HateXplain2.0: An Explainable Hate Speech Detection Framework Utilizing Subjective Projection from Contextual Knowledge Space to Disjoint Concept Space

Md Fahim [1] **Md Shihab Shahriar** [2] **Mohammad Sabik Irbaz** [2] **Syed Ishtiaque Ahmed** [3] **Mohammad Ruhul Amin** [4]

Abstract

Finetuning large pre-trained language models on specific datasets is a popular approach in (Natural Language Processing) NLP classification tasks. However, this can lead to overfitting and reduce model explainability. In this paper, we propose a framework that uses the projection of sentence representations onto task-specific conceptual spaces for improved explainability. Each conceptual space corresponds to a class and is learned through a transformer operator optimized during classification tasks. The dimensions of the concept spaces can be trained and optimized. Our framework shows that each dimension is associated with specific words which represent the corresponding class. To optimize the training of the operators, we introduce intra- and inter-space losses. Experimental results on two datasets demonstrate that our model achieves better accuracy and explainability. On the HateXplain dataset, our model shows at least a 10% improvement in various explainability metrics.

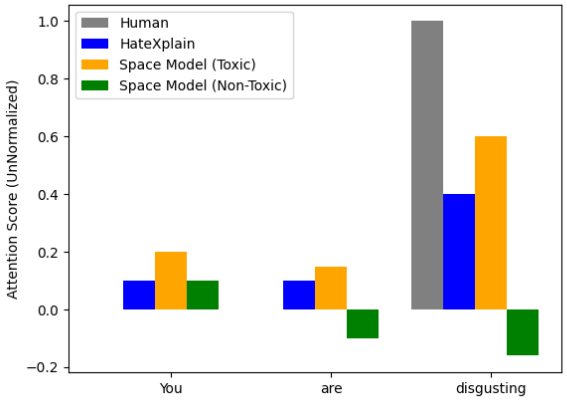

Figure 1: Comparison of Attention Scores of Space model with HateXplain and Human Rationales for a toxic sentence. For binary toxic comment classification, the Space model provides both non-toxic and toxic attention scores for each individual word of a sentence.

1. Introduction

Large Language Models (LLMs) such as BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), DeBERTa (He et al., 2021), and so on, have demonstrated their effectiveness in Natural Language Processing (NLP) tasks, particularly in understanding context and cultural nuances in specific languages. These models have shown powerful capabilities in solving a wide range of NLP tasks, particularly in their pretraining stage. To leverage the language-specific information acquired during pretraining for specific tasks such as Text Classification, Text Generation, and Question Answering, these models are typically fine-tuned with all the information they have captured.

However, the fine-tuning stage has revealed challenges in effectively transferring the knowledge of LLMs to various downstream tasks. These challenges encompass issues such as anisotropy problem (Ethayarajh, 2019), catastrophic forgetting (Xu et al., 2020), overfitting (Huang et al., 2021), and a lack of interpretability.

The proliferation of the internet has led to increased participation in online communities, social media platforms, and other digital spaces. However, with the increase of on-

[1]Center for Computational & Data Sciences, Independent University Bangladesh (IUB) Dhaka-1229, Bangladesh [2]Islamic University of Technology, Boardbazar, Gazipur, Bangladesh [3]University of Toronto, Toronto, ON, Canada [4]Fordham University, New York, USA. Correspondence to: Md Fahim <fahimcse381@gmail.com>, Md Shihab Shahriar <shihab1069@gmail.com>, Mohammad Ruhul Amin <mamin17@fordham.edu>.

AI & HCI Workshop at the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. 2023. Copyright 2023 by the author(s).

Model	Text	Label
Human Annotator	The jews are again using holohoax as an excuse to spread their agenda. Hitler should have eradicated them	HS
BiRNN-HateXplain	The jews are again using holohoax as an excuse to spread their agenda. Hitler should have eradicated them	HS
BiRNN-SpaceModel	The jews are again using holohoax as an excuse to spread their agenda. Hitler should have eradicated them	OF
BERT-HateXplain	The jews are again using holohoax as an excuse to spread their agenda. Hilter should have eradicated them	OF
BERT-SpaceModel	The jews are again using holohoax as an excuse to spread their agenda. Hilter should have eradicated them	HS

Table 1: Comparing Predicted Rationales: Our Space Model vs Human Annotator and HateXplain Models. The tokens marked in green are identified as important for the prediction by both the human annotator and the model. Whereas, the tokens marked in orange were found important by the model, but not by the human annotators. Here we can see BiRNN-SpaceModel predicts more wrong rationales than human where BERT-SpaceModel predicts fewer but more correct rationales.

line users, we also see a significant rise in harmful, abusive, or offensive comments, posts, and other forms of toxicity. These comments and posts can be directed against individuals or groups based on their race, gender, sexual orientation, religion, or other personal characteristics. Toxicity classification is thus an essential aspect for fostering healthy and respectful interactions online, and LLMs in the field of NLP have the potential to play a vital role in this regard. It is crucial to develop better fine-tuning approaches to ensure better performance on detecting abusive comments, threats, racism, and bias on online platforms such as blogs, articles, and social media.

To tackle this issue, we introduce a novel technique called the **Space Model** as a solution. In Figure 1, we provide an illustration of how the Space Model operates on a sentence. When dealing with two class labels (hate and non-hate), our model generates two attention scores for each word: one for the hate class and another for the non-hate class. These attention scores represent the contribution of each word towards predicting the respective class by the model.

To disjoint the concept spaces during training, we introduce two different losses in our model. We evaluate the performance of our Space Model on a hate speech classification dataset, where we observe a significant improvement over the vanilla model. In addition, we report explainability metrics, which demonstrate that our model provides better explainability and word-level attention scores. To generate the contextual knowledge space, we use both transformer-based and non-transformer-based text encoders. We illustrated our model effectiveness in the HateXplain (Mathew et al., 2021) dataset as it provides us the rationales also. As we experiment and analyze our space model on this HateXPlain Dataset for detecting hate speech, we also refer our space model as **HateXplain2.0**. We illustrate our model effectiveness in Table 1 for a given sentence with the HateXplain model and human annotator. We observed that our BiRNN-SpaceModel predicts more wrong rationales than human, as a result, it assigns wrong label (offensive speech - OF). On the other hand, our BERT-SpaceModel predicts less but more correct rationales than BiRNN-SpaceModel, as a result, we get correct label prediction (hate speech – HS).

In summary, our Space Model provides a new approach for fine-tuning LLMs for text classification tasks. Unlike other methods, our model offers improved explainability by generating word-level attention scores for each word in a sentence for each individual class. These per-class attention scores not only provide better interpretability but also aid in understanding the model's training, errors, and rationales, which are more human-interpretable.

2. Related Work

The task of finding effective approaches for fine-tuning LLMs has been a prominent topic in the field of NLP. Several research works have been conducted to address this challenge. For instance, Lee et al. introduced the MixOut technique (Lee et al., 2019), which successfully mitigates the issue of catastrophic forgetting during fine-tuning. Instead of updating all the weights at each iteration, MixOut randomly selects a small portion of the large LLMs and updates the weights of that specific part. Similarly, Xu et al. proposed a technique called ChildTraining (Xu et al., 2021), which utilizes the Fisher Information Matrix to identify the target-specific portion of the LLMs during fine-tuning, rather than randomly selecting a portion. Another recent work in this area is CAMERO (Liang et al., 2022), where weight sharing is implemented between the bottom layers of models, and various perturbations are applied to the hidden representations.

In recent years, several datasets have been published for hate speech detection tasks. Wasseem and Hovy (Waseem & Hovy, 2016) provided a dataset that focuses on disambiguating racism and sexism. The dataset contains 16,914 tweets, with 3,383 labeled as sexist, 1,972 as racist, and 11,559 as neither. Davidson et al. (Davidson et al., 2017) released a dataset for distinguishing between hate speech, offensive language, and normal speech. This dataset consists of 25,000 tweets collected using the Twitter API, with offensive language being the majority label (77%), followed by hateful language (6%), and the remaining tweets categorized as normal. Founta et al. (Founta et al., 2018) created a dataset with 80,000 tweets, differentiating between abusive, hateful, normal, and spam content.

More recent datasets have been collected from various sources and incorporate diverse criteria for hate speech classification. Zampieri et al. (Zampieri et al., 2020) introduced the Offensive Language Identification Dataset (OLID), which is considered one of the most well-established datasets for hate speech containing 14k tweets where 4.5k tweets labeled as offensive. It employs a three-level hierarchy annotation schema that includes offensive language detection, categorization, and target identification. Kiela et al. (Kiela et al., 2021) presented a dataset of 40,000 entries, with 54% of them labeled as hateful. They also employed a binary labeling schema for identifying the type and target of hate speech. Multiple annotators provided feedback for the annotation process in this dataset. The HateXplain dataset is the most recent addition to hate speech detection datasets and includes human rationales.

Various machine learning models have been utilized for hate speech classification, including logistic regression, decision trees, random forests, and more. Some approaches incorporate word embedding techniques such as Word2Vec (Mikolov et al., 2013), FastText (Bojanowski et al., 2017), and GloVe (Pennington et al., 2014) to obtain word embeddings from text data, in combination with architectures like CNN (Zhang & Wallace, 2015), LSTM, and GRU (Chung et al., 2014). Currently, transformer-based models such as BERT and different variations of BERT have been outperforming other models in the hate speech classification task. Some research performed a comparison among different BERT models. Another work by Caselli et al. introduced HateBERT (Caselli et al., 2020), a re-trained BERT model for abusive language detection in English. The model was trained on a large-scale dataset of Reddit comments in English from communities banned for being offensive, abusive, or hateful. In all cases, HateBERT outperformed the corresponding general BERT model.

3. Dataset

The paper by Mathew et al. introduces the HateXplain (Mathew et al., 2021) dataset that covers different aspects of hate speech. The dataset uses a 3-class classification (i.e., hate, offensive, or normal), the target community (i.e., the community that has been the victim of hate speech/offensive speech in the post), and the rationales (i.e., the portions of the text on which their labeling decision as hate, offensive or normal, is based). The authors collected text from Twitter and Gab and uses annotators from Amazon Mechanical Turk (MTurk) to label the texts. 3 annotators annotate each text along with the rationales. They are the first dataset to introduce a word and phrase-level annotation. They annotate more than 20k texts of which 5,935 are identified as hate speech, 5,480 as offensive, and 7,814 as normal. The authors fixed the train, test, and validation sets for the dataset. The dataset consists of 31% hate speech class, 29% offensive class, and the rest normal class. Each post is also annotated on a word level as to which words are responsible for the post being labeled as hate speech or offensive. They used BERT and BiLSTM-Att to benchmark their dataset and added a loss term based on the human rationales provided to modify the model. They named their modified model as BERT-HateXplain and BiLSTM-HateXplain respectively.

4. Methodology

Our proposed model uses BERT to generate contextual word embeddings for a given input sentence and then projects them onto two conceptual spaces, a hate space, and a non-hate space, for classification. The training objective is to learn embeddings that are far apart from each other in the conceptual space for hate and non-hate words, and also prevent the embeddings from converging to the same word embedding within the conceptual space. Our model is trained using an inter-space loss and an intra-space loss along with binary cross-entropy loss for classification.

4.1. Contextual Word Embeddings

We use BERT or LSTM (Hochreiter & Schmidhuber, 1997) as text encoder to obtain contextual word embeddings for the input sentence s. BERT is a transformer-based neural network architecture pre-trained on a large corpus of text data to learn contextualized word representations. Given an input sequence of tokens, BERT computes a sequence of contextualized word embeddings that capture the meaning of the words based on their context within the sentence.

Let the embedding for the i^{th} word in the sentence be denoted as $e_i \in R^d$, where d is the dimensionality of the embeddings. The contextual word embeddings are obtained by passing the preprocessed sentence s through BERT, resulting in an output matrix $E = [e_1, e_2, ..., e_n] \in R^{d \times n}$, where n is the number of words in the input sentence.

4.2. Conceptual Word Embedding Matrix / Concept Space

We have two conceptual word embedding matrices, one for hate words and another for non-hate words. We also refer to them as concept spaces consisting of conceptual word embeddings. The hate word embedding matrix $H \in R^{d \times m}$ consists of the embeddings for m conceptual hate words and the non-hate word embedding matrix/ non-hate concept space $N \in R^{d \times m}$ consists of the embeddings for m conceptual non-hate words. Each column of the embedding matrix represents the embedding of a particular word in the vocabulary. Individual concept words do not inherently carry

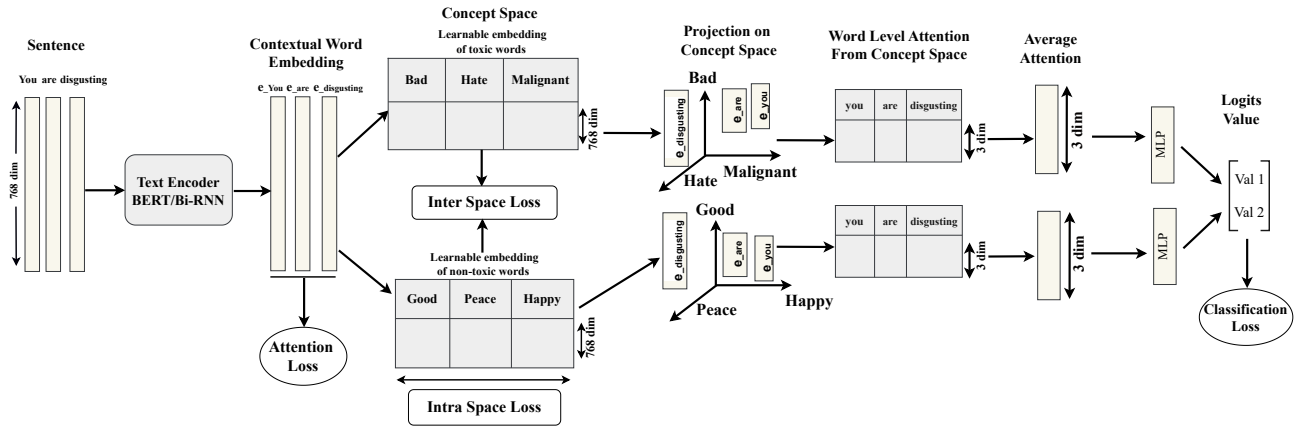

Figure 2: Model Architecture of SpaceModel. After feeding a sentence (e.g. You are disgusting) into a text encoder, the contextual embeddings are extracted with dim 768. Then the concept space for each class consisting of conceptual words is created where no. of conceptual words defines the space dimension. (e.g. For binary toxicity classification two concept spaces are created for each class. The space for the toxic class has the embeddings of 3 concept words (e.g. Hate, Bad, Malignant) while the space for the non-toxic class has the embeddings of the concept words (e.g. Good, Peace, Happy)). A projection is applied to the contextual embeddings over each concept space which measures similarities between them. From the similarities word-level attention scores are measured & mean of those scores are extracted for sentence-level representation. The mean attentions are then fed into linear layers for classification.

concrete semantic interpretations, but rather, they contribute to the formation of clusters consisting of semantically related words. These concept words are not selected based on their explicit meanings; instead, our focus lies solely on their embeddings. By identifying the closest words to these concept words within a sentence, we aim to uncover the underlying meanings associated with the concept words. The model will learn the conceptual word embedding during training.

For multi-class classification having c classes we consider c conceptual word embedding matrices/ concept spaces $S_0, S_1, ..., S_{c-1}$. Each matrix $S_k \in R^{d \times m}$ consists of the d dimensional word embeddings for each of m words of that class. All the matrix S_k will be learned during training.

4.3. Projection

We project the contextual word embeddings onto the conceptual spaces using cosine similarity. For multi-class classification having c classes. For a specific class k the projection of the i^{th} word embedding e_i onto the word concept space of that class S_k is denoted as $p_{i,S_K} \in R$ and is calculated as follows:

$$p_{i,S_k} = (e_i^T S_k)/(||e_i|| \cdot ||S_k||) \tag{1}$$

where $||e_i||$ denotes the L2 norm and $||S_k||$ is the Frobenius norm of Matrix S_k.
Now if we have a sentence s with n words and d dimensional embedding then our contextual word embedding matrix will be $E_s \in R^{d \times n}$. If we apply the projection of E_s in the concept space S_k, we will find a projection matrix $P_{s,k}$ where $P_{s,k} \in R^{m \times n}$ for each concept space S_k. Each entry in $P_{s,k}$ matrix defines the cosine similarity between the word embeddings of the sentence and concept word embeddings of the concept space which indicates the attention scores of the input words w.r.t concept words in that concept space. We call these attention scores word-level attention scores from space.

4.4. Inter-space loss

We introduce an inter-space loss to ensure that the embeddings of the concept spaces are orthogonally apart from each other. The loss L_{inter} is designed to encourage the model to find disjoint sets of concept spaces.

For k-th class, we find the mean of the concept word embeddings (a.k.a mean of concept space S_k) denoted as μ_k then we calculate the sum of inter-space loss for each pair of conceptual spaces as the total inter-space loss.

$$L_{inter} = \sum_{k=0}^{k=c-1} \sum_{l=0}^{l=c-1} \frac{1}{1 - \frac{\mu_k \mu_l}{||\mu_k|| \cdot ||\mu_l||}} \tag{2}$$

4.5. Intra-space loss

To ensure that the embeddings don't converge to the same word embedding inside the conceptual space, we introduce an intra-space loss. The loss L_{intra,S_k} in S_k concept space for k-th class encourages the concept word embeddings to be dissimilar from each other. If there are m different concept words in S_k concept space then the variance of that

space will be

$$\text{Var}(S_k) = \frac{1}{m}\sum_{i=1}^{m}(w_i - \tilde{w}^2) \tag{3}$$

where w_i represents the i-th column of the concept space matrix S_k, and $\tilde{w}$ is the mean vector of S_k defined as $\tilde{w} = \frac{1}{m}\sum_{i=1}^{m} w_i$. Then the intra-space loss for S_k concept space is calculated as follows:

$$L_{intra,S_k} = \frac{1}{\text{Var}(S_k)} \tag{4}$$

The total intra-space loss is computed as follows:

$$L_{intra} = \sum_{k=0}^{k=c-1} L_{intra,S_k} \tag{5}$$

4.6. Classification

After projecting the contextual word embeddings of a sentence s onto the concept spaces S_k, we find the projection matrix $P_{s,k} \in R^{m\times n}$ which give us word-level attention scores for the input words in concept space S_k. To find the sentence-level representation for an input sentence in that space, we compute the mean of the word level attention scores for each space. For a input a sentence s we will get a sentence level representation $m_{s,k} \in R^m$ in S_k concept space as follows:

$$m_{s,k} = \frac{1}{n}\sum_{i=1}^{n} P_{s_i,k} \tag{6}$$

After finding sentence-level representations from a concept space, we pass it into Multi-Layer Perceptron (MLP) to find class logit value. Then we use Cross Entropy (CE) loss as classification loss in the model.

4.7. Attention Loss

To train the model to focus on the most relevant words for each classification, we can use a loss function that incorporates the human rationales as a base for comparison with the attention vectors of each word. One such loss function that we can use is the attention-based rationale loss:

$$L_{att} = -\sum_{i} y_i \log(p_i) + (1-y_i)\log(1-p_i) \tag{7}$$

where y_i is a rationale label indicating whether the ith word in a sentence is part of the rationale, p_i is the attention weight assigned to the ith word by the last layer of the model, and the summation is overall words and sentences in the training set. The rationale labels y_i are obtained from human annotations

The attention-based rationale loss encourages the model to assign high attention weights to the words that are part of the human rationales and low weights to the words that are not part of the rationales.

We minimize the total loss given as L in the following equation:

$$L = L_{CE} + \lambda_1 L_{inter} + \lambda_2 L_{intra} + \lambda_3 L_{att} \tag{8}$$

where λ_i is a hyperparameter that controls the weight given to the losses. The final loss function is a combination of the classification loss, the inter-space loss, the intra-space loss, and the attention-based rationale loss, with appropriate weighting factors for each term.

5. Experiment

We evaluate the effectiveness of our space model in both HateXplain the 3-class dataset and the 2-class dataset. The experiment was done not only on the model performance but also for the explainability results. Additionally, we conducted experiments to investigate and analyze various properties of the space model, providing a comprehensive understanding of its behavior and capabilities.

5.1. Preprocessing & Settings

For both datasets, we use some basic text preprocessing like removing the punctuation, emojis, and URL if any. To get the hidden representation from the text, we use LSTM & BERT model as the encoder. For the HateXplain dataset, the exact model architecture is used to find the text representation. As the BiRNN-Attn model & BERT-HateXplain-Attn model was the benchmarking model for the dataset, we choose these two model architectures for experimenting with our space model. For the BiRNN-Attn-based model, a Bidirectional LSTM model is used and an attention mechanism based on a learnable context vector is applied to the representation from the LSTM. For the BERT model, we use *bert-base-uncased* variants for our space model as an encoder.

We use Adam optimizer with learning rate of 10^{-3} for BiRNN-Attn model and $2 * 10^{-5}$ for BERT-HateXplain-Attn model with batch_size = 32. For most of our experiments, we set the no. of conceptual words (a.k.a space dim) for each space as 128. We experiment with different space dimensions like 4, 10, 32, 64, 128, 256, 512. Among them, space dim 128 produces better results. The ablation study on the concept space dimension is shown in Appendix B in Table 6. For the loss function, in most of our experiments,

Model [Token Method]	Performance			Explainability				
				Plausibility			Faithfulness	
	Acc.↑	Macro F1↑	AUROC↑	IOU F1↑	Token F1↑	AUPRC↑	Comp.↑	Suff.↓
BiRNN-HateXplain[LIME]	0.629	0.629	0.805	0.174	0.407	0.685	0.343	-0.075
BiRNN-SpaceModel [LIME]	0.618	0.614	0.781	0.195	0.334	0.568	0.335	0.090
BiRNN-HateXplain [Attn]	0.629	0.629	0.805	0.222	0.506	0.841	0.281	0.039
BiRNN-SpaceModel [Attn]	0.624	0.612	0.786	**0.353**	**0.544**	0.848	0.208	**-0.0025**
BERT-HateXplain [LIME]	0.698	0.687	**0.851**	0.112	0.452	0.722	0.500	0.004
BERT-SpaceModel [LIME]	0.695	0.688	0.812	0.277	0.466	0.729	**0.579**	0.053
BERT-HateXplain [Attn]	0.698	0.687	**0.851**	0.120	0.411	0.626	0.424	0.160
BERT-SpaceModel [Attn]	**0.701**	**0.693**	0.826	0.133	0.515	**0.881**	0.538	0.035

Table 2: Comparison of performance and explainability metrics of our Space Model with HateXplain model on 3 class HateXplain Dataset. Attention and LIME token method are used for explainability measures. Our Space model outperforms HateXplain in almost all scenarios in explainability metrics. The space model with BERT gives a boost in model performance and the space model with Bi-RNN performs better in explainability metircs

we use Classification Loss along with InterSpace Loss, Intra Space Loss & Attention Loss which are described in section 4. The effect of those losses of our space is shown in Table **??** based on both performances of the model & explainability of the model. An in-depth discussion on the impact of the losses in our model is described in Analysing the Effect of the Losses section C in the Appendix.

In this case, we consider only HateXplain 3-Class dataset for our ablation study as it uses both performance & explainability metrics. For this study, we only consider the BERT model as a text encoder and attention method for selecting the tokens for explainability. We use a single Nvidia Tesla P100 GPU for our training. Our model takes around 8 minutes for a single epoch. The values of $\lambda_1, \lambda_2, \&, \lambda_3$ were set to 1.

5.2. Evaluation Metrics

The authors of the HateXplain paper evaluate their model for both performance and explainability using two different types of metrics. We adopt those metrics for judging our space model also in HateXplain Dataset.

Performance Metrics

For calculating model performance on the prediction, we use a few performance-based metrics. For the HateXplain dataset we use Accuracy, Marco F1 Score &AUROC (Area Under ROC) metrics for model performance.

Model Name	Performance		
	Acc.↑	Macro F1↑	AUROC↑
BERT HateXplain	0.845	0.841	0.882
Space Model + BERT [Attn]	**0.893**	**0.892**	**0.924**

Table 3: Benchmarking on HateXplain Dataset (2 Class). We use the attention-based token method for this experiment. Space model gives around 5% improvement in the model performance than the BERT HateXplain model

Explainability Metrics

In order to evaluate the explainability of our proposed model, we utilize the metrics provided by the HateXplain dataset and the ERASER benchmark by DeYoung (DeYoung et al., 2019). The ERASER benchmark measures explainability through the metrics of plausibility and faithfulness. Plausibility assesses how convincing the model's interpretation is to humans, while faithfulness measures the accuracy with which a model reflects the true reasoning process.

To measure plausibility, we use Intersection-over-Union (IoU) F1 Score, Token F1 Score, and Area Under Precision-Recall Curve (AUPRC). IoU is calculated at the token level by dividing the size of the overlap between a model's prediction and the ground truth rationale by the size of their union. If the overlap is greater than 0.5, it is considered a match, and the F1 score is determined using these partial matches, referred to as IoU F1. Additionally, token-level precision

and recall are measured through the Token F1 Score. To evaluate the plausibility of soft token scoring, we also use AUPRC, which is determined by sweeping a threshold over the token scores.

To evaluate faithfulness, we use the metrics of comprehensiveness and sufficiency as defined in the Eraser evaluation benchmark. Comprehensiveness is measured by creating a contrasting example by removing a predicted rationale and feeding it into the model. The difference in the model's predictions between the original and contrast examples is measured. Sufficiency is measured by creating a synthetic sentence only using the predicted rationale for a given sentence and comparing the model's predictions for the sentence and the synthetic sentence.

5.3. Experimental Results

Our evaluation of the space model involved experiments conducted on the HateXplain dataset, both for the 2-class and 3-class scenarios. We compared the performance of the HateXplain model and our Space Model using classification metrics, as shown in Table 2. By applying the subjective projection to the models from the HateXplain dataset, we obtained improved performance metrics with our space model, as demonstrated in Table 2.

From table 2, the BiRNN models outperform BERT in explainability metrics, although they tend to generate more false rationales. In terms of token methods, the attention-guided token method outperforms the LIME (Ribeiro et al., 2016) attention approach in our model. The attention loss incurred with the attention-guided token method proves beneficial in achieving more accurate space attention scores. Table 1 presents a comparative study of the predicted rationales, revealing that our space model offers superior rationales compared to other models. Across various model types and token methods, our model consistently surpasses the Baseline scores for the HateXplain 3-class dataset by a substantial margin.

In addition, we applied our space model to the 2-class HateXplain dataset and compared it with the BERT-HateXplain-Attn model, as shown in Table 3. Once again, our space model exhibited superior performance in this scenario. These experiments provide strong evidence that our space model not only achieves better performance but also enhances explainability in comparison to existing models. We observed significant enhancements in the explainability measurements, with an average improvement of approximately 10% with the baselines in the HateXplain 3 class dataset.

5.4. Analysis of Attention Scores from Concept Space

To analyze the properties of the concept space, we conducted our experiment on the HateXplain 2-class dataset,

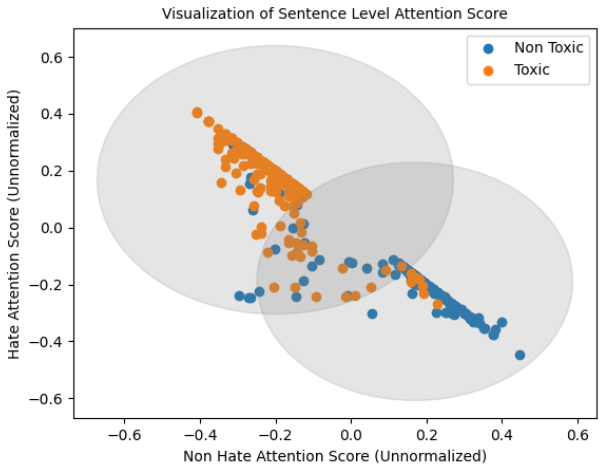

Figure 3: Visualization of Sentence Level Attention Scores (which is the mean of concept words attention scores for a concept space) from the Concept Space of Space Model on 2 class HateXplain dataset.

which consists of toxic and non-toxic labels. As a result, we obtained two distinct concept spaces using our Space Model: one for toxic and another for non-toxic. For each word/token in a sentence, our model provides two attention scores. To obtain sentence-level attention scores, we compute the average of these scores. Figure 3 illustrates that our model assigns opposite attention scores for different classes. The attention scores for the toxic and non-toxic classes exhibit clear separation and form clusters, contributing to improved performance and interpretability of the model. Figure 3 also highlights some intersections between the clusters and a few outliers. These outliers correspond to misclassified samples, and the reasons for misclassification can be attributed to either the model or the labels in the dataset. Some examples of those misclassified samples are provided in Appendix Section F for further discussion.

In addition, we computed the **Rank Correlation** to evaluate the relationship between the ground truth attention scores and the attention scores derived from the concept spaces. Specifically, for the hate samples in the HateXplain 2-class test dataset, we observed an average rank correlation of 26.48% between the ground truth and the attention in the hate space. Similarly, for the non-hate samples, the average rank correlation scores were 21.28%.

To get more intuition about the attention scores derived from the concept space, we measured average hate attention scores and average non-hate scores for each word based on

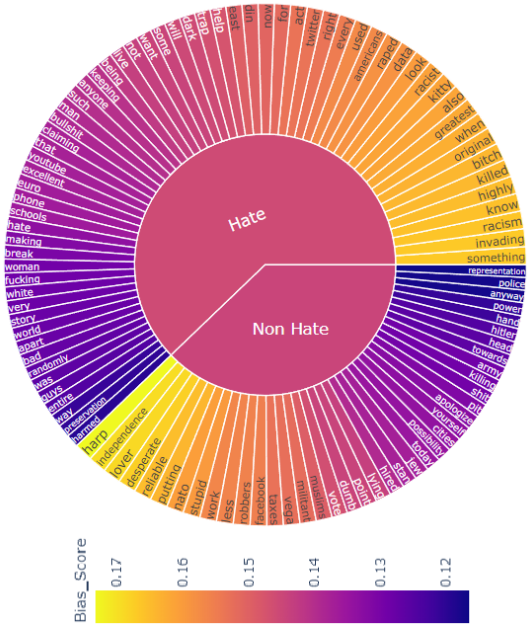

Figure 4: SunBrust Plot of Words Based on the Bias Score in Eqn 9. The labeling for the words is based on that bias score. If the bias score of a word is greater than 0 then it is labeled as a Non-Hate word otherwise labeled as a Hate word in the plot.

their occurrences in different sentences. Then we define a **Bias_Score** for each word as the following:

$$\text{Bias\_Score}_{\omega} = \frac{\sum \text{Non-HateAttn}_{\omega}}{n_{\omega}} - \frac{\sum \text{HateAttn}_{\omega}}{n_{\omega}} \quad (9)$$

where n_{ω} is the no. of occurrence of word ω in test dataset. Following the equation in 9, we refer a word bias to Non-Hate class if its bias score is greater than 0 otherwise the word is biased to Hate class.

In figure 4, we show a sunburst plot for a few words with their bias score and the biases to the classes. To get a more clear view of the sunburst plot, the figure 8 is in the Appendix.

From the sunburst plot in fig 8, we can see some hate words like $racism, killed, bitched, racist, fucking$ and so on in the Hate class. Though there are some non-hate words belonging to the hate class, we get a good amount of hate, non-hate words via *Bias_Score*.

We also experiment with our model efficiency on another dataset and also pretrained & finetuned the space model trained on HateXplain 2 class dataset on that dataset. The results are shown in Appendix A.

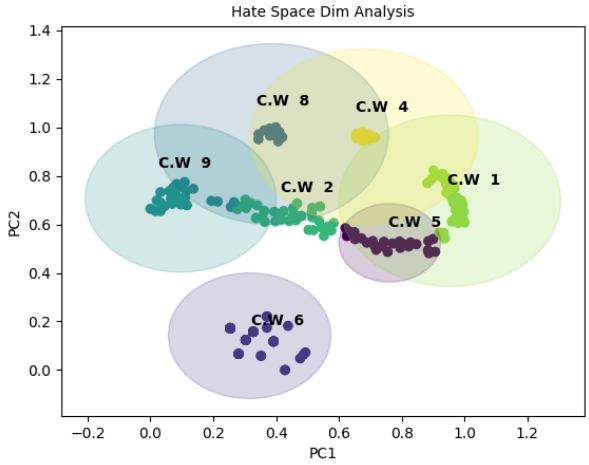

Figure 5: Visualization of Hate Concept Space with Space Dim 10. Here, C.W stands for Concept Words in that space. Within the hate space, the concept words form distinct clusters that partially overlap with each other.

5.5. Concept Words Analysis

We conducted an analysis of concept words on the two-class variation of the HateXplain dataset, considering a space dimension of 10 for simplicity. To perform this experiment, we selected sentences from the test set and compared the contextual word embeddings of each word with the closest conceptual word embedding in the hate space, based on cosine similarity.

In Figure 5, we present a 2D t-SNE plot illustrating the projected word representations of the sentences' words in the hate space. The plot reveals that the words tend to cluster around specific conceptual word embeddings, providing insights into the types of words represented by these conceptual word embeddings. This analysis offers an explanatory perspective on the hate space and its association with different word categories. We also did the same experiment for non-hate concept space. The result is shown in Appendix E where we discussed more about the interpretation of concept words.

6. Conclusion

In conclusion, this paper presents a novel approach, *Subjective Projection on Conceptual Space* named as **Space Model**, for fine-tuning pre-trained natural language processing models for text classification tasks. By learning the representations of conceptual words that are specific to a particular class, our proposed model not only improves performance but also enhances explainability. Overall, the re-

sults of our experiments suggest that our proposed approach can create a more effective representation for task-specific classes and provide a new method for adapting the knowledge of pre-trained language models. These findings open up opportunities for further research on improving the performance and explainability of downstream tasks through novel fine-tuning strategies.

While the current implementation of our model is promising, future research could explore incorporating manifold or geometric-based losses to further enhance the separation of conceptual spaces.

Limitations

The performance of the space model depends on the direction of the conceptual spaces. If the conceptual spaces are orthogonal to each other, then the space model gives better results. But when we have more classes (such as 10 classes), we need to build more conceptual spaces and optimize them. In that case, it becomes difficult to train the space model. Besides, there are overlapping common words between the classes which also hamper the space model. Implementing better loss functions & adding more manifold base strategy to the model may solve those issues.

References

Biswas, A., Masud, Z., Mokhberi, M., Kteily-Hawa, R., Goldstein, A., Gillis, J. R., Rayana, S., and Ahmed, S. I. Covid-stigma: A dataset of anti-asian stigmatizing tweets during covid-19.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5: 135–146, 2017.

Caselli, T., Basile, V., Mitrović, J., and Granitzer, M. Hatebert: Retraining bert for abusive language detection in english. *arXiv preprint arXiv:2010.12472*, 2020.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Davidson, T., Warmsley, D., Macy, M., and Weber, I. Automated hate speech detection and the problem of offensive language. In *Proceedings of the international AAAI conference on web and social media*, volume 11, pp. 512–515, 2017.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

DeYoung, J., Jain, S., Rajani, N. F., Lehman, E., Xiong, C., Socher, R., and Wallace, B. C. Eraser: A benchmark to evaluate rationalized nlp models. *arXiv preprint arXiv:1911.03429*, 2019.

Ethayarajh, K. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. *arXiv preprint arXiv:1909.00512*, 2019.

Founta, A. M., Djouvas, C., Chatzakou, D., Leontiadis, I., Blackburn, J., Stringhini, G., Vakali, A., Sirivianos, M., and Kourtellis, N. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Twelfth International AAAI Conference on Web and Social Media*, 2018.

He, P., Gao, J., and Chen, W. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Huang, S., Xu, D., Yen, I. E., Wang, Y., Chang, S.-E., Li, B., Chen, S., Xie, M., Rajasekaran, S., Liu, H., et al. Sparse progressive distillation: Resolving overfitting under pretrain-and-finetune paradigm. *arXiv preprint arXiv:2110.08190*, 2021.

Hutto, C. and Gilbert, E. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media*, volume 8, pp. 216–225, 2014.

Kiela, D., Bartolo, M., Nie, Y., Kaushik, D., Geiger, A., Wu, Z., Vidgen, B., Prasad, G., Singh, A., Ringshia, P., et al. Dynabench: Rethinking benchmarking in nlp. *arXiv preprint arXiv:2104.14337*, 2021.

Lee, C., Cho, K., and Kang, W. Mixout: Effective regularization to finetune large-scale pretrained language models. *arXiv preprint arXiv:1909.11299*, 2019.

Liang, C., He, P., Shen, Y., Chen, W., and Zhao, T. Camero: Consistency regularized ensemble of perturbed language models with weight sharing. *arXiv preprint arXiv:2204.06625*, 2022.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Mathew, B., Saha, P., Yimam, S. M., Biemann, C., Goyal, P., and Mukherjee, A. Hatexplain: A benchmark dataset for explainable hate speech detection. In *Proceedings of the*

AAAI Conference on Artificial Intelligence, volume 35, pp. 14867–14875, 2021.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Pennington, J., Socher, R., and Manning, C. D. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

Ribeiro, M. T., Singh, S., and Guestrin, C. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.

Waseem, Z. and Hovy, D. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*, pp. 88–93, 2016.

Xu, R., Luo, F., Zhang, Z., Tan, C., Chang, B., Huang, S., and Huang, F. Raise a child in large language model: Towards effective and generalizable fine-tuning. *arXiv preprint arXiv:2109.05687*, 2021.

Xu, Y., Zhong, X., Yepes, A. J. J., and Lau, J. H. Forget me not: Reducing catastrophic forgetting for domain adaptation in reading comprehension. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2020.

Zampieri, M., Nakov, P., Rosenthal, S., Atanasova, P., Karadzhov, G., Mubarak, H., Derczynski, L., Pitenis, Z., and Çöltekin, Ç. Semeval-2020 task 12: Multilingual offensive language identification in social media (offenseval 2020). *arXiv preprint arXiv:2006.07235*, 2020.

Zhang, Y. and Wallace, B. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*, 2015.

Appendix

A. Transfer Learning Experiment Result

Recently, Biswas et al. (Biswas et al.) created anti-Asian COVID-19 related stigma and hate speech dataset. After COVID-19, Asian descent have been facing stigma and hate speech in both offline and online communities, particularly on social networks like Twitter. The aim of this dataset is to detect those stigma and hatespeech. The dataset contains 9289 posts collected from Twitter of which are tweets showing hate speech or stigma towards people of Asian descent. The tweets were collected over the course of six months in 2020 using terms relevant to COVID-19. Two annotators mark the tweets as stigmatizing (positive), non-stigmatizing (negative), or irrelevant (unknown).

We investigate the effect of space model in Covid19 Stigma Dataset. We first create a BERT baseline model for this dataset. For the baseline, we use BERT model & take prediction from the [CLS] token. Then we apply our space model on that dataset in exact same configuration as used for HateXplain 2 class dataset. Our model works better in this case also. The result is shown in Table 4.

Model Name	**F1 Score**
BERT Baseline	0.7510
Space Model	**0.7824**

Table 4: Benchmarking on Covid19 Stigma dataset

As we described above in Section 4, the conceptual space dynamically creates the embedding of conceptual words such as happy, peace so it is expected that our model should be able to give a descent performance if it is applied to another dataset. To claim the fact, we pretrained our space model in the HateXplain dataset. Then we observe the effect of our pretraining model in the Covid19 Stigma dataset. We find that our space model also works well in this case. The result is shown in Table 5.

Approach	**Performance**		
	F1 Score↑	Precision ↑	Recall↑
Transfer Learning	0.549	0.577	0.524
Fine Tuning	0.756	0.776	0.734

Table 5: Performance of Pretrained Space Model on Covid19 Stigma dataset

B. Ablation Study

In the table 6, an ablation study on concept space dimension (or num of concept words in each concept space) is shown. We encountered the model performance for this ablation study. We get a better score when space dim = 128. We use *bert-base-case* as text encoder for this experiment.

Space Dimension	**Performance**		
	Acc.↑	Macro F1↑	AUROC↑
Space Dim 4	0.688	0.682	0.825
Space Dim 10	0.682	0.6744	0.823
Space Dim 32	0.686	0.679	0.824
Space Dim 64	0.692	0.685	0.821
Space Dim 128	**0.701**	**0.693**	**0.826**
Space Dim 256	0.689	0.682	0.823
Space Dim 512	0.694	0.685	0.821

Table 6: Ablation Study for Contextual Space Dimension for HateXplain 3 class Dataset

C. Analysing the Effect of the Losses

We also did an experiment to investigate the impact of different losses on our model. Table 7 presents the results, showcasing the effects of these losses on both model performance and explainability scores.

When using only the classification loss, we observed decent model performance but poor explainability scores. However, incorporating the inter-space loss along with the classification loss significantly improved the explainability scores. The inter space loss with the classification loss help much to boost the explainability scores. But as their no word level loss to direct the space representations so classification along with inter space loss is lower than the classification loss with attention loss. If we add attention loss with classification and inter space loss, we get better scores in both perfomance scores and explainability scores. It is because adding attention loss, the model get a good direction at word level which helps building more better representations for the concept words in the concept spaces. By including attention loss, the model gained valuable word-level guidance, leading to improved concept word representations in the concept spaces.

Loss Name	**Performance**			**Explainability**				
				Plausibility			Faithfulness	
	Acc.↑	Macro F1↑	AUROC↑	IOU F1↑	Token F1↑	AUPRC↑	Comp.↑	Suff.↓
Clf	0.677	0.667	0.821	0.092	0.425	0.409	0.497	0.0302
Clf + Inter SL	0.680	0.673	0.804	0.102	0.447	0.729	0.483	0.0394
Clf + Attn	0.698	0.688	0.819	0.130	0.508	0.871	0.524	0.0321
Clf + Inter SL + Intra SL	0.687	0.678	0.805	0.121	0.473	0.834	0.492	**0.0257**
Clf + Attn + Inter SL	0.688	0.683	0.816	0.126	0.500	0.876	0.519	0.0387
Clf + Attn + Inter SL + Intra SL	**0.701**	**0.693**	**0.826**	**0.133**	**0.515**	**0.881**	**0.538**	0.0355

Table 7: Impact of different losses on 3 Class HateXplain Dataset of our Space Model. We use BERT as text encoder for this experiment and attention explainability. *Clf* means the classification loss where as *Attn* means attention loss and *SL* means the space loss.

However, during our experiments with the (clf + attention loss + inter space loss) combination, we encountered the issue of data collapse. In this scenario, the model generated very similar embeddings for the concept words, resulting in a loss of diversity. If we have T different concept words/ dimensions in a space, we are expecting T different embeddings from that space. As in our model, for getting sentence level representations we considered average space attention score, after

few iterations the space model are generating almost very much similar embeddings for those T concept words. To address this problem, we introduced the inter-space loss. The addition of this loss to the (clf + attention loss + inter space loss) combination yielded significant improvements in both model performance and explainability scores. The effects of these losses are summarized in Table 7.

D. Impact of the Space Losses

In our study, we employ two types of space losses, namely Inter Space Loss and Intra Space Loss, as outlined in Section 4. The Inter Space Loss is employed to ensure orthogonality among the conceptual spaces, while the Intra Space Loss is utilized to address the issue of data collapse. This experiment aims to evaluate the efficacy of these losses in fulfilling their respective objectives. If two concept spaces are orthogonal to each others, then their cosine similarity between them will be -1. If the cosine similarity is towards -1 that means they are trying two orthogonally separable.

For the inter space losses, we calculated mean attention scores for a sentence for all the concept spaces. Then we calculated the cosine similarity between the spaces. Then we average the cosine similarity scores for all sentences in the dataset. Table 8 shows the result for both HateXplain 3 and 2 class dataset

Dataset	**Epoch**	**Average Cosine Similarity**		
		Hate & Non Hate	Hate & Offensive	Offensive & Non Hate
HateXplain 3 Class	1	0.758	0.903	0.808
	2	-0.204	0.029	0.149
	3	-0.186	-0.026	0.090
	4	-0.248	-0.045	0.076
	5	-0.413	-0.205	-0.133
HateXplain 2 Class	1	0.426	-	-
	2	-0.898	-	-
	3	-0.876	-	-
	4	-0.927	-	-
	5	-0.945	-	-

Table 8: Average Cosine Similarity between the Concept Spaces

On the other hand, the Intra Space Loss was employed to address the issue of data collapse by promoting differentiation among concept word embeddings within a concept space. As depicted in Figure 5 and Figure 6, it can be observed that each concept space exhibits distinct clusters, indicating that the dimensionality of the concept word embeddings is not collapsing into a singular region but rather forming separate and distinguishable clusters.

E. Space Dim/ Concept Word Analysis

As previously discussed in Section 4, direct interpretability of concept words is challenging due to the difficulty in extracting words from learnable contextualized embeddings. However, the inverse process, generating embeddings for a given word within the model, can be easily achieved. In order to interpret the concept word embeddings, we identify words whose embeddings closely resemble the concept word embeddings. Subsequently, we employ techniques similar to those used in Topic Modeling to interpret the concept words.

To do this experiment, for each token in a sentence within each concept space, we extracted the index that gives the maximum cosine similarity of the concept word embeddings from the concept embedding matrix. The cosine similarity was calculated between the token's contextual representations and the concept word embeddings.For HateXplain 2 class test dataset, the result of this experiment is shown at the table **??** for hate concept space and table **??** for non-hate concept space.

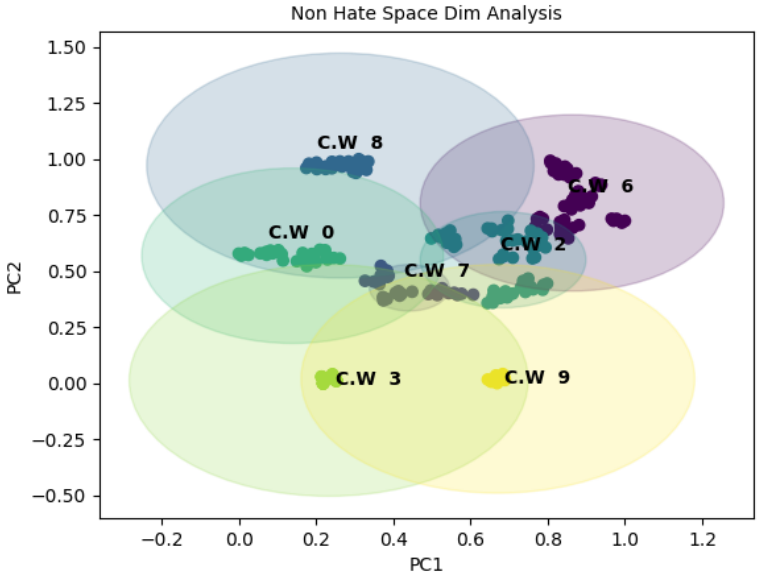

Figure 6: Visualization of Non-Hate Concept Space with Space Dim 10. Here, C.W stands for Concept Words in that space. Those concept words make separable overlapping clusters within that space.

Concept Words No	Word List	Concept Words No	Word List
5	getting, white, listening, dumb, refuge, folks	6	think, touched, folks, voice, lady
6	raped, screaming, disgusting, prison, sex, fuck	7	blame, black, labour, sheep, worry
8	pig, pork, females, fuck, evil	8	tolerate, bitch, kill, independence, keep, neck
9	dutch, jihad, virus, muslims, race, asian	2	fingers, homosexual, arab, love, daughter
2	even, divorce, isabelle, crack, tech	0	trump, stick, real, business, mandela
1	traitor, jews, hispanic, arab, kill	3	text, call, old, fully, "
4	terrorist, gangs, islam, blacks, racism	9	not, hate, dash, under, their
	Hate Concept Words		Non-Hate Concept Words

Figure 7: Word Level Interpretation for the Concept Spaces. We are showing top 5 words for each index.

For each index in the concept word embeddings, we store the corresponding tokens representations from that space. For hate concept space, we store the hate attentions from hate concept space for those tokens that represent the concept words embeddings. So, each tokens have a embedding size of space_dim size. We project the embedding into 2D using t-SNE algorithm for visualization. We followed same procedure for the non-hate space also. Then we find the cluster in those 2D plots. The results are shown in Figure 5 and Figure 6.

In Figure 6 we show the 2d TSNE plot of the projected word representations in the non-hate space and show that they also cluster around one of the conceptual word embeddings. This shows that the model learned the conceptual word embeddings to correctly explain which words in the sentence are attributing to what kind of hate or non-hate. As expected, we found separable clusters for each dimension of both concept space with the intersections.

F. Misclassified Samples

We conducted an analysis on the misclassified samples within the HateXplain 2-class dataset. Our goal was to identify these specific samples. To achieve this, we initially extracted all the samples labeled as non-hate class. Next, we filtered out the sentences from these samples based on two conditions: (1) the hate attention score is higher than the non-hate attention

score, and (2) the rank correlations of the non-hate attention scores are negatively correlated with the ground attention scores. We repeated the same procedure to identify misclassified samples within the toxic class.

For those misclassified sentences we generated VADER (Valence Aware Dictionary and sEntiment Reasoner) (Hutto & Gilbert, 2014) score for checking the ground truth label provided in HateXplain dataset.

Class Label	Space Dimension	Mean Hate Attention	Mean Non-Hate Attention	VADER Score
Toxic	those are called niglets	-0.334	0.242	100% Neutral
	was engels a kike	-0.352	0.296	100% Neutral
	keep waiting sheboon	-0.378	0.364	100% Neutral
Non-Toxic	cheating is not the wave grow up you dumbass nigga	0.271	-0.262	47.4% Negative
	i fucking hate you	0.192	-0.082	57.1% Negative
	same with arabs lol i am arab but white as fuck	0.277	-0.259	30.4% Negative

Table 9: Misclassify Samples with their ground truth label in HateXplain, Mean Hate Attn Score, Mean Non-Hate Attn Score, VADER Score

G. SunBrust Plot

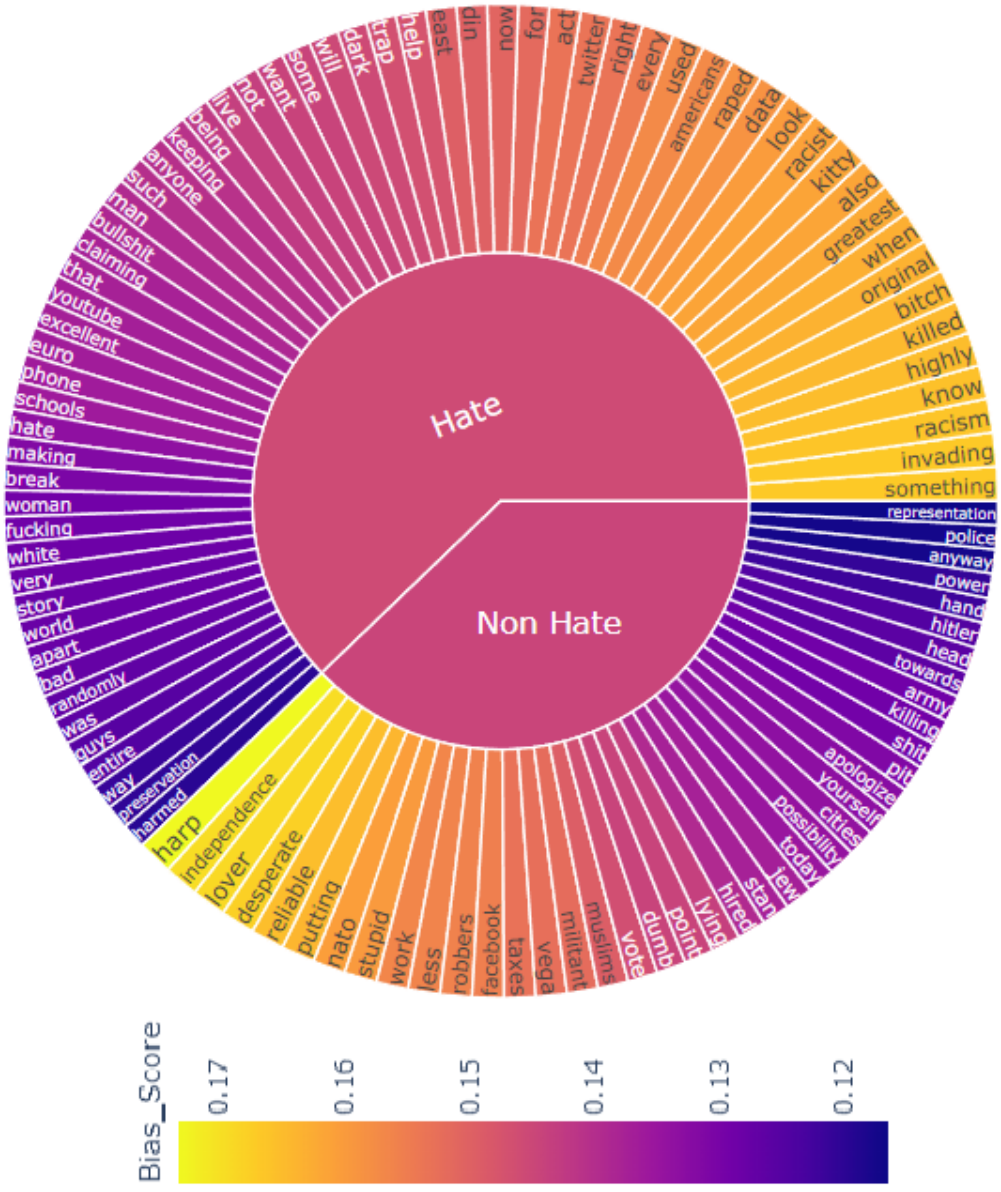

Figure 8: SunBrust Plot for Bias Score in HateXplain 2 cls dataset