EDAL: Entropy based Dynamic Attention Loss for HateSpeech Classification

Md Fahim, Amin Ahsan Ali, M Ashraful Amin, A K M Mahbubur Rahman
Center for Computational & Data Sciences
Independent University, Bangladesh
Dhaka-1229, Bangladesh
fahimcse381@gmail.com, {aminali,aminmdashraful,akmmrahman}@iub.edu.bd

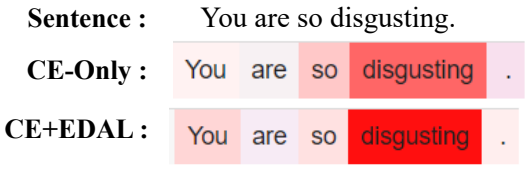
Abstract

Hate speech classification is crucial in addressing harmful content on social media and online platforms. Building robust classifiers poses challenges that include model interpretability and handling dynamic hate speech. We propose the *Entropy based Dynamic Attention Loss* (EDAL) loss to enhance model interpretability by leveraging an additional attention layer. EDAL encourages informative attention scores and improves the model performance of pretrained model during the fine tuning stage of the downstream tasks. Extensive experiments on six diverse datasets validate the effectiveness of EDAL in enhancing classification performance while preserving interpretability. Moreover, experiments on various pretrained models demonstrate that EDAL significantly improves their performance during finetuning. Overall, EDAL shows promise in creating more transparent and reliable hate speech classifiers, fostering a safer online environment.

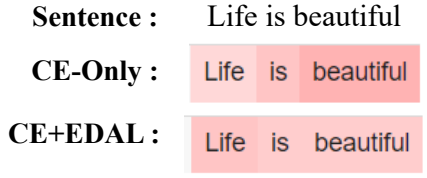
1 Introduction

The rise of harmful and offensive content on social media and online forums has made hate speech classification a crucial task in NLP. Automated systems for detecting and mitigating hate speech play a vital role in fostering safer online spaces. Hate speech classification involves categorizing text into hateful and non-hateful classes. Automating this process helps take timely actions to curb discrimination, hostility, and violence in social space as well as it empowers content moderators and platform administrators.

A fundamental aspect of building robust and reliable hate speech classifiers is the prelude for model explainability. As hate speech classification models are increasingly integrated into real-world applications, there is a growing demand for understanding the decision-making process of these models. Model explainability provides insights into the features and patterns considered by the classifier while making predictions. Therefore, model explainability fosters transparency and trust in the system's decisions. Additionally, explainable models enable stakeholders to identify potential biases and make informed decisions about content moderation.



(a) Attention Heatmap Visualization for Hate Sentence



(b) Attention Heatmap Visualization for Non-hate Sentence

Figure 1: Attention Heatmap Visualization of Sample Sentences for Both Losses

Language models, particularly large-scale pretrained models like BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) etc. have significant impact on the field of NLP. These Language Models (LMs) have shown remarkable performance across various tasks including hate speech classification. However, building effective classifiers faces challenges such as handling dynamic hate speech, addressing class imbalance, and ensuring model interpretability. The opaque nature of these models makes it challenging to comprehend the rationale behind their predictions, thus limits their applicability in sensitive domains like hate speech classification.

In this research paper, we introduce an intuitive approach to enhance the interpretability of hate speech classifiers using the *Entropy-based Dynamic Attention Loss* (**EDAL**) technique. EDAL

uses an additional attention layer that measures the the attention scores for each of the words of a sentence. In this way EDAL gives better explainability. EDAL loss addresses the issue of model opacity by encouraging the attention scores of the classifier to be more informative and aligned with the specific linguistic features that contribute to hate speech classification. By leveraging the EDAL loss, we aim to achieve higher model interpretability without compromising the classification performance.

In Figure 1, heatmaps illustrating the attention scores for a hate sample sentence and a non-hate sample sentence are presented. In the heatmap, higher attention scores are represented by a deeper shade of red. Notably, when applying EDAL loss to a hate sentence, the model allocates increased attention to the toxic spans compared to the Cross Entropy (CE) approach. On the other hand, for the non-hate one, EDAL provides almost similar attention to each of the tokens and attention is more refiner than CE loss. This observation signifies that the EDAL loss enhances the model's focus on specific aspects relevant to hate speech identification.

In our study, we have extensively demonstrated the efficacy of the EDAL technique across six diverse datasets. For each dataset, we integrated the EDAL loss alongside the CE loss during finetuning that results in notable improvements in the performance metrics of pretrained language models. The impact of EDAL was observed across various evaluation criteria that further validated its effectiveness in enhancing model capabilities and overall classification performance. This comprehensive analysis showcases the versatility and robustness of the EDAL loss that makes it a promising idea for improving pretrained language models during finetuning across different domains and datasets. We show the model performance and attention score measurement for detecting hatespeech text. The EDAL loss can easily be extended to any text classification tasks.

2 Related Work

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, 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 4K are identified as hate speech, 5.5K as offensive, and 7.8K as normal. 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 introduce an attention loss with their ground truth attentions.

(Cao et al., 2019) introduces an attention loss for small sample image classification tasks. This attention loss focuses on the misclassified samples and incorporates a soft label approach to handle confused categories. (Li et al., 2020) propose the use of dice loss as an alternative to the standard cross-entropy objective for data-imbalanced NLP tasks. (Zayed et al., 2023) investigate the relationship between the entropy of the attention distribution and the model's performance and fairness. They introduce a method called Entropy-based Attention Temperature scaling (EAT) to modulate the entropy of the model's attention maps. This modulation is achieved through temperature scaling after training, and the hyperparameter β is chosen based on the validation set to strike a balance between performance and fairness.

Recent work by (Attanasio et al., 2022) also introduces an entropy based attention regularization techniques for mitigating bias the unintended biases. In that work, they used internal attention of BERT layers to discourage overfitting to training-specific terms. For a layer of BERT, they found mean attention of positions for different heads and passed the attention into softmax operator for finding entropy for that layer. A sum of entropy from different layers of BERT were used with the main classification loss.

3 Methodology

3.1 Finding Contextual Representations

After the input sentence S undergoes tokenization, resulting tokens: $\{x_1, x_2, \ldots, x_n\}$ are fed into a text encoder such as LSTM (Hochreiter and Schmidhuber, 1997) or BERT (Devlin et al., 2019). This process aims to find contextual representations $H = \{h_1, h_2, \ldots, h_n\}$ where h_i is the contextual representation of x_i.

For transformer-based text encoders, specific to-

kenization is tailored to the model's architecture. In contrast, for non-transformer models, an embedding layer is employed to derive the token embeddings.

3.2 Additional Attention Layer

Once the contextual representations H are obtained, we proceed to incorporate an additional attention layer. The primary purpose of introducing this layer is to compute learnable attention scores for each token. For the token x_i, the attention score is denoted as α_i, and its calculation is as follows:

$$\alpha_i = \textit{softmax}(W \cdot h_i + b), \\ i = 1, 2, \ldots, n$$

For each sentence, we calculate an attention score for each token within that sentence. This results in a set of *attention_scores* $= \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ corresponding to the tokens in the given sentence.

These attention scores collectively represent the overall attention distribution across the sentence indicating the relative importance or relevance of each token to the context of the entire sentence.

3.3 Classification

After finding attention scores for each tokens, we find the context vector for the sentence S by multiplying the contextual representations of token x_i with its attention score α_i.

$$c = \sum_{i=1}^{T} \alpha_i \cdot h_i$$

After obtaining the context vector, it is further processed through a linear layer to perform the classification task. This linear layer is for mapping the contextual information into the appropriate output format for the specific classification objective.

$$z = W_c \cdot c + b_c$$

3.4 EDAL Loss

In the binary hate speech classification problem, we designate the target class 0 to represent the *non-hate* category, while the other class serves as the *hate* class that will be our positive or interest class. When a sentence belongs to the target class 0 (non-hate), it indicates that there are no specific words or word-spans within the sentence that are responsible for expressing hate. Consequently, the model should assign similar attention scores to all the words in that sentence.

```
def edal_loss(attention_scores, target):
"""
attention_socres: shape (B, L, 1)
target: shape (B, 1)

"""
# Compute the entropy of the distribution
entropy = -torch.sum(attention_scores * torch.log(
                                attention_scores), dim=1)

# Apply different objectives based on the target class
class_p_indices = (target == 0).nonzero(as_tuple=True)[0]
class_not_p_indices = (target != 0).nonzero(as_tuple=True)[0]

entropy_p_loss, entropy_not_p_loss = 0.0, 0.0

if len(class_p_indices) > 0:
    # Maximize entropy for class 'p' or 0
    entropy_p_loss = - torch.mean(entropy[class_p_indices])

if len(class_not_p_indices) > 0:
    # Minimize entropy for other classes
    entropy_not_p_loss = torch.mean(entropy[class_not_p_indices
                                ])

# Compute the overall loss
edal_loss = entropy_p_loss + entropy_not_p_loss

return edal_loss
```

Figure 2: Pytorch Code for EDAL Loss calculation

That means, we aim to achieve a uniform distribution of attention scores for class 0, indicating that all words in non-hate sentences should receive similar attention from the model. On the other hand, for class 1 (hate), we expect a non-uniform distribution of attention scores as certain words or word-spans play a significant role in determining hate speech. We know that a uniform distribution has maximal entropy[1] meaning that all elements are equally likely.

To enforce this behavior in the model, we introduce the **Entropy-based Dynamic Attention Loss (EDAL Loss)**. This loss function is designed to encourage the model to produce attention distributions with high entropy for class 0 and non-uniform distributions for class 1. By optimizing the EDAL Loss, we ensure that the model pays consistent attention to all words in non-hate sentences while focusing on the most relevant words for hate speech classification.

To do that, first, we calculate the entropy of the attention scores obtained from the attention layer in Section 3.2 for a batch with size B. If the attention scores for each batch are represented as $\mathbf{A} = B \times L$, where L is the *sentence_length*, then A_i denotes the attention scores for the i-th sample in the batch given by $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$.

[1] https://stats.stackexchange.com/questions/66108/why-is-entropy-maximised-when-the-probability-distribution-is-uniform

The entropy of $\mathbf{A}_i$ is calculated as:

$$\textit{Entropy}(\mathbf{A}_i) = -\sum_{j=1}^{n} \alpha_j \log(\alpha_j)$$

If the target class is 0 (let's denote it as class p), the goal is to maximize the entropy of the attention scores for class p. This is achieved by averaging the negative entropy values for the samples in class p:

$$\textit{Entropy}_p(\mathbf{A}) = -\frac{1}{N_p} \sum_{i \in \text{class } p} \textit{Entropy}(\mathbf{A}_i)$$

where N_p is the number of samples in class p, and $\mathbf{A}_i$ is the entropy of attention scores for the i-th sample in class p.

On the other hand, if the target class is not p, the goal is to minimize the entropy of the attention scores for other classes. This is achieved by averaging the entropy values for samples in classes other than p. Let $N_{\text{not } p}$ be the number of samples in classes other than p:

$$\textit{Entropy}_{\text{not } p}(\mathbf{A}) = \frac{1}{N_{\text{not } p}} \sum_{i \in \text{class not } p} \textit{Entropy}(\mathbf{A}_i)$$

If target represents the ground-truth target class indices, the overall *EDAL* loss is the sum of the above two entropy terms:

$$\mathcal{L}_{EDAL}(\mathbf{A}, \text{target}) = \textit{Entropy}_p(\mathbf{A}) + \textit{Entropy}_{\text{not } p}(\mathbf{A})$$

EDAL can easily be extended to multi-class classification problem. A pytoch like code for calculating *EDAL* is provided in Figure 2. The final loss for hatespeech calculation is a combination of *Cross Entropy* based loss and EDAL base loss:

$$\mathcal{L} = \mathcal{L}_{CE} + \lambda * \mathcal{L}_{EDAL}$$

4 Experimental Setup

4.1 Dataset

To evaluate the performance of *EDAL* loss, we experiment with six different hatespeech dataset. (Davidson et al., 2017) released a dataset of 25k tweets collected via the Twitter API to discern hate speech, offensive language, and normal speech. (Founta et al., 2018) constructed an 80k-tweet dataset classifying content as abusive, hateful, normal, or spam. (Zampieri et al., 2020) 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. (Vidgen et al., 2021) 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. (de Gibert et al., 2018) introduced another hatespeech dataset which contains 10k texts where 11% data was hate. (ElSherief et al., 2021) created an implicit hate speech dataset collected from Twitter. There were around 20k samples whereas around 5k were implicit hate samples.

Dataset	**Train**		**Test**	
	Class 0	Class 1	Class 0	Class 1
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 1: Dataset Size after Preprocessing

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 (Class 0), and the merged class as the hate class (Class 1). Following this conversion, the dataset sizes are listed in the Table 1.

4.2 Preprocessing & Settings

In each dataset, we performed some essential text preprocessing steps including removing punctuation, emojis, and URLs, if any. For obtaining the hidden representations from the text, we utilized two different models: LSTM and BERT as the text encoders.

When using the LSTM-based model, an embedding layer with an embedding dimension of 128 was employed to convert the tokens into vector rep-

resentations. The LSTM model's hidden dimension was set to 256. We used a learning rate of 0.005 and a batch size of 64 for this configuration.

On the other hand, for the BERT model, we utilized the *bert-base-cased* variant that enables us to extract contextual representations through fine-tuning. In this case, the hidden dimension of the BERT model was set to 768. The learning rate for BERT was 2×10^{-5}, and a batch size of 32 was used.

Both configurations employed the AdamW optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.99$. To ensure robustness, we performed five-fold cross-validation and three different random seeds. Additionally, we set $\lambda = 10$ for all experiments. An ablation study investigating the effect of different λ values is presented in Table 3. All experiments were conducted using Python (version 3.8) and PyTorch, leveraging the free NVIDIA Tesla K80 GPU available in Google Colab, as well as a single NVIDIA Tesla P100 GPU provided by Kaggle.

4.3 Evaluation Metrics

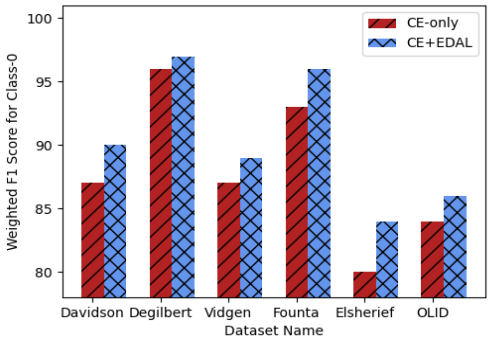
To compare the model performance on the predictions, we use the following performance-based metrics:

- **Accuracy**: This metric measures the proportion of correctly classified samples over the total number of samples.
- **Weighted F1 Score**: The weighted F1 Score takes into account the class imbalance by computing the F1 Score for each class and then weighting the scores by the number of samples in each class.
- **ROC-AUC**: The ROC-AUC (Receiver Operating Characteristic - Area Under the Curve) metric evaluates the model's ability to distinguish between positive and negative samples across different probability thresholds.

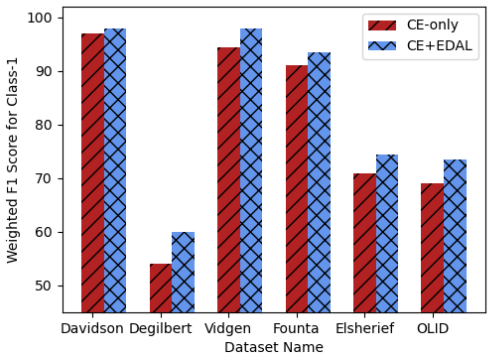
5 Results and Analysis

5.1 Model Performance with EDAL Loss

The performance comparison of the BERT model fine-tuned with both cross-entropy (CE) loss and *EDAL* loss across six different datasets is presented in Table 2. We reported the performance of the models in test dataset in that table. For each dataset, the BERT model trained with both losses exhibits superior performance compared to the one trained solely with CE loss. Notably, the proposed *EDAL* loss consistently yields a substantial improvement of 1-3% in accuracy and ROC-AUC for all datasets. Similarly, in terms of weighted F1 score, *EDAL* results in a notable improvement of 2-4%.



(a) F1 Score for Class-0 Predictions



(b) F1 Score for Class-1 Predictions

Figure 3: Experiment on classwise F1 score

These findings indicate that *EDAL* effectively enhances the model's attention mechanism leading to improved performance across various evaluation metrics. To further illustrate this, we report the class-wise F1 scores for all datasets in Figure 3. The figure demonstrates that *EDAL* significantly boosts the model's performance, particularly for class 0, surpassing the performance of the model fine-tuned with only CE loss by a significant margin.

As observed from Table 2, it is evident that *EDAL* shows limited effectiveness when applied to LSTM-based models. In comparison to CE loss, the performance of *EDAL* is relatively lower. The reason behind this disparity lies in the fact that, for LSTM models, the word embeddings are trained from scratch, which hinders *EDAL*'s ability to precisely determine which words or spans of words should receive higher attention. In contrast, BERT

Dataset	Experiment	Performance Metrics		
		Accuracy	F1 Score	ROC-AUC
Davidson	LSTM	92.8	87.1	96.6
	LSTM + EDAL	91.4	85.1	94.8
	BERT	96.3	96.2	97.1
	BERT + EDAL	**97.2**	**98.4**	**98.8**
Degilbert	LSTM	87.9	58.4	71.8
	LSTM + EDAL	90.4	60.7	76.1
	BERT	92.6	91.9	93.2
	BERT + EDAL	**93.3**	**93.1**	**94.2**
Vidgen	LSTM	76.9	76.3	77.5
	LSTM + EDAL	73.8	73.8	78.9
	BERT	87.5	91.6	94.8
	BERT + EDAL	**87.9**	**94.1**	**96.1**
Founta	LSTM	91.4	92.7	92.9
	LSTM + EDAL	92.0	91.4	95.0
	BERT	93.9	92.2	96.9
	BERT + EDAL	**95.3**	**94.9**	**97.8**
Elsherif-implicit	LSTM	71.1	69.2	76.2
	LSTM + EDAL	70.8	68.7	75.9
	BERT	79.3	76.6	84.4
	BERT + EDAL	**82.7**	**80.2**	**87.5**
Olid	LSTM	68.6	63.8	65.4
	LSTM + EDAL	70.9	64.9	68.5
	BERT	79.5	78.8	84.8
	BERT + EDAL	**80.3**	**80.5**	**86.6**

Table 2: Model performance of CE and CE+EDAL loss in six different Hatespeech dataset based on three different evaluation metrics. For each dataset, CE+EDAL loss with BERT model outperforms the other approach.

models benefit from pre-trained word embeddings enabling *EDAL* to function more effectively. *EDAL* loss helps the pretrained model for getting better performance during finetuning in a custom dataset.

5.2 Effects of λ and Batch Size

Our proposed method incorporates a hyper-parameter λ that determines the relative contribution of the *EDAL* loss to the overall loss function. To assess the impact of different λ values, we conducted an experiment on the OLID dataset. The model performance, specifically class-wise F1 scores on the OLID test dataset is reported in Table 3 for various λ values. Our observations indicate that both lower and higher values of λ hinder the model's predictive capabilities. However, an optimal value of λ can significantly enhance the model's performance.

λ	**Class-0 F1**	**Class-1 F1**
0.1	85.1	66.2
0.5	85.7	67.8
1	85.3	68.3
10	85.5	67.3
100	84.7	66.4

Table 3: Ablation Study of λ in OLID Dataset

The choice of batch size in deep learning can have a notable impact on the model's loss function and overall performance. To investigate this effect,

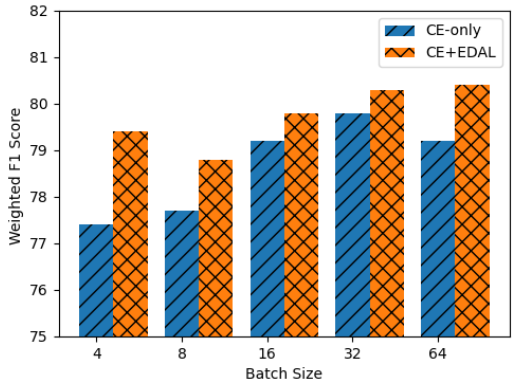


Figure 4: EDAL Performance on the variation of batch size

we conducted an experiment on the OLID dataset using CE loss and CE+EDAL. We measured the weighted F1 score on the test dataset with different batch sizes and the results are depicted in Figure 4. When using only CE loss, the model's performance exhibits fluctuations with varying batch sizes. On the other hand, when EDAL loss is combined with CE, the performance remains relatively stable across different batch sizes. This observation suggests that EDAL loss is less affected by changes in batch size compared to CE loss.

The graph in Figure 4 indicates that EDAL loss provides a consistent improvement in performance regardless of the batch size used. This robustness to batch size variations is advantageous as it allows the model to achieve reliable results without the need for fine-tuning the batch size.

5.3 Experiments with LLMs

As mentioned in Section 5.1, the EDAL proves to be beneficial for pretrained models during fine-tuning. To demonstrate this, we conducted an experiment on the OLID dataset involving eight different Large Language Models (LLMs). The detailed results are presented in Table 6 in Appendix A. From the table, it is evident that EDAL consistently outperforms CE loss for each of the pretrained models except for the *deberta-base* model, where the difference is relatively small. The substantial margin of improvement in performance achieved by EDAL highlights its effectiveness in enhancing model capabilities during fine-tuning.

For these experiments, we set the hyperparameter λ to 10 and used a batch size of 16 across all models. However, it is important to note that model-specific hyperparameter optimization could potentially further enhance the performance of EDAL leading to even better results.

5.4 Attention Score Analysis

Another experiment was conducted to measure the effectiveness of EDAL Loss in word-level attention. For this experiment, we selected the OLID dataset. As mentioned earlier, for non-hate sentences, the attention scores should be uniform. Following the approach taken by HateXplain(Mathew et al., 2021), we considered that the attention score for each token in a non-hate sentence should be $1/sentence\_length$. We then measured the distance between the predicted attention scores and $1/sentence\_length$.

Given a sentence s with n tokens, we have the predicted attention values for each word, denoted as α_i, where i is the index of the word in the sentence. We want to measure the difference between the predicted attention of each word and $1/n$, and then find the average difference for each word in the sentence. The predicted attention values for the sentence are represented as the list $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_n]$, and the attention value for each word can be accessed as $\alpha[i]$. The average difference for each word in the sentence is calculated as follows:

$$\text{Average\_Difference}(s) = \frac{1}{n}\sum_{i=1}^{n} |\alpha_i - \frac{1}{n}|$$

where $|\cdot|$ represents the absolute value. We calculate the average difference score for all the sentences in test dataset for both hate (class-1) labelled sentence and non-hate (class-0) labelled sentence. For hate labelled sample this score should be close to zero and for the other one the score should be far from zero.

Loss	Class-0 ↓	Class-1 ↑
CE	0.051	0.1722
CE + EDAL	0.025	0.1987

Table 4: Attention Score Difference Measurement

The analysis of Table 4 reveals that the inclusion of EDAL loss has notably improved the alignment of the model with the desired behavior. Specifically, in the non-hate class, the EDAL loss effectively guides the attention scores towards values close to zero while in the hate class, the same loss function encourages the attentions to diverge further

Task	Dataset	Experiment	Accuracy ↑
Sentiment Classification (Bangla)	SentNoB	BanglaBERT	74.46
		BanglaBERT+EDAL	75.59
Sentiment Classification (English)	IMDB	BERT	84.68
		BERT+EDAL	87.04
Intended Sarcasm (English)	iSarcasm	BERT	86.23
		BERT+EDAL	87.97
Fake News Detection (Bangla)	BanFake	BanglaBERT	96.65
		BanglaBERT+EDAL	98.43
Question Classification (English)	TREC	BERT	97.63
		BERT+EDAL	98.72

Table 5: Extending EDAL for different text classification task for both Bangla and English Laguage

from zero. In comparison to the model with only CE loss, the utilization of EDAL loss achieves a better balance in attention allocation, contributing to a more refined and contextually-aware model performance.

5.5 Extending EDAL for Any Classification Problem

EDAL can be easily adapted to any text classification problem. Table 5 showcases the performance of EDAL across various text classification tasks. In this experiment, we employed the Bangla Sentiment Classification dataset named SentNoB (Islam et al., 2021) and the Bangla Fake News Classification dataset called BanFakeNews (Hossain et al., 2020). For these tasks, we utilized BanglaBERT (Bhattacharjee et al., 2022) with Cross Entropy (CE) loss and BanglaBERT with both CE and EDAL. Additionally, we employed the IMDB Movie Review dataset (Maas et al., 2011), the Question Classification dataset TREC (Li and Roth, 2002), and the Intended Sarcasm Detection dataset named ISarcasm (Oprea and Magdy, 2020) for the English language, using the BERT model.

From Table 5, it is evident that EDAL consistently outperforms vanilla BERT across different classification datasets and languages. This demonstrates the versatility of EDAL, making it applicable to any language and text classification problem.

6 Conclusion

In this paper, we introduce the Entropy based Dynamic Attention Loss (EDAL loss) to address the challenges of model interpretability and to enhance classifier transparency. EDAL loss is calculated by using an additional attention layer. Through extensive experimentation, we demonstrate the effectiveness of EDAL in improving classification performance while maintaining a clear understanding of the decision-making process. The attention scores are more refined with the EDAL loss that helps the pretrained model for detecting the hate content. Our research emphasizes the necessity of model interpretability and highlights EDAL as a promising solution for building robust and transparent hate speech classifiers. Importantly, EDAL's utility extends beyond hate speech and can applied to various NLP tasks.

Acknowledgements

This project has been jointly sponsored by Independent University, Bangladesh and the ICT Division of the Bangladesh Government.

References

Giuseppe Attanasio, Debora Nozza, Dirk Hovy, and Elena Baralis. 2022. Entropy-based attention regularization frees unintended bias mitigation from lists. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1105–1119, Dublin, Ireland. Association for Computational Linguistics.

Abhik Bhattacharjee, Tahmid Hasan, Wasi Ahmad, Kazi Samin Mubasshir, Md Saiful Islam, Anindya Iqbal, M. Sohel Rahman, and Rifat Shahriyar. 2022. BanglaBERT: Language model pretraining and benchmarks for low-resource language understanding evaluation in Bangla. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1318–1327, Seattle, United States. Association for Computational Linguistics.

Jie Cao, Yinping Qiu, Dongliang Chang, Xiaoxu Li, and Zhanyu Ma. 2019. Dynamic attention loss for small-sample image classification. In *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 75–79. IEEE.

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

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. 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.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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. Association for Computational Linguistics.

Mai ElSherief, Caleb Ziems, David Muchlinski, Vaishnavi Anupindi, Jordyn Seybolt, Munmun De Choudhury, and Diyi Yang. 2021. 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. Association for Computational Linguistics.

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

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

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

Md Zobaer Hossain, Md Ashraful Rahman, Md Saiful Islam, and Sudipta Kar. 2020. BanFakeNews: A dataset for detecting fake news in Bangla. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2862–2871, Marseille, France. European Language Resources Association.

Khondoker Ittehadul Islam, Sudipta Kar, Md Saiful Islam, and Mohammad Ruhul Amin. 2021. SentNoB: A dataset for analysing sentiment on noisy Bangla texts. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3265–3271, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. 2020. Dice loss for data-imbalanced NLP tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 465–476, Online. Association for Computational Linguistics.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

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

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2021. Hatexplain: A benchmark dataset for explainable hate speech detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14867–14875.

Silviu Oprea and Walid Magdy. 2020. iSarcasm: A dataset of intended sarcasm. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1279–1289, Online. Association for Computational Linguistics.

Bertie Vidgen, Tristan Thrush, Zeerak Waseem, and Douwe Kiela. 2021. 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. Association for Computational Linguistics.

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

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. 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). International Committee for Computational Linguistics.

Abdelrahman Zayed, Goncalo Mordido, Samira Shabanian, and Sarath Chandar. 2023. Should we attend more or less? modulating attention for fairness. *arXiv preprint arXiv:2305.13088*.

A Model Experiment

We also experiment the effectiveness of our EDAL loss in different pretrained models while finetuning them in OLID dataset. Eight different language models [BERT (Devlin et al., 2019) (both cased and uncased versions), RoBERTa (Liu et al., 2019) (both base and large model), Electra (Clark et al., 2020), DeBerta (He et al., 2021) (v1 base and v3 base), and XL-Net (Yang et al., 2019)] were used in this experiment. The results are listed down below:

Model Name	**Loss**	**Performance Metrics**		
		Accuracy	F1 Score	ROC-AUC
bert-base-cased	CE	79.5	78.8	84.8
	CE + EDAL	80.3	80.5	86.6
bert-base-uncased	CE	78.8	77.6	84.0
	CE + EDAL	79.5	79.1	85.8
roberta-base	CE	79.4	79.5	86.5
	CE + EDAL	81.2	80.1	87.1
roberta-large	CE	79.2	79.3	85.5
	CE + EDAL	80.4	80.6	86.2
electra-base	CE	79.5	78.4	85.2
	CE + EDAL	81.4	80.7	87.5
deberta-base	CE	80.1	79.8	86.5
	CE + EDAL	80.0	79.3	86.3
deberta-v3-base	CE	80.7	80.1	86.1
	CE + EDAL	81.5	81.7	87.6
xlnet-base	CE	80.0	80.0	86.6
	CE + EDAL	81.5	81.8	87.3

Table 6: Effects of Different Models in OLID Dataset