334
Random Oversampling (RO)	0.587	0.254	0.305
AdaSyn (AD)	0.598	0.240	0.336
LLM-Augmented (LA)	0.599	**0.312**	**0.436**

Table 13. Decision Tree-Frequency F_1 scores

Resampling Method	Binary	5-class	4-class
None (NO)	0.601	0.251	0.318
Random Undersampling (RU)	**0.622**	0.283	0.301
SMOTE-Tomek (ST)	0.589	0.243	0.328
Random Oversampling (RO)	0.596	0.256	0.337
AdaSyn (AD)	0.602	0.244	0.302
LLM-Augmented (LA)	0.581	**0.325**	**0.478**

A.4 Naïve Bayes

Table 14. NB-Bag of Words F_1 scores

Resampling Method	Binary	5-class	4-class
None (NO)	**0.625**	0.287	0.352
Random Undersampling (RU)	0.594	0.212	0.312
SMOTE-Tomek (ST)	0.539	0.253	0.375
Random Oversampling (RO)	0.608	0.282	0.353
AdaSyn (AD)	0.538	0.287	**0.418**
LLM-Augmented (LA)	0.565	**0.303**	0.335

Table 15. NB-TF-IDF F_1 scores

Resampling Method	Binary	5-class	4-class
None (NO)	0.559	0.253	0.347
Random Undersampling (RU)	0.575	0.255	0.343
SMOTE-Tomek (ST)	0.535	0.254	0.341
Random Oversampling (RO)	0.540	0.255	0.343
AdaSyn (AD)	0.528	0.255	0.346
LLM-Augmented (LA)	**0.578**	**0.267**	**0.356**

Table 16. NB-Frequency F_1 scores

Resampling Method	Binary	5-class	4-class
None (NO)	0.558	0.243	0.336
Random Undersampling (RU)	**0.584**	0.250	0.297
SMOTE-Tomek (ST)	0.535	0.250	0.336
Random Oversampling (RO)	0.548	**0.255**	0.334
AdaSyn (AD)	0.528	0.249	0.336
LLM-Augmented (LA)	0.576	0.245	**0.342**