HateXplain Space Model: Fusing Robustness with Explainability in Hate Speech Analysis

Md Fahim
Center for Computational & Data Sciences
Independent University, Bangladesh
Dhaka-1229, Bangladesh
fahimcse381@gmail.com

Md Shihab Shahriar
Islamic University of Technology
Gazipur, Bangladesh
shihab1069@gmail.com

Mohammad Ruhul Amin
Fordham University
New York, USA
mamin17@fordham.edu

Abstract

In the realm of Natural Language Processing, Language Models (LMs) excel in various tasks but face challenges in identifying hate contexts while considering zero-shot or transfer learning issues. To address this, we introduce Space Modeling (SM), a novel approach that enhances hate context detection by generating word-level attribution and bias scores. These scores provide intuitive insights into model predictions and aid in the recognition of hateful terms. Our experiments across six hatespeech datasets reveal SM's superiority over existing methods, marking a significant advancement in refining LM-based hate context detection.

1 Introduction

Language Models (LMs) such as BERT [4], RoBERTa [12], and so on, have showcased their prowess in Natural Language Processing (NLP) tasks. They excel in grasping intricate contextual nuances and cultural subtleties within specific languages, proving invaluable in tasks ranging from Text Classification and Text Generation to Question Answering. Despite their remarkable pretraining capabilities, the process of fine-tuning these models for specific tasks has posed challenges. Issues such as the anisotropy problem [6], catastrophic forgetting [16], overfitting issues[11], and the need for interpretability have emerged during this stage. To get rid of those aforementioned problems and retain the generalization capabilities of LMs in the pretraining stage, zero-shot domain adaptation, prompt-based approaches, and other techniques are becoming research interest in the NLP domain.

Though those recent approaches have shown remarkable results in different NLP tasks but faces problem while identifying hate contexts [7]. The discrepancy arises because the datasets used for training the LMs are from the general domain that lacks sufficient hate-related context or contains a limited amount of hate speech and abusive language within its specific domain [9]. Besides, the model explanation is very crucial for this task. Interpretable models like LIME [14], SHAP [13], and so on, are model agnostic and use another independent module to explain model prediction by perturbing the input where implicit model explanation is missing. In our research, we introduce a novel approach named **Space Modeling** (SM) to enhance the precision of hate context detection while maintaining the effectiveness of LMs. Our model not only accurately identifies hate speech but also provides the underlying rationale behind its classification. Through the implicit generation of

This research paper contains discussions of sensitive topics that might be distressing to some readers.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

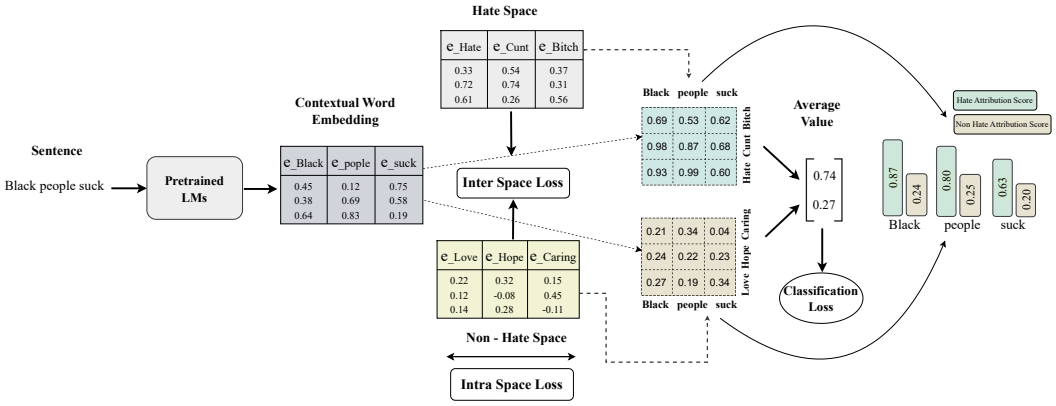


Figure 1: Model Architecture of Space Model. For a example sentence, we will find word level attribution scores for both hate and non-hate class from the class spaces which enhance model explainability

word-level attribution scores for every class label within a sentence, our method offers a more intuitive explanation of model predictions. Additionally, the model computes bias scores for words, enabling the identification of highly hateful or commonplace terms within a specific dataset. Experiment on six different hatespeech datasets shows that our model outperforms zero-shot classification by a huge margin and transfer learning-based approach by a good margin.

2 Method Description

Figure 1 shows the overall model architecture of our Space Model for an example sentence. If we consider a text classification problem having c classes. For a sentence s, after passing it into the pre-trained LM model we will find contextual word embeddings $E = [e_1, e_2, ..., e_n]$ where n is the no. of words and d represents the dimensionality of the embeddings. For a class k we consider m different words which define the class most. If we pass those m words into pre-trained LM and extract their embeddings and concatenate them, we will find a space $S_k \in R^{d \times m}$ for class k. For each class we will find a class space $S_0, S_1, ..., S_{c-1}$. All the S_k is predefined.

2.1 Space Model

We project the contextual word embeddings onto the class spaces using cosine similarity. For a specific class k the projection of the i^{th} word embedding e_i onto the class 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 $||.||$ denotes the L2 norm. If $E_s \in R^{d \times n}$ then the projection of E_s in the class space S_k, we will find a projection matrix $P_{s,k}$ where $P_{s,k} \in R^{m \times n}$ for each class space S_k. Each entry in $P_{s,k}$ matrix defines the cosine similarity between the word embeddings of the sentence and space word embeddings of the class space which indicates the attribution scores of the input words w.r.t space words in that class space. For each word in a sentence, we will get an attribution score for each class by considering the mean of column-wise attribution scores of $P_{s,k}$ matrix.

We define two types of space models, i) **Supervised Space Model** (SSM) and ii) **Semi-Supervised Space Model**(Semi-SSM). SSM replicates the Zero-Shot Text classification techniques where no training is required. We fix the class space S_k. In SSM, for a sentence, we simply pass it into LM and find cosine with the class spaces. The maximum average score of class space defines the class of the sentence. In Semi-SSM, we allow to train the embeddings of class spaces. That means, we only train the vectors of the concept space. To do so, we use classification loss along with the inter and intra space losses. For this variation, for a sentence, we pass it into LM and find cosine with the

class spaces. The average of cosine for each class space is extracted and combined for logits value for classification. Semi-SSM replicates transfer learning techniques where the LMs remain frozen.

2.2 Inter and Intra space loss

During Semi-SSM, intra-space loss and inter-space loss are introduced. The job of inter-space loss is to ensure that the embeddings of the class 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 class 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.

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

To ensure that the embeddings don't converge to the same word embedding within the class 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 $\text{Var}(S_k) = \frac{1}{m}\sum_{i=1}^{m}(w_i - \tilde{w}^2)$ where w_i represents the i-th column of the class space matrix S_k, and $\tilde{w}$ is the mean vector of S_k. Then the intra-space loss for S_k concept space is calculated as: $L_{intra,S_k} = \frac{1}{\text{Var}(S_k)}$ The total intra-space loss is computed as: $L_{intra} = \sum_{k=0}^{k=c-1} L_{intra,S_k}$.

We minimize the total loss given as L in the following equation: $L = L_{CE} + \lambda_1 L_{inter} + \lambda_2 L_{intra}$ where λ_i is a hyperparameter that controls the weight given to the losses.

Dataset	**Experiment**	**Performance Metrics**		
		Accuracy	Macro Precision	Macro Recall
OLID	Zero Shot (No-training)	54.79	46.41	46.97
	SSM (No-training)	64.51	63.59	64.96
	Transfer Learning (Freezing BERT)	66.96	67.87	53.04
	Semi-SSM (Freezing BERT)	67.92	63.65	58.53
Davidson	Zero Shot	64.51	50.30	50.44
	SSM	75.15	62.84	68.30
	Transfer Learning	87.14	83.05	65.49
	Semi-SSM	89.93	84.31	68.24
Founta	Zero Shot	55.66	58.26	58.48
	SSM	78.86	77.69	78.86
	Transfer Learning	90.82	90.61	89.74
	Semi-SSM	86.72	86.91	86.25

Table 1: Model performance of Zero Shot, SSM, Transfer Learning, and Semi-SSM on three different datasets (OLID, Davidson, Founta). The model performance of those models on three more datasets (Degilbert, Elsherif, Vidgen) is reported in Appendix A.3 Table 4.

3 Result and Analysis

To evaluate the performance of our method, we experiment with six different hatespeech datasets. The dataset description is provided in the Appendix A.1. We converted the three-label datasets into binary label datasets for simplicity. We created two different class spaces one for hate and another for non-hate. Eleven different hate words are considered and we get their embeddings from a pre-trained BERT model. By merging them, we create a hate space. We create a non-hate space using a similar

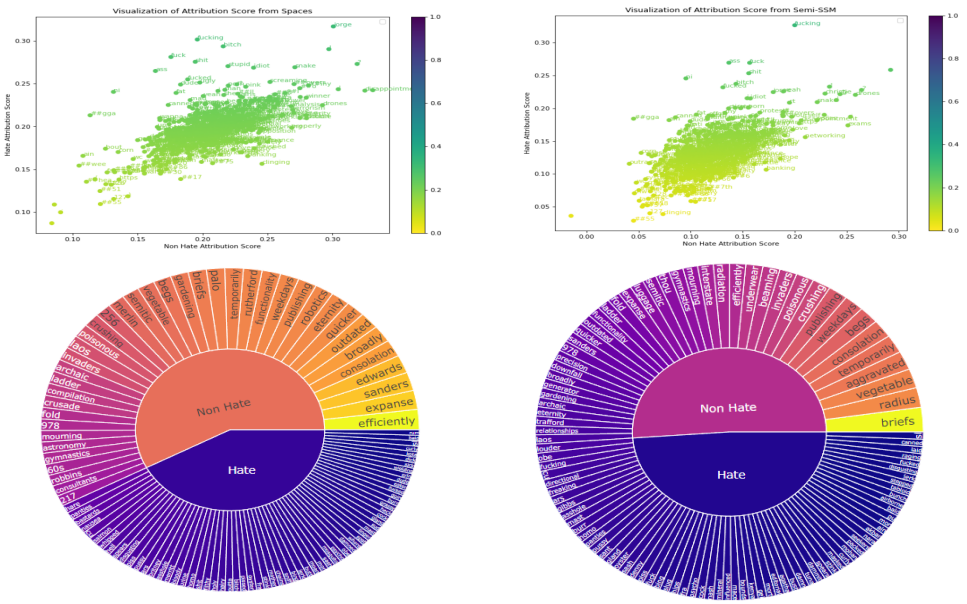


Table 2: Explainability Analysis of the Space Model on the Founta Dataset based on Attribution Score and SunBrust plot for both SSM (left) and Semi-SSM (right) model.

process. More details about the creation of hate and non-hate space can be found in the Appendix A.2.

3.1 Performance Analysis

The performance analysis is shown in Table 1 where four different models were experimented for a dataset. For zero-shot text classification, the performance of LMs is directly evaluated on test data, and no training was involved. Similarly, the SSM model is evaluated where no training is required and the performance on test data is shown. For transfer learning, a trainable classification head is inserted at the top of a BERT model with frozen parameters and the classification head model is trained. In contrast, Semi-SSM also incorporates training the weights of the Space Model only keeping the BERT model parameters frozen.

For each dataset from Table 1, it is seen that our SSM model outperforms zero-shot techniques with an improvement of 10-20% for different performance metrics. In the case of Semi-SSM, it beats transfer learning techniques for four out of six datasets with a good margin. But for the remaining two datasets (Founta & Vidgen), Semi-SSM slightly underperforms. The reason behind this is that the words we choose for creating class spaces may not be a good representative of that class. Furthermore, we are also interested in investigating the effect of losses and different LMs. In Appendix B.1 the effect of loss and in Appendix B.2 the effect of choosing different LMs are reported.

3.2 Explainability Analysis

In our model, for any given sentence, we can get hate attribution and non-hate attribution for every word as we discussed in Section 2.1. In Figure 2 we plot the attribution scores for different words/tokens that we get from the space model while experimenting on the Founta test dataset for both SSM and Semi-SSM. It is common to see that one word/token may repeat several times and get different hate/non-hate attribution scores. In this case, we simply consider an average of those attribution scores for visualization. From that plot, we can see that the hate attribution score is higher than the non-hate one for the hate words like fucking, bitch, shit, idiot, and so on. We also plotted a sunburst plot in Figure 2 for both SSM & Semi-SSM which is based on bias score. The **Bias_Score** for each word is defined as $\text{Bias\_Score}_\omega = \frac{\sum \text{Non-HateAttn}_\omega}{n_\omega} - \frac{\sum \text{HateAttn}_\omega}{n_\omega}$ where n_ω is the

no. of occurrence of word ω in test dataset. Following the equation in 3.2, we refer a word bias to the Non-Hate class if its bias score is greater than 0 otherwise the word is biased to the Hate class. In the sunburst plot in Figure 2, we encountered some offensive words such as fuck, pussy, and asshole in the hate class. More analysis on explainability is reported in Appendix C.

4 Conclusion

In this work, we propose a space modeling technique for detecting hatespeech classification by preserving the generalization capabilities of LMs. Space Model not only gives a boost in the model performance but also introduces an interpretable framework that gives a better intuition about the rationale for a sentence being hate or non-hate. Experimenting on different datasets shows the capability of our model. Space model can be considered for the hatespeech detection in the future.

Future Work

While the current implementation of our model is promising, future research could explore two different aspects. The losses that are used for the space model can be improved further by incorporating manifold or geometric-based losses.

Besides, a dynamic approach for choosing the words that are used for creating the class spaces can be explorable. In the current work, the class spaces are fixed for different datasets. Choosing dataset-specific words for creating class spaces would be a good approach for the space model.

References

[1] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020.

[2] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of the international AAAI conference on web and social media*, volume 11, pages 512–515, 2017.

[3] Ona de Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. Hate speech dataset from a white supremacy forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 11–20, Brussels, Belgium, October 2018. Association for Computational Linguistics.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina 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)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[5] Mai ElSherief, Caleb Ziems, David Muchlinski, Vaishnavi Anupindi, Jordyn Seybolt, Munmun De Choudhury, and Diyi Yang. Latent hatred: A benchmark for understanding implicit hate speech. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 345–363, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

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

[7] Paula Fortuna, Monica Dominguez, Leo Wanner, and Zeerak Talat. Directions for NLP practices applied to online hate speech detection. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11794–11805, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[8] Antigoni Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Twelfth International AAAI Conference on Web and Social Media*, 2018.

[9] Goran Glavaš, Mladen Karan, and Ivan Vulić. XHate-999: Analyzing and detecting abusive language across domains and languages. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6350–6365, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[10] Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021.

[11] Shaoyi Huang, Dongkuan Xu, Ian EH Yen, Yijue Wang, Sung-En Chang, Bingbing Li, Shiyang Chen, Mimi Xie, Sanguthevar Rajasekaran, Hang Liu, et al. Sparse progressive distillation: Resolving overfitting under pretrain-and-finetune paradigm. *arXiv preprint arXiv:2110.08190*, 2021.

[12] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[13] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[14] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " 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*, pages 1135–1144, 2016.

[15] Bertie Vidgen, Tristan Thrush, Zeerak Waseem, and Douwe Kiela. Learning from the worst: Dynamically generated datasets to improve online hate detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1667–1682, Online, August 2021. Association for Computational Linguistics.

[16] Ying Xu, Xu Zhong, Antonio Jose Jimeno Yepes, and Jey Han Lau. Forget me not: Reducing catastrophic forgetting for domain adaptation in reading comprehension. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

[17] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

[18] Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. SemEval-2020 task 12: Multilingual offensive language identification in social media (OffensEval 2020). In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1425–1447, Barcelona (online), December 2020. International Committee for Computational Linguistics.

A Appendix

A.1 Dataset Description

[2] released a dataset of 25k tweets collected via the Twitter API to discern hate speech, offensive language, and normal speech. [8] constructed an 80k-tweet dataset classifying content as abusive, hateful, normal, or spam. [18] introduced the Offensive Language Identification Dataset (OLID), one of the most established datasets for hate speech, encompassing 14k tweets, with 4.5k labeled as offensive. It utilizes a three-level annotation schema for offensive language detection, categorization, and target identification. [15] proposes a human-and-model-in-the-loop process for dynamically generating datasets comprising 40k entries, with 54% identified as hateful, adopting a binary labeling schema to identify the type and target of hate speech. [3] introduced another hatespeech dataset which contains 10k texts where 11% data was hate. [5] created an implicit hate speech dataset collected from Twitter. There were around 20k samples whereas around 5k were implicit hate samples.

To ensure consistency in the dataset labels, we unified the class labels across different datasets. Some datasets had 3 class labels, while others had 2 class labels. To achieve this, we merged the *hate* and

offensive classes into a single *hate* class. Subsequently, we considered normal text as the non-hate class, and the merged class as the hate class. Following this conversion, the dataset sizes are listed in Table 3.

Dataset	**Train**		**Test**	
	Non Hate	Hate	Non Hate	Hate
Davidson	3328	16498	835	4122
Degilbert	7026	746	1755	188
Vidgen	9179	11697	2237	2982
Founta	43148	25624	10703	6491
Elsherief	10617	6567	2674	1622
OLID	7107	3485	1733	915

Table 3: Dataset Size after Converting Multiclass into Binary Class

A.2 Creating Hate and Non-Hate Spaces

For creating a predefined hate space, we considered 11 different hate words. Those words are: ['Moist', 'Cunt', 'Panties', 'Fuck', 'Hate', 'Nigger', 'Pussy', 'Ass', 'Motherfucker', 'Bitch', 'Damn']. The words are chosen from Indy: The most offensive American swear words ranked [1], the ladders most hated words [2]. Those individual words are passed into pretrain LMs (like BERT and so on). Extracting the embeddings from the LMs for those words, we just simply combine them to define hate space.

Dataset	**Experiment**	**Performance Metrics**		
		Accuracy	Macro Precision	Macro Recall
Degilbert	Zero Shot	44.62	53.43	59.37
	SSM	65.72	56.64	67.96
	Transfer Learning	90.37	45.16	50.00
	Semi-SSM	92.31	59.78	65.86
Elsherif-implicit	Zero Shot	46.44	45.69	45.43
	SSM	59.31	56.85	56.91
	Transfer Learning	69.60	68.44	63.39
	Semi-SSM	71.48	69.37	65.84
Vidgen	Zero Shot	44.93	44.26	44.19
	SSM	51.56	50.32	50.33
	Transfer Learning	72.39	71.96	71.08
	Semi-SSM	68.29	67.83	67.47

Table 4: Model performance of SSM and Semi-SSM in three more different dataset.

Similarly, for defining non-hate space, we took 11 different words which are mainly opposite to the hate words. Those words are ['Love', 'Peace','Kindness', 'Happiness', 'Respect', 'Friendship', 'Appreciation', 'Hope', 'Encouragement', 'Support', 'Caring']. We consider the opposite words of the most hated words from Favourite Word Poll [3]. To define the non-hate space, we extract embeddings from the LMs for these specific words and straightforwardly combine them.

[1] https://www.indy100.com/viral/us-worst-swear-words-ranked-b1827546-2656995131

[2] https://www.theladders.com/career-advice/these-are-the-9-most-hated-words-in-the-english-language

[3] https://forreadingaddicts.co.uk/polls-and-discussion/your-top-50-most-hated-words/

A.3 Model Performance on the Other Datasets

Model Performance of different models in Degilbert, Elsherif, and Vidgen datasets are reported in Table 4.

B Ablation Study

Dataset	Experiment	Performance Metrics		
		Accuracy	Macro Precision	Macro Recall
OLID	Semi-SSM	67.92	63.65	58.53
	- Inter Space Loss	66.13	63.28	52.89
	- Intra Space Loss	66.38	63.10	54.29
	- Both losses	64.89	62.83	51.73
Davidson	Semi-SSM	89.93	84.31	68.24
	- Inter Space Loss	88.01	83.68	66.14
	- Intra Space Loss	88.29	84.02	67.23
	- Both losses	87.77	83.12	65.97
Founta	Semi-SSM	86.72	86.91	86.25
	- Inter Space Loss	85.33	82.45	82.86
	- Intra Space Loss	85.90	84.38	85.41
	- Both losses	80.31	81.02	79.62

Table 5: Effect of Losses in Semi-SSM Model

B.1 Effects of Losses

While training the Semi-SSM model, we introduce two different losses for creating separable word embeddings. An experiment was done to find out if the losses are useful for the space model. For this experiment, we consider three different datasets (OLID, Davidson, and Founta). For each dataset, we first excluded inter-space loss from the Semi-SSM model, then intra-space loss was excluded. Finally, another experiment was done without considering both intra and inter-space loss. The results is shown in Table 5. If we drop inter-space loss, the performance of the model is decreased. The same goes for intra-space loss but the drop in the performance is less than the removing inter-space loss. It indicates that inter-space loss is more important for the Semi-SSM model. If we exclude both losses, the performance drops around 2-5% which concludes that both losses are important for our Semi-SSM model.

B.2 Effects of Different LMs

We also experimented with the effect of LMs. We consider the OLID dataset for this experiment. We reported the Macro F1 score for this experiment. Eight different language models [BERT [4] (both cased and uncased versions), RoBERTa [12] (both base and large model), Electra [1], DeBerta [10] (v1 base and v3 base), and XL-Net [17]] were used in this experiment. The results are listed down below:

LMs	Zero Shot	SSM	Transfer Learning	Semi-SSM
bert-base-uncased	46.27	63.23	46.92	67.93
bert-base-cased	38.86	52.02	39.56	60.02
roberta-base	43.35	59.28	49.78	53.96
roberta-large	50.01	60.82	52.56	65.33
electra-base	42.11	54.55	52.02	62.56
deberta base	42.38	57.29	51.26	61.31
deberta -v3 base	47.92	63.37	54.37	65.65
xlnet-base-cased	50.42	64.38	59.39	70.05

Table 6: Macro F1 Score for OLID Dataset. For this experiment we report the macro F1 score.

C Explainability of Space Model

C.1 Nearest Words of Spaces

To do this experiment, for each token in a sentence, we extracted the index (it represents the word embeddings of a word in class spaces) that gives the maximum cosine similarity of the word embeddings from the class space matrix. The cosine similarity was calculated between the token's contextual representations and the hate or non-hate word embeddings in the class spaces. The result of this experiment is shown at the table 7 for hate class space and table 8 for non-hate class space for Semi-SSM model tested on Founta Dataset.

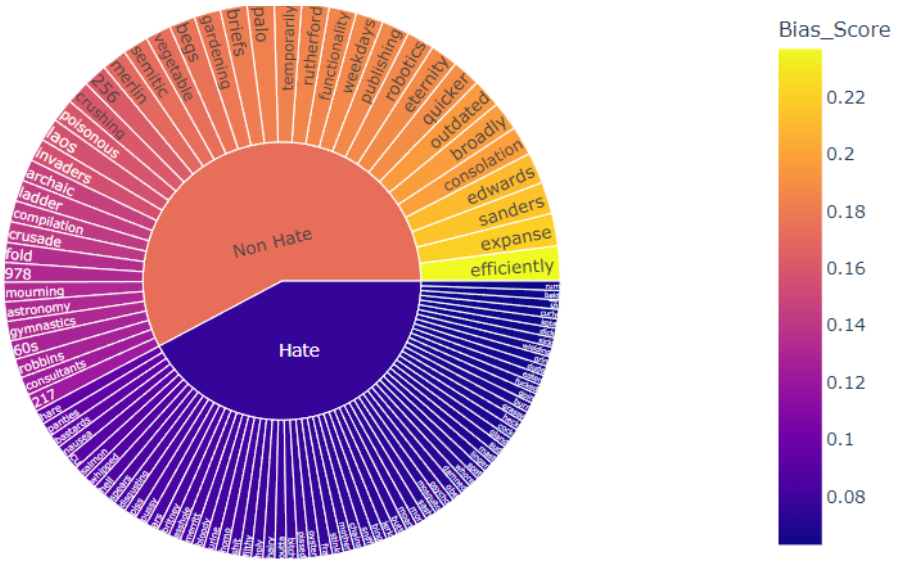
Words from Hate Space	Neighbour Word List
Moist	trump, syria, terrorism, block, islam
Cunt	fucking, steal, drunken, woman, terror
Panties	girls, porn, clips, virginity, outfit
Fuck	fuck, fucked, drugs, fucking, drunk
Hate	disappointment, betrayal, stuck, fake, seldom
Nigger	beat, ruined, muslim, reject, starving
Pussy	corrupt, child, pussy, cruel, clap
Ass	ass, fox, idiots, period, crunch
Motherfucker	ugly, fucked, slams, america, idiot
Bitch	bitch, idiot, flirt, evil, slapped
Damn	dumb, annoying, lost, leaked, damn

Table 7: Neighbouring Word List for Hate Space Words for Semi-SSM Model in Founta Dataset

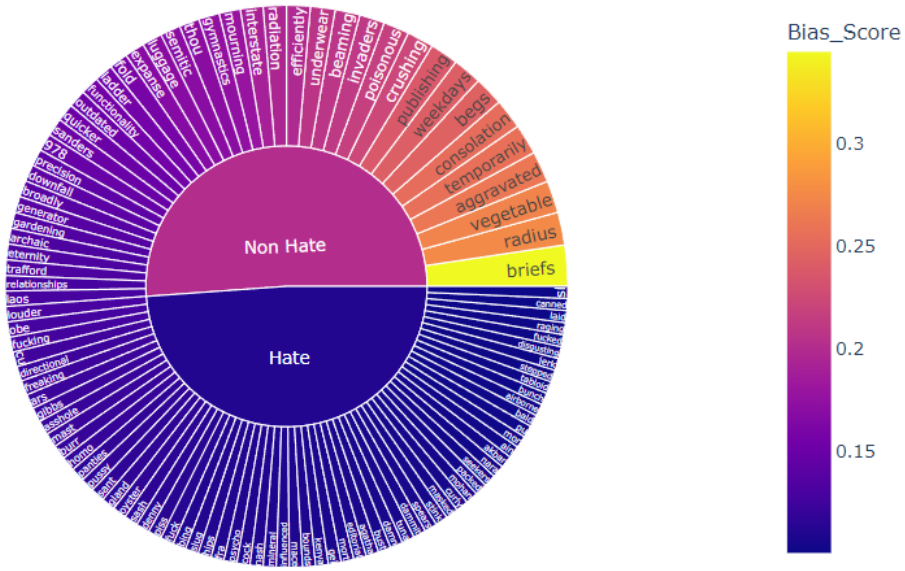
Words from Non-Hate Space	Neighbour Word List
Love	leaders, christian, pink, haven, ass
Peace	peace, syria, republican, democracy, health
Kindness	nice, icy, behavior, honesty, dramatic
Happiness	glad, brain, bachelor, kidding, laughter
Respect	history, willingness, loyal, stupid, honesty
Friendship	friends, members, chat, colleges, partnership
Appreciation	artist, highlights, recognized, photograph, reception
Hope	dude, stacks, wish, candidates, winner
Encouragement	ambitious, america, female, immunity, learned
Support	video, ticket, blocks, banking, finance
Caring	looking, response, reflective, ignoring, charging

Table 8: Neighbouring Word List for Non-Hate Space Words for Semi-SSM Model in Founta Dataset

C.2 SunBrust Plot Analysis



(a) Sunbrust Plot for SSM Model



(b) Sunbrust Plot for Semi-SSM Model

Figure 2: SunBrust Plot for Bias Score on Founta Test Dataset