Responsible Data Science (DS-GA 1017) Final Project
Auditing A BERT-Based Toxicity Detector[1]

Johnny Ma (jlm10003), Akshit Gandhi (amg9556)

May 2022

[1]Data found here: https://www.kaggle.com/competitions/jigsaw-unintended-bias-in-toxicity-classification/overview. Model found here: https://www.kaggle.com/code/yuval6967/toxic-bert-plain-vanila/notebook

Background

As internet access becomes more widespread, digital forums such as Twitter and online communities such as Facebook are exposed to more cultures, personalities, and ultimately, more forms of racism, hatred, or toxicity. To promote the "civil exchange of ideas" and make the internet "stronger and safer for everyone", the Jigsaw team at Google has developed several toxicity detection models, which are used by companies to moderate their communities by automatically detecting and removing toxic comments.

While these systems can help these communities thrive, these state-of-the-art toxicity models are trained on historically data labeled by crowd-workers, and are therefore susceptible to human bias and worse performance on historically under-represented groups. If we see disparate performance on different identity groups, implementing a toxicity detector as a forum moderator can amplify pre-existing biases, causing more emergent bias against marginalized groups. In addition, new toxic phrases are constantly appearing, and several forms of toxicity are subtle and may be co-opted by nefarious agents at any time (the letter Z, OK symbol, etc.). Therefore, it is crucial to audit toxicity models in order to prevent automated moderation (or failure of moderation) and disparate treatment of identity groups in online communities, especially those protected by free-speech or other human rights laws.

For this project, we will focus on a simple BERT based model fine-tuned on the Jigsaw training data. Auditing this transformer based model will give us insight in how large language models learn and potentially exhibit bias when trained on annotated toxicity data, and annotated text in general. As many currently implemented online moderation systems are likely using or will likely go on to adopt this state of the art architecture, this analysis can unravel how these systems, while well-meaning, may be censoring and discriminating against specific sub-groups on the margin.

Input and Output

In 2017, the Civil Comments platform released the available online comments on the platform publically for research summing to nearly 2 million comments. Jigsaw sponsored this effort and extended annotation of this data by human raters for various toxic conversational attributes.

In the dataset, the comment_text column contains the text of the individual comment. Each comment in Train has a toxicity label (target), and the model predicts the target toxicity for the Test data. To produce the "target" toxicity label, annotators were asked to rate the toxicity of each comment as one of the following categories: Very Toxic, Toxic, Hard to Say, Not Toxic. Each comment was shown to up to 10 annotators. The data also has several additional toxicity subtype attributes which are as follows: severe_toxicity, obscene, threat, insult, identity_attack, and sexual_explicit. From preliminary analysis, we found that the vast majority of included comments are non-toxic, a scenario likely mirroring real life internet communities.

To limit our analysis to the general problem of toxicity detection, we will focus on predicting only the target variable of "target" which measures the inter-annotator toxicity of a comment. This target variable is transformed into a binary "toxic" vs. "not toxic" via a threshold of 0.5, which is provided by the competition.[1] The distribution of toxic vs. non-toxic comments in the test set can be seen in Figure 1.

Additionally, a subset of comments has been labeled with a variety of identity attributes, representing the identities that are mentioned in the comment. The annotators were asked to select all identities that were mentioned in the comment to obtain the identity labels. The columns corresponding to identity attributes that had over 500 comments in the training set are listed as follows: male, female, homosexual_gay_or_lesbian, christian, jewish, muslim, black, white, psychiatric_or_mental_illness. We use these identity labels to examine the disparate performance of the model on various identity sub-groups. The majority of these labels are exclusive, outside of the largest white/black and male/female mirrored categories, indicating that comments

[1] This threshold was a point of contention for several competitors that saw improvements when NOT using this same threshold for training. To keep things consistent with the implementation we are using, we will adopt this binary cut-off.

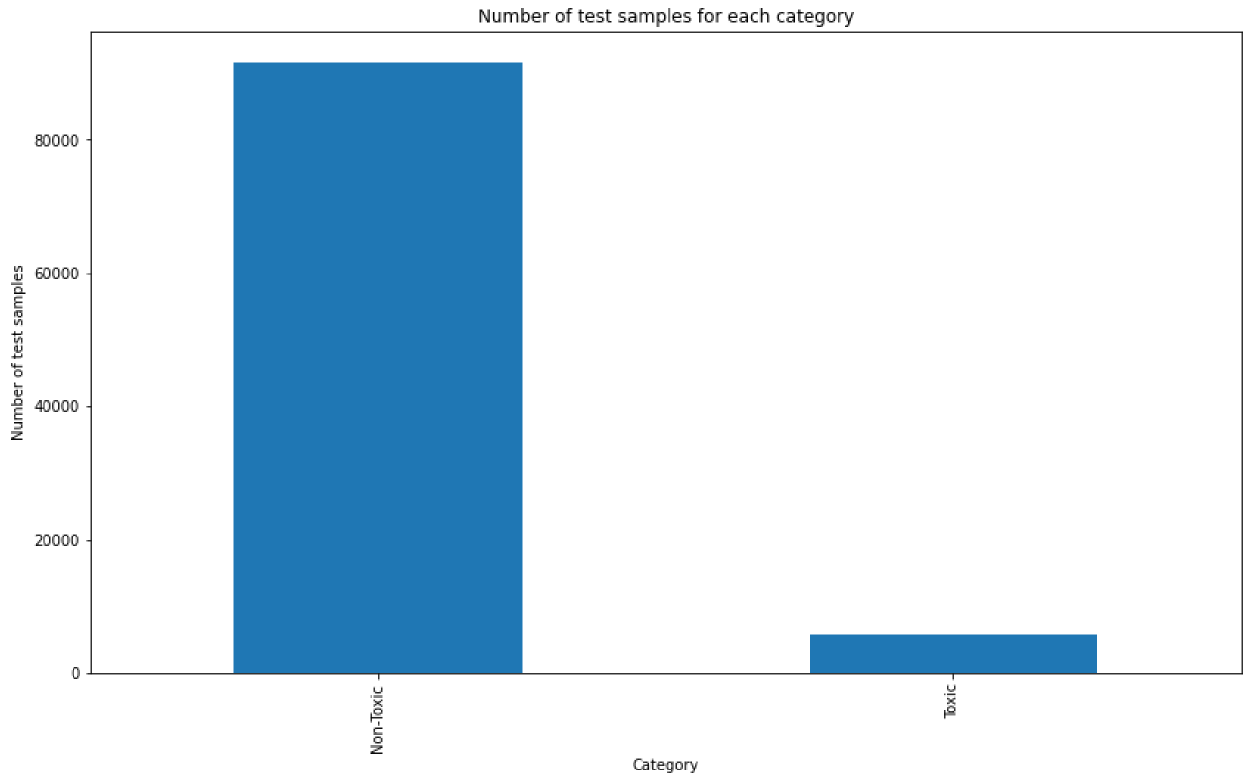


Figure 1: Distribution of toxic vs. non-toxic comments in the test set. Very un-balanced, mirroring real online communities.

are likely to focus on one specific identity group. This can be seen in Figure 2.

The following features have 1.4 million missing values in the training dataset: black, christian, female, homosexual_gay_or_lesbian, jewish, male, muslim, psychiatric_or_mental_illness, accounting for approximately 72% of the data not including these identity labels. All other features contain no missing values. All features are of float datatype except the comment_text feature which contains only string values.

We mostly restrict our analysis to within subgroups, so while we train the toxicity detector on a large number of comments, we are only able to evaluate the model's performance on the relatively small subset of comments that are tagged by annotators. Additionally, we focus on the aforementioned identity groups as they contain over 500 comments and were explicitly pointed out as key to the evaluation for the competition itself.

Finally, we will be using the "test_public_expanded.csv" released by Kaggle after the competition had concluded. This is the same public facing test set used to evaluate competitors' performance and produce the leader boards, but obviously did not contain the actual target variable prediction (binary prediction derived from the same 0.5 threshold) and did not reveal the identity groups. The missing values for the identity groups are roughly equivalent across classes (about 72% missing), with both datasets having an equal distribution of missings over all subgroups. We can therefore consider the train and test datasets to be drawn from the same data generating process.

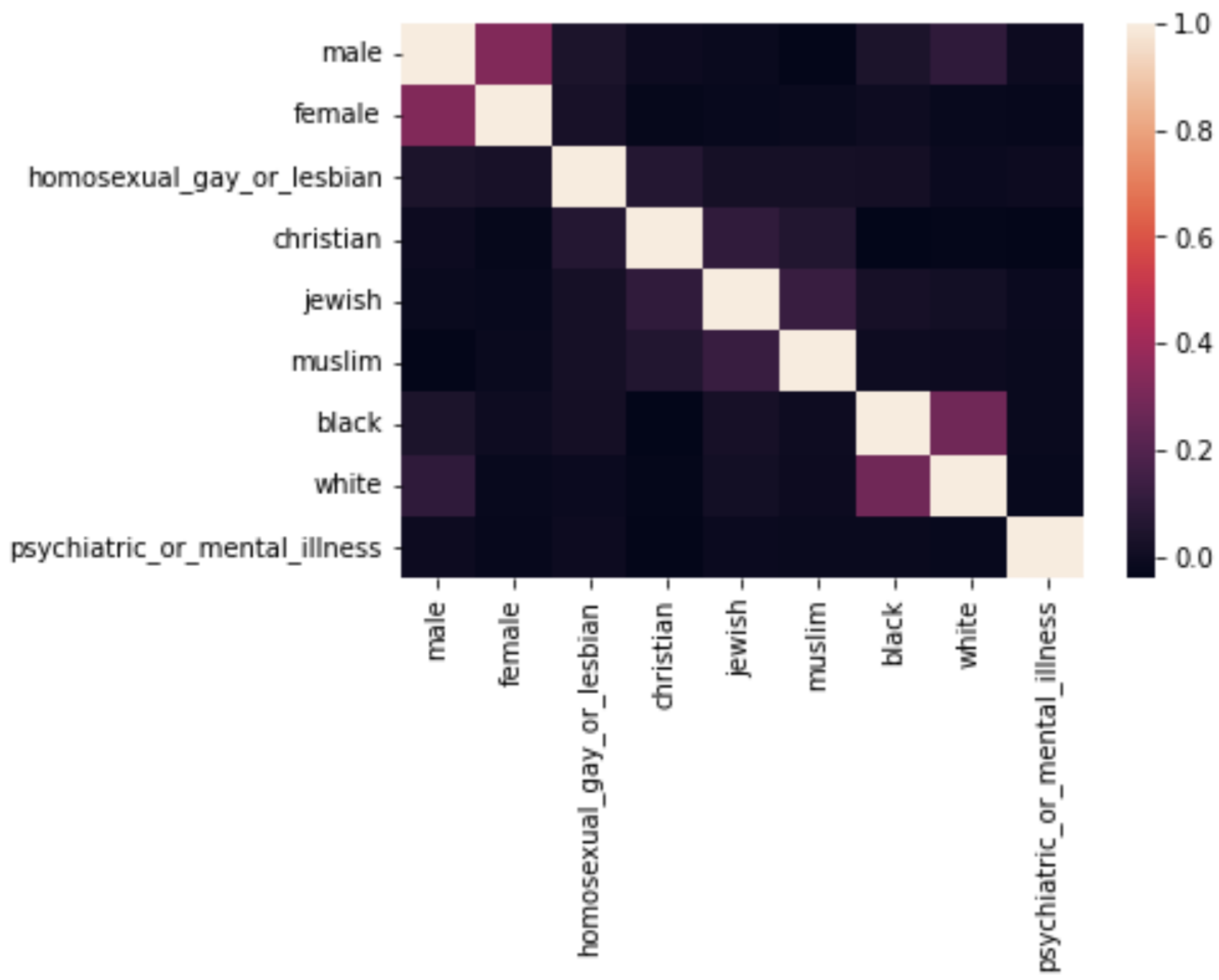


Figure 2: Heatmap of correlation between identity subgroups within comments. There is some overlap is related subgroups.

Implementation and Validation

Data Cleaning and Pre-Processing

The toxicity training and fully public test datasets released by the competition are formatted identically; they contain the aforementioned features and target variable of ”target” that measures the inter-annotator toxicity of a given comment. We transform this continuous measure of toxicity into a binary ”toxic” and ”not-toxic” variable using the threshold cut-off of 0.5 as provided by the competition and used in several of the popular implementations (including our own). Note that this produces a highly skewed dataset, where most comments are considered non-toxic.

For the input to the model, we exclusively use the ”comment_text” which contains a string type instance of a comment on the Civil Commons forum. We use the BERT Tokenizer provided by HuggingFace [2] that is built on the WordPiece tokenizer. We chose a sequence length of 220 to tokenize the comments, with default settings (lowercase words, padding, etc.). A simple visualization of the word counts on the comments, as seen in Figure 3, suggests that there are few comments that would exceed the sequence length, but a majority (if

[2]https://github.com/huggingface/transformers/blob/v4.18.0/src/transformers/models/bert/tokenization_bert.py#L117

not all) that would be padded to the full sequence length. Notably, our implementation separates the padded ”0”s using a special token ([SEP]), and begins each comment with a special token ([CLS]) that denotes a classification task. This same pre-processing is applied equally to the train and test dataset.

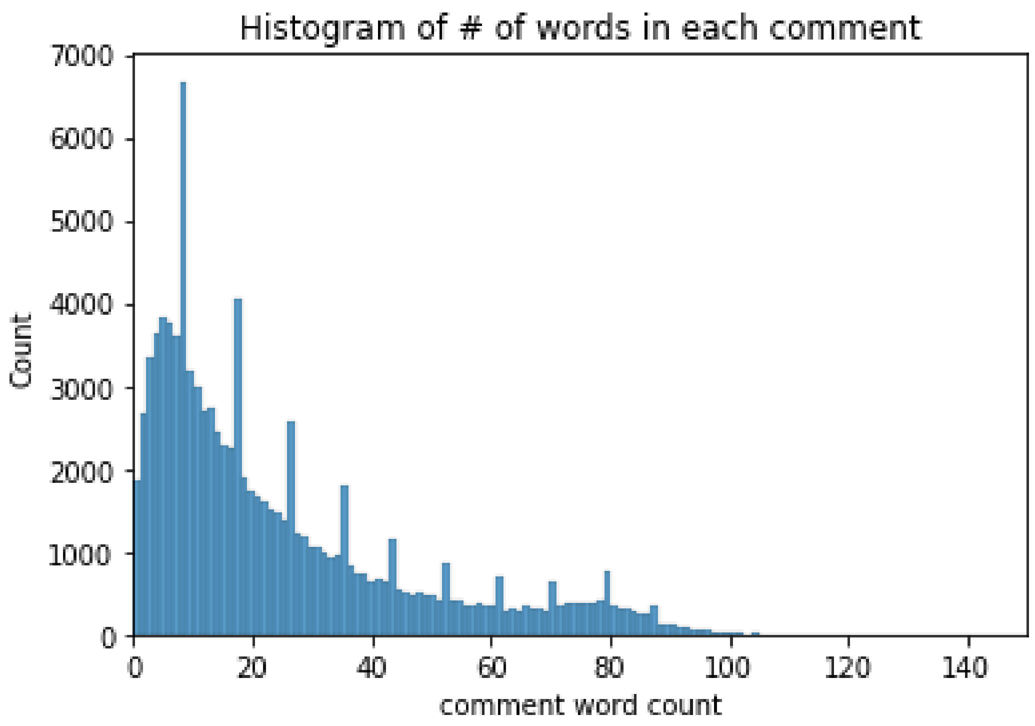


Figure 3: Distribution of comment word counts using CountVectorizer. Most comments are short.

Implementation Details

The implementation we are auditing is a typical data pipeline that trains a transformer based model namely a pre-trained BERT model. The implementation begins by importing a BERT model checkpoint pre-trained by Google[3] with 12 layers and a hidden state embedding size of 768, trained originally on the Wikipedia and BooksCorpus [4]. This is commonly refered to as ”BERT BASE” and has seen wide success and adoption within the Natural Language Processing community.

This pre-trained model is then fine-tuned on the training dataset (a subset of around 1 million points, as this training dataset is divided into an extra validation set) for the downstream task of toxicity detection, specifically binary classification of comments to a toxicity label. We apply the Bert Tokenizer with special padding to the input text. We use a learning rate of 2e-5, a batch size of 32, and binary cross entropy loss over 1 epoch of the training data, which exposes the pre-trained model to almost 2 million labeled comments. The training uses the apex package for training and a scaled loss, which seems to empirically perform the best with the specific hyperparameters (as indicated by the comments section of the linked notebook). The total training time is around 6 hours, which is difficult to benchmark as the authors do not release their hardware specification. In total, this training process is very typical (hence the notebook title ”BERT Plain Vanilla”) for supervised learning tasks with a binary target label, and is easily reproducible given the full set of hyperparameters.

[3]Training details and model download available at `https://huggingface.co/google/bert_uncased_L-12_H-768_A-12`

[4]See the original BERT paper `https://arxiv.org/pdf/1810.04805.pdf` for additional training details

Finally, inference is performed in the same manner as the training. By passing batches of tokenized comments, the model returns logit probabilities, which are transformed into a continuous measure of toxicity and then binarized as described above.

Validation

As this model came from a Kaggle competition, there is typically a reliance on validation by placement on the leader board based on unseen test set performance. However, the author of the notebook did perform a validation by splitting the original training data in about one half (though they did this mostly to reduce the training set size to ensure the model can train within the competition's time limit). Therefore, we have some preliminary results on the model performance prior to the release of the test set. We use the AUC score in order to focus on classification of positive instances, or toxic comments, which is the most relevant metric for online moderation, the key use case of toxicity detection. This metric also helps account for the incredibly skewed data distribution of toxicity, which is to be expected for this task.

The model achieves an AUC score of 0.930 on the validation dataset, which is roughly in line with the eventual score of 0.947 on the test set. While there is no agreed upon bound for how well a toxicity detector must perform to be considered accurate and production worthy, the model performs far above the typical 0.5 baseline in a AUROC. We will examine the few examples that the automated system shows the failure state on in order to get a better understanding of the consequences of implementing this generally successful model.

Outcomes

We will be using metrics based on the AUC scores as our primary method of evaluation. As this toxicity detector pipeline is intended to be used as an automated online moderation system, flagging potentially toxic comments for removal (either with human approval or automatically, or typically a combination of the two), we are most interested in the behavior of the model surrounding positive instances. By combining the **False Positive Rate**, or $\frac{TP}{TP+FN}$ with the **False Positive Rate**, or $\frac{FP}{FP+TN}$ into a single metric, we use calculate the AUROC using different thresholds to plot both the TPR and FPR on a single graph, with the AUC representing the probability that the classifier will rank a positive outcome higher than a random negative example. This should capture the nature of how well an automated moderation system is doing; both at correctly detecting toxic comments and correctly **not** classifying a negative instance as positive.

Conveniently, the competition provides a bevy of evaluation metrics built off of AUROC[5], specifically for measuring performance on identity groups. We re-created this metrics using the code provided in the Kaggle competition kernel. We will cover these metrics, the intention behind them, and their values over the test set in the following sections.

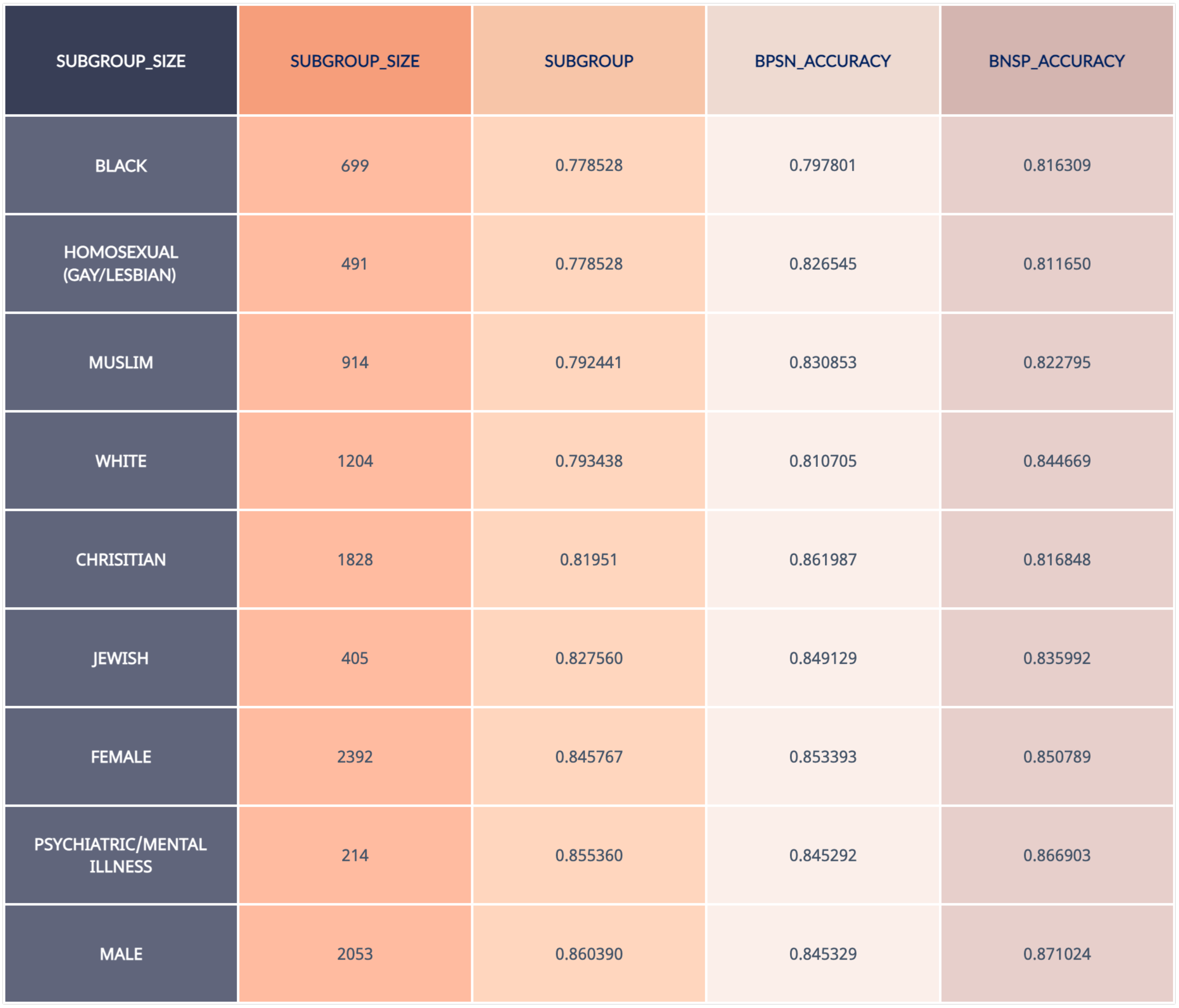
Fairness

There are three metrics that we examine as measures of subgroup performance:

1. **Subgroup AUC:** Calculate model performance on data instances from specific subgroups.
2. **BPSN (Background Negative, Subgroup Positive)**: Calculate model performance on a restricted subset of the data that contains non-toxic examples that are labeled with the given identity (background negative) and toxic examples that are not (subgroup positive). Higher values means the model does not confuse non-toxic examples that mention the identity with toxic examples that do not, due to the nature of AUC.

[5]See `https://www.kaggle.com/competitions/jigsaw-unintended-bias-in-toxicity-classification/overview/evaluation`

3. **BNSP (Background Negative, Subgroup Positive) AUC**: Calculate model performance on examples that are toxic and mention the identity. and non-toxic examples that do not. A higher value means that the model does not confuse toxic examples that mention the identity with non-toxic examples that do not.



SUBGROUP_SIZE	SUBGROUP_SIZE	SUBGROUP	BPSN_ACCURACY	BNSP_ACCURACY
BLACK	699	0.778528	0.797801	0.816309
HOMOSEXUAL (GAY/LESBIAN)	491	0.778528	0.826545	0.811650
MUSLIM	914	0.792441	0.830853	0.822795
WHITE	1204	0.793438	0.810705	0.844669
CHRISITIAN	1828	0.81951	0.861987	0.816848
JEWISH	405	0.827560	0.849129	0.835992
FEMALE	2392	0.845767	0.853393	0.850789
PSYCHIATRIC/MENTAL ILLNESS	214	0.855360	0.845292	0.866903
MALE	2053	0.860390	0.845329	0.871024

As seen in the table, the subgroup accuracy would be lower for groups, namely Black, Homosexual, Muslim, White, Christian, and Jewish. On the contrary, the subgroups like Female, Mentally Ill, and Male have higher subgroup accuracy. For instance, it is easy to determine the toxicity level of the comment if the user belongs to the male subgroup. There are few significant changes in the patterns of performance across groups when looking at the various AUC metrics.

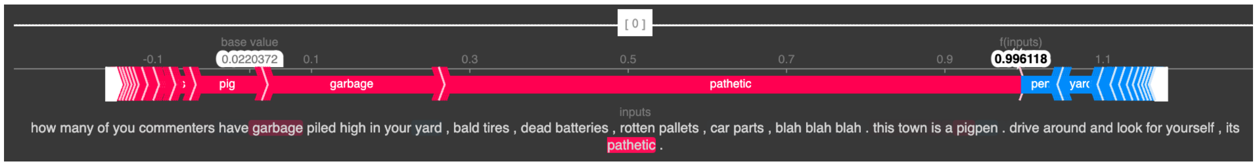
Moreover, the subgroup size for these subgroups somewhat correlates with the performance of the model for that subgroup. The model learns hidden representations more precisely when there are a sufficient number of samples to learn. Specifically, the model learns to detect toxicity in a comment related to mental illness easily than others.

In addition, we see lower performance on subgroups that may have less explicitly identifying or more subtle terms associated with toxicity. For example, there are currently in the English language a large collection of ways to refer to LGBTQ+ members that may or may not be toxic, whereas the vocabulary for toxicity against members who are mentally ill may be more limited and specific. However, it is difficult to validate these claims without examining the individual weights of the aforementioned terms, which we will look at using SHAP analysis.

BERT SHAP Analysis: Don't Say Gay?

In addition to the above fairness metrics, we examine the model directly using SHAP, or Shapley Additive explanations. This approach provides a local explanation of any prediction using a game theoretic approach to assigning credit for each tokenized word towards the eventual prediction of the model. As the BERT implementation is a straight-forward text-in prediction-out type of model, we can apply SHAP analysis to the toxicity detection model.

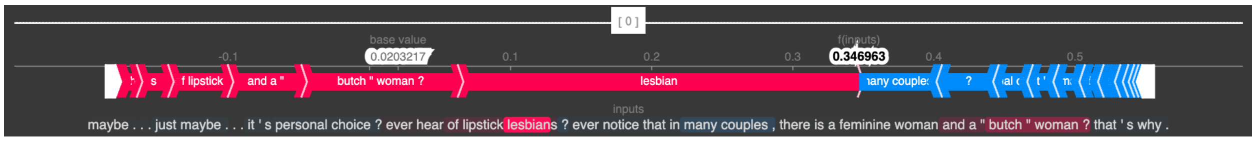
We specifically select three cases to examine; a comment that is successfully detected as toxic and two failure cases of the model, where a toxic comment is missed by the system and where a non-toxic comment is flagged.



how many of you commenters have garbage piled high in your yard , bald tires , dead batteries , rotten pallets , car parts , blah blah blah . this town is a pigpen . drive around and look for yourself , its pathetic .

Figure 4: Example of a comment correctly labeled as toxic.

In Figure 4, we see a toxic comment that is correctly captured by the BERT system. The tokens that contribute to the toxic classification are in red and ordered by their magnitude of the contribution according to the local models of SHAP, and vice versa for blue tokens. We see behavior that we expect from a negative sentiment classifier; that is, highly negative terms such as "pathetic", "garbage", and "pig" are regarded as contributing greatly to the toxic prediction, whereas mentions of "pens" and "yards" fight against this ultimately correct classification.The negative terms overwhelm the positive ones, as even the positive terms are only possibly positive due to their association with pet-ownership or home-ownership, and given the context of the sentence would actually have a negative connotation, as the speaker is comparing commentators to animals in a very toxic and belittling manner. While there are no real surprises here, other than the inability of the transformer model to fully display contextual understanding, we look at two failure cases to better understand cases where the model can produce undesired behavior, especially around minority groups.



maybe . . . just maybe . . . it ' s personal choice ? ever hear of lipstick lesbians ? ever notice that in many couples , there is a feminine woman and a " butch " woman ? that ' s why .

Figure 5: Example of a missed toxic comment, focusing on LGBTQ+ groups.

In Figure 5, we see a toxic comment that is incorrectly deemed non-toxic by BERT. The mere mention of the word "lesbian" is seen as the most influential feature towards a toxic classification, which already is quite problematic. Simply stating the word "lesbian" should not be an indicator of toxicity unless the context demonstrates a hateful commentator. In this case, the other typically ignorant terms such as "butch women" are correctly labeled as non-toxic. However, we also see that referencing "couples" is seen as non-toxic, which generally makes sense in most contexts and arguably is a positive mention here, as at least the commentator has the decency to recognize lesbians as being valid couples. Without additional context, it is still relatively difficult to understand the intentions of the commentator, though the model should capture this as a toxic comment rather than as somewhat toxic (0.347 ¡ 0.5).

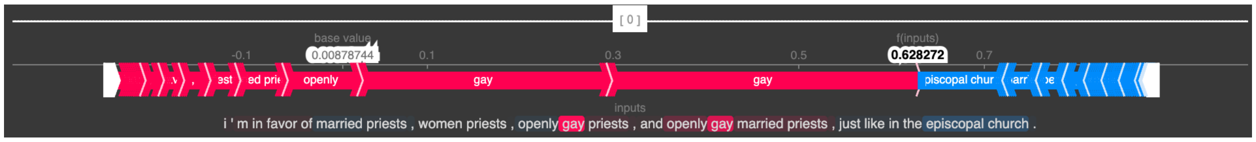


Figure 6: Example incorrectly labeled as toxic, focusing on LGBTQ+ groups. Positive context mentions of protected groups should not be considered toxic.

Finally, in Figure 6 we see the most egregious error of the model as it incorrectly labels a very positive affirmation of LGBTQ+ acceptance as a toxic comment. The top features, simply the words ”gay” and ”openly”, drive the model to go above the 0.629 threshold for toxicity alone. This seems to imply that the model views the terms as inherently toxic, likely due to the dataset it is trained on and historical, pre-existing societal bias against gay individuals and perhaps gay priests in particular. Given this example, we would likely see a greater censorship of any topics or forums discussing LGBTQ+ terms (as compared to say, religious discussion that are seen as largely neutral from ”church”), causing a great mistreatment of these protected groups that are now seen as ”toxic” when merely spelling out their sexual orientation. This could lead to these groups feeling unwelcome in internet forums and un-free to discuss their identities, which is contrary to the intentions of toxicity detectors.

Summary

Toxicity detection in online communities grows in importance as more diverse groups gain access to the internet and spend increasingly more time interacting with others over the web. As the volume of online messaging expands, there is more demand for fully automatic moderation, in order for the social media company to ensure a pleasant user experience and to avoid exposing the current generation of human moderators to continuous toxic content.

The Kaggle implementation is evidence that large transformer model based implementations, trained on expertly annotated data collected by one such social media company, can achieve great success as an automatic moderator as it flags the vast majority of toxic comments. The training dataset and evaluation metric of AUROC are well suited to developing a model for automatic community moderation. However, additional tagged text data with additional context can likely further improve the model, given slightly improved metrics when training for more epochs. In addition, the difference in performance over subgroups may be attributed to the lack of toxic comments targeted against these groups (though most tags are missing rather than 0, so there may be more identity based toxic comments in the dataset than are explicitly tagged as such). This can be addressed by tagging more identity labels and possibly including these tags into the model directly and optimizing to have similar performance over these subgroups, as opposed to the current method of evaluation through mean AUROC on subgroups.

Given the high AUROC scores from both internal validation and the test set evaluation, it is certainly helpful to have the toxicity detector in place to flag toxic comments, and the volume of messages the system is able to process quickly simply cannot be matched by human annotators. The differences in performance over the subgroups is not as severe as other applications, and is possibly due to the inherent varying difficult of the problem and a lack of data specific to each subgroup rather than any overall architecture failure.

Using SHAP to create local explanations for specific model failures, we do see some troubling signs of bias. For example, the model appears to weigh even positive mentions of sexual identity as highly toxic, which is certainly undesired behavior. However, as the model does not explicitly use any identity labels outside of the text itself, it is more difficult to equalize treatment of identity groups without knowing the full context of the comments. While setting every weight of sensitive words to 0 would theoretically help this problem, it would likely render the algorithm useless to detect actual toxic speech. As the trade-off between incorrectly flagging non-toxic comments and letting more toxic comments pass greatly favors being harsher with the tagging (as evidenced by a somewhat low threshold of 0.5), it is likely that most systems will ignore the false positives in favor of overall performance.

The impressive performance of the automatic toxicity detector on high quality test data is likely sufficient for large scale social media companies to adopt these language models to create online moderation tools. Given the poor performance over specific sensitive subgroups and a bleak picture of model failures at the local level, we do not recommend that such systems be given free reign to moderate online communities. A temporary solution as more training data is gathered and the models are improved can be to use the model's predictions with a human-in-the-loop. Very high probability toxicity (the majority of the toxic comments) can be automatically flagged, with cases more on the margin (around the 0.5 threshold) can be sent out for manual review. This would save the human moderator from being exposed to blatantly toxic comments and utilize their better understanding of nuance and context to accept or reject the model's decisions. Their work can be used to further improve the model over time, which will prove invaluable as the flood of internet activity continues to swell, with no end in sight.