Innovators at SemEval-2024 Task 10: Revolutionizing Emotion Recognition and Flip Analysis in Code-Mixed Texts

Abhay Shanbhag*, **Suramya Jadhav***, **Shashank Rathi**
Siddhesh Pande, Dipali Kadam
Pune Institute of Computer Technology
{abhayshanbhag0110, 2018suramyajadhav, shashankrathi2}@gmail.com,
siddheshspande@gmail.com, ddkadam@pict.edu

Abstract

In this paper, we introduce our system for all three tracks of the SemEval 2024 EDiReF Shared Task 10, which focuses on Emotion Recognition in Conversation (ERC) and Emotion Flip Reasoning (EFR) within the domain of conversational analysis. Task-Track 1 (ERC) aims to assign an emotion to each utterance in the Hinglish language, a code-mixed language between Hindi and English, from a predefined set of possible emotions. Tracks 2 (EFR) and 3 (EFR) aim to identify the trigger utterance(s) for an emotion flip in a multi-party conversation dialogue in Hinglish and English text, respectively. For Track 1, our study spans both traditional machine learning ensemble techniques, including Decision Trees, SVM, Logistic Regression, and Multinomial NB models, as well as advanced transformer-based models like XLM-Roberta (XLMR), DistilRoberta, and T5 from Hugging Face's transformer library. In the EFR competition, we developed and proposed two innovative algorithms to tackle the challenges presented in Tracks 2 and 3. Specifically, our team, Innovators, developed a standout algorithm that propelled us to secure the 2nd rank in Track 2, achieving an impressive F1 score of 0.79, and the 7th rank in Track 3, with an F1 score of 0.68.

1 Introduction

With advancements in science and technology, the rise of social media has increased remote conversations with different people, resulting in a great deal of linguistic diversity. India is the country with the highest number of users on multiple social media platforms like Facebook, WhatsApp, Instagram, etc. Hinglish remains the most widely used code-mixed language on social media platforms.

A primary challenge associated with code-mixed languages revolves around the misidentification of parts of speech (POS) Atrey et al., 2012. This issue arises when individuals attempt to simultaneously utilize the vocabulary of both languages, leading to the failure of current state-of-the-art machine learning algorithms. Another significant problem identified in code-mixed language is the absence of context within conversations. Unlike traditional emotion detection ML models for pure languages, where a single sentence might suffice to detect emotion, this approach proves inadequate for code-mixed languages like Hinglish. In Hindi-based conversations, context plays a pivotal role in determining emotion Bansal and Lobiyal, 2021.

The data provided by the organizers of SemEval 2024 Task 10 Kumar et al., 2024 for the task comprised conversational episodes, each containing multiple utterances from different speakers. For Track 1 Kumar et al., 2023b, the data included a list of speakers and their utterances, with emotion being the target variable. In contrast, Track 2 Kumar et al., 2022 and Track 3 Kumar et al., 2023a provided utterances and emotions, with triggers as our target variable. Upon examining the training data, we identified an imbalance in the emotion classes, particularly illustrated in Table 1. To address this discrepancy, we applied a range of sampling techniques to effectively rectify the imbalance. Further details about the data are discussed in Section 2.

For Track 1, we employed two approaches: ensemble methods and the transformer approach. In the ensemble methods, we utilized classic ML models such as Random Forest, SVM, Multinomial Naive Bayes, and Logistic Regression, complemented by hyperparameter tuning. For our transformer approach, our main strategy involved creating a pipeline consisting of two main parts: the first deals with converting Hinglish to English, and the second detects emotion from the English output provided by the first. Thus, the pipeline takes Hinglish as input and outputs the corresponding emotions.

For tracks 2 and 3, where we had to detect emotion flips in Hinglish and English conversations, respectively, we developed an algorithm that identifies the last emotion flip of every user. The algorithm takes entire episodes as input and outputs the presence of triggers.

Upon evaluating our approach on the testing set with F1-score as the evaluation metric, we obtained a score of 0.28 for Track 1, 0.79 for Track 2, and 0.68 for Track 3.

The rest of the paper is organized as follows: Section 2 talks discusses the dataset provided by organizers for all three tracks , and Section 3 deals with existing research for several code-mixed tasks focusing on Hinglish text. Further in the paper, we discuss our

*first author, equal contribution

Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024), pages 634–641
June 20-21, 2024 ©2024 Association for Computational Linguistics

EMOTION	TRAIN	TEST	VALID
Neutral	3,909	656	633
Joy	1,596	349	228
Sadness	819	155	126
Anger	558	142	118
Fear	542	122	88
Contempt	514	82	74
Surprise	441	57	66
Disgust	127	17	21
TOTAL	8,506	1,580	1,354

Table 1: Figure showing distribution and count of emotions for Track 1.

proposed solutions in Section 4. Section 5 gives the experimental setup .Then Section 6 describes the performance of the different approaches along with key findings. Finally in Section 7 we have concluded our discussion.

2 Background

The dataset provided for Track 1 was supplied by the organizers. It consisted of episodes, each containing several sets of utterances in Hinglish. For every utterance, the dataset included the speaker responsible for the utterance, all formatted in JSON. Table 2 offers a glimpse into the Track 1 dataset for one of the episodes.

For Track 2, the data was similar but included an additional column for triggers. A trigger was set to 1 for the last emotion flip of every speaker, while it remained 0 for all other utterances. The primary distinction for Track 3 was the language of the utterances, which was English.

Upon analyzing the dataset, we identified eight emotions: Neutral, Joy, Sadness, Anger, Fear, Contempt, Surprise, and Disgust.

In addition to the organizer's data, we utilized the Hinglish-Top dataset. This dataset features two columns: English (en) and Hinglish (hi-en). We primarily employed this dataset for the Hinglish-to-English conversion component within our pipeline architecture.

3 Related Work

The task of emotion detection and classification has been extensively researched in the context of monolingual data. However, studies focusing on code-mixed text, especially in Indian languages like Hindi mixed with English, are limited due to the scarcity of sufficient data and the absence of a standardized approach for processing code-mixed text.

Foundational research on emotion identification within social media content written in a code-mixed Hindi-English pattern was conducted by Sasidhar et al., 2020. They compiled a dataset of 12,000 code-mixed Hindi-English texts from various sources, annotating them with emotions such as happiness, sadness, and anger. Their study utilized feature vectors generated by a pretrained multilingual model, and the classification models were derived from deep neural networks. Notably, the CNN-BiLSTM approach achieved a classification accuracy of 83.21%, outperforming other models tested in their research.

Wadhawan and Aggarwal, 2021 introduced a deep learning-based technique to recognize emotions in Hindi-English code-mixed tweets. This technique leverages transformer-based models along with bilingual word embeddings produced by Word2Vec and FastText techniques. Their experimentation with CNNs, LSTMs, bi-directional LSTMs, and a variety of deep learning models and transformers, including BERT, RoBERTa, and ALBERT, revealed that the transformer-based BERT model surpassed all others, achieving an accuracy of 71.43% according to their findings.

Bohra et al., 2018 focused on detecting hate speech in social media content that mixes Hindi and English codes, using two distinct classifiers: the Random Forest Classifier and the Support Vector Machines (SVMs). Due to the large feature vectors generated by their study, they employed the chi-square feature selection technique to reduce the size of their feature vector to 1,200. Their findings indicated that SVMs, when utilizing all attributes, outperformed the Random Forest classifier with a maximum accuracy of 71.7% . Additionally, they discovered that Word N-Grams were more effective with the Random Forest Classifier, while Character N-Grams achieved the best results in SVM.

Patil et al., 2023 conducted a comparative analysis of numerous transformer-based language models pre-trained through unsupervised methods, focusing on Hindi and English with mixed codes. Their study included non-code-mixed models such as AlBERT, BERT, and RoBERTa, as well as code-mixed models like HingBERT, HingRoBERTa, HingRoBERTa-Mixed, and mBERT. Models based on HingBERT, specifically trained on authentic code-mixed text, yielded state-of-the-art results on related datasets.

Employing the SentiMix code-mixed dataset, Ghosh et al., 2023 proposed a transformer-based multitask framework for sentiment identification and emotion classification. They enhanced the pre-trained cross-lingual embedding model, XLMR, using task-specific data to improve overall efficiency and leverage transfer learning more effectively.

Singh, 2021 discusses the outcomes of various methods used for sentiment analysis on Hinglish-written social media content, with Twitter serving as a primary example. The data was converted using Fasttext embeddings, count vectorizers, one hot vectorizers, doc2vec, word2vec, and tf-idf vectorizers. Singh employed a range of machine learning techniques, including SVM, CNN, Decision Trees, Random Forests, Naïve Bayes, Logistic Regression, and ensemble voting classifiers, to create the models. The evaluation was based on the F1-score (macro), with the ensemble voting classifier achieving the highest F1-score of 69.07%.

Speaker	Utterances	Emotions
Indu	Wo great hoga! Thanks!	Joy
Monisha	Me abhi tumhare liye new bana deti hun!	Joy
Indu	momma! hath chhodiye dad!	Sad
Monisha	Oh no! Kya hua?	Sad
Indu	Aaj to bhot awful day tha!	Sad

Table 2: Utterances Example from training

		Train	Test	Valid
TRACK 2	No. of episodes	4,893	385	389
	No. of utterances (unique in brackets)	98,777 (10,460)	7,690 (3,650)	7,642 (3,577)
	Avg. utterances per episodes (approx.)	20	20	20
TRACK 3	No. of episodes	4,000	1,002	426
	No. of utterances (unique in brackets)	35,000 (7,831)	8,642 (2,107)	3,522 (924)
	Avg. utterances per episodes (approx.)	9	9	8

Table 3: Track 2 and Track 3 episode-emotion distribution

4 System Description

4.1 Transformer Approach

To translate Hinglish to English and subsequently identify emotions from the translated text, we have developed a two-stage pipeline leveraging the power of transfer learning and pre-trained models from Hugging Face

In the first stage, we utilize the model developed by sayanmandal [1] as our foundational model from Hugging Face. This choice was motivated by its initial proficiency in translating between Hindi and English. To tailor its capabilities more closely to our Hinglish dataset, we applied transfer learning techniques, training it on the Hinglish TOP. dataset[2] by Agarwal et al., 2023.This process resulted in a notable improvement in translation accuracy, as evidenced by achieving a BLEU score of 18.0863%. The model adeptly takes Hinglish as input and outputs the corresponding English text, laying the groundwork for the subsequent emotion analysis.

For the second stage, the English text output from the first model is processed to extract emotional context. We employed the model of j-hartmann [3] from Hugging Face as the baseline for this task. Originally, this model, based on distilRoBERTa, was trained on a diverse array of datasets sourced from Twitter, Reddit, student self-reports, and TV dialogue utterances. However, it did not include 'contempt' among the eight emotion classes specified by the project's guidelines. Therefore, we adapted and further trained this model to recognize the additional emotion class, ensuring a comprehensive analysis of the emotional spectrum in the translated English text.

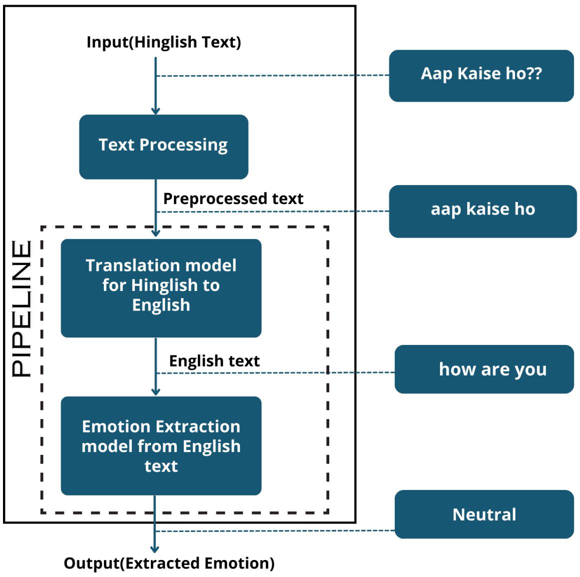

Figure 1: Transformer Architecture along with an example

4.2 XLM-Roberta

XLM-Roberta Conneau et al., 2020 has the ability to process text in Hinglish, a smooth blend of Hindi and En-

[1] sayanmandal/t5-small_6_3-hi_en-to-en
[2] Hinglish TOP dataset
[3] j-hartmann/emotion-english-distilroberta-base

glish, since it is proficient in over 100 languages, including Hindi and English. Its deep linguistic knowledge, reinforced by 2.5 terabytes of training data, enhances its comprehension of Hinglish's emotional nuances. In our work, we trained XLM-Roberta on a particular Hinglish emotion detection dataset using pre-trained weights. It was able to perform better and comprehend Hinglish emotions better as a result.XLMR model helped us to improve the overall performance significantly.

4.3 T5

T5 Raffel et al., 2020 demonstrates an impressive ability to comprehend the subtleties of Hinglish, a language that combines Hindi and English.It served as a good option for translating Hinglish because of its encoder-decoder architecture, which can easily handle code-switching, non-standard syntax, and transliteration. In our work, we fine-tuned the T5 model proposed by $sayanmandal$[2] on Hugging face with hyperparameters given in Table 6 on the external Hinglish TOP Dataset [2], which comprises 3,92,439 translations of Hinglish text into English. As a result, the model outperformed generic models in its ability to comprehend the particular complexities and differences in the dataset.

4.4 Distilroberta

DistilRoBERTa is computationally efficient and perfect for evaluating the frequently enormous amounts of translated text data because it is smaller as compared to RoBERTa and is capable of recording long-range dependencies in text. DistilRoBERTa was pre-trained on two enormous text corpora: BookCorpus and the English Wikipedia making the model more exposed to a wider range of linguistic patterns and improving its understanding of the semantic relationships found in text, both of which help the model identify different emotions. We used the j-hartmann [3] model of Hugging Face in our approach to recognize emotion from translated Hinglish text to English because of its inherent ability to recognize emotions. This helped us navigate any possible emotional nuances that were offered during translation, which strengthened our pipeline approach and increased the accuracy of the detection.

4.5 Random Forest

In Random Forest every tree conducts an independent examination of the data and makes predictions using predetermined feature criteria. A majority vote among all trees determines the final decision, providing resistance against noise and overfitting.Based on certain features like Word frequencies, part-of-speech tags, and sentiment lexicons, the model branches out and divides the input recursively until it reaches leaf nodes, which stand for expected emotions. The layered structure allows you to investigate the characteristics that contribute most to various emotion categories, providing you with a certain level of interpretability. Furthermore, we received higher results from the Random Forest Cutler et al., 2012 trials than from several other methods.

4.6 SVM

Support Vector machines (SVMs) are a useful tool for emotion identification applications because they can quickly scan high-dimensional text input and generate respectable results even with a limited amount of training data. SVMs Evgeniou and Pontil, 1999 excel at determining which feature space hyperplane most effectively separates different emotional classes, capturing the key characteristics that set each emotion apart.SVM proved to have a pretty decent F1 score as compared to other ensemble methods. It is so because of its robust hyperplane-based classification approach. The algorithm then uses statistical techniques to select the optimal line to split the different groups represented in Hearst et al., 1998.

4.7 MNB

Multinomial Naive Bayes (MNB) Kibriya et al., 2005 is a computationally efficient method for handling large datasets with great appropriateness. Using word frequency, it determines the likelihood that a text belongs to each emotion class. The steps in MNB include calculating the likelihood of every word, utilizing the Bayes theorem, and normalizing the probabilities. The final probabilities, which indicate the likelihood that a text belongs to each emotion, are produced by subtracting the estimated probability for each class from the total of the probabilities for all classes.

4.8 Logistic Regression

The linear classification model Logistic Regression Maalouf, 2011 offers a trustworthy and understandable solution for our emotion detection challenge. To forecast the likelihood of each emotion class, it uses a linear combination of input features extracted from the data. The objective variable (or output) in a classification problem, y, can only accept discrete values for a specific set of features (or inputs), X Cox, 1958. Only when a decision threshold is added does logistic regression transform into a classification technique based on the sigmoid function.

4.9 UnderSampling and Oversampling

In our experimental endeavors, we explored both oversampling and undersampling techniques Mohammed et al., 2020 to address class imbalances within our training dataset. The necessity for such interventions became evident as the 'neutral' class dominated the dataset—outnumbering instances of emotions like 'sad' and 'anger' by nearly double, and 'disgust' by an astounding factor of thirty. This disproportion threatened to skew the learning process, potentially biasing the model towards the overrepresented classes.

To mitigate this, we employed oversampling strategies, particularly focusing on the minority classes. By replicating instances from these underrepresented categories, we aimed to achieve a more equitable distribution across all emotions. This technique not only

Algorithm 1

```
Require: A dictionary of episode data with each entry containing a speaker, utterances, and the emotion associated with that speaker.
Ensure: A list indicating the trigger points, where each trigger point is set to 1.0 in case of a flip trigger and 0.0 elsewhere.
Initialization:
Initialize context: A dictionary to store each speaker's emotions and their indices like {emotion : indices}.
Initialize lastFlipForEverySpeaker: An empty list to store the indices of the last emotion change for each speaker.
Build Context:
for each speaker in the episode data do
    if the speaker is not in context then
        initialize their context
    end if
    append a dictionary {emotion: index} to the context for the current speaker
end for
Identify Last Emotion Changes:
for each speaker in the context do
    Initialize lastFlip to 0 and lastEmo to 'null'.
    for each emotion index in the speaker's context do
        Extract emotion and index from the context.
        if lastEmo is not equal to emotion then
            Set lastFlip to index.
            Set lastEmo to emotion.
        end if
    end for
    Append lastFlip - 1 to lastFlipForEverySpeaker.
end for
Initialize Trigger List:
Initialize trig as a list of 0.0s with a length equal to the number of speakers.

Mark Trigger Points:
for each speaker index do
    if the speaker's index is in lastFlipForEverySpeaker then
        set the corresponding element in trig to 1.0.
    end if
end for
return trig as the list of trigger points.
```

Algorithm 2

```
Require: A dictionary episodes with keys 'speakers' and 'labels'.
Ensure: A list of triggers for each episode, where each trigger list has a 1.0 for the second last conversation and 0.0 for the rest.
Algorithm:
Determine the number of speakers in the episode.
Initialize an empty list named trig to store trigger flags.
for each speaker in the episodes['speakers'] list do
    if the speaker is not the second-to-last one then
        append "0.0" to the trig list, indicating a non-trigger condition.
    else
        append "1.0" to the trig list, indicating a trigger condition.
    end if
end for
return a tuple consisting of the trig list from the episodes dictionary.
```

prevented the majority class from monopolizing the learning dynamics but also ensured that the model received ample exposure to each emotion. As a result, the capability of our model to accurately recognize emotions that were previously underrepresented saw significant improvement. Conversely, undersampling was also considered a method to harmonize the dataset. This approach involves reducing the instances of the majority class to match the numbers of the minority classes, thereby leveling the playing field. However, while undersampling can effectively reduce bias towards overrepresented classes, it also entails the risk of losing valuable information by discarding data.

Overall, we concluded that oversampling helped in the training process by giving each emotion class equal weight, and unlike undersampling, there was no loss of data.

4.10 Metrics Used F1 Score

For evaluating our model, we used the F1 score as our metric, which is given as the harmonic mean of precision and recall.

$$F1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{1}$$

Here, precision is the number of samples correctly predicted out of the number of samples predicted in that category. Recall is the number of samples predicted correctly out of the number of samples present for that class.

5 Experimental Setup

5.1 Data Preprocessing

Data preprocessing steps like lowercasing,stopword removal,punctuation removal and stemming were per-

Figure 2: Data Preprocessing Overview

formed before feeding text data into ensemble learning methods as shown in Figure 2. Emotion classification improves model performance by reducing noise, normalizing text, and enhancing feature extraction because of data prerocessing.

5.2 Vectorisation for ensemble methods

Word2Vec translates words into numerical vectors that represent their relationships to other words as well as their meanings. These vectors are valuable because they convey semantic understanding rather than merely indicating word presence. CBOW Mikolov et al., 2013, which predicts words based on context, and Skip-gram are two significant Word2Vec architectures. In our experiment, Word2Vec was utilized to convert processed text into vectors, which were then inputted into ensemble learning models for emotion prediction.

6 Result

	Track1	Track2	Track3
Our F1 Score	0.28	0.79	0.68
Our Rank	27	2	7
Max F1 Score	0.78	0.79	0.79

Table 4: Leaderboard Results

6.1 Key Findings

Our models demonstrated enhanced efficiency when the input data was augmented using minority oversampling. The input data exhibited a significant class imbalance, leading the model to predominantly recognize the dominant class. To address this issue, we employed both undersampling and oversampling techniques. Oversampling notably improved model performance, as indicated in Table 5, because it enabled the model to learn about the minority classes more effectively. We adjusted all feature sizes to match that of the dominant class size (in this case, 'neutral').

As illustrated in Figure 3 and Figure 4, our Algorithm 1 approach for Tracks 2 and 3 showed a considerable number of false positives, or negative samples incorrectly predicted as positive, amounting to 959. This figure was substantially higher than that observed in Algorithm 2, which was only 68. The count of false negatives in Algorithm 1 was comparable to that in Algorithm 2 (68 and 99, respectively), though Algorithm 1

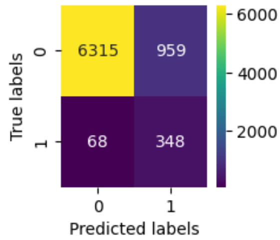

Figure 3: Confusion Matrix for Algorithm 1 on test data of track 2

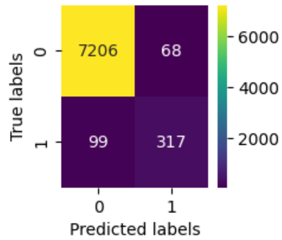

Figure 4: Confusion Matrix for Algorithm 2 on test data of track 2

was slightly more effective than Algorithm 2 in reducing false negatives.

In summary, instances of being erroneously classified as 0s were marginally higher in Algorithm 2, whereas instances of 0s being wrongly classified as 1s were significantly higher in the case of Algorithm 1.

7 Conclusion

In our participation in SemEval 2024 Task 10, we embraced two approaches: first, ensemble methods, and next, a transformer pipeline for our experiments in Track 1. Our analysis revealed a compelling insight: even marginal enhancements in translation accuracy can lead to substantial improvements in emotion classification outcomes. This underscores not only the importance of the sentence itself but also the critical role of contextual understanding in accurately leveraging this foundational insight. We developed and proposed two distinct algorithms designed to adeptly navigate the challenges of emotion flip recognition in Tracks 2 and 3.

Furthermore, our experiments highlighted the effectiveness of oversampling as a strategy to counteract the dataset's imbalance—a challenge characterized by a striking 30:1 ratio between dominant and minority classes. This technique emerged as a performance enhancer, enabling our models to achieve a more balanced understanding and representation of all emotional classes. Through these methodical and strategic efforts, we contributed valuable insights to the field and also demonstrated our algorithms' potential to transform emotion recognition practices.

APPROACH	F1 SCORE
DT	0.2495
DT (Undersampled)	0.2255
DT (Oversampled)	0.2578
SVM	0.2297
SVM (Undersampled)	0.2602
SVM (Oversampled)	**0.2830**
Multinomial Naive Bayes	0.1945
MultinomialNB (Undersampled)	0.2209
MultinomialNB (Oversampled)	0.2623
Logistic Regression - Softmax	0.2242
Logistic Regression - Softmax (Undersampled)	0.2584
Logistic Regression - Softmax (Oversampled)	0.2809
Logistic Regression - OvR	0.2242
Logistic Regression - OvR (Undersampled)	0.2584
Logistic Regression - OvR (Oversampled)	**0.2809**
Random Forest Classifier	0.2418
XLMR Approach	0.2626
Pipeline Approach	**0.2688**

Table 5: F1 Scores for Different Approaches used in Track 1

Parameter	Value
learning_rate	2e-05
train_batch_size	32
eval_batch_size	64
seed	42
gradient_accumulation_steps	2
weight decay	0.01
optimizer (Adam with betas)	(0.9, 0.999)
epsilon	1e-08
lr_scheduler_type	linear
num_train_epochs	100
mixed_precision_training	Native AMP

Table 6: Hyperparameters for Fine Tuning

References

Anmol Agarwal, Jigar Gupta, Rahul Goel, Shyam Upadhyay, Pankaj Joshi, and Rengarajan Aravamudhan. 2023. CST5: Data augmentation for code-switched semantic parsing. In *Proceedings of the 1st Workshop on Taming Large Language Models: Controllability in the era of Interactive Assistants!*, pages 1–10, Prague, Czech Republic. Association for Computational Linguistics.

Shree Atrey, T. Prasad, and G. Krishna. 2012. Issues in parsing and pos tagging of hybrid language. pages 20–24.

Mani Bansal and D. Lobiyal. 2021. Context-based machine translation of english-hindi using ce-encoder. *Journal of Computer Science*, 17:825–843.

Aditya Bohra, Deepanshu Vijay, Vinay Singh, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. A dataset of hindi-english code-mixed social media text for hate speech detection. In *Proceedings of the second workshop on computational modeling of people's opinions, personality, and emotions in social media*, pages 36–41.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

David R Cox. 1958. The regression analysis of binary sequences. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 20(2):215–232.

Adele Cutler, D Richard Cutler, and John R Stevens. 2012. Random forests. *Ensemble machine learning: Methods and applications*, pages 157–175.

Theodoros Evgeniou and Massimiliano Pontil. 1999. Support vector machines: Theory and applications. In *Advanced Course on Artificial Intelligence*, pages 249–257. Springer.

Soumitra Ghosh, Amit Priyankar, Asif Ekbal, and Pushpak Bhattacharyya. 2023. Multitasking of sentiment detection and emotion recognition in code-mixed hinglish data. *Knowledge-Based Systems*, 260:110182.

Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28.

Ashraf M Kibriya, Eibe Frank, Bernhard Pfahringer, and Geoffrey Holmes. 2005. Multinomial naive bayes for text categorization revisited. In *AI 2004: Advances in Artificial Intelligence: 17th Australian Joint Conference on Artificial Intelligence, Cairns, Australia,* *December 4-6, 2004. Proceedings 17*, pages 488–499. Springer.

Shivani Kumar, Md Shad Akhtar, Erik Cambria, and Tanmoy Chakraborty. 2024. Semeval 2024 – task 10: Emotion discovery and reasoning its flip in conversation (ediref). In *Proceedings of the 2024 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics.

Shivani Kumar, Shubham Dudeja, Md Shad Akhtar, and Tanmoy Chakraborty. 2023a. Emotion flip reasoning in multiparty conversations. *IEEE Transactions on Artificial Intelligence*, pages 1–10.

Shivani Kumar, Ramaneswaran S, Md Akhtar, and Tanmoy Chakraborty. 2023b. From multilingual complexity to emotional clarity: Leveraging commonsense to unveil emotions in code-mixed dialogues. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9638–9652, Singapore. Association for Computational Linguistics.

Shivani Kumar, Anubhav Shrimal, Md Shad Akhtar, and Tanmoy Chakraborty. 2022. Discovering emotion and reasoning its flip in multi-party conversations using masked memory network and transformer. *Knowledge-Based Systems*, 240:108112.

Maher Maalouf. 2011. Logistic regression in data analysis: an overview. *International Journal of Data Analysis Techniques and Strategies*, 3(3):281–299.

Tomas Mikolov, Kai Chen, G.s Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Proceedings of Workshop at ICLR*, 2013.

Roweida Mohammed, Jumanah Rawashdeh, and Malak Abdullah. 2020. Machine learning with oversampling and undersampling techniques: overview study and experimental results. In *2020 11th international conference on information and communication systems (ICICS)*, pages 243–248. IEEE.

Aryan Patil, Varad Patwardhan, Abhishek Phaltankar, Gauri Takawane, and Raviraj Joshi. 2023. Comparative study of pre-trained bert models for code-mixed hindi-english data. In *2023 IEEE 8th International Conference for Convergence in Technology (I2CT)*, pages 1–7. IEEE.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

T Tulasi Sasidhar, Premjith B, and Soman K P. 2020. Emotion detection in hinglish(hindi+english) code-mixed social media text. *Procedia Computer Science*, 171:1346–1352. Third International Conference on Computing and Network Communications (CoCoNet'19).

Gaurav Singh. 2021. Sentiment analysis of code-mixed social media text (hinglish). *ArXiv*, abs/2102.12149.

Anshul Wadhawan and Akshita Aggarwal. 2021. Towards emotion recognition in Hindi-English code-mixed data: A transformer based approach. In *Proceedings of the Eleventh Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 195–202, Online. Association for Computational Linguistics.