Journal of Industrial Information Technology and Application
Vol.10 No. 02 *ISSN(Online): 2586-0852*

AI for Digital Well-Being: Real-Time Moderation of Harmful Online Content

*C. Kalpana, Manoj Devare and Bhagya P Bijay Kumar**

Institute of Information Technology Amity University Mumbai, India

Abstract. The platforms for online communication have expanded rapidly. These are creating new opportunities for interaction but also increasing the chances of offensive and harmful content. These types of messages affect the mental health of the user, which also affects the social well-being of the individual user. This shows the importance of strategies for moderating such issues. In this work, we present a real-time framework for detecting and blocking harmful text using Natural Language Processing (NLP) and machine learning techniques. The text is represented with Term Frequency–Inverse Document Frequency (TF–IDF) features and classified using Random Forest, Support Vector Machine, and XGBoost algorithms. From our experiments and comparison the tuned XGBoost model performed the best, reaching 90% accuracy with an F1-score of 0.75. The per-sample prediction latency was of only 0.004 seconds.

Keywords; AI based message detection, social well-being, Machine Learning, Natural Language Processing, TF-IDF

Cite this paper as : C. Kalpana, Manoj Devare, and Bhagya P Bijay Kumar (2026) "AI for Digital Well-Being: Real-Time Moderation of Harmful Online Content", Journal of Industrial Information Technology and Application, Vol. 10. No. 02, pp. 1308-1320

* Corresponding author : ckalpana@mum.amity.edu
Received: Jan. 6. 2026 Accepted: Feb. 19. 2026 Published: Jun. 30. 2026

1. Introduction

The rapid expansion of digital communication platforms such as social media, online forums, and instant messaging applications has changed the way how individuals interact and share information. Even though these platforms promote a global way for communication, openness, and help in community building, they have equally big problems like the spread of toxic and harmful content. Toxic comments, including hate speech, abusive language, and personal attacks, these types of messages not only degrade the quality of the discussions but also cause various problems for the users, like severe mental health issues and psychological harm [1], [2]. Left unmoderated, such content may escalate into cyberbullying, misinformation campaigns, and even offline conflicts. If not moderating such content, it may lead to an increase in online conflicts, may cause misinformation, and may even cause cyberbullying.

In traditional moderation approaches, it included methods such as manual review or rule-based keyword filters. These are insufficient for modern digital environments, as they generate millions of user-generated comments every second. Manual moderation of these will be slow and not efficient, while keyword filters can fail to capture the actual context, or the sarcasm, and the evolving nature of the toxic expressions. Therefore, there is an increasing need for systems that can accurately and efficiently detect toxicity in real time.

This research presents a real-time toxicity detection framework that uses Natural Language Processing (NLP) and machine learning models. Unlike many existing offline studies, our approach focuses on the low-latency prediction, making it suitable for live chat and interactive platforms. We will use a TF–IDF representation of text and then compare several tuned models, including Random Forest, Support Vector Machines, and XGBoost after tuning. After these comparisons, the best-performing model is selected and deployed through a Streamlit-based application. This will ensure immediate feedback to the users and moderators. This real-time mechanism is designed to make and promote healthier digital environments by reducing harmful content before it spreads further.

2. Related work

Previous research was done for detecting the harmful content, using very simple statistical models to even advanced deep learning architectures; in these studies, the majority relied on classical machine learning approaches such as logistic regression and naïve Bayes, combined with the bag of words or using tf-idf features. These resulted in a reasonable accuracy, but they lacked in understanding the context, like sarcastic expressions of the toxicity.

In some of the research, as deep learning evolved, some of the techniques, like Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM), and Convolutional Neural Networks (CNNs), were used by many for classifying the texts, including hate speech, and detecting the cyberbullying. Such models improved the performance and also found the contextual patterns in the language. Recently, transformer-based models like BERT and its extensions have also been used for some studies to classify this toxic text, as they have the ability to encode semantic context at scale. But these are highly computational and are less suitable for real-time applications where latency is important.

There are various benchmark datasets created to support the toxicity research. The Wikipedia Detox dataset and the Jigsaw Toxic Comment Classification Challenge dataset are among the most widely used. These have a large collection of annotated use comments. Studies on the basis of these datasets confirm that even though deep learning models achieve higher accuracy, lightweight classical algorithms provide a balance between speed and accuracy, which is an important aspect for real-time deployment.

Existing studies also explore deployment strategies for practical use. Some works suggested ideas for browser plugins or web services for detecting the hate speech, while others researched moderation tools for live-streaming chats and online gaming environments. Most of the approaches focus primarily on accuracy, with limited focus on the latency-performance balance which is actually necessary for real-time systems.

Our work builds upon this foundation by prioritizing both accuracy and efficiency. By evaluating multiple machine learning algorithms under latency constraints, and by demonstrating an interactive real-time system, this research addresses the gap between high-performing but slow models and lightweight but less reliable methods.

Table 1. Literature Review

Author(s) & Year	Dataset Used	Techniques / Models	Key Findings	Limitations
Kwok & Wang (2013)	Twitter (tweets against racial groups)	Naïve Bayes + n-grams	Demonstrated feasibility of ML for hate speech detection	Context ignored; simple unigram features
Davidson et al. (2017)	Twitter (25k tweets)	Logistic Regression + TF–IDF	Classified hate/offensive/neutral with ~90% accuracy	Struggled with sarcasm and implicit toxicity
Jigsaw/Google (2018)	Wikipedia Toxic Comment Challenge (Kaggle)	Logistic Regression, SVM, Ensembles	Large-scale benchmark dataset for toxicity classification	Mainly offline; latency not studied
Noever (2018)	Jigsaw Dataset	62 classifiers (ML families)	Provided extensive benchmarking of traditional ML for toxicity detection	Did not address real-time performance
Almerekhi et al. (2022)	Reddit conversations (top 100 subreddits)	Bi-LSTM, fine-tuned BERT	Detected "toxicity triggers" before escalation; high contextual performance	High computational cost; not suited for real-time use
Oikawa et al. (2022)	Live-streaming chat data	Stacking + shallow ML classifiers	Achieved low-latency toxicity detection in live chats without GPUs	Limited generalization beyond live streaming
Lin et al. (2023, ToxicChat)	User–AI conversations	BERT-based contextual models	Highlighted hidden challenges of toxicity in AI–user interactions	High resource demand; offline focus
Yang et al. (2023, ToxBuster)	In-game chat logs	Metadata + contextual ML models	Detected real-time in-game toxicity with ~90% precision, proactive detection	Domain-specific; scalability issues
Khandekar et al. (2023)	Kaggle dataset (~40k comments)	Bi-LSTM + NLP preprocessing	Achieved 86.9% accuracy in detecting "unethical" text with a Flask app	Focus on accuracy; no latency analysis

3. Novelty

The novelty of this work is that it focuses on detecting the toxicity in real time using a lightweight but effective model with the help of machine learning. Different from other research, instead of improving the classification accuracy in offline settings, this study maintains a low-latency performance and prediction accuracy. This makes the system practical for live chats. The main highlights of this work include

The main highlights of this work include: Comparison of machine learning models based on speed and accuracy to choose the best one: Implementing various machine learning models like Random Forest, Support Vector Machine, and XGBoost using TF-

IDF features and threshold tuning, and comparing these algorithms' performance on the basis of accuracy using F1 score, latency, and training time.

Real-time toxicity detection framework: Creating a pipeline that is capable of getting input from the user and processing it and classifying it later as toxic or not within milliseconds, thus ensuring no delay in live conversations.

Lightweight and efficient model for real word implementation: Unlike the expensive approaches such as deep learning, this system uses a classical model that can achieve better accuracy while maintaining a lower time for feedback, making it more suitable for applications where real-time moderation is required maintaining significantly lower inference times, making it suitable for real-world chat moderation.

Interactive application for taking actions in advance for prevention: A Streamlit and Python-based application providing the user and moderators an interface to see the inputs, get instant feedback, and also see the chat history.

Support for sustainable digital interactions: By providing an efficient mechanism to identify and mitigate harmful content instantly, the proposed system contributes to creating healthier online environments aligned with the broader goal of sustainable and responsible technology use.

4. Methodology

A. ***Dataset***

For this study, we used the publicly available dataset- Hate Speech and Offensive Language dataset by Davidson et al. (2019) [2]. The dataset has tweets that are voted and labelled by at least three CrowdFlower users, and then labelled by them as 0-hate speech, 1-offensive language, and 2-neither. The dataset is available in CSV and pickle data form. This dataset will help to distinguish the comments and give back an alert if offensive or hateful after training the dataset.

Reference Deep learning models for evaluation: To compare and evaluate how our proposed lightweight models perform, we briefly considered the mostly used current models of deep learning. These models like BERT are accurate but very slow and they need many resources thus its not suitable for real-time detection [5], [7]. For comparison, we also considered BERT [10], a transformer-based architecture widely used in toxicity detection tasks.

LSTM overview: An LSTM is a special type of network that can handle the sequential data, making it suitable for capturing contextual cues in offensive language.

Formulation: For an input sequence $x = (x_1, x_2, \ldots, x_T)$, the LSTM computes hidden states ht using input, forget, and output gates:

$$f_t = \sigma!\left(W_f\,[h_{t-1}, x_t] + b_f\right), \qquad i_t = \sigma!\,(W_i\,[h_{t-1}, x_t] + b_i)$$

$$i_t = \sigma!\,(W_i\,[h_{t-1}, x_t] + b_i),$$
$$i_t = \sigma!\,(W_i\,[h_{t-1}, x_t] + b_i)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c_t}, \qquad h_t = o_t \odot \tanh(c_t)$$

where f_t, i_t, o_t represents the input and output gate activations, c_t denotes the cell state, and h_t denotes the hidden state at time t.

BERT overview: BERT uses transformer-based self-attention to encode bidirectional context. For a sequence, $x = (x_1, \ldots, x_T)$ the hidden representation of token i is: where Q,K,V are query, key, and value projections, and self-attention is computed as:

$$h_i = \text{Attention}\left(QW_Q, KW_K, VW_V\right)$$

$$\text{Attention}(Q,K,V) = \text{softmax!}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

$$\text{Attention}(Q,K,V) = \text{softmax!}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where Q, K, V are the query, key, and value projections, d_k is the dimension of the key vectors.

*B. **Feature Extraction***

Machine learning models cannot process and understand the raw text on their own. So, TF-IDF (Term Frequency – Inverse Document Frequency) is used for analyzing the text and to know how important each word is in the tweet from the dataset. It helps for better performance of machine learning models since it provides weighted counts. We used TF–IDF as a feature extraction method, which has been used commonly in prior toxicity detection studies [2], [3]

$$w_{t,d} = \text{TF--IDF}(t,d) = \text{tf}(t,d) \cdot \log!\left(\frac{N}{df(t)}\right)$$

where N is the total number of documents, tf(t,d) is the term interval for term t in document d, and df(t) is the document recurrence.

C. *Applied Machine learning techniques*

In this paper, we implemented three models for comparing which among them will perform best and provide efficient results. The models were Random Forest, Support Vector Machine, XGBoost with threshold tuning. Each model will be trained and compared on the basis of metrics like accuracy, F1 score, etc, In our work we observed that it provides better result for the selected dataset.

To improve the detection, a threshold tuning decision was applied while applying the XGBoost algorithm. Threshold tuning done was:

$$T = \{t_{hate} = 0.25, t_{off} = 0.33, t_{neither} = 0.33\}$$

A tweet was classified into category Ci if:

$$C_i = argmax(P_i \; where \; P_i > t_i)$$

where Pi here represents the predicted probability for class i and t_i is its corresponding threshold. Tweets which were not meeting any threshold was set defaulted to the most probable class. If none of the class probabilities met its threshold, then the tweet will be labelled with the highest predicted probability.

SVM Classifier:

$$f(x) = sign!\,(w^T x + b)$$

Here w denotes the weight vector, b is the bias, and x represents the input data point i.e the TF–IDF vector of the input text. SVM finds the hyperplane maximizing the margin between toxic and non-toxic classes.

Random Forest: Given training data $D = \{(x_i, y_i)\}_{i=1}^{n}$ Random Forest trains k decision trees on bootstrap samples of D. For classification:

$$\hat{y} = mode\{h_1(x),\, h_2(x),\, \ldots,\, h_k(x)\}$$

where $h_j(x)$ is the prediction of tree j. XGBoost with Threshold Tuning XGBoost optimizes an objective function:

$$\mathcal{L} = \sum_{i=1}^{n} l!\left(y_i,\, \widehat{y_i^{(t-1)}} + f_t(x_i)\right) + \Omega(f_t)$$

In this formula, l represents the loss function,
f_t is the tree added at step t and $\Omega(f)$ is the
regulates the complexity of the model.
For threshold tuning, we classify:

$$\hat{y} = C_i, if P_i \geq \theta_i$$

$$argmax_j P_j, otherwise$$

where P_i is the predicted probability for class i_i, and θ_i is the tuned threshold.

D. System Architecture and System Workflow

The dataset is pre-processed, and then TF-IDF is used to convert the text into a numeric form for highlighting important words, and then the vector file is saved. For training the dataset, Random Forest, SVM, and XGBoost were used and compared. To improve the precision and recall value, threshold tuning is performed. After testing each model, the best among them is selected. The thresholds value is saved along with the top performed model.

The user interface is made using Streamlit and python and deployed using Streamlit cloud. In the system, the user can enter a message, and while sending the message, the model predicts the probability of each class, and if none of the classes meet the threshold value, then the highest probability class is assigned. If the user enters an offensive or hate message, then the message is restricted and a toast alert is provided to the user by the system else the message can be send.

While deep learning models (LSTM, BERT) achieve high accuracy, their training and inference times are pointedly higher. In latency-sensitive scenarios such as live chat moderation, lightweight models like SVM and XGBoost strike a better balance between accuracy (90% F1) and real-time responsiveness (<0.005s per sample). Our study prioritizes practical deployment efficacy over marginal accuracy gains.

5. Results

The AI system ensures that the messages that are harmful, toxic, or offensive that affect the social well-being is restricted and a toast alert is provided. In Figure 1, the comparison of accuracy and F1-score is shown with XGBoost tuned showing

the highest accuracy of 0.90 and F1- score of 0.75. Figure 2 shows the confusion matrix of the XGBoost model after Threshold tuning.

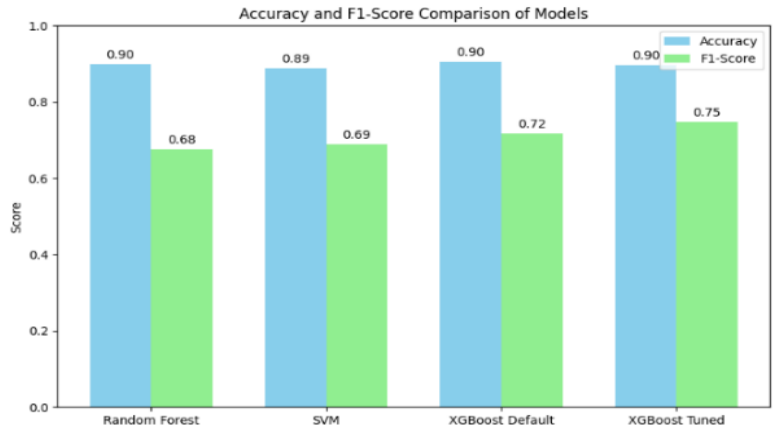

Figure 1. Accuracy and F1-score comparison of

- **Accuracy** – It defines theoverall correctness.
- **Precision** – It measures the accuracy of the model's positive predictions.

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

- **Recall (Sensitivity)** – ability to correctly identify actual toxic messages.

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

- **F1-score** – harmonic mean of precision and recall.

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

- **ROC (Receiver Operating Characteristic) Curve** – It plots the True Positive Rate (Recall) against the False Positive Rate. A better model will have the curve closer to the top-left corner
- **AUC (Area Under the Curve)** – numerical measure of separability (1 = perfect, 0.5 = random guessing).
- **Latency** – average time for a model to predict one sample. Important in real-time applications like chat moderation.

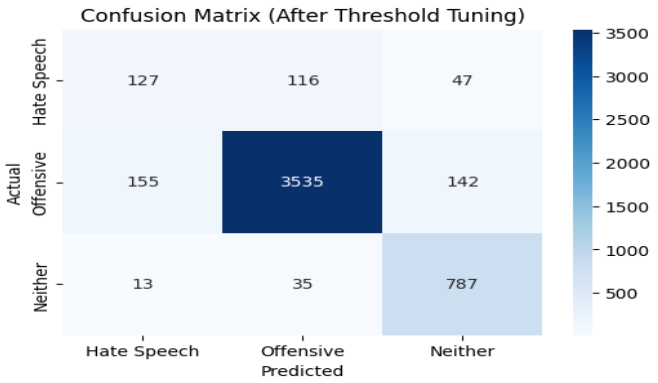

Figure 2. Confusion Matrix after Threshold Tuning

Table 2 shows the performance and the time taken for training, prediction time per sample. It shows that its lightweight, and can do prediction with lower latency, and its less computationally expensive than deep learning. XGBoost tuned takes 15.62 seconds for training and 0.004 seconds for predicting per sample, thus best and efficient for prediction in real time.

Table 3. Model performance comparison on the basis of training time, accuracy and prediction latency.

	Model	Training Time	Prediction Time per Sample (ms)	Accuracy
0	Random Forest	24.67	0.092	0.8995
1	SVM	0.12	0	0.8884
2	XGBooost Default	17.79	0.005	0.9046
3	XGBooost Tuned	15.62	0.004	0.9046

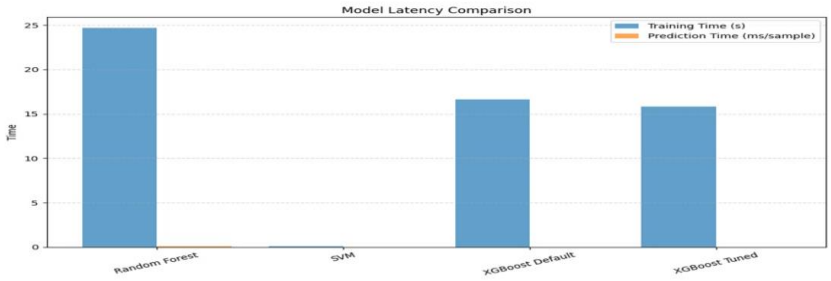

Figure 3. Comparison of model latency.

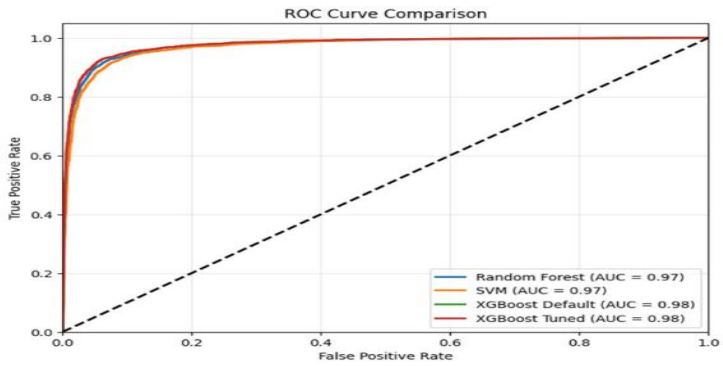

Figure 4. Comparison on the basis of ROC curve

Figure 4 shows the roc curve comparison of the models. Both XGboost default and tuned have achieved the AUC of 0.98. But along with that considering the other factors i. latency, accuracy and the f1-score, XGboost tuned is selected.

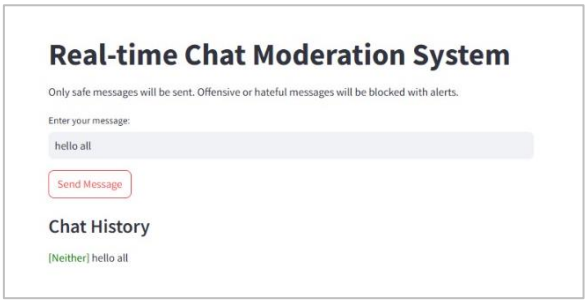

Figure 5. AI-Powered Real-Time Chat Moderation – Output 1

In Figure 5, it shows the user enters a text, and when the message is send, it does not raise any alert since not offensive, and the text is visible in the chat history.

Figure 6. AI-Powered Real-Time Chat Moderation – Output 2

Figure 6 shows that the text entered by the user is flagged, and a toast alert is received by the user when the message is send that the message is blocked and it is offensive, and the text is not displayed in the chat history.

Table 4. Comparison table

Study	Dataset	Model	Accuracy	F1	AUC	Latency
Davidson et al. (2017)	Twitter (25k tweets)	Logistic Regression + TF–IDF	90%	0.8	–	Not reported
Almerekhi et al. (2022)	Reddit (subreddits)	Bi-LSTM, BERT	91%	0.90	0.93	High (GPU required)
Yang et al. (2023, ToxBuster)	In-game chat logs	Contextual ML	90%	0.89	–	Moderate
Proposed (This Work)	Davidson dataset	XGBoost (tuned, TF–IDF)	90%	0.75	0.88	0.004s/sample

In Figure 3, it shows the user enters a text, and when the message is send, it does not raise any alert since not offensive, and the text is visible in the chat history. Table 2 illustrates the comparison of existing models with the proposed model. Figure. 4 shows that the text entered by the user is flagged, and a toast alert is received by the user when the message is sent that the message is blocked and it is offensive, and the text is not displayed in the chat history. In addition to accuracy and F1-score, we evaluate models using precision, recall, ROC–AUC, and prediction latency. For real-time moderation, the response from the system is also important, so these metrics together help to know how correct the classification is along with how quickly the system responds, thus overall providing an idea of how the system works.

6. Conclusion And Future Scope

Our proposed AI-powered chat moderation system demonstrates efficiency in detecting the offensive, harmful, or toxic messages that may affect the mental health and well-being of an individual. The system works well with the help of TF-IDF and with 90% accuracy using the XGBoost model among the various machine learning algorithms compared. It also has a very low prediction latency. The system provides instant alerts, ensuring intervention, handling such contents before itself ensuring safety during online communication.

While the current study focuses on the English language dataset, in the future, we can add models for detecting offensive words in other languages, thus making it more effective for social well-being and for making it global. The model can also be used for other platforms like comment sections, emails, etc. These add-ons will help the system to expand its role for encouraging respectful and healthful interactions online.

References

[1] Kwok and Y. Wang, "Locate the hate: Detecting tweets against blacks," Proc. AAAI Conf. Artificial Intelligence, vol. 27, pp. 1621–1622, 2013.

[2] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated Hate Speech Detection and the Problem of Offensive Language," in *Proc. 11th Int. AAAI Conf. Web and Social Media (ICWSM '17)*, Montreal, Canada, 2017, pp. 512–515.

[3] Jigsaw/Google, "Toxic Comment Classification Challenge," Kaggle, 2018. [Online]. Available: https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

[4] M. Noever, "Machine learning suites for online toxicity detection," arXiv preprint arXiv:1810.01875, 2018.

[5] N. Almerekhi, S. J. Joty, and E. Zaghouani, "Detecting toxicity triggers in Reddit conversations," Proc. Int. AAAI Conf. Web and Social Media (ICWSM), 2022.

[6] M. Oikawa, T. Kajiwara, and M. Okumura, "Low-latency toxicity detection in live chats," Proc. 60th Annual Meeting of the Association for Computational Linguistics (ACL), 2022.

[7] Z. Lin, J. Kocoń, and D. Wróblewska, "ToxicChat: The hidden challenges of toxicity detection in AI-user conversations," Proc. 61st Annual Meeting of the Association for Computational Linguistics (ACL), 2023.

[8] L. Yang, et al., "ToxBuster: Real-time toxicity detection in online games," Proc. IEEE Int. Conf. Big Data, 2023.

[9] A. Khandekar, R. Patil, and S. Deshmukh, "Detecting unethical text using Bi-LSTM and Flask deployment," Proc. Int. Conf. Intelligent Systems and Data Science (ICISDS), 2023.

[10] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019, pp. 4171–4186.