INVENTION DISCLOSURE FORM

1) TITLE: EthiCheck: Multilingual Hate Speech Detector

2) INVENTOR(S)/STUDENT(S): all fields in this column are mandatory to be filled

A.	Full name	Kunwar Aditya Singh
	Mobile Number	+91 9559299441
	Roll number	13
	UID/Registration Number	12104742
	Permanent Address	6a, Narayan Nagar, Sector 11, Indira Nagar, Lucknow, India
B.	Full name	Harsirat Singh Chahal
	Mobile Number	+91 7888500625
	Roll number	15
	UID/Registration Number	12105521
	Permanent Address	L-1601, Jalandhar Heights-2, 66 Feet Road, Pholriwal, Jalandhar-144005, Punjab, India
C.	Full name	Avijeet Singh Bhati
	Mobile Number	+91 9462168788
	Roll number	44
	UID/Registration Number	12111692
	Permanent Address	House no. 14 Near Hanuman Temple Veer Durgadas colony, Jodhpur, Rajasthan, India
D.	Full Name	Pushkar Singh Gangwar
	Mobile Number	+91 6386622458
	Roll Number	49
	UID/Registration Number	12106173
	Permanent Address	91, kaimganj, farrukhabad, uttar Pradesh
E.	Full name	Piyush Bhatia
	Mobile Number	+91 8968147507
	Roll number	51
	UID/Registration Number	12104522
	Permanent Address	GT Road Goraya-144409, Punjab, India

3) DESCRIPTION OF INVENTION:

A. **PROBLEM ADDRESSED BY THE INVENTION:**

Our invention, the 'EthiCheck', is a real-time, multilingual hate speech detector. It features customizable sensitivity thresholds and advanced contextual analysis to differentiate hate speech from satire. Sentiment analysis and user profiles aid in understanding the emotional tone and user behaviour. Additionally, the system supports reporting and moderation, promoting a safer online environment. It's culturally sensitive, considering regional nuances. In essence, the 'EthiCheck' is a powerful tool for monitoring and addressing hate speech in digital communication, enhancing online inclusivity and respect.

B.

S. No	Patent no	Title	Abstract	Research Gap
1.	US9792279B2	Methods and systems for analysing communication situation based on emotion information	Provided is a method of recommending a sticker through an emotion analysis. The method of recommending a sticker through an emotion analysis, include: by a server, performing a surface analysis on the last utterance between the first user terminal and the second user terminal; performing an emotion analysis on the last utterance using a result of the surface analysis; extracting a dialog context factor including a surface analysis result and an emotion analysis result on a certain number of continuous utterances including the last utterance between the first user terminal and the second user terminal; selecting a sticker to be recommended to the first user using the dialog context factor; and providing the selected sticker for the first user terminal.	1. A method of recommending a dialogue sticker group by use of a server that is connected to a database, a first user terminal and a second user terminal through a network and relays an utterance inputted to a messenger between the first user terminal and the second user terminal, the utterance including at least one of a text and an image, the method comprising: 2. 2. A non-transitory computer readable medium storing one or more sequences of pattern data for recommending a dialogue sticker group by use of a server that is connected to a database, a first user terminal and a second user terminal through a network and relays an utterance inputted to a messenger between the first user terminal and the second user terminal, the utterance including at least one of a text and an image, wherein execution of the one or more sequences of the pattern data by one or more processors causes the one or more processors to perform the steps of:
2.	US20150278195A1	Text data sentiment analysis method	A method and system for text data analysis by performing deep syntactic and semantic analysis of text data and extracting entities and facts from the text data based on the results of deep syntactic and semantic analysis, including extraction of	1. A method of text data analysis, including: obtaining text data; performing deep syntactic and semantic analysis of text data; extracting entities and facts from text data based on the results of deep syntactic and semantic analysis, including extraction of sentiments using a sentiment lexicon constructed

			sentiments using a sentiment lexicon constructed upon a semantic hierarchy. The data analysis can include determining sign of the extracted sentiment, aggregate function of the text data, analyzing social mood, and classifying the text data.	upon a semantic hierarchy. 2. The method of claim 1, further including the step of determining the sign of the extracted sentiments. 3. The method of claim 1, further including the step of determining the aggregate function of text data. 4. The method of claim 1, further including the step of identifying social networks based on the extracted entities and facts.
3.	CN104008091B	A kind of network text sentiment analysis method based on emotion value	The present invention relates to a kind of network text sentiment analysis method based on emotion value.The operating procedure of this method is as follows : First, pretreatment is carried out to text, paragraph splits, and punctuate is replaced etc..Second, clause is analyzed, and is judged query clause and exclamation clause in segmentation text, is weighted process to emotion value.3rd, emotion value is mated, and carries out emotion word coupling according to the sentiment dictionary for predefining good emotion value to each segmentation of text, emotion value is brought into.4th, emotion assignment is carried out to the emotion word for matching, segmentation emotion value is obtained.5th, emotion value is calculated, and each segmentation	a kind of network text sentiment analysis method based on emotion value, it is characterised in that analytical procedure is as follows : (1) Text Pretreatment : Punctuation mark is processed and sentence segmentation ; (2) clause analysis : Judge query clause and exclamation clause in segmentation text, process is weighted to emotion value ; (3) emotion word coupling : Each segmentation is analyzed one by one, mate emotion value dictionary, by the word not matched as Neutral word processing ; (4) emotion word assignment : Emotion assignment is carried out to the emotion word for matching, and the emotion word after emotion assignment is connected Connect, obtain segmentation emotion value ; (5) emotion value is calculated : Each segmentation emotion value is combined calculating, whole sentence emotion value is obtained ; (6) emotion value correction : Emotion value correction is carried out according to text size to whole text ;

			emotion value is combined calculating, whole sentence emotion value is obtained.6th, emotion value correction carries out after emotion value is disposed, all emotion values being modified according to certain rule to each segmentation.7th, Sentiment orientation is analyzed, and carries out feeling polarities analysis according to emotion value after drawing emotion value.The analysis method can more accurately analyze the emotion information of Chinese text.	(7) Sentiment orientation judges : Sentiment orientation is judged according to emotion value interval, including front, neutral and negative

C. **DETAILED DESCRIPITION (technical as well as Non-Technical):**

EthiCheck is a sophisticated language analysis system implemented in Python, leveraging a combination of libraries, algorithms, and technologies to effectively detect and address hate speech and unethical language in digital communication.

(1) NLP Libraries: The system makes extensive use of popular Python NLP libraries, including:

a) NLTK (Natural Language Toolkit): NLTK is employed for text preprocessing, tokenization, and part-of-speech tagging.

b) spaCy: spaCy is used for more advanced NLP tasks, such as named entity recognition and dependency parsing.

c) TextBlob: TextBlob is utilized for sentiment analysis, providing insights into the emotional tone of text.

2) VADER Sentiment Analysis: EthiCheck incorporates the VADER (Valence Aware Dictionary and sEntiment Reasoner) sentiment analysis tool, which is specifically designed for social media text. VADER provides a pre-trained sentiment analysis model that can determine the emotional intensity of

text, helping EthiCheck understand the sentiment and emotional tone of the content.

3) Machine Learning Framework: EthiCheck leverages the scikit-learn library for machine learning tasks. This includes training and deploying machine learning models for hate speech classification.

4) Deep Learning Framework: For more complex NLP tasks, EthiCheck incorporates TensorFlow and Keras. These libraries are used to build and train deep learning models, such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs), for contextual analysis.

5) Multilingual Models: EthiCheck uses pre-trained multilingual NLP models like BERT, GPT, or XLM-R to extend support for multiple languages. These models provide excellent language understanding and analysis capabilities across diverse linguistic contexts.

6) Customizable Thresholds: EthiCheck's threshold customization is implemented through a user-friendly interface and configuration files. The system allows users to adjust various parameters and fine-tune sensitivity settings.

7) Real-time Monitoring: The system uses asynchronous programming techniques to continuously monitor digital communication channels, ensuring real-time analysis and detection of potential hate speech.

8) Adversarial Text Detection: EthiCheck employs various techniques such as adversarial training and input perturbation to detect adversarial text, ensuring robustness against evasion attempts.

9) Reporting and Moderation Tools: Reporting and moderation functionalities are supported by database systems, enabling efficient storage and retrieval of reported content. Elasticsearch, MongoDB, or SQL databases may be used for these purposes.

10) Cultural Sensitivity: To implement cultural sensitivity, EthiCheck may use custom models or datasets specific to regional languages and dialects. It also employs techniques like word embeddings to capture cultural nuances.

11) Cloud Integration: EthiCheck can be deployed on cloud platforms like AWS, Google Cloud, or Azure for scalability and reliability. Integration with cloud services may be used for real-time data processing and analysis.

In summary, EthiCheck combines the power of Python, popular NLP libraries, machine learning and deep learning frameworks, the VADER sentiment analysis tool, and cloud technology to create a versatile and efficient language ethical checker that caters to the multilingual, real-time, and culturally

sensitive needs of online communities. It's an amalgamation of cutting-edge technology to ensure a safer and more inclusive online environment.

D. RESULTS AND ADVANTAGES:

1. Enhanced Multilingual Support: Unlike many previous systems, EthiCheck provides robust multilingual support, enabling users to monitor and address hate speech in diverse languages, making it a versatile solution for global online communities.

2. Real-time Detection: EthiCheck excels in real-time monitoring, promptly identifying and responding to potentially harmful content, which is crucial for ensuring the safety of online spaces.

3. Contextual Analysis and Sentiment Insight: The system's contextual analysis, coupled with VADER sentiment analysis, provides a nuanced understanding of text, reducing false alarms and improving the accuracy of hate speech detection.

4. Cultural Sensitivity: EthiCheck's cultural sensitivity algorithms ensure that the system respects regional language nuances, minimizing unintended filtering of legitimate expressions, a feature often overlooked in prior models.

5. User Customization: EthiCheck empowers users with customizable thresholds, allowing them to fine-tune the system's sensitivity to align with their individual preferences, granting a personalized experience.

6. Efficient Reporting and Moderation Tools: The system simplifies content reporting and moderation, making it more efficient for platform administrators and users, contributing to a safer and more respectful online environment.

4) POTENTIAL DRAWBACKS:

a) False Positives: EthiCheck may occasionally flag content as hate speech or unethical language when it is not, potentially leading to user frustration and concerns over censorship.

b) Overlooking Subtle Hate Speech: The system may struggle to detect more subtle forms of hate speech or content that employs coded language, allowing such content to go undetected.

c) Bias and Fairness Concerns: Like many AI systems, EthiCheck could inherit biases from its training data, potentially resulting in the unfair classification of certain groups or content.

d) Resource Intensiveness: Real-time monitoring and advanced language analysis can be resource-intensive, potentially requiring substantial computational power and infrastructure.

e) Adaptation Challenges: Adversarial text techniques may evolve faster than the system can adapt, necessitating continuous updates to remain effective against evasion attempts.

f) Privacy Concerns: The creation of user profiles for behavior monitoring raises privacy concerns, potentially leading to objections from users concerned about their online privacy.

g) Moderation Challenges: While reporting and moderation tools are valuable, they may also face challenges in terms of timely content review and determining the appropriate action to take.

h) Multilingual Challenges: Multilingual support, while an advantage, may also present challenges in maintaining language-specific models and datasets for accurate detection in various languages.

i) User Interface Complexity: Customizable thresholds and advanced features may make the user interface complex, potentially leading to usability issues for some users.

j) Scalability Challenges: Real-time monitoring in EthiCheck demands substantial resources, which can be problematic for smaller platforms with limited infrastructure.

5) CODE WORKING:

This Python script represents a robust and multifaceted text classification system, designed to serve as a pivotal component in various applications involving text analysis, content moderation, and ethical language monitoring. The code ingeniously combines an array of libraries, advanced NLP techniques, and machine learning methodologies to achieve its objectives. Here, we delve into a detailed analysis of the code, its core functions, and its technical intricacies.

1. Import Libraries:

re (Regular Expressions): While not actively employed in this code segment, the inclusion of the re library showcases the code's readiness for text pattern matching, an important aspect in more complex NLP tasks.

nltk (Natural Language Toolkit): This library is a cornerstone for a multitude of NLP operations. From text preprocessing to stopwords management, nltk forms the linguistic foundation of the code.

2. NLTK Data Path Configuration:

The code starts with a forward-thinking approach, customizing the data path for NLTK. By extending the data path to a user-specified directory, it ensures the accessibility of essential language resources. This enables flexibility and adaptability, making it well-suited for a variety of use cases and language datasets.

3. Specify Stopwords Path:

The variable stopwords_path is introduced to find the path to the "english" subdirectory within the "stopwords" dataset. While seemingly minor, this detail exemplifies the code's diligence in sourcing stopwords, a critical element in NLP for filtering common, low-information words. It underscores the code's meticulous language-specific approach.

4. Sample Dataset:

The data variable serves as a miniature yet highly illustrative dataset. Each data point consists of a text sample and its corresponding label (0 for Ethical, 1 for Harmful). This small dataset highlights the code's practicality for creating training sets and initiating real-world applications with modest data volumes.

5. Data Preprocessing:

The heart of the code lies in its data preprocessing phase, orchestrated by the preprocess_text function. This function performs a series of transformative steps, introducing a cascade of NLP concepts to enhance the quality of the dataset.

The code's commitment to quality starts with the conversion of text to lowercase, ensuring consistent handling of text regardless of its original case. This case normalization step is vital for accurate text analysis, as it minimizes inconsistencies due to case variations.

Another crucial transformation entails the removal of non-alphabetical characters, effectively purifying the text of special characters, symbols, and punctuation. This phase marks a fundamental step in text cleaning, guaranteeing that the code's subsequent analysis is based on refined text data.

The code's text tokenization process, i.e., splitting text into individual words, is pivotal. It lays the foundation for a word-level analysis, facilitating the calculation of word frequencies and their contribution to the overall document.

The removal of common English stopwords represents another dimension of this code's sophistication. By eliminating these ubiquitous words, the code filters out the most frequent, yet often semantically uninformative, words such as "the," "and," "is," which could otherwise dominate the text analysis.

The culmination of these steps is a fully preprocessed dataset. The code retains the preprocessed text in the corpus variable, while labels are extracted and stored separately in the labels variable.

6. Text Vectorization:

Text data, now meticulously preprocessed, undergoes an evolution of its own as it is transformed into numerical vectors. The introduction of TfidfVectorizer from the scikit-learn library heralds the onset of this transformation.

The code chooses the TF-IDF (Term Frequency-Inverse Document Frequency) vectorization approach, a technique that evaluates the importance of each word in each document relative to the entire dataset. By creating numerical representations of text data, this stage opens doors to machine learning models and text classification.

7. Classifier Training:

In this code, simplicity meets efficacy as it introduces a text classification model: MultinomialNB (Multinomial Naive Bayes). The model is not only capable of high-performance classification but is also relatively easy to implement. The code skillfully combines the model's simplicity with its powerful classification abilities.

8. Content Checking Function:

The check_content function, a focal point of this code, is the gateway to real-time content classification. It crystallizes the entire preprocessing, vectorization, and classification journey into a user-friendly function.

The function begins by conducting preprocessing, mirroring the text preprocessing steps applied to the training data. It elegantly aligns real-time text with the processed training data for consistency in analysis.

The vectorized_text variable encapsulates the user's input, vectorized using the previously trained vectorizer. The result is a numerical representation of the input text, aligning it with the training data's structure.

The code's prediction, at the heart of the function, derives the ethicality of the text input, resulting in the classification of "Ethical" or "Potentially Harmful." This binary classification

showcases the code's suitability for applications where content must be swiftly categorized based on ethical considerations.

9. User Input and Testing:

The code's interactivity shines through as it invites user input. It beckons the user to contribute text for real-time classification, illustrating its practicality in content moderation and ethical language monitoring.

The code executes the check_content function, initiating the analysis of the user's text input. The result is promptly stored in the result variable, conveying the classification outcome.

The code responds with a summary of its analysis: the user's input text, the classification outcome, and an additional layer of transparency—the path to the NLTK stopwords data directory. This transparency underscores the code's commitment to detail and clarity.

In conclusion, this Python code transcends the realm of a mere script; it embodies a versatile and adaptable text classification system, poised for applications in content moderation, ethical language monitoring, and text analysis. Its elegance lies in the harmonious fusion of diverse libraries and techniques, supported by a thorough understanding of NLP concepts and principles. By offering an elaborate set of tools for data preprocessing, feature engineering, model training, and real-time classification, it exemplifies a framework that empowers developers to create solutions that align with the ethical considerations of our digital age. In an era where the quality of online content and the promotion of respectful digital discourse are paramount, this code represents a step toward a safer and more inclusive online environment.

6) REFERENCES:

1. Geeks for Geeks
2. Linguamatics
3. Dataquest
4. YouTube
5. Google Patents
6. Levity.ai
7. Serokell.io