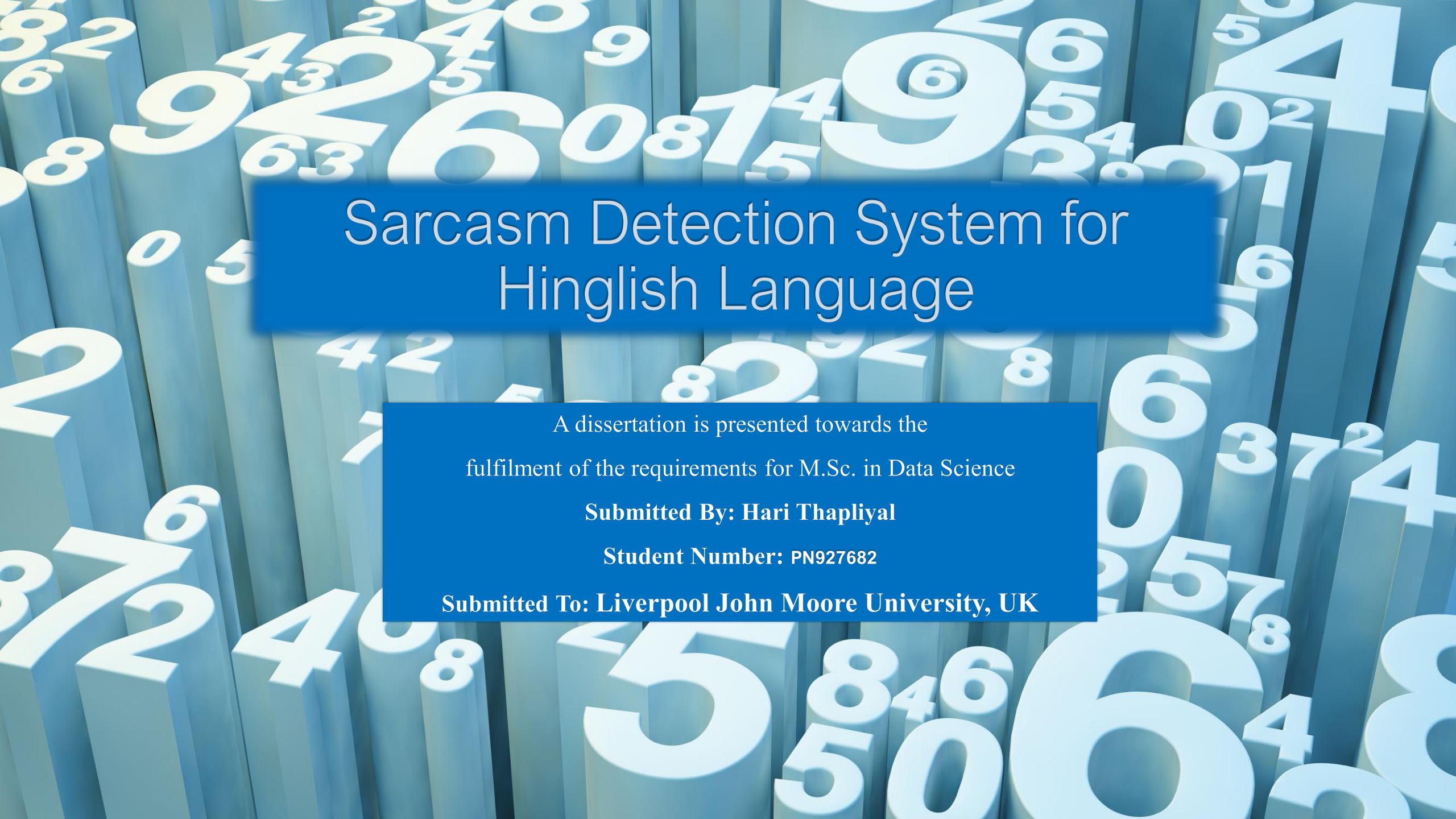


Sarcasm Detection System for Hinglish Language

A dissertation is presented towards the

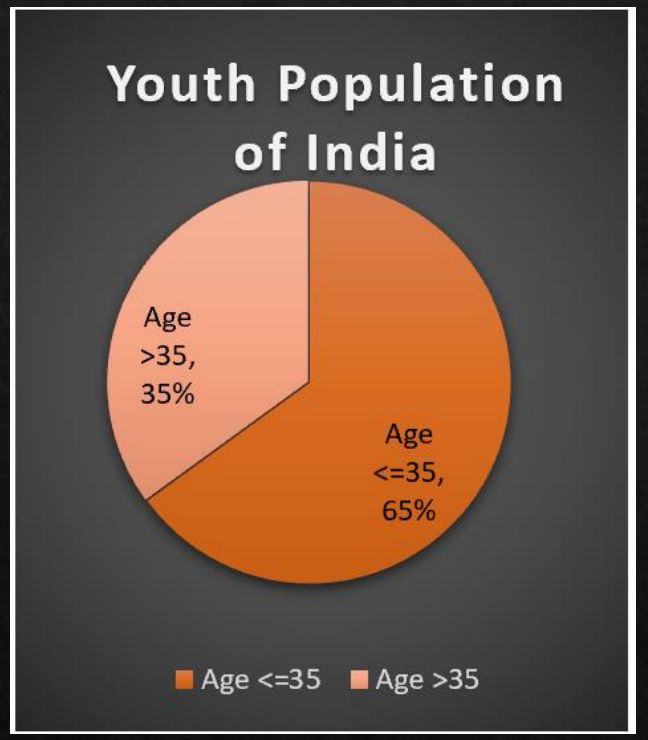
fulfilment of the requirements for M.Sc. in Data Science

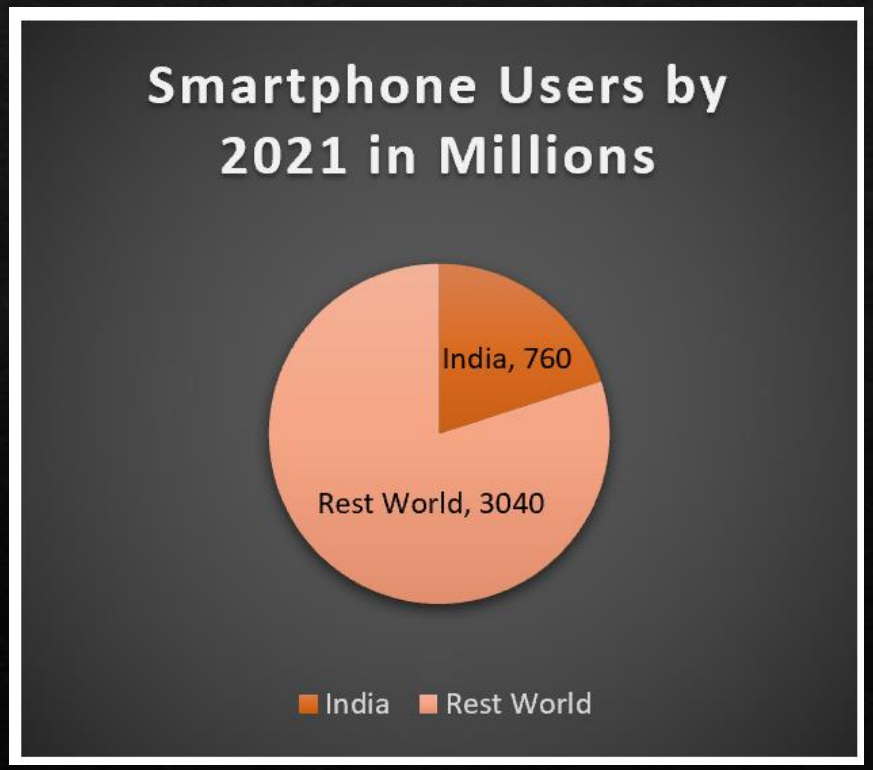
Submitted By: Hari Thapliyal

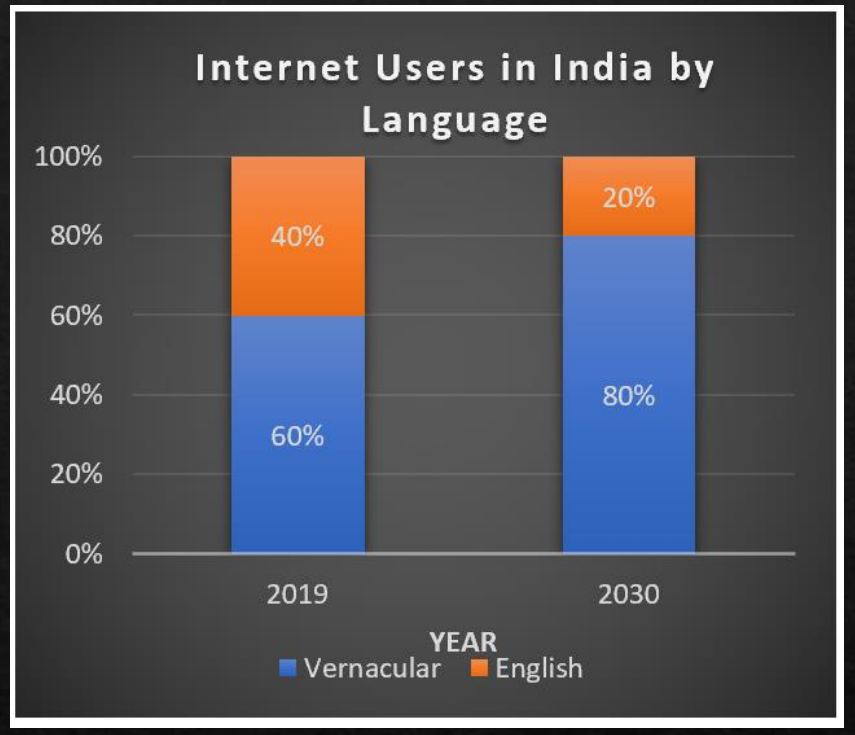
Student Number: PN927682

Submitted To: Liverpool John Moore University, UK

Introduction

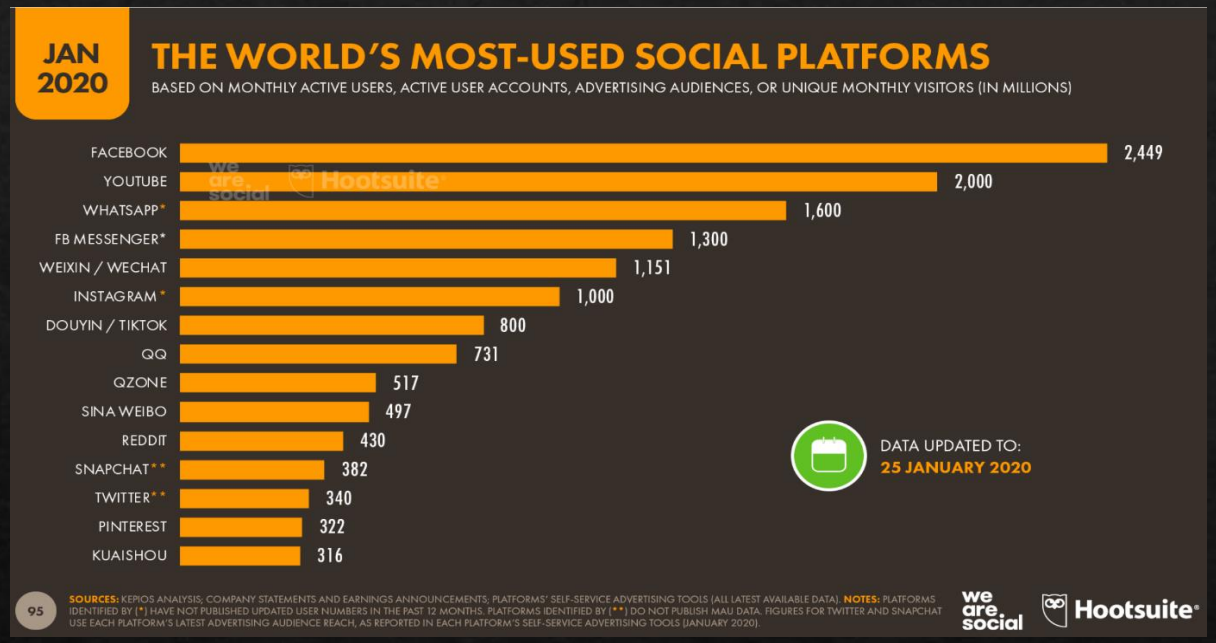


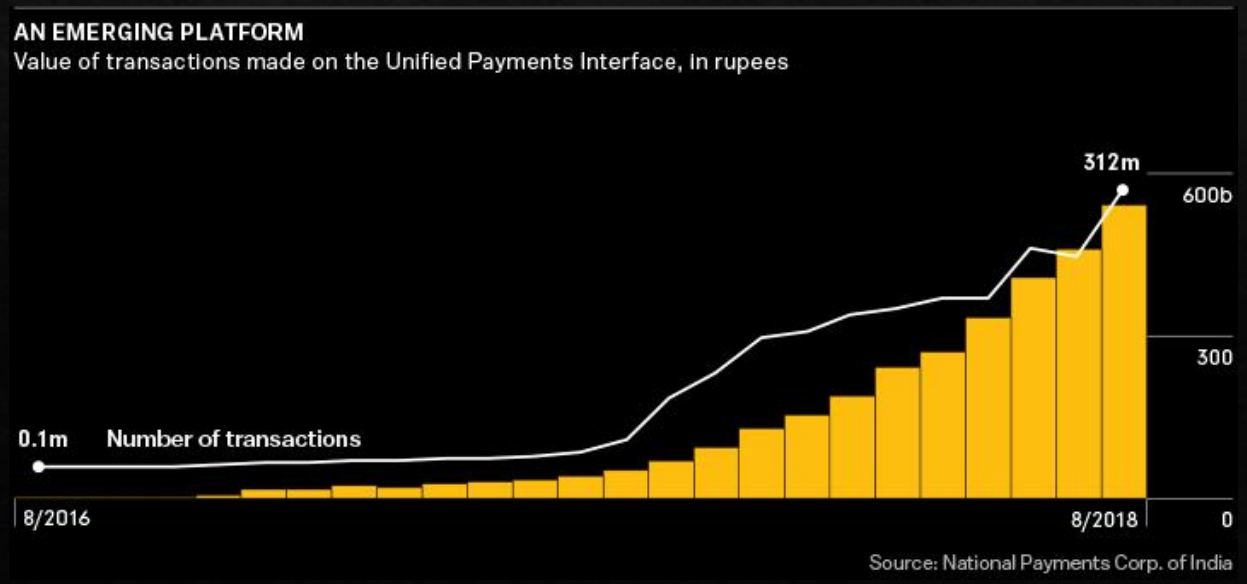


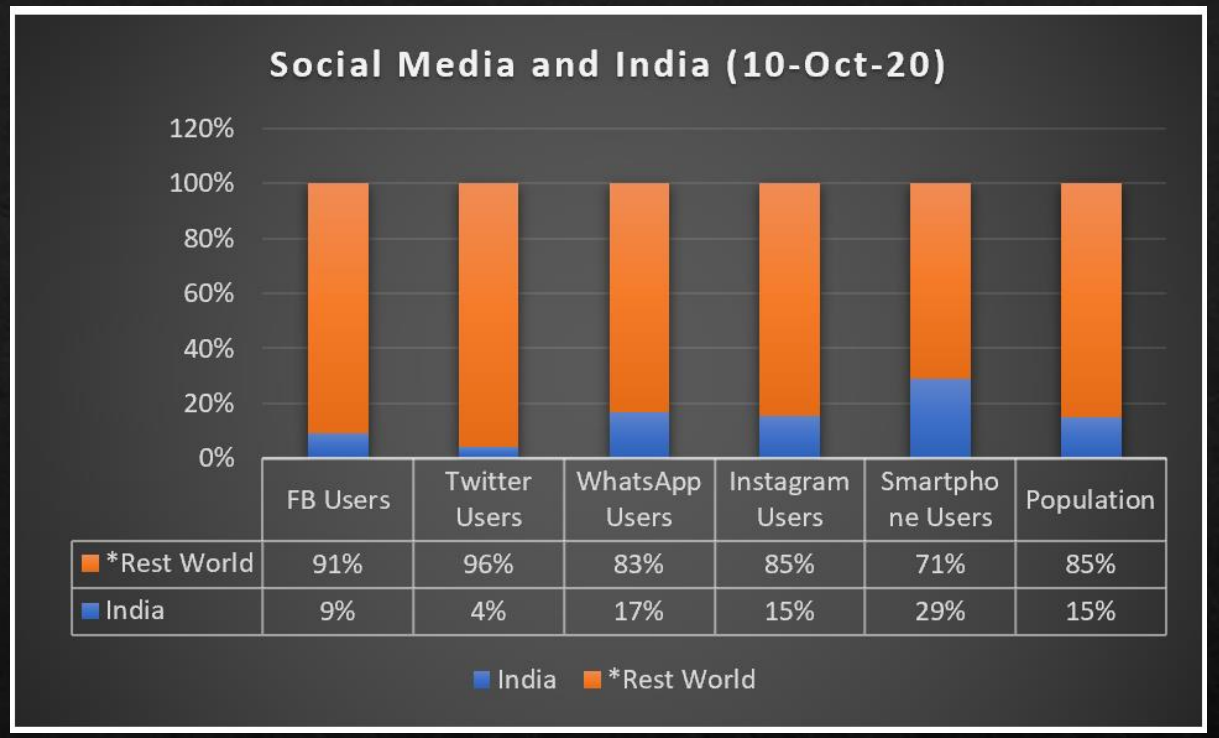


Source: https://en.wikipedia.org/wiki/Internet_in_India

Introduction





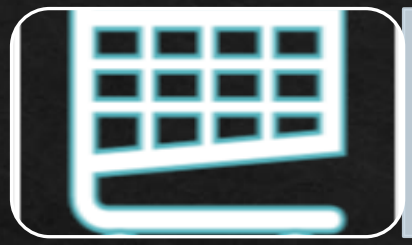


	FB Users	Twitter Users	WhatsApp Users	Instagram Users	Smartphone Users	Population
*Rest World	91%	96%	83%	85%	71%	85%
India	9%	4%	17%	15%	29%	15%

https://fortunly.com/statistics/whatsapp-statistics/#gref
https://www.omnicoreagency.com/facebook-statistics/
https://www.omnicoreagency.com/twitter-statistics/
https://backlinko.com/instagram-users

*Rest Word- In some case data of all countries were not available so we have taken top 10 countries

Problem Statement



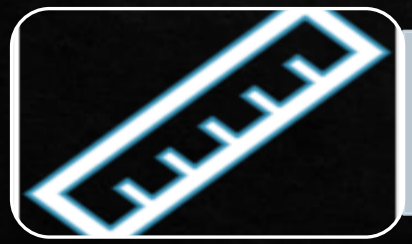
- Digital India program of GOI, recent COVID19 pandemic, youth population, surge in literacy and education, affordable cost of mobile-phone and internet connection are the catalyst to huge content surge in Indian languages.
- People are buying online, consuming service online and giving feedback online.

HINGLISH

- Rise of a new language called Hinglish. This is written in two scripts (Devanagari and Roman) and adopts words from multiple language. Most of the words are taken from Hindi language and written in Devanagari script.

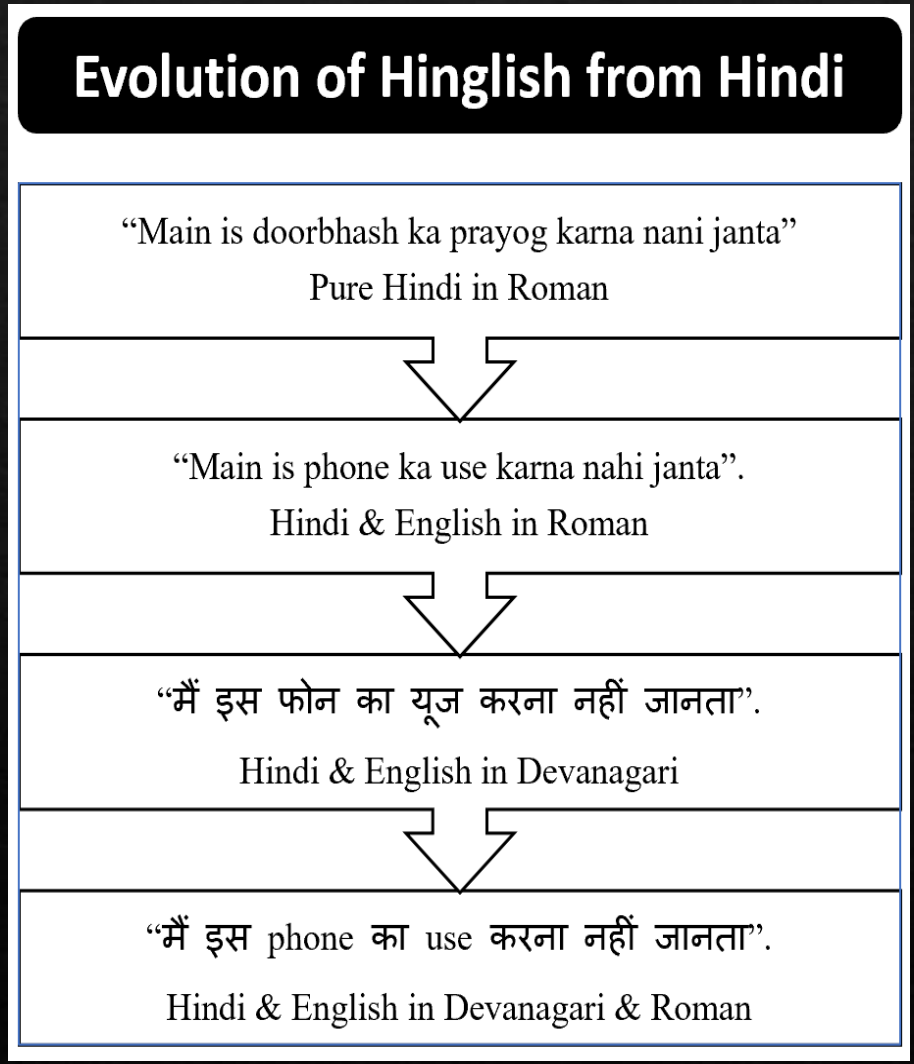
- Hindi people population is using this language in their all communication, specially to give feedback and social media communication.
- Generally, more educated people have more diplomatic language for the feedback.

- Lots of work has been done for sentiment analysis for English language but less work has been for sarcasm detection. Although some work has been done for sarcasm detection in Hindi language but no significant work is available for sarcasm detection in Hinglish language.



- Keeping this volume, demand and need in mind, we want to develop a sarcasm detection system for Hinglish language which can work for all social media content, reviews, comments, and feedbacks.

Evolution of Hinglish



Aim & Objective

- To create Hinglish language dataset with minimum 2000 sentences, which can be used for training and testing a sarcasm detection system of Hinglish Language
- To develop a sarcasm detection models
- To check the effectiveness of Transfer learning for our work.
- To understand which embedding model or library works best for Hinglish language.

Literature Review

- We reviewed 32 research papers on Sentiment Analysis, Emotion Detection, and Sarcasm Detection.
- Most of the work was done using English dataset. Some work has been done in Hindi but that is limited to twitter dataset.
- We didn’t find any Sarcasm detection work using Hinglish language.
- Depending upon dataset different metrics has been used to evaluate the performance of these systems. In most cases where dataset is balance Accuracy is used to measure the performance.
- Accuracy of these systems varies from 55.59% to 99.79% depending upon data source, domain, text script (Roman, Devanagari, Tamil etc.) and language (English, Hindi, Marathi, Tamil, German, Spanish etc.) used for model building

https://github.com/Hari-thapliyal/SDSHL/blob/master/docs/final/Summary-of-Sarcasm-Papers.pdf

Literature Review

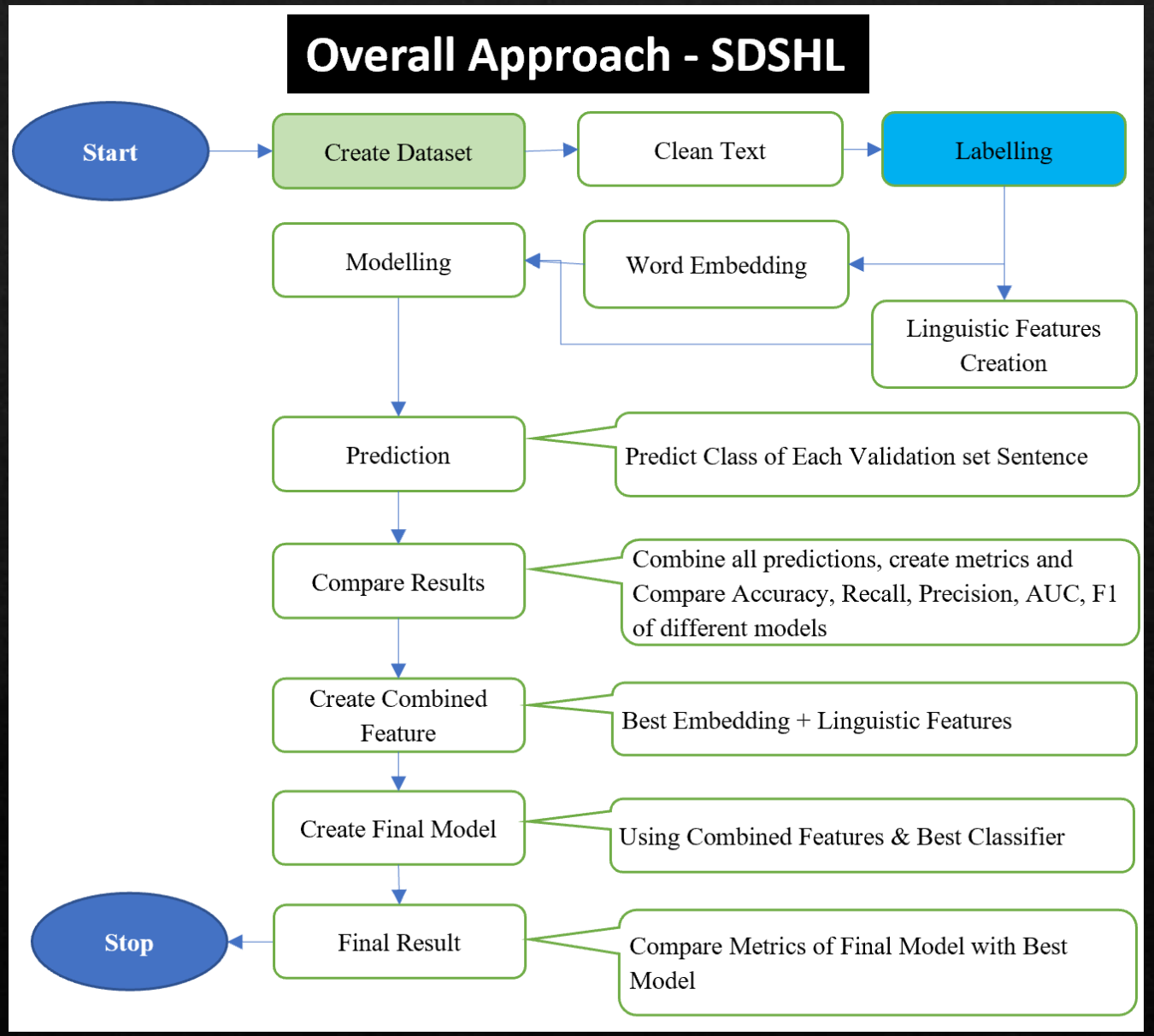
Classification Type - Feature Type
Discussed in Section Number

		Feature Types		
		LFS	Embedding	Both
Classification Type	Rule Based	2.5.1	x	x
	Classical ML Algorithms	2.5.2	2.5.3	2.5.4
	CNN	2.5.5	2.5.6	2.5.7
	Transformers	x	2.5.8	x
	Transfer Learning	x	2.5.9	x

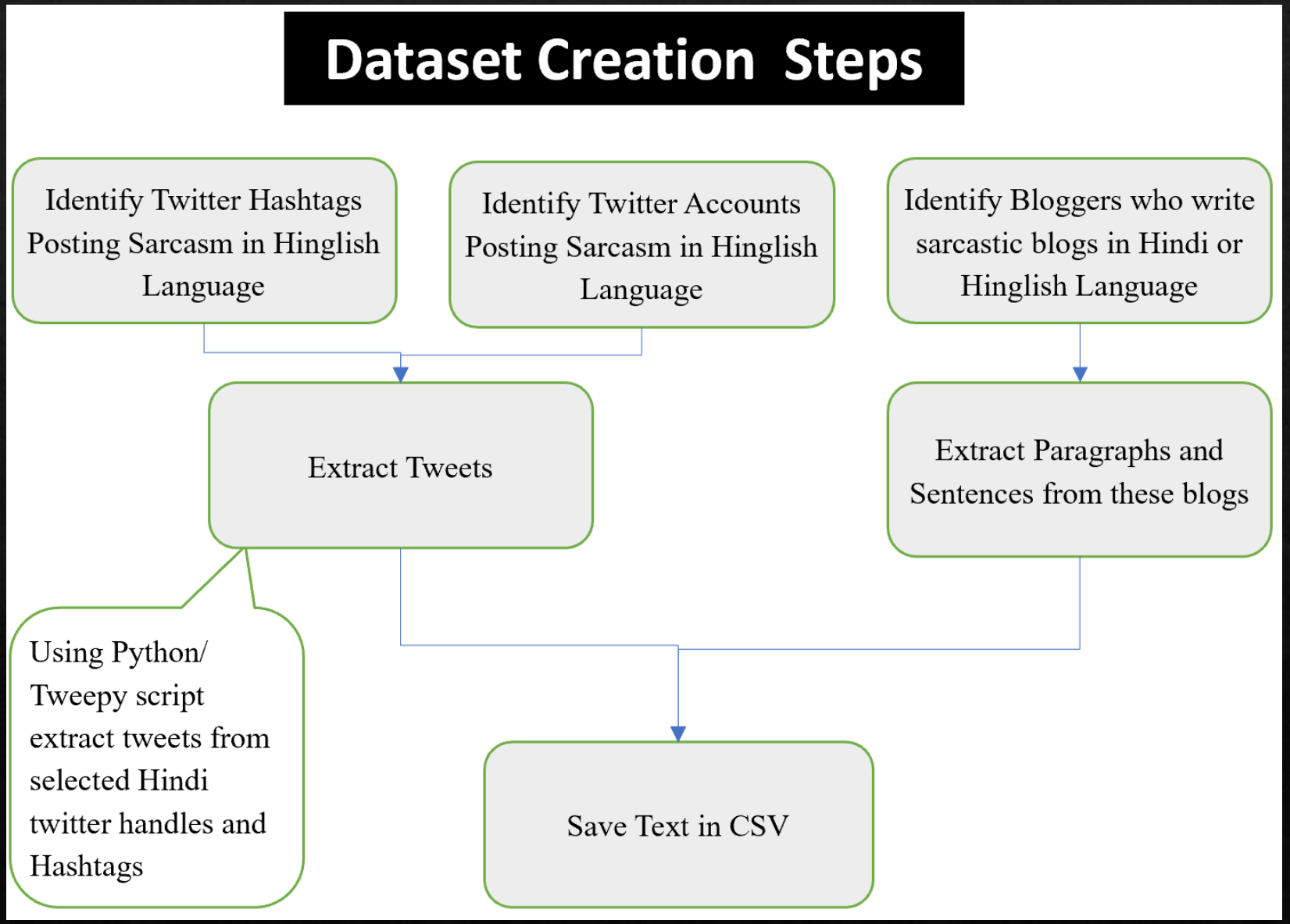
Type of Sarcasm Detection Systems

Architecture Based	• Rule based, Classical ML, CNN Based, Transformer Based
Domain Based	• Politics, IT, Finance, Medical, Law etc.
Mode of Communication Based	• Text, Voice, Body Language, Image, Multi-modal
Time of Detection Based	• Batch, Realtime
Language and Script Based	• English, German, Spanish, Hindi, Tamil, Bangla etc

Methodology



Dataset Creation Steps



Classifier, Embedding, TL Techniques

Classifiers

1. Logistic Regression (LR)
2. Light Gradient Boosting Method (LGBM)
3. Naïve Bayesian (NB)
4. AdaBoost (ADB)
5. Support Vector Machine (SVC)
6. Gradient Boost Classifier (GBC)
7. Random Forest Classifier (RFC)
8. XGBoost (XGB)
9. Decision Tree (DT)
10. Perceptron

Word Embedding

No Transfer

1. TFIDF
2. Word2Vec
3. BOW
4. fastText

Transfer Embedding

1. IndicBERT
2. Multilingual BERT
3. fastText Wiki
4. fastText Indicnlp/ IndicFT

Feature Engineering

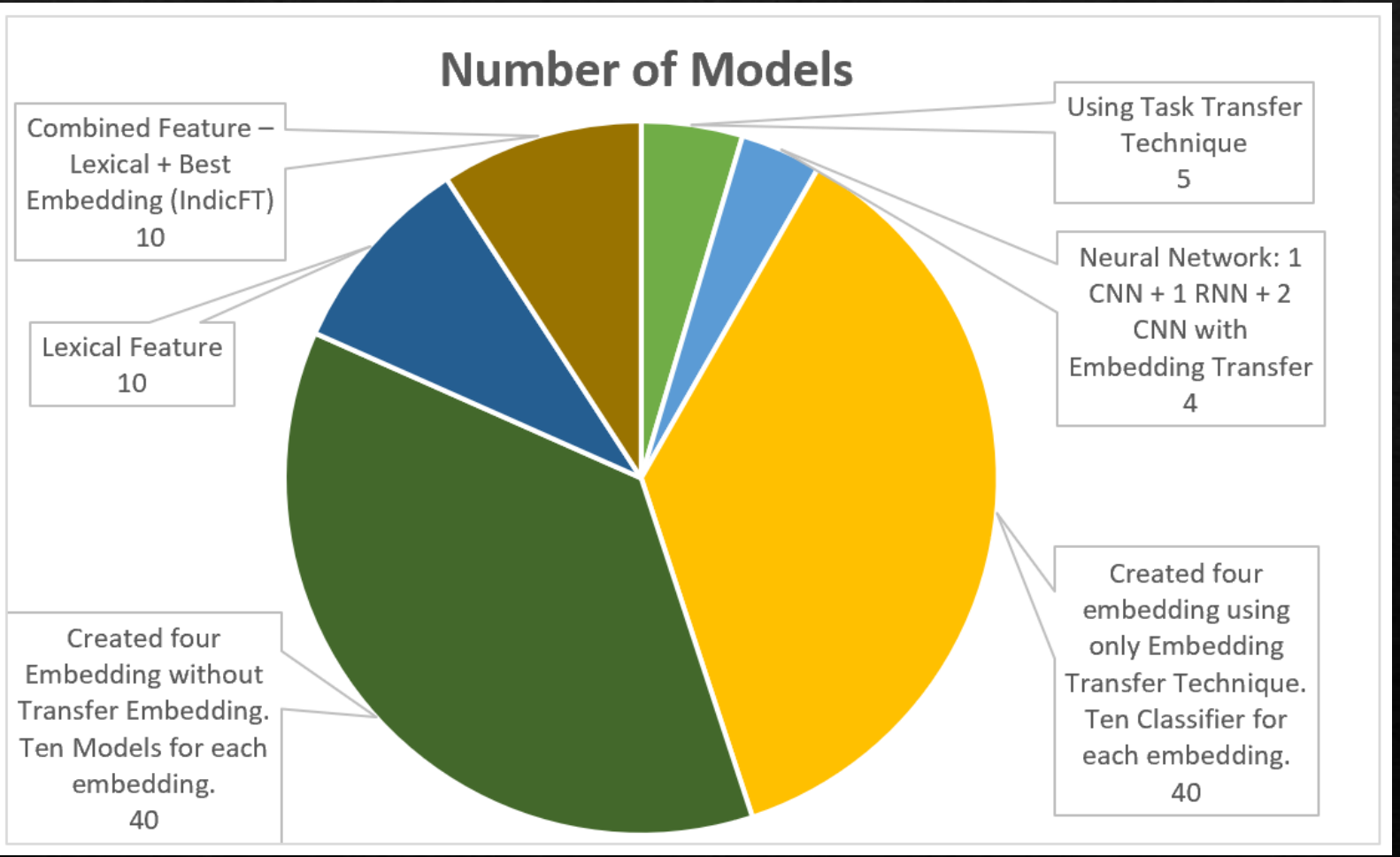
1. Lexical Feature
2. Combined = IndicFT + LexicalFeature

Task Transfer

1. mBERT (Pyrotch)
2. mBERT (Transformer)
3. IndicBERT
4. IndicFT
5. fastTextWiki

Models Types

Twelve classifiers are used, ten embedding used. Six approaches used to create 109 models are summarised in the graph.



Results – Best Models

Top 10 Best Models

Classifier	Embedding Name	AUC	Accuracy	Recall	Precision	F1
NB	fastTextWiki	0.80	0.76	0.78	0.75	0.76
TT	fastTextWiki	0.81	0.76	0.71	0.79	0.75
NB	IndicFT	0.77	0.74	0.70	0.76	0.73
LR	IndicFT	0.78	0.74	0.70	0.75	0.73
SVC	IndicFT	0.79	0.74	0.71	0.76	0.73
ADB	IndicFT	0.79	0.74	0.72	0.76	0.74
XGB	IndicFT	0.79	0.74	0.70	0.76	0.73
NB	Combined	0.79	0.74	0.76	0.74	0.75
PyrotchTT	mBERT	0.80	0.74	0.69	0.76	0.72
SVC	fastTextWiki	0.81	0.74	0.67	0.79	0.72

Task Transfer Learning

Embedding Name	AUC	Accuracy	Recall	Precision	F1
fastTextWiki	0.81	0.76	0.71	0.79	0.75
mBERT (Pytorch)	0.80	0.74	0.69	0.76	0.72
IndicFT	0.81	0.74	0.71	0.76	0.74
mBERT (Transformer)	0.60	0.58	0.65	0.57	0.61
IndicBERT (Transformer)	0.61	0.58	0.63	0.57	0.60

Best Classifier for Best Embedding

Embedding Transfer: fastText Wiki

Classifier	AUC	Accuracy	Recall	Precision	F1
NB	0.80	0.76	0.78	0.75	0.76
TT	0.81	0.76	0.71	0.79	0.75
SVC	0.81	0.74	0.67	0.79	0.72
LR	0.81	0.72	0.66	0.75	0.70
XGB	0.78	0.71	0.64	0.74	0.69
RFC	0.79	0.71	0.65	0.74	0.69
LGBM	0.78	0.70	0.63	0.72	0.67
ADB	0.79	0.70	0.65	0.73	0.69
GBC	0.78	0.69	0.62	0.72	0.67
CNN	0.74	0.65	0.74	0.63	0.68
Perceptron	0.63	0.63	0.36	0.78	0.49
DT	0.64	0.63	0.63	0.63	0.63

Lexical Features are Not Good

Lexical Feature Engineering

Classifier	AUC	Accuracy	Recall	Precision	F1
LGBM	0.69	0.66	0.70	0.64	0.67
GBC	0.71	0.66	0.71	0.65	0.68
SVC	0.72	0.66	0.69	0.66	0.67
RFC	0.72	0.66	0.75	0.64	0.69
LR	0.74	0.66	0.57	0.70	0.63
ADB	0.68	0.62	0.63	0.62	0.63
DT	0.64	0.61	0.60	0.61	0.61
XGB	0.64	0.60	0.63	0.59	0.61
NB	0.69	0.58	0.32	0.67	0.43
Perceptron	0.50	0.50	0.00	0.00	0.00

Conclusion

- Two embedding transfer fastTextWiki and IndicFT both gives competitive results 76% and 74% accuracy respectably with NB classifier. Both of these are fastText based embedding.
- Task transfer gives the best results, highest accuracy 76% when fastTextWiki pretrained model is used for classification
- IndicBERT or mBERT Task transfer with transformer implementation are not giving good results.
- All other models which were created using our own embedding could not perform good with any classifier used.
- The best result of Lexical features is with LGBM classifier. Accuracy; 66%. Lexical features are not effective in sarcasm detection for Hinglish text.
- NB & SVC are the best classifier provided good embedding are used.
- Transliteration is quite complicated task and it need separate focus otherwise in future also we may not see enough improvement in Hinglish language-based models.

Limitation

- We included only two scripts Devanagari and English. If text is in any other script, we will not get good results.
- Trained only on general text and not related to any specific domain
- Training dataset has more twitter data and less blog data but performance on blog data is better than twitter data. We are assuming that this may be because of high structure and consistency in blog text than twitter text.

Future Recommendation

Dataset	• Our dataset has only 2000 sentences. To make a stable model we need more data for this sarcasm classification task. Hence, in future work we should focus on expending the dataset with Hinglish text.
Classifier & Task Transfer	• NB & SVM are good classifier provided good embedding is chosen • Task transfer yields good results when fastTextWiki or IndicFT or mBERT (pytorch) models are used • We should try other transformers like GPT
Embedding	• We used mBERT, fastTextWiki, IndicFT and IndicBERT models to finetune and transfer embedding. These embeddings are Hindi based but we need to develop Hinglish based embedding from scratch
Transliteration	• Before we start fine tuning our model using Embedding transfer techniques, we should transliterate all text in Devanagari script. Hindi/Non-English word in Roman script words and English words in Roman should be transliterated into Devanagari

नमस्ते

नमः ते