GUS-Net: Social Bias Classification in Text with Generalizations, Unfairness, and Stereotypes

Maximus Powers* Umang Mavani*† Harshitha Reddy Jonala*† Ansh Tiwari*†
Hua Wei†

Abstract

GUS-Net is a fine-tuned BERT model designed to improve social bias detection in text by identifying biases across three semantic categories: generalizations, unfairness, and stereotypes. Unlike previous named-entity recognition models that classify tokens with a single broad label, GUS-Net takes a multi-label token classification approach. This allows for the detection of overlapping and nested biases at the token level. A synthetic dataset was developed to train the model, to offer balanced and robust coverage of nuances in varied domains. GUS-Net demonstrated strong performance, particularly in detecting stereotypes, offering a more detailed and precise analysis of biased language. By capturing the structural complexity of social biases, GUS-Net represents a significant step forward in natural language processing for bias detection.

1 Introduction

The detection of bias in natural language processing (NLP) [2] is an important task, particularly with the increasing use of large language models (LLMs) [13] in domains like education [5] and business [11]. Bias can significantly influence public perception and decision-making, often subtly reinforcing stereotypes or propagating discriminatory practices. While explicit bias which refers to clearly expressed prejudice or favoritism, is easy to define, implicit bias involves more subtle and often unconscious associations or attitudes. Therefore, identifying and mitigating implicit bias in the text is challenging: what is perceived as biased can vary greatly depending on the context including the perspectives of viewers and speakers. For example, consider the phrase "hard-working immigrants". To some, this phrase may appear positive, acknowledging the effort and diligence of immigrants. However, from another perspective, it might be perceived as implicitly biased, suggesting that immigrants are expected to work harder than others to be valued or accepted. This subtle implication can be seen as reinforcing a stereotype that separates immigrants from native citizens, placing an undue burden of proof on their worthiness. This subjectivity underlines the complexity of implicit bias detection, making it a critical area of research within NLP [3, 10, 7].

While the implicit nature of bias can manifest in subtle forms, such as the choice of words, framing of narratives, or the omission of certain viewpoints, traditional approaches to bias detection have typically relied on human annotators to label datasets [7, 8]. While this method has been instrumental in creating foundational resources, human annotators may unconsciously bring their biases into the annotation process, and it is often challenging for individuals to step outside their own ideological frameworks, whether political, cultural, or otherwise. This limitation can result in a narrow, one-dimensional understanding of bias, particularly when annotators struggle to consider perspectives divergent from their own.

Moreover, existing datasets for bias detection are often limited in scope, both in terms of the types of biases they cover and the perspectives they incorporate. For example, the Dbias model [12] utilized the MBIC dataset, which contained only 1,700 sentences. This dataset's small size and narrow focus limited the model's ability to generalize across different domains and types of bias. While the NBias framework [7] expanded Dbias by incorporating the entity "BIAS" in named entity recognition (NER) tasks, it still primarily focused on explicit biases, missing the structural elements of implicit bias like generalizations and stereotypes. Despite studies of robust annotations conducted by trained experts [9], all the previous bias datasets rely on human annotators, which means that datasets often lack the diversity of viewpoints necessary to capture the implicit bias.

In response to these challenges, this paper innovates the process of bias detection by utilizing generative AI and automated agents to build an optimal dataset. Using the synthetic data generated by these

*Ethical Spectacle Research.
†Arizona State University.

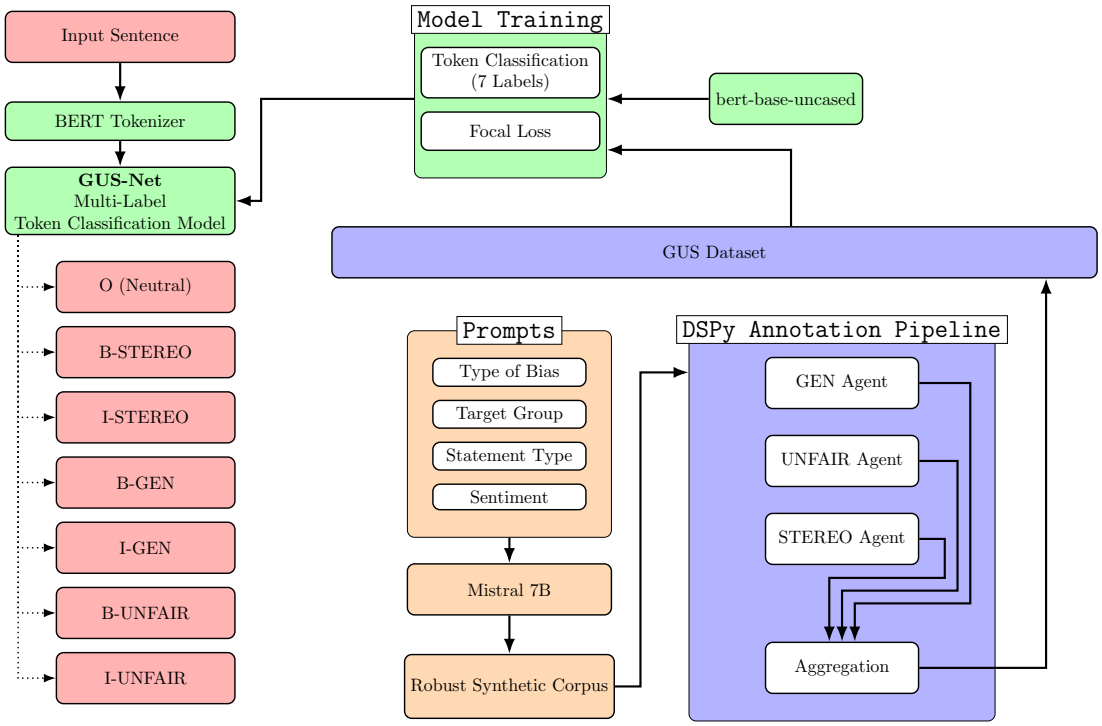


Figure 1: Proposed GUS-Net Architecture

automated agents, we further fine-tune the pre-trained model BERT, for the task of multi-label token classification, as shown in Figure 1. This approach improves upon traditional methods by combining LLM reasoning and the powerful contextual embeddings provided by pre-trained models, resulting in more accurate and comprehensive bias detection across various types of text. Our extensive experiments demonstrate that the proposed method outperforms state-of-the-art techniques in terms of dataset diversity and annotation depth. The fine-tuned model not only achieves superior performance on traditional metrics, such as accuracy and F1-score but also provides a more nuanced understanding of the biases present in various texts. The main contributions of this paper are:

- We generate a synthetic corpus containing biases in varied domains, which is annotated by a team of LLM agents.
- We train an NLP model for multi-label named-entity recognition, enhancing bias detection specificity and insight.
- We conduct experiments to demonstrate the contributions of our methods in relation to existing approaches, showcasing improvements in accuracy, F1-score, and the depth of bias detection.

2 Related Works

The most prominent natural language processing (NLP) techniques for social bias detection/classification primarily focus on: (1) Training dataset annotation, (2) Transfer learning using general-purpose NLP models, and (3) Named-entity recognition (NER).

These methods have shown promise in accurately identifying explicit bias, yet fall short in a few key areas:

- Approaching "bias" with a single definition isn't inclusive of varying viewpoints.
- Dataset quality is limited by an annotator's expertise and subjective opinion, often at the mercy of the organizer's objectivity.
- Binary sequence classification of bias is a broad abstraction, and doesn't offer insight to bias mitigation.
- Named-entity recognition models are more specific than sequence classification, yet still approach bias with a one-size-fits-all definition.

A 2022 model for news article bias detection, Dbias, was trained on the MBIC dataset of 1,700 sentences [8]. Nbias, published in 2023, utilized named-entity recognition but implemented just one new entity, "BIAS" [7].

Both papers trail blazed methods for improving bias detection accuracy, but modern synthetic data generation techniques can unlock more granular predictions and offer solutions to previous limitations.

2.1 Ethical Dataset Construction Manually annotating a training corpus for a bias detection model involves deep reflection and contextual understanding. Firstly, it should encompass a broad spectrum of demographic and ideological diversity, without over-representation or under-representation of any groups. An ideal dataset is also presently relevant while remaining conscious of historical biases. Finally, the logistics of accurately annotating such a dataset come with human obstacles of their own.

The BABE (Bias Annotations By Experts) dataset is a robust dataset for media bias research, and establishes a precedent for responsible bias labelling. In 2022, Spinde et al. published 3,700 sentences, annotated by trained experts to achieve higher annotation quality and inter-annotator agreement than crowd sourced datasets [9]. BABE includes word and sentence-level annotations, providing a detailed analysis of bias, balanced across various topics and outlets. Versions of BERT were fine-tuned on BABE, and BERT (with distant supervision) received the highest macro F1-score of 0.804 [9], at binary classification.

2.2 Transfer Learning with BERT The versatility of pre-trained NLP models has pushed transfer learning to become a cornerstone of NLP tasks. By training architectures like BERT [1] (Bidirectional Encoder Representations from Transformers) generally on an extensive corpus, then refining it for a task (e.g. bias detection), developers have access to powerful token representations. BERT, released in a paper by Devlin et al. in 2018, has an architecture that allows it to capture the context of all surrounding words in each token's embedding. These representations of words and sentences can be used to make many forms of predictions, for instance, bias classifications in text. Using BERT for transfer learning also generalizes well across different domains, which makes it a powerful building block of bias detection.

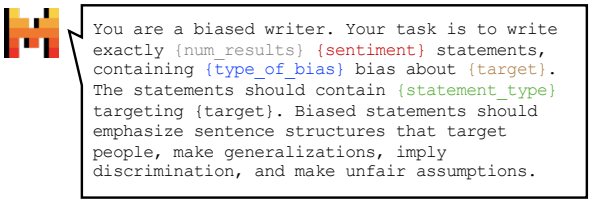
BERT has been employed in various bias detection frameworks, including those focused on media bias. In 2022, the Dbias model illustrated the effective use of BERT for bias detection in news articles. Raza et al. fine-tuned a version of BERT on the MBIC dataset, leveraging contextual embeddings to classify bias at the sentence-level, achieving an F1 score of 0.75 [8]. BERT and its variants have proven the effectiveness of their embeddings for NLP tasks, and more specifically, for bias detection. It will serve as a strong foundation for the future of bias detection systems.

2.3 Named-entity Recognition (NER) Named-entity recognition (NER) is a machine learning approach to identify words or multi-word entities in text. The concept has long been used in social bias analysis, for example, studying patterns between entities such as pronouns and descriptors in HR communications. Amazon was widely criticized for not making use of this technique to ensure safety in 2015, after their hiring tool was discovered to favor men [10].

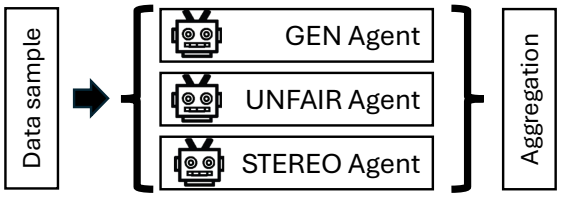
The Dbias framework, released by Raza et al. in 2022, is composed of multiple modules. The first is bias detection at the sentence level. Then, text sequences (i.e. sentences) that are classified as biased get passed to the second stage: an NER model that identifies the biased words in the text. Finally, the third and fourth stages mask and replace the biased words. RoBERTa and Spacy core web transformer (trf) pipeline [8] are the foundations of their NER model, fine-tuned on the MBIC dataset (1,700 records), with annotations for biased entities. This dataset, while insightful, is small and the annotation was outsourced to micro-jobbers on Amazon Mechanical Turk. Dbias's use of NER demonstrates its potential for bias entity recognition.

Raza et al. went on to publish "Nbias: A Natural Language Processing Framework for Bias Identification in Text" in 2023 [7]. This research paper improved upon the earlier paper's second-stage NER architecture, training BERT with the entity "BIAS" and achieving a much-improved F1 score of 0.869-0.903. The key to Nbias's NER performance improvement was the use of a more comprehensive dataset than MBIC, used in Dbias. Datasets from diverse domains were incorporated, such as healthcare (MIMIC, MACCROBAT), news/social media (BABE), and HR (Job Hiring/Recruitment dataset). Their total dataset size is 41,100 records. 20% were manually annotated, then those labels were used to train a BERT model to label the rest. This semi-autonomous labeling technique is bottle-necked by the accuracy of the BERT model trained for labeling, though the resulting F1-score proved this method largely effective.

Raza et al. show strong potential for NER in bias identification at the word level. The use of a more diverse and comprehensive dataset contributed to a 1-8% accuracy improvement from previous methods. These results inspire further thought into the use of generative AI for data collection and more advanced semi-autonomous labeling, enabling a more complex NER model.



(a) Corpus Generation



(b) Multi-agent Annotation

Figure 2: Overview of dataset generation pipeline, which includes (a) corpus generation with Mistral [4] though specifications on different arguments, and (b) multi-agent annotation with DSPy [6].

3 Methodology

3.1 Dataset Generation With modern synthetic training data labeling techniques, we can create a comprehensive dataset encapsulating our novel entities, while avoiding the labor intensive and potentially subjective human annotation process. In addition to a synthetic data annotation pipeline, we also use a language model to synthetically generate the underlying corpus for better coverage and structural consistency of the entities we're looking to classify. The overall dataset generation pipeline can be found in Figure 1.

3.1.1 Corpus Creation Since no existing dataset includes the specific entities of generalizations, unfairness, and stereotypes, we created a synthetic corpus of statements and questions. To ensure coverage of many domains, we created four lists to use in prompting: Type of bias, target group, statement type, and sentiment (as shown in Table 1).

Using Mistral-7B [4], chosen for its lack of guardrails, we created prompts with combinations of values from each of the lists. The same process was applied with different prompt templates and types of statements for fair statements, and questions. The prompt template used for biased statement generation is included in Figure 2a.

For 1000 fair statement generations, `Sentiment` was modified to "slightly positive yet fair" or "slightly negative yet fair." Responses were given in JSON format for easier parsing and storage. Alternatively, an authentic corpus could be used for annotation and training. However, the synthetic corpus created offers two key advantages: a broader scope that balances domains and underlying content that is dense in our targeted parts of speech.

3.1.2 Data Annotation We utilized gpt-4o [**?**] and the Stanford DSPy [6] framework to annotate the generated sentences with entity labels. We created an annotation pipeline to label one entity's presence in a text sequence at a time, then aggregate the labels as shown in Figure 2b.

DSPy modules were compiled ahead of annotation for each entity, enabling prompts that contain the entity

Table 1: Detailed list of arguments for corpus creation.

Type Of Bias	Target	Statement Type	Sentiment
Racial	['white people', 'black people', 'asian people', 'hispanic people', 'indigenous people', ...]	Stereotypes, Unfair Generalizations, False Assumptions, Discriminatory Language, Offensive Implications	Positive, Negative
Religious	['christians', 'muslims', 'jewish people', 'hindus', 'buddhists', 'sikhs', 'atheists', 'agnostics'		
Gender	['men', 'women', 'boys', 'girls', 'females', 'males', 'non-binary people', ...]		
Age	['children', 'teenagers', 'young people', 'middle aged people', 'old people', ...]		
Nationality	['immigrants', 'refugees', 'people from developing countries', 'people from Western countries', ...]		
Sexuality	['straight people', 'gay people', 'bisexual people', 'asexual people', 'LGBTQIA+ people', ...]		
Socioeconomic	['working class people', 'middle class people', 'upper class people', 'poor people', 'rich people', ...]		
Educational	['uneducated people', 'highly educated people', 'people with non-traditional education', ...]		
Disability	['people with physical disabilities', 'people in wheelchairs', 'people with mental disabilities', ...]		
Political	['republicans', 'democrats', 'independents', 'conservatives', 'liberals', 'progressives', ...]		
Physical	['tall people', 'short people', 'fat people', 'skinny people', 'ugly people', 'hot people', ...]		

Table 2: Definitions used by annotator agents, and the corresponding entity labels.

Class	Prompt Definition	Entities
Generalizations	*Any broad generalization of a group or ubiquitous classifiers, including adjectives/descriptors.*	B-GEN, I-GEN
Unfairness	*Any harsh or unjust characterization or offensive language.*	B-UNFAIR, I-UNFAIR
Stereotypes	*Any statement (multiple words) that contains a stereotype targeting a group of people, both explicitly and unconsciously.*	B-STEREO, I-STEREO
Neutral		O

definition and four examples of correct annotations for that entity. In effect, each annotation is done by an agent supported by an LLM for a single type of entity (e.g., GEN, UNFAIR, STEREO), resulting in a list of NER tags for each sentence. The definitions are provided in Table 2.

Central to the DSPy "programming not prompting" paradigm are the examples included in the prompt. The examples included in our pipeline were handwritten and curated by our researchers and advisors, which differs from DSPy's intended use in assisting to create optimal synthetic examples. Also included in our pipeline were `Suggestions`, to ensure correct word/label alignment with backtracking where necessary.

Each of the three lists of individual entity labels was then systematically aggregated into a single two-dimensional list, where each sub-list contains one or multiple tags for each token. Specifically, each token would be categorized into one or multiple semantic part-of-speech categories with B/I/O (Beginning, Inside, Outside) labels, as shown in Table 2.

In total, we annotated 3739 sentences, each annotated for multi-label token classification training. A summary of the annotated dataset is shown in Table 3.

Table 3: Statistics of GUS Dataset.

# records	3739	# tokens	77157
# statements	2045	# questions	1694
# B-GEN	6736	# I-GEN	3270
# B-UNFAIR	1899	# I-UNFAIR	2154
# B-STEREO	2353	# I-STEREO	12465
# O	47312		

3.2 Proposed Model To efficiently and accurately identify social biases in text, we propose a multi-label token classification model. As shown in Figure 1, we fine-tune a pre-trained model, `bert-base-uncased` [1] to preform the multi-label classification.

Rather than implementing a single entity meant to capture all definitions and nuances of "bias," we achieve more granular and accurate insights with these entities, chosen for their individual semantic clarity and collective comprehensiveness of social bias. Using a multi-label BIO format to represent token-level annotations enables predictions of nested entities that can span multiple words. For example, stereotypes often span a full sentence that begins with a generalization, to which some unfairness is assigned.

3.2.1 Model Architecture The GUS-Net model is a multi-label token classification system designed to identify social bias across three categories: generalizations, unfairness, and stereotypes. It outputs 7 labels, which allows the model to capture the nuanced structure of biased language in text sequences. These labels not only enable the identification of individual bias categories but also provide flexibility for overlapping and nested biases.

Input processing. We tokenized all sentences using the pre-trained BERT tokenizer, ensuring that token splits (such as sub-words) inherited the correct entity labels from the parent word. Each text sequence was padded to a maximum length of 128 tokens to ensure consistent input size. Since sentences are rarely longer than 128 tokens, we reduced the BERT input size from the default 512 tokens, representing a 16x reduction in self-attention elements to be processed. Correspondingly, the NER tags were converted into a $(128, 7)$ dimensional vector, where each of the 7 elements represents a binary label (0 or 1) for the respective entity type. These vectors were padded with -100 values up to the full sequence length of 128 tokens, with the -100 values being ignored during the loss calculation.

Model fine-tuning. We fine-tune a pre-trained transformer model, specifically `bert-base-uncased`, due to its ability to capture deep contextual relationships between words, which is crucial for identifying implicit biases [1]. BERT's bidirectional nature allows it to process the entire input sequence, ensuring that each token is evaluated in the context of its surrounding words. This feature is particularly valuable in detecting subtle and complex forms of social bias.

The model is implemented with the Hugging Face `transformers` library. Input text is tokenized using the pre-trained BERT tokenizer, which breaks down sentences into sub-word units while preserving word boundaries. Each token sequence is padded to a length of 128 tokens, and the corresponding labels are mapped to ensure proper alignment. By reducing the input size from the default 512 tokens to 128, we optimize the model's computational efficiency without sacrificing performance for typical sentence lengths in the dataset. The fine-tuned model is available here: (GUS-Net Model).

3.2.2 Loss Function Given the significant class imbalance in our dataset, where certain entities like **STEREO** are underrepresented compared to frequent entities like **O**, we employed **focal loss** to address this challenge. Standard binary cross-entropy (BCE) tends to focus on the majority class, leading to poor performance on rare classes. Focal loss extends BCE by introducing two parameters:

- **Alpha ($\alpha = 0.75$)**, which balances the contribution of positive and negative samples, ensuring under-

represented entities receive more focus.

- **Gamma** ($\gamma = 3$), which down-weights well-classified examples, allowing the model to focus on harder-to-predict instances.

The focal loss function is defined as:

$$\text{FL}(p_t) = -\alpha_t(1-p_t)^\gamma \log(p_t)$$

where p_t is the predicted probability for the true class. The term $(1-p_t)^\gamma$ reduces the impact of well-classified examples, helping the model prioritize rare or difficult examples. The loss was calculated over all tokens, with -100-padded tokens ignored during computation. The mean loss of the tokens was used for the text sequence loss.

4 Experiments

4.1 Settings

4.1.1 Task Description and Metrics Multi-label classification with GUS. GUS is the generated dataset comprising 3,700 sentences, evenly split between biased and fair statements. Each sentence was annotated with one or more labels.

Metrics We evaluated the model using a variety of single-label and multi-label metrics, as shown in Table 4, to assess its ability to identify biased entities across token sequences:

- **Hamming Loss**: Measures the fraction of incorrect labels over all tokens in the sequence, accounting for multi-label classification.

$$\text{Hamming Loss} = \frac{1}{L}\sum_{i=1}^{L} 1(y_i \neq \hat{y}_i)$$

where L is the number of tokens, y_i is the true label for the i-th token, and $\hat{y}_i$ is the predicted label.

- **Precision**, **Recall**, and **F1-Score**: These metrics were calculated for each entity class individually and as a macro-average.

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Recall} = \frac{TP}{TP+FN}$$

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

where TP is true positives, FP is false positives, and FN is false negatives.

To account for the class imbalance, we evaluated individual entity class metrics in addition to the macro-average performance of the model. Further, treating B- and I- tags as a single entity (e.g., B-GEN and I-GEN predictions both used to calculate "Generalizations" metrics) allowed us to focus on the model's ability to detect the presence of each biased entity, rather than evaluating the boundaries.

4.1.2 Hardware and Environment All experiments were conducted on a single NVIDIA T4 GPU with 16GB of memory, hosted on Google Colab, utilizing under 10GB of RAM. The codebase was implemented using `PyTorch` and the `Transformers` library, and executed on Ubuntu 20.04 with Python 3.8. We employed `pytorch-lightning` to streamline the training loops and logging mechanisms.

4.1.3 Hyperparameters We trained our BERT-based multi-label token classification model with seven output classes for **17 epochs**, using a **batch size of 16** and an initial **learning rate of** 5×10^{-5}. The AdamW optimizer with weight decay was utilized, along with a linear learning rate scheduler featuring a warm-up ratio of 0.1. Focal loss was employed as the loss function, with $\alpha = 0.65$ and $\gamma = 2$ to handle class imbalance. The classification threshold for all labels was set at 0.5. The original dataset was partitioned into training (70%), validation (15%), and test (15%) splits, maintaining similar distributions of the biased entity types across splits.

4.2 Results

4.2.1 Overall Performance Table 4 shows that our model balances performance across entity classes. The model performs exceptionally well on the Stereotypes and Neutral classes, and still moderately well at predicting Unfairness despite it being underrepresented in the dataset.

Table 4: Overall Model Performance on Test Set

Entity Class	Precision	Recall	F1
Generalizations	0.78	0.72	0.74
Unfairness	0.69	0.49	0.61
Stereotypes	0.89	0.90	0.90
Neutral	0.93	0.97	0.95
Macro Average	0.82	0.77	0.80
Hamming Loss		0.0528	

4.2.2 Baseline Comparison Finding a reasonable baseline for our novel model comes with complications, since there isn't a directly comparable model with our outputs. Instead, we re-implemented Nbias, with a multi-label architecture, and fine-tuned/evaluated it on the GUS dataset. The most important contribution this comparison highlights is the use of focal loss instead of binary cross-entropy, seen in Table 5.

Table 5: Baseline Comparison of Nbias Architecture (Multi-Label Implementation) Performance on Test Set

Metric	GUS-Net	Nbias
Generalizations F1	**0.74**	0.70
Unfairness F1	**0.61**	0.19
Stereotypes F1	**0.90**	0.89
Neutral F1	**0.95**	0.95
Macro Average F1	**0.80**	0.68
Hamming Loss	**0.05**	0.06

Furthermore, we can approximate a separate comparison between (X)the number of biased words per sentence in the **BABE** dataset and (Y)the number of positive (non-'O') label classifications made by our model. Although this is not a perfect one-to-one comparison, it allows us to assess whether our model's predictions align and scale with the definition of bias represented in the BABE dataset, as shown in Figure 3.

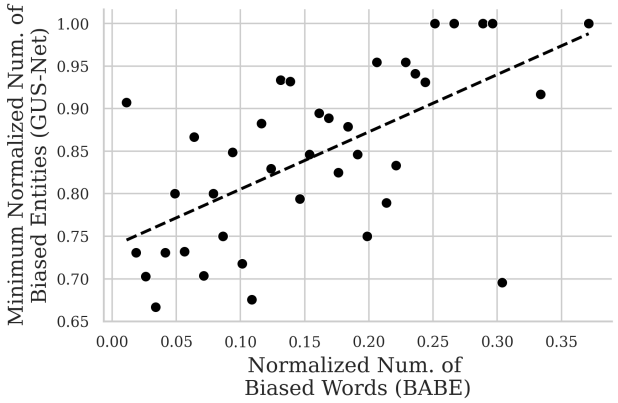


Figure 3: Scatter plot showing the minimum normalized biased entities versus normalized number of biased words, along with the trend line.

To perform this comparison, we first filtered the train split of the BABE dataset to include only sentences classified as biased. Since our model labels multiple entity types (GEN, UNFAIR, and STEREO), and the BABE dataset does not distinguish between different forms of bias, we adjusted for imbalance by binning the results and using the minimum number of GUS entities found in each bin. Additionally, both the number of biased words from BABE and the entities predicted by our model were normalized by sentence length.

Normalizing by sentence length is crucial as it accounts for variability in sentence structures, ensuring that longer sentences do not artificially inflate the GUS-Net entity count or the BABE number of biased words. This comparison provides a general sense of whether our model captures social bias in a manner consistent with the definition used in the BABE dataset.

In Figure 3, the scatter plot displays a positive correlation between the normalized number of biased words from the BABE dataset and the normalized minimum number of biased entities predicted by our model. The linear regression trend line indicates that our model's internal understanding of bias aligns with the definition of bias in the BABE dataset.

4.2.3 Ablation Study We conducted an ablation study to evaluate the impact of different configurations on the model's performance. Table 6 presents the macro-average Precision, Recall, F1-score, and Hamming Loss for the following settings:

- Our proposed **GUS-Net** model.
- Training on an authentic corpus (**BABE**) with synthetic annotations from the same pipeline used on our synthetic corpus.
- Using the **Binary Cross-Entropy** loss function.
- A maximum input length of **512** tokens.

Table 6: Ablation Study Results: Comparing Binary Cross-Entropy, Input Length, and Dataset

Metric	GUS	BABE	BCE	512
Precision	**0.82**	0.02	0.93	0.78
Recall	**0.77**	0.22	0.63	0.70
F1-Score	**0.80**	0.05	0.68	0.73
Hamming Loss	**0.05**	0.26	0.06	0.07

The results in Table 6 demonstrate that our proposed architecture, **GUS-Net**, outperforms other configurations across nearly all key performance metrics. Specifically, GUS-Net achieves the highest macro-average Precision (0.82) and F1-Score (0.80), and the lowest Hamming Loss (0.05), indicating its superior ability to correctly identify and classify entities with

minimal misclassifications. The high Precision and F1-Score suggest that GUS-Net is particularly effective at reducing false positives while maintaining a strong balance between Precision and Recall.

By substituting focal loss for BCE, the model shows a moderate Precision of 0.65, but upon further inspection of the metrics for each entity individually, we can see that the macro-average metrics are distorted by the class imbalance of 'O' tags. Essentially, the model learns to focus on predicting 'O' tags correctly, instead of the new classes we're interested in. This distortion emphasizes the importance of using a loss function and architecture, like those in GUS-Net, that are specifically designed to handle class imbalance, ensuring a more accurate and reliable model performance. Interestingly, using the BABE dataset as the underlying corpus for annotation and training, showed very poor results. This is likely because our test set was designed to span many domains, while the BABE corpus was gathered specifically from news articles.

4.3 Parameter Sensitivity Study To find the optimal focal loss parameters α and γ we tested various values for each while holding the other constant. We used the F1-Score of each entity class, for an evaluation that accounts for the varying levels of entity representations in the test dataset. As you can see in Table 7 and Table 8, the best preforming values were $\alpha = 0.65$ and $\gamma = 2$.

Table 7: F1-Scores at varying α values, while $\gamma = 2$.

α	**0.1**	**0.2**	**0.4**	**0.65**	**0.8**
Generalizations F1	0.19	0.40	0.56	**0.74**	0.71
Unfairness F1	0.01	0.14	0.35	**0.61**	0.54
Stereotypes F1	0.60	0.81	0.83	**0.90**	0.83
Neutral F1	0.87	0.91	0.94	**0.95**	0.91
Macro Average F1	0.42	0.57	0.67	**0.80**	0.75
Hamming Loss	0.09	0.08	0.07	**0.05**	0.09

Table 8: F1-Scores at varying γ values, while $\alpha = 0.65$.

γ	**0.5**	**1**	**2**	**3**	**4**
Generalizations F1	0.74	0.73	**0.74**	0.74	0.71
Unfairness F1	0.55	0.48	**0.61**	0.57	0.57
Stereotypes F1	0.90	0.89	**0.90**	0.88	0.87
Neutral F1	0.95	0.95	**0.95**	0.94	0.94
Macro Average F1	0.78	0.76	**0.80**	0.78	0.77
Hamming Loss	0.05	0.05	**0.05**	0.06	0.06

4.4 Case Study To demonstrate our model's labels and generalizability, we've included an example of religious bias from the GUS Dataset in Table 9. We've also included an example of religious bias (on a different group), labeled by GUS-Net. The results in Table 10 showcase GUS-Net's outputs, and ability to generalize.

Table 9: Labeled Dataset Example

Token	**Labels**
All	B-STEREO, B-GEN
Christians	I-STEREO, I-GEN
believe	I-STEREO
in	I-STEREO
an	I-STEREO
outdated	I-STEREO, B-UNFAIR
fairy	I-STEREO, I-UNFAIR
tale	I-STEREO, I-UNFAIR
called	I-STEREO
the	I-STEREO
Bible	I-STEREO

Table 10: Model Output Example

Token	**Labels**
Atheists	B-STEREO, B-GEN
are	I-STEREO
so	I-STEREO
close-	I-STEREO, B-UNFAIR
minded	I-STEREO, I-UNFAIR

5 Conclusion and Discussion

The proposed GUS-Net model addresses limitations in existing bias detection methods by focusing on the nuanced identification of social biases with semantic categories of generalizations, unfairness, and stereotypes. Moreover, GUS-Net uses a mutli-label token classification architecture, based on bert-base-uncased, that allows entities to span multiple tokens and be nested within each other. Previous models such as Nbias, classified bias under a single, broad entity. Instead, GUS-Net approaches bias with three entities, offering a more granular and precise detection of social biases. This enables better insights into the structural components of biased language. Our results demonstrate that GUS-Net performs well at classifying tokens as each of the entities, with a notable strength in detecting stereotypes. In sum, GUS-Net contributes the field of bias detection in NLP by incorporating a fine-grained and multi-faceted view of biased language.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *Bert: Pre-training of deep bidirectional transformers for language understanding*, in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2019, pp. 4171–4186.

[2] D. Hovy and S. Prabhumoye, *Five sources of bias in natural language processing*, Language and linguistics compass, 15 (2021), p. e12432.

[3] A. Z. Jacobs, S. L. Blodgett, S. Barocas, H. Daumé III, and H. Wallach, *The meaning and measurement of bias: lessons from natural language processing*, in Proceedings of the 2020 conference on fairness, accountability, and transparency, 2020, pp. 706–706.

[4] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, *Mistral 7b*, 2023, `https://arxiv.org/abs/2310.06825`, `https://arxiv.org/abs/2310.06825`.

[5] E. Kasneci, K. Sessler, S. Küchemann, M. Bannert, D. Dementieva, F. Fischer, U. Gasser, G. Groh, S. Günnemann, E. Hüllermeier, et al., *Chatgpt for good? on opportunities and challenges of large language models for education*, Learning and individual differences, 103 (2023), p. 102274.

[6] O. Khattab, A. Singhvi, P. Maheshwari, Z. Zhang, K. Santhanam, S. Haq, A. Sharma, T. T. Joshi, H. Moazam, H. Miller, M. Zaharia, and C. Potts, *Dspy: Compiling declarative language model calls into self-improving pipelines*, arXiv preprint arXiv:2310.03714, (2023), `https://arxiv.org/abs/2310.03714`.

[7] S. Raza, M. Garg, D. J. Reji, S. R. Bashir, and C. Ding, *Nbias: A natural language processing framework for bias identification in text*, Expert Systems with Applications, 237 (2024), p. 121542.

[8] S. Raza, D. J. Reji, and C. Ding, *Dbias: detecting biases and ensuring fairness in news articles*, International Journal of Data Science and Analytics, 17 (2024), pp. 39–59.

[9] T. Spinde, M. Plank, J.-D. Krieger, T. Ruas, B. Gipp, and A. Aizawa, *Neural media bias detection using distant supervision with babe–bias annotations by experts*, arXiv preprint arXiv:2209.14557, (2022).

[10] T. Sun, A. Gaut, S. Tang, Y. Huang, M. ElSherief, J. Zhao, D. Mirza, E. Belding, K.-W. Chang, and W. Y. Wang, *Mitigating gender bias in natural language processing: Literature review*, arXiv preprint arXiv:1906.08976, (2019).

[11] M. Vidgof, S. Bachhofner, and J. Mendling, *Large language models for business process management: Opportunities and challenges*, in International Conference on Business Process Management, Springer, 2023, pp. 107–123.

[12] S. Zhang, Y. Shen, Z. Tan, Y. Wu, and W. Lu, *De-bias for generative extraction in unified ner task*, in Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2022, pp. 808–818.

[13] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, et al., *A survey of large language models*, arXiv preprint arXiv:2303.18223, (2023).