Universidad Distrital Francisco José de Caldas

Facultad de Ingeniería

Jigsaw Unintented Biad in Toxicity Classification

Andrey Camilo Gonzalez Caceres
Hugo Mojica Angarita
Laura Paez Cifuentes

May 16, 2025

Abstraction

This paper presents the system analysis and design for the Kaggle competition "Jigsaw Unintended Bias in Toxicity Classification," which involves identifying potentially toxic words in certain conversations. We address the challenges of detecting seven types of toxicity, while reducing false positives and managing disagreements among annotators. We incorporate features such as IdentityAttackChecker and AnnotatorWeightCalculator to help us manage the chaos that can arise in our system, which in a given case would be certain words that could be considered toxic, but in a non-offensive context. We heavily leverage the Python ecosystem, as it allows us to focus on fairness metrics and system reliability with limited resources.

Introduction

Jigsaw Unintended Bias in Toxicity Classification is a competition proposed by Kaggle and Google. It consists of an identifier of toxic comments that may be posted online. One challenge we must address is the fact that in certain contexts, a word or phrase may be considered toxic that isn't actually offensive. This also depends on the identity. Specific functions will be defined for this purpose.

We approach this competition as a complex system, defining each of its different properties, taking into account the chaos the system can present and controlling it (such as words that may be offensive but aren't in that context). The system, like all systems, was represented as an input (the word or phrase to be rated), processing (the process of determining whether the word is toxic or not), and output (the final decision on whether it can be represented as toxic or not).

We found that our system can experience certain variations due to environmental factors that would severely affect it, for example, human subjectivity and ambiguity in certain data. For this competition, we will need to use a modular architecture that allows us to manage these factors.

We observe that the language and the different annotations can have chaotic behavior, and that likewise the smallest change in any of the annotations can have a significant impact on the final result. In order to control this systemic chaos, we will apply some strategies such as controlling false positives in the data and correctly managing the sensitivity of our system.

For data processing, we will use Python, which allows us to use extremely useful libraries such as NumPy, Pandas, and Scikit-learn, and because the Python ecosystem is perfect for this competition. This is done with the goal of ensuring that it is a system designed with attention to detail and can be improved over time and used with larger amounts of data, as all complex systems should be.

Methods and Materials

For the design of the system and application of the tools with which the data will be processed to determine the toxicity of online comments, we will use different tools that in this case are the most useful to us, we present them below:

Technology Selection:

For data processing, we will use the Python programming language, as it is the most widely used language in data science due to its flexibility, scalability, and ease of use. Additionally, we will be able to use widely used libraries in this field, such as NumPy, Pandas, Scikit-learn, and other specialized metrics tools like Subgroup AUC, BPSN, AUC and BNSP AUC that would be used to evaluate the fairness or justice of the model, especially identity comments(religion, gender, etc).

Figure 1: Scikit-learn, Pandas, NumPy, Python logos

NumPy: It would be used to solve low-level mathematical operations, especially vector and matrix manipulation. It is essential in the weight calculation phases, operations on feature vectors, and numerical sensitivity assessments.

Pandas: It would facilitate the loading, inspection, and transformation of the train.csv, test.csv, and individual annotation files. It would allow us to apply filters, group data by identities, and prepare balanced training sets.

Scikit-learn: It would be primarily used to implement linear regression models, such as logistic regression, which act as base models. These models allow for a clearer interpretation of the coefficients associated with each input variable (such as words or tokens) and are useful for evaluating the influence of differ ent features on toxicity prediction.

System Architecture:

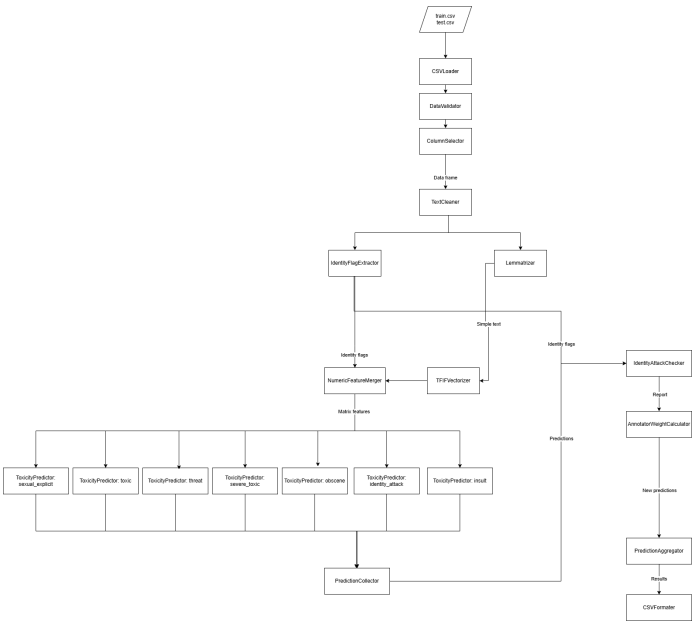

Figure 2: High-Level Arquitecture reference

Labels:

• **VSCLoader:** Has the responsibility for extraction. It extracts the data from the files.

• **DataValidator:** Has the responsibility to validate the data, that the key columns exist.

• **ColumnSelector:** Transforms the received data by re moving the columns that are not relevant.

• ColumnSelector: Transforms the received data by re moving the columns that are not relevant.

• TextCleaner:Transforms the received data by removing the text noise.

• IdentityFlagExtractor: Transforms the identities it de tects as a vector.

• Lemmatizer: Transforms the received data by a more simplified text.

• **TFIDVectorizer:** Transforms the input text to a vector.

• **NumericFeatureMerger:** Transforms the input vectors to a matrix, stored in a dataframe.

• **ToxicityPredictor(X):**runs an individual model for toxi city prediction.

• **PredictorCollector:**Transforms each individual predic tion, leaving a single dataframe.

• **IdentityAttackChecker:** Validates if there are false pos itives in the identities.

• **AnnotatorWeightCalculator:** Transforms the possible false positive to a better prediction.

• **PredictionAggregator:** validates if there are any possible inconsistencies in the final prediction.

• **CVSFormatter:** outputs in the format required by Kaggle.

System requirements:

Based on the above summary, the systemic requirements that are important in our design will be:

• Detect 7 distinct toxicity types

• Reduce false positive identity classifications

• Handle the disagreements between annotators, reducing the percentage of inconsistencies

• As for the evaluation metrics, it is important to prioritize
the fairness of the subgroups of identities, since the competition will restrict the dataset to only comments that men-

tion a subgroup of identities.

• Runtime limits, to have a proper performance it will be necessary to be able to handle batches of 1000 comments without exceeding 16 GB of RAM, by using lightweight models.

• For reliability you need to minimize data, for example by validating file integrity.

Relationship Mapping Graph

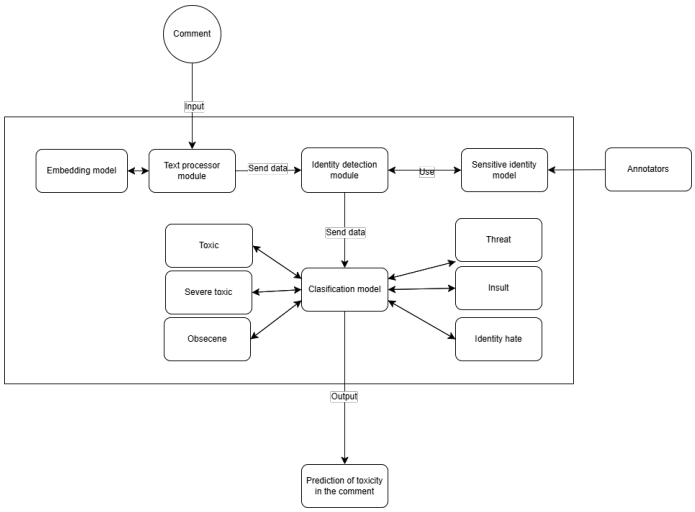

Figure 3: Relationship mapping graph

Chaos in the system:

From a theoretical perspective, the chaos in this system arises primarily from human subjectivity when performing toxicity annotations. The fact that the target value is derived from a fraction of annotators who consider a comment toxic introduces heightened sensitivity to small variations in indi vidual perceptions. Ambiguous, ironic, or sarcastic comments can be interpreted in opposite ways by different annotators, generating a chaotic dynamic that directly impacts the trained model.

Chaos Management Model

To model this sensitivity and inherent chaos in the system, we have considered the use of two specific classes that encapsulate the main concepts for handling some situations that may arise:

• **IdentityAttackChecker:** This class aims to identify subtle or implicit patterns of attacks against identity groups, considering the high variability in how these are perceived. It focuses on identifying cases in which language can have a negative impact, although not always explicitly. This helps capture signals that, although weak, can be amplified in the presence of bias.

• **AnnotatorWeightCalculator:** Since not all annotators contribute equally to the evaluation, this class focuses on modeling the influence (or weight) of each one on the final result. Factors such as their tendency to classify more comments as toxic or their bias toward certain iden tity groups can be quantified and used to adjust the final target value.

Conclusion

Through a systemic analysis of the Jigsaw Unintended Bias in Toxicity Classification competition, we identified the main technical, ethical, and social challenges involved in designing a toxic comment classification model. The subjectivity of human annotations and the presence of identity-related biases prompted us to present a structured and context-sensitive approach. We proposed a systems-based architecture that allows for proper data preparation, the application of machine learning techniques, and the ability to manage the system's chaotic behavior.

The proposed implementation is supported by tools from the Python ecosystem and specialized fairness metrics, used to maintain system balance and ensure it is not affected by certain contexts. We can also observe that the objective of our competition is the identification of seven types of toxicity, but it also has certain minimum requirements for full processing.

Bibliography

[1] *¿Que es la regresion lineal?* AWS Amazon. URL: https://aws.amazon.com/es/what-is/linear-regression/ (visited on 05/10/2025).

[2] *Jigsaw Unintended Bias in Toxicity Classification*. Kaggle. URL: https://www.kaggle.com/competitions/jigsaw-unintended-bias-in-toxicity-classification (visited on 04/16/2024).

[3] *LAS TEORÍAS DEL CAOS Y LOS SISTEMAS COMPLEJOS*. Universidad Autónoma de Madrid. URL: https://repositorio.uam.es/bitstream/handle/10486/684812/EM_7_2.pdf (visited on 04/10/2024).

[4] C. Sierra. *ud-public/courses/advanced-programming/project/Paper Guidelines.pdf at main · EngAndres/ud-public*. GitHub. URL: https://github.com/EngAndres/ud-public/blob/main/courses/advanced-programming/project/Paper_Guidelines.pdf (visited on 06/15/2025).

[1] [3] [4] [2]