

Highlights

Literature Review on Sarcasm Detection In Hinglish

Hari Thapliyal

- Created a dataset for Hinglish Language Sarcasm Detection.
- Created various embeddings using different techniques.
- Evaluated various embedding for Sarcasm Detection in Hinglish language text.
- Evaluated various Classifiers and Transformer to classify Sarcastic text of Hinglish language.

Literature Review on Sarcasm Detection In Hinglish

Hari Thapliyal (Researcher)

^adasarpAI, Lord Krishna Green, Doon University Road, Dehradun, 248001, Uttarakhand, India

ARTICLE INFO

Keywords:

Hinglish Language Text,
Sarcasm Detection in Hinglish Language,
Embedding for Hinglish Language,
NLP for Hinglish Language

Abstract

Hindi is third ¹ most spoken language on our planet. Like English which is written in Roman script, Hindi also does not have its own script but almost all the Hindi speaking people write Hindi in Devanagari script. Hinglish is a mix language and it is spoken by Hindi speaking, English educated people and they can add words from other Indian languages during their conversation. Unlike Hindi Hinglish has its own script and this script is called Hinglish script. Hinglish script has characters borrowed characters from Roman and Devanagari scripts. (Wikipedia) states that 65% of Indian population is under 35 years age. Several disruptions like low cost mobile phone, extremely cheap data, digital India initiatives by government of India has caused huge surge in Hinglish language content. Hinglish language context is available in audio, video, images, and text format. We can find Hinglish content in comment box of online product, news articles, service feedback, WhatsApp messages, social media like YouTube, Facebook, twitter etc. To engage with consumer, it is extremely important to analyse the sentiments, but to perform sentiment analysis it is not possible to read every comment or feedback using human eyes. With the increasing number of education and sophisticated people in Indian society it is obvious that people do not say negative things directly even when they want to say. Generally, an educated mind is more diplomatic than less educated. Due to this reason educated people use more sarcastic language; they say negative things in positive words. Thus, it becomes necessary to identify the true sentiments in the text available on social media or product review or comment pages. In this paper we are demonstrating a system which can help in automatic sarcasm detection in Hinglish language. In this work no word, either Indian language words written in Roman or English word written in Devanagari is translated or transliterated. We developed our dataset with the help of 3 Hinglish language speakers. Out of 10 embeddings created in this work, 4 embeddings were finetuned using transfer embedding, another 4 embedding were built using training data and standard libraries, one from lexical features and one from lexical features + best embedding. In this work we used ten classification libraries for classification work and developed 109 classification models, including 4 classification models developed using neural network. We analysed the performance of those models against the embedding and classifier used. Our best model with fastTextWiki embedding and Naïve Bayesian classifier gives 76% accuracy, 78% recall, 75% precision, 76% F1 score and 80% AUC.

1. LITERATURE REVIEW

Lot of work has been done in English language sarcasm detection and authors mentioned different challenges in sarcasm detection, although results are not that great as for any other classification or other sentiment analysis problems. Challenges exists because of context understanding, missing context, domain, culture, different words, or expression used by people to flip the meaning etc. There is not much work done in Hinglish Language Sarcasm detection. Hinglish language has a separate set of challenge like mixing script, mixing language, highly morphological words, using same morphology on English language words, meagre size of corpus etc.




Let us take one English verb “do”, in Hindi, it can be used like कर्ता (noun), करता (verb with male), करती (verb with female), करूंगा (future tense with male), करूंगी (future tense with female), करेंगे (future tense with plural), किया (did, done), करो (request,

must do) करें (please do) etc. these all are with different gender, mood and tenses. However, in English we have inflection like do, does, did, done. These inflections in Hindi are such that even without using pronoun sentence is meaningful. For example, करता है = वह करता है. Even without pronoun वह sentence is correct, complete, and meaningful. While this is not true in the case of English language.

Now, let's take another example but this time we take noun “Ram”. राम का, राम ने, राम को, राम द्वारा, राम में, राम पर, राम के लिए, राम पर and many times you will see letters are written together. We never see any word like “ByRam” in English but in Hindi रामने and राम ने both have same meaning. In sanskrit we call it Vibhakti (विभक्ती)

2. History of Auto Sarcasm Detection

In this section we are focusing on a brief history of automatic sarcasm detection in the English and Hindi language. Different approaches are used by various researchers over the period of last 20 years. Recently we see surge of machine learning algorithms, neural

 hari.prasad@vedavit-ps.com (H. Thapliyal)
 www.dasarpai.com (H. Thapliyal)
 orcid(s): 0000-0001-7907-865X (H. Thapliyal)

networks, and transformers in sarcasm detection. In this document we are trying to analyze features used, algorithms used and results achieved by different researchers.

In 2002, (Turney, 2002) in their work “The perfect solution for detecting sarcasm in tweets #not” presents an unsupervised learning-based algorithm for classification of review in English language. Their dataset was opinion survey of product. They used Semantic Orientation (SO) to perform this work. SO of a phrase is calculated using adverbs and adjectives used in the phrase. The experiments were done for text of various domains like automobiles, banks, movie review and travels. The results of this experiment vary from domain to domain between 66 to 84%. The power of this SO in Hinglish language sarcasm detection can used and verified.

In “Clues for detecting irony in user-generated contents” (Carvalho, Sarmiento, Silva and Oliveira, 2009) used emoticons, heavy punctuation marks, quotation marks and positive interjections, onomatopoeic expressions for laughter to predict the sarcasm. They concluded that if sentence has more laughter related feature than there are high chances it will be ironic. But heavy use of punctuation and quote is also sign of sentence being sarcastic.

In “Semi-supervised Recognition of Sarcastic Sentences in Twitter and Amazon” (Davidov, Tsur and Rappoport, 2010) used English language dataset which has twitter and amazon reviews. They created features based on Meta Tag (User, Company, Product, Title, Author), Link, HashTags, Punctuation. They used KNN classifier and they found F1 on amazon data is 78% and F1 on twitter data is 83%.

In “Identifying Sarcasm in Twitter: A Closer Look.” (González-Ibáñez, Muresan and Wacholder, 2011) used English Tweets and found Accuracy with SVM classifier varies between 55.59% to 75.78% depending upon tweet format. They used following features. • Lexical Features: unigram, dictionary based (Linguistic Processes (e.g., adverbs, pronouns), Psychological Processes (e.g., positive and negative emotions), Personal Concerns (e.g, work, achievement), and Spoken Categories (e.g., assent, non- fluencies)) + WordNet Affect + interjection + punctuation • Pragmatic Features: positive emotions like smily, negative emotions like frowning face. ToUser like @Name • χ^2 test to select features

In “Baselines and bigrams: Simple, good sentiment and topic classification” (Wang and Manning, 2012) claims that although SVM and NB both are good for text classification. However, NB does better than SVM when document is small size but in case document size is bigger than SVM performs better than NB.

(Liebrecht, Kunnean and Van den Bosch, 2013) demonstrated a sarcasm detection system in “The perfect solution for detecting sarcasm in tweets #not”. This

system was developed for tweets in Dutch language. They used 78,000 sarcastic tweets, along with normal tweets dataset, while adding normal tweet ensured that none of the normal tweet is part of sarcastic dataset. Split the sarcastic tweets into train-test and added with normal tweet into train dataset to train the model. Then test the model using test dataset which has only sarcastic tweets. Their experiments leads to AUC of .79. This paper gives an overall approach of building sarcasm detection system in other than English language. But it does not address the problem which Hinglish language has. Their test train split and model training approach looks good for non-English language.

(Asghar, Kundi, Khan and Ahmad, 2014) developed a system to detect negative, positive, and neutral sentiments for English language tweets. In their work “Lexicon-Based Sentiment Analysis in the Social Web” they claimed their system can detect and score the slang used in the tweet. This system has Accuracy of 92% for binary classification and 87% for multinomial classification. An approach to get tweets clean text is discussed for English language tweets.

In “Sparse, contextually informed models for irony detection: Exploiting user communities, entities and sentiment.” (Wallace, Choe and Charniak, 2015) claimed that capturing previous and following comments on Reddit improves classification performance. They used POS features like NNP, sentiment, subreddit (like progressive or conservative; atheism or Christianity). They used reddit irony corpus.

In “Sarcasm Detection on Twitter: A Behavioral Modeling Approach.” (Rajadesingan, Zafarani and Liu, 2015) mentioned that Users behavioral information is also beneficial as it captures useful contextual information in Twitter post. They created 335 SCUBA in following categories Sentiment Score, Sentiment Transition between past and present, Sarcasm as a complex form of expression, emotion (mood, frustration, affects and sentiments), language familiarity, sarcasm familiarity, environment familiarity, written expression related, structural variation. With logistic regression they achieved highest accuracy of 83.46%.

In “Figurative messages and affect in twitter: Differences between# irony,# sarcasm and# not.” (Sulis, Fariás, Rosso, Patti and Ruffo, 2016) evaluated novel set of sentiment, structural and psycholinguistic features. They used different lexical resources developed for English language. They used classification algorithms NB, DT, RF, LR, SVM. They used english language resources like HL, GI, DAL, ANEW, SS, EmoSN, EmoLex, SN, LIWC, EWN, SWN, AFINN. They found Irony shows highest experience of joy, anticipation and trust. It shows least behaviour of sadness, fear, surprise, disgust, anger.

In “Irony detection in twitter: The role of affective content” (Fariás, Patti and Rosso, 2016) presented

the performance of knn classifier which was fed with a feature set that captures a wide range of linguistic phenomena like structural, emotional. They used SVM, DT, NB classifiers and achieved accuracy between 73-96% depends upon datasets and classifier used.

In "An Empirical, Quantitative Analysis of the Differences between Sarcasm and Irony" (Ling and Klinger, 2016) explored syntactic and sentiment related features. They demonstrated that separating sarcasm from normal text with 90% accuracy but separating irony from sarcasm achieved only 79% accuracy. In this experiment they used English language twitter data.

In "Unsupervised irony detection: A probabilistic model with word embeddings" (Nozza, Fersini and Messina, 2016) introduced a novel unsupervised probabilistic modeling approach to detect irony. They developed a Topic Irony Model (TIM) and used word embedding to develop a unsupervised irony detection system. This experiment was conducted on education, politics and humor domain. F1 score achieved using TIM+WE system is between 82.92 to 88.34% depending upon domain.

Sarcasm is the major factor which can flip the meaning of a written or spoken phrase. To avoid the negativity people use positive words to communicate negative message, (Desai and Dave, 2016). In their work "Sarcasm detection in hindi sentences using support vector machine" they have used libsvm algorithm for multiclass classification. This paper uses 5 grades of sarcasm Non-Sarcastic, Mild Positive Sarcastic, Extreme Positive Sarcastic, Mild Negative Sarcastic and Extreme Negative Sarcastic. This work demonstrates the accuracy between 60% to 84% depending upon, whether sentence has any clue like emoticon, tag etc of sarcasm. This work suggests usage of lexical, pragmatic, and linguistic features along with emoticons, hashtag, punctuation marks to detect the sarcasm.

In "Harnessing Online News for Sarcasm Detection in Hindi Tweets" (Bharti, Babu and Jena, 2017) mentions that sarcasm detection is one of the most complex work in Hindi Language and the reason for that is words in Hindi language are rich in morphology. This paper discusses a system to sarcasm detection in Hindi tweet but for that it is taking help of online news related to the tweet. This work demonstrates accuracy of 70.4%

In "A novel automatic satire and irony detection using ensembled feature selection and data mining" (Ravi and Ravi, 2017) used Linguistic, Semantic, Psychological, unigram features. They further created group of features like LIWC features (L), TAALES Features(T), Unigram Features(D), then created ensemble feature subset. To select the useful feature they used IG, GR, Chi, CORR, TSTAT. They used English language text from Newswire, Satire news articles and Amazon. Experimenters used SVM (Linear, RBF, Sigmoid, Polynomial), LMT, LR, RF, NB, BN, MLP classifiers. They

achieved the highest F1 score 96.58% with (L+T+D features) + GR feature selector + SVM RBF Classifier

In "Exploring the fine-grained analysis and automatic detection of irony on twitter" (Hee, Lefever and Hoste, 2018) used SVM classifier in combination of lexical, semantic and syntactic features passed through an SVM classifier and it outperformed LSTM deep neural network approaches.

(Potamias, Siolas and Stafylopatis, 2020) published their work "A Transformer-based approach to Irony and Sarcasm detection". In this paper they mentioned figurative language (FL) is ubiquitous and irony and sarcasm detection is challenging. They observed that social media users violates grammar rules and heavily use figurative language to communicate. Author states that classical machine learning algorithms such as k-Nearest Neighbors (KNN), SVM, and tree-based models (Decision Trees, Random Forest) are inappropriate for real world applications, due to their demand of hand-crafted feature and exhaustive pre-processing to make text clean for performing any NLP task. To develop a reasonable KNN or SVM based model, there should be a lot of effort to embed sentences on word level to a higher space. After this only a classifier may recognize some patterns. They used Deep Learning methodologies for sarcasm detection. They conducted various experiments using ELMo, FastText, XLNet, BERT-based, BERT-uncased, RoBERTa, UPF, ClaC, DESC, USE and NBSVM.

As per (Zhang, Wu, Zhao, Li, Zhang, Zhou and Zhou, 2020b) in their work "Semantics-aware BERT for Language Understanding" existing language representation models like ELMo, GPT and BERT exploit plain context-sensitive features such as character or word embeddings. These models rarely consider incorporating structured semantic information which can provide rich semantics for language representation. Authors proposed semantics-aware BERT and claims it is a simple in compare to BERT and more powerful. SemBERT (large) could predict the sentiment with accuracy as 94.5% on SST2. This looks impressive results however this limited to English language. In the absence of sufficient corpus size in Hinglish we cannot perform similar experiment using BERT.

In their work "Multi-Rule Based Ensemble Feature Selection Model for Sarcasm Type Detection in Twitter" (Sundararajan and Palanisamy, 2020) attempted to detect the sarcasm and further classify sarcasm into four categories namely rage, rude, polite and deadpan. Authors exploited a twitter dataset for this work. They extracted four types of features Lexical, Intensifiers, Pragmatic, Uppercase words. They extracted following 20 features from this tweet dataset: 1. noun count, 2. verb count, 3. positive intensifier, 4. negative intensifier, 5. bigram, 6. trigram, 7. skip gram, 8. unigram, 9. emoji sentiment, 10. sentiment score, 11. interjections,

12. punctuators, 13. exclamations, 14. question mark, 15. uppercase, 16. repeat words count, 17. positive word frequency, 18. negative word frequency, 19. polarity flip, and 20. parts of speech tagging. Using these features authors developed 8 models using following 8 algorithms 1. Random Forest, 2. Naive Bayes, 3. Support Vector Machine (SVM), 4. K-Nearest Neighbor (KNN), 5. Gradient Boosting (GBC), 6. AdaBoost, 7. Logistic Regression, and 8. Decision Tree. Their final goal is not to classify tweet as sarcastic or non-sarcastic tweet. They wanted to classify sarcastic tweet into four different categories. They reported accuracy of sarcasm detection as 92.7% and accuracy of classification of sarcastic tweet among these four categories varies between 86.61% to 99.79% depending up type of emotion.

For categorization of the sarcastic tweet first they deployed certain rules to create linguistic features (L), sentiment features (S), contradictory features (C). Following this they used different combination of these features to create different ensembles models.

Authors obtained tweets through Twitter API (Tweepy and Twython) on the basis of the following hash tags: #sarcasm, #sarcastic, #Sarcasm, and #notSarcasm. A total of 76,799 tweets are collected, tweets that are non-English and non-Roman script are removed. During the pre-processing steps they removed any URL, Retweet, Link related text from the tweet.

3. Sarcasm Detection Systems (SDS)

Sarcasm is perception of the human receiver about some inputs. "Input" can be of four types. First type of input is text format written in social media, book, newspaper etc. Second type of input can be vocal tone, expressed in some voice communication over phone, face to face meeting, stage show, etc. Third kind of input can be image appearing on some public hoarding, newspaper article, blog post, social media etc. Forth kind of input can be body language of human during face to face interaction or in video.

To understand a message correctly following conditions should be met successfully.

- Speaker speaks in the language which listener can understand
- Listener understand the background
- Listener has technical knowledge about the subject

Beauty is in the eye of beholder. If receiver missed the sarcastic intent of input due to any reason, then will you call that statement sarcastic? This is philosophical debate and, in our work, we will be focusing on text which is marked as sarcastic by different annotators. From receiver's perspective input received can be any of the following four types.

All Weather Sarcastic: Every civilized person will treat those statements as sarcastic. For example, "I like when you treat me like a slave". No matter what the context is, what language is used to communicate this text everybody will say this is sarcastic statement. No other information is required, sentence has complete information and almost all human agree to this.

Conditionally Sarcastic I: More information is required to classify a sentence is sarcastic or not. In the presence of that important information we can confidently say this is normal or sarcastic sentences. This more information may be related to profession, culture, rules, law of the land etc. For example, "I love to beat drum at 5 am in the morning". Some cultures, profession forces their follower, community members to do eating, praying, singing, playing activities at a time so in that profession or community's context it may be normal. Otherwise it is sarcastic.

Conditionally Sarcastic II: Sometimes we need individual event and person specific information to detect sarcasm in sentence and this information cannot be generalized even for the same person at other time. For example "First thing in morning I like to do is cleaning potty of Ruby". If receiver know the context that the speaker is mother and Ruby is new born baby then speaker may say it is sarcastic or non-sarcastic depends upon receivers individual like or dislike. Here even after understanding the full context it is depending upon the receiver who does the classification. But if receiver knows that that speaker is busy CXO and Ruby is his pet name. Then receiver will say it is definitely a sarcastic statement.

Non-Sarcastic: Normal sentences with straight forward meaning without hiding any intention and no scope of different interpretation.

Sarcasm detection system is one which can flag an "input" provided to the system as sarcastic or non-sarcastic. In the context of our project "input" mean text and no other type of input like body language, image, video, speech etc. Even with text as "input" we are particularly dealing with one or two liner text appearing on social media or day to day communication. We are not dealing with long chain of text like a paragraphs, a page, a chapter or a book. We are interested in developing state of art sarcasm detection system for Hinglish language. Systems takes input as one or two sentences with full context and returns True or 1 if Hinglish sentence is sarcastic else returns false or 0. If the context is missing, then system may fail to predict correctly.

3.1. Generic Text-based Sarcasm Detection System (GTSDS)

There are many dimensions of complexity in any sarcasm detection. General purpose text-based sarcasm detection system means a system which can

detect sarcasm in any text. Before building a GTSDC we need to answer following question.

- Can we develop one SDS which can detect sarcasm expressed in all human language like English, Hindi, Japanese, Chinese, Spanish etc.?
- Can we develop one SDS where text written in any script like Devanagari, Roman, Hebrew, Chinese etc. can be identified?
- Can we develop one SDS which can detect sarcasm from text, written in simple language vs figurative language which is full of proverbs and coded words?
- Can we develop one SDS which can detect sarcasm from the text, which is using words from any business domain like politics, philosophy, medical practitioners, lawyers etc.?
- Can we develop a SDS which can detect sarcasm from the text written by the people of different culture like British, North American, Indian, Japanese etc?

Building a general-purpose text-based sarcasm detection is extremely complex task. In this work we are trying to develop a general purpose SDS where the text of two scripts and words from multiple language like Hindi, Sanskrit, Urdu, Punabi, Marathi, Bhojpuri, Avadhi are used. We are aware this is not a complete generic purpose text-based sarcasm detection system and but a small step towards that.

3.2. Feature Engineering in Sarcasm Detection Systems

Researchers have extracted various features from the given text to detect whether sentence is sarcastic or not. These features can be grouped under following categories.

- Lexical: unigram, bigram. These can be created using words or characters.
- Pragmatic: These features are created using emoticons, punctuations and capital letters used.
- Incongruity: Incongruity in the sentences is detected using novel approaches.
- Polarity: Polarity of the noun, adverb, adjectives are counted.
- Syntactical: These features are based on POS (part of speech).
- Idiosyncratic: Sentences are analysed for repetition of any specific word by the speaker. Many people have habit of say words like “I know”, “you know”, “yah yah”, “absolutely”, “like” etc.

- Prosodic: Analysing pattern of words in the sentence, how a specific words is written to emphasis something. For example “It is soooooooooo beauuuuuuuuuuuuuuuutiful”.
- Features based on the Author’s or reader’s profile data: Gender, nationality, religion, education, ideology, familiarity of language etc.
- Features based on the environment: Datetime, current news, messages in past, present state of mind etc.
- Hashtag & @users: Different hashtags used and different users tagged in the message
- Slang: Number of slang used, ratio of slag to normal words, nature of slang word etc.
- Profanity: Any dirty, abusive, naughty, offensive words

There are many creative ways to create hundreds of features under above categories. We will refer all these features as Linguistic Features of the Sentence (LFS)

In their work, (Joshi, Bhattacharyya and Carman, 2018) have used 3 types of features POS, Named Entities, Unigram to predict the disagreement. (Sharma, Sangal, Pawar, Sharma and Bhattacharyya, 2014) in their work “A Sentiment Analyzer for Hindi Using Hindi Senti Lexicon” suggests using bootstrap approach to extract sentiment words from Hindi Wordnet. It has given encouraging results of 87% accuracy in sentiment analysis. We are going to test usefulness of this approach in sarcasm detection.

We have prepared a “[Summary of Papers on Sarcasm Detection](#)”. This presents a summary of these features used by different researchers and the performance reported by them. If you interested to read more, you can refer to the [github repository](#).

3.3. Approaches to Develop SDS

Over the period of last 20 years different approaches are adopted by different researchers. Broadly these can be categorized into following categories. In the following subsections we are analyzing features explored, algorithms used, and results gained by the different researchers. If you want to more about these then you can refer to our work “Summary of Papers on Sarcasm Detection” and History of Sarcasm Detection. Table below presents the summary of approaches used to develop SDS. Numbers written in the cells of the table are section number following the table.

3.3.1. Rule based Approaches (A)

In this approach researcher depends upon the content and context-based of the text. They extracted various Linguistic Features of the Sentence (LFS).

Classification Type - Feature Type Discussed in Number Mentioned in the Section Title			
Classification Type	Feature Types		
	LFS	Embedding	Both
	Rule Based	A	-
	Classical ML Algorithms	B	C
	CNN	E	F
	Transformers	-	H
	Transfer Learning	-	I

Figure 1: Classification Types & Feature Types Mapping

Some experimenters have demonstrated a good performance on sarcasm detection work without using any machine learning algorithm. "Lexicon-Based Sentiment Analysis in the Social Web" by (Asghar et al., 2014) didn't use any classical or neural network based algorithm for this work. They could achieve 95% accuracy by using a) Lexical features- unigram using chi-square test, (b) Pragmatic- emoticons, punctuation marks, capital words, (c) Explicit congruity- related to polarity changed, and (d) Implicit incongruity features.

Just using rule based approaches (Bharti, Babu and Raman, 2018) achieved 87% accuracy on Hindi language tweets and (Sharma et al., 2014) could achieve 85-89.5% accuracy on Hindi language product reviews.

3.3.2. Classical Machine Learning with Linguistic Features (B)

In this approach we can use LFS but classification is done using the classification machine learning algorithms like LR, SVM, RF etc. Many experiments are done using this approach.

(Fafias et al., 2016) demonstrated 73-96% accuracy using classifiers like SVM, DT, NB and feature engineering approaches. (Suhaimin, Hijazi, Alfred and Coenen, 2017) shown 82.5% accuracy with non-linear SVM. Both of these experiments are done on English tweets.

(Sundararajan and Palanisamy, 2020) used English twitter data and shown 86.61% to 99.79% accuracy using classifiers like Random Forest, Naive Bayes, Support Vector Machine, K-Nearest Neighbor, Gradient Boosting, AdaBoost, Logistic Regression, and Decision Tree. They extracted 20 features from the dataset.

In an another interesting work on Instagram image (English text), (Kumar, Singh and Kaur, 2019) has developed a sarcasm detection system with 73% to 88% accuracy. They extracted features like Number of negative words, number of positive words, POS tag, hashtag from the dataset.

3.3.3. Classical Machine Learning with Embedding (C)

In this approach we need not to explore any LFS. Using word embedding word vectors are created and they can be used for creating classification model. During literature survey we could not find any papers which solely rely on word embedding for creating the models.

3.3.4. Classical Machine Learning with Embedding & Other Features (D)

Using this approach, we create LFS along with word embedding for every sentence. (Kumar et al., 2019) used tokens using Classical language toolkit, unigram, bigram. They also used fastText and TF-IDF embedding. Authors used SVM linear kernel, LR, RF, Shallow CNN + Bi-Directional LSTM for classification purpose. In their work "BHAAV- A Text Corpus for Emotion Analysis from Hindi Stories" they were trying to classify emotions in Hindi language sentences. They claim that they could get an accuracy of 62%.

3.3.5. Deep Learning with Linguistic Features (E)

In this approach deep learning neural networks are explored for classification but instead of using word embedding Linguistic features of the sentence are used. (Liu, Ott, Goyal, Du, Joshi, Chen, Levy, Lewis, Zettlemoyer and Stoyanov, 2019) in their work "A2Text-net: A novel deep neural network for sarcasm detection" used CNN and created a novel architecture for sarcasm detection. They tested their model on different dataset and got different results. The results vary between 71%-90% F1 score.

3.3.6. Deep Learning with Word Embedding (F)

Deep learning approaches includes those experiments where experimenters have used CNN, RNN, GRU, LSTM or any variation of neural network. They transformed the text input into vectors using different embedding techniques like TF-IDF, word2vec, fastText etc. (Subramanian, Sridharan, Shu and Liu, 2019) used GRU on English language twitter and facebook dataset and show 89.36% accuracy on twitter dataset and 97.97% accuracy on facebook dataset.

3.3.7. Deep Learning with Embedding & Other Features (G)

In this approach, researchers created features using both the word embedding and LFS. For classification researchers used deep learning networks like CNN. In "CARER: Contextualized Affect Representations for Emotion Recognition" (Saravia, Liu, Huang, Wu and Chen, 2018) used BoW, char n-gram, TF-IDF, Word2Vec, fastText(ch), word-cluster, enriched patterns, Twitter-based pre-trained word embeddings and reweight them via a sentiment corpus through

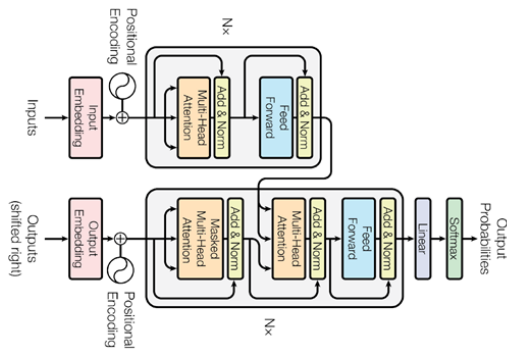


Figure 2: Transformer Architecture

Source: (Vaswani et al., 2017)

distant supervision. Authors used CNN for the classification and claimed an accuracy of 81% using their novel architecture named CARER.

3.3.8. Transformer Based (H)

(Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser and Polosukhin, 2017) in their work "Attention Is All You Need" proposed a novel architecture which named Transformer Model Architecture. A transformer has two units first is encoder, and second is decoder. Subcomponents of transformer architecture are positional encoding, multi-headed attention, feed forward network, masked multi-headed attention, fully connected dense layer and finally Softmax layer.

Several companies are taking lead and exploiting this architecture to build state of art model for NLP tasks. Below is the list of some selected transformer-based models by various companies.

1. GPT from OpenAI by in their paper "Improving Language Understanding by Generative Pre-Training"
2. BERT from Google by (Devlin, Chang, Lee and Toutanova, 2018) in their paper "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding"
3. XLNet from Google & CMU by (Yang, Dai, Yang, Carbonell, Jaime Salakhutdinov and V, 2019) in their work "XLNet: Generalized Autoregressive Pretraining for Language Understanding"
4. ALBERT from Google Research by (Lan, Chen, Goodman, Gimpel, Sharma and Soricut, 2019) in their work "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations"
5. T5 (from Google) by (Raffel, Shazeer, Roberts, Lee, Narang, Matena, Zhou, Li and Liu, 2019) in their work "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer"
6. ELECTRA from Google Research & Stanford University by (Clark, Luong, Le and Manning, 2020) in their work "ELECTRA: Pre-training text

encoders as discriminators rather than generators"

7. RoBERTa from Facebook by (Liu et al., 2019) in their work "Robustly Optimized BERT Pretraining Approach"
8. DialoGPT from Microsoft Research by (Zhang, Yang and Zhao, 2020c) in their work "DialoGPT: Large-Scale Generative Pre-training for Conversational Response Generation"
9. DistilBERT from HuggingFace by (Sanh, Debut, Chaumond and Wolf, 2019) in their work "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter".

(Potamias et al., 2020) developed novel architecture RCNN-RoBERTa in their work "A transformer-based approach to irony and sarcasm detection". They developed this architecture using an existing transformer RoBERTa. As mentioned above RoBERTa is developed by facebook research for natural language processing tasks. This novel architecture by authors could predict the sarcasm with 85% to 94% accuracy. In this paper authors has compared performance of various kind of transformers like ELMo, USE, NBSVM, FastText, XLnet, BERT base cased, BERT base uncased, RoBERTa base model, UPF, ClaC, DESC to compare the performance of their novel architecture.

3.3.9. Transfer Learning Approaches (I)

Training a new model from scratch is expensive and time-consuming work. Therefore, recent trends of Transfer Learning are picking up. Some companies or universities who have plenty of resources to develop new models using large amount of data develops the model of various size. They release the models, which need lesser resources to run, for the consumption by other researchers, who have lessor resources & time at their disposable. These released models are called pre-trained models. We can use models as is or with some fine tuning, based on our need.

The pre-trained models are developed using corpus of some language, some task and text from some domain. The beauty of these models is we can fine-tune them using our data for a task which we want to accomplish. This is called transfer learning. Using transformer-based system we can perform three kind of transfers namely task transfer, language transfer and domain transfer. In the NLP world task means like classification, comprehension, text generation, next word or sentence prediction etc. When we say task transfer it means a model which is created for let us say classification task can be finetuned for next word prediction or any other task. To use the model, we need to convert the text into vectors using embedding provided by the transformer. Recently we see a surge of models of various size in the NLP marketplace. Researcher community is happily adopting those for

their experiments and getting good results compare to other approaches and techniques mentioned earlier.

3.4. Approaches to Handle Key Challenges in Sarcasm Detection

3.4.1. Handling Figurative Languages

Figurative language is the language used by intellectuals or those who have a good command over language. If you take a literal meaning of a sentence written in figurative language you will not get anything useful and meaningful. Many times educated people of the society want to communicate some idea or message but they use simile or old proverbs or chose words which are not directly related to the situation but the gist of that incident or proverb has parallel to the situation in hand. Figurative language is work of intellectual caliber and many times it is not easy even for human to understand the message. For example “My daughter is apple of my eyes” मेरी बेटी मेरे आंख का तारा है” If you miss the presence of figurative language in this sentence then you will miss the meaning of this sentence.

(Potamias et al., 2020) in their work “A transformer-based approach to irony and sarcasm detection” claims their novel architecture RCNN-RoBERTa performs well on the figurative language. (Nozza et al., 2016) in their work “Unsupervised Irony Detection: A Probabilistic Model with Word Embeddings” claims that if we integrate probabilistic models like TIM with word embedding then we get promising results in detecting irony and sarcasm.

3.4.2. Handling Limited Data in Sarcasm

Although we did not find research work which tells how much percentage of our day to day communication is sarcastic, but we know from our day to day communication that percentage is very less. You just observe yourself or family members around for one day and count how many times you used sarcastic language. Due to this reason, we do not have enough good size dataset of sarcastic communication. Hindi & Hinglish being one of the least NLP resource languages has too little data to build a good sarcasm detection system.

Researchers takes either of the two approaches to handle imbalance dataset. In first approaches they do not take more non-sarcastic sentences than they have sarcastic ones, this is under sampling of non-sarcastic sentences. In second approach they do over sampling of sarcastic sentences. But it is extremely complicated to create sarcastic sentences with raw data. It is easy to get non-sarcastic sentences but to build our dataset we will not go for collecting more than 1000 non-sarcastic sentences, because we are planning to have 1000 sarcastic sentences in our dataset.

In either of the cases if dataset size is small for the training purpose, we use cross validation techniques.

In this technique we create multiple fold of the same dataset using random sampling and then use the fold for the training purpose. Let us say our data set has only 1000 records and it is balanced dataset. If we create a 5 folds cross validation for the training purpose then 5 folds of 200 records will be created from these 1000 records. These 5 folds should have same distribution of the classes. Every time we create new folds there will be different set of records in those folds. After 5 folds are created, we can use 4 folds for training and 1-fold for validation purpose. Thus, we run train our model 5 times and every folds gets opportunity to become validation set. If our experiments, we will use 5 fold cross validation to know the best parameters.

3.4.3. Handling Out of Vocabulary Issues (OOV)

To create word embedding vector which represents all the possible words and their possible usage in different context we need a huge corpus. Not only this, if we have huge corpus of political news or short moral stories that will not represent the same words which are used in the context of medical, physics, philosophy, finance etc. For example, “Interest of various stakeholders is increasing in the recent peace talk process”. This is a statement from normal news. But “Banks are continuously increasing interest and it is making capital more costly” is a statement from financial news. Same word “interest” in financial news has different context than when it is used in normal life. To make sure that final word embedding represent all the possible context we need to include corpus of all the possible domain’s data. But this is difficult task as of today. Because of limited good quality corpus from all the domain of business, science, technology, culture etc.

Due to this reason, at training or prediction time, when we are looking for a word vector for a new context and if word embedding is not available then that word becomes OOV word. When our dataset has many OOV words then training task will not be able to generate a model which can perform NLP, NLU task with good results. Similarly, if word is available at the time of training but it is not available at the time of validation or in real environment then due to OOV NLP, NLU task performance will be poor, and nobody will use that model.

OOV problems becomes serious when we are using a dataset for training which has words from multiple languages and multiple scripts are used to write those words. This is the typical case of Hinglish language especially in social media or whatsapp communication between Indians. Although there is no silver bullet solution for this OOV problem but if do following we can address this problem to a large extend.

- A. Use large corpus
- B. Use corpus of different domains
- C. Use corpus which has text written in multiple scripts

- D. Use corpus which has words from multiple languages
- E. Instead of creating context-based vector for words, create subwords from the word and create context vector of those. This is the approach used by fastText of Facebook.

3.5. Embedding

Computers cannot understand text so we need to convert them into numbers. But how to convert a word, phrase, sentence, dialogue, paragraph, chapter, news article, book or encyclopaedia in number? Broadly there are two approaches one is frequency based and another is prediction based.

TF-IDF: Term frequency inverse document frequency is frequency based embedding approach. This is a numerical statistic technique that is intended to reflect how important a word is in a collection or document. TF-IDF numbers of a word imply a strong relationship with the document they appear in, it suggests that if that word were to appear in a query, the document could be of interest to the user, (Ramos, 2003).

CBOW: Continuous bag of words is a prediction-based technique. It predicts the probability of the word if a context is given. Context window is number of words around the word. Context window of size one means one word left and one word right of the main word. (Wang, Xu, Chen and He, 2017)

Skip-gram: Skip gram is another prediction-based technique. If we want 3 gram one skip, skip-gram from a sentence "I hit the tennis ball" then we get following skip-grams "I hit the", "hit the tennis", "the tennis ball". This gives us good context understanding. However with this approach a problem of sparsity of the word becomes more severe, (Brunt, 1987).

3.5.1. Absolute Embedding

Word embedding like TF-IDF are absolute word embedding approaches. In these approaches word meaning is fixed irrespective of the context a word is used. We know from our experience that meaning of same word can change from one domain to another and one context to other. For example, "गया गया गया". English meaning "Gaya went to Gaya". First word is subject, second word is a verb, and the third word is a location. Absolute embedding approaches cannot handle this kind of text and because of wrong vector the classification task will be incorrect.

3.5.2. Contextual Embedding using full word

Three popular and most used absolute embedding vectors are glove, word2vec and freebase. Glove840B is pretrained word vector with 940 billion tokens. This is developed by Stanford university. Word2vec-GoogleNews-vectors-negative300.bin.gz is pretrained word vector with 100 million tokens. Freebase [freebase-vectors-skipgram1000.bin.gz is pretrained word vector with 1.4

million tokens. Word2vec and Freebase are developed by google using google news dataset. In the contextual embedding different meaning of one word in different context can be represented by different vector of the same word. Contextual embedding is done using skip-gram and CBOW. Full word is used to develop this kind of embedding. Issue with this kind of embedding is OOV. If you create word vector using this embedding post lemmatization of word then context is not fully represented but OOV problem will be less. If you develop word vector using this embedding without lemmatization, then OOV problem will be more and matrix will be too sparse.

3.5.3. Contextual Embedding using subwords

As discussed above contextual Embedding using full word cause OOV problem during the training. To address that problem this technique, create subwords from a word and then create word vector of those subwords. Final word vector is sum of all these vectors. fastText uses this technique to create word vector. Fasttext treats each word as composed of character ngrams. So the vector for a word is made of the sum of this character n grams. Let's say there is a word "apple" in the sentence so to get the word vector of "apple" we need to sum all vectors of the n-grams of apple "<ap", "app", "appl", "apple", "apple>", "ppi", "pple", "pple>", "ple", "ple>", "le>". Assuming ngram-min is 3 and ngram-max is 6. This embedding technique also uses n-gram and CBOW for creating word vector.

In their paper, "Adaptive GloVe and FastText Model for Hindi Word Embeddings", (Gaikwad and Haribhakta, 2020) states that AGM gives better results than GloVe and FastTextWeb. They also mentioned that FastText embeddings which are trained on FastTextHin (Hindi Monolingual corpus) produce better results than FastTextWeb. Google research has introduced a multilingual BERT which is capable of working with more than 100 languages (Romano).

3.6. Types of Sarcasm Detection System

Sarcasm detection systems can be classified in following ways

- Architecture Used: Based on the architecture used to develop the system.
 - Rule Based
 - Classical Machine Learning Based
 - Neural Network Based
 - Transformer Based
- Domain Specific: Based on
 - e domain it serves.
 - Health
 - Education

- Travel
- Social
- Generic (It is extremely difficult to build a generic SDS)
- Mode of Communication Based: This classification is based on the mode of inputs it can accept to perform the classification.
 - Text Based Systems: They can process only online or offline text is used as an input.
 - Voice Based Systems: They can process only voice signals.
 - Video Based Systems: They can process only videos.
 - Image Based Systems: They can accept only images.
 - Multimodal System: These systems can take any form of input to perform the classification. It is challenging task to build a SDS which can take all type of inputs, as mentioned above.
- Time of Detection Based
 - Realtime Systems: In real time message can be classified as sarcasm or not. For example, as soon as message is delivered on whatsapp, twitter, facebook receiver get a different kind of tick message that it is sarcastic message.
 - Batch Systems: At the end of day or any other frequency, based on the need, all the messages or text can be processed in batch to know how many of them were sarcastic.
- Language Script Based: This classification is based upon spoken Language used to write message and written script used to write message.
 - Language Specific: Only for specific language like German or Japanese or Hindi etc.
 - Multiple Language: Can support any spoken language of the world. It is very challenging work to develop such a model.
 - Script Specific: Only for a specific script like Roman or Devanagari or Chinese etc.
 - Multiple Scripts: Can support any script of the world. It is incredibly challenging work to develop such a model which supports all the scripts of the world

3.7. Summary

Thus, we see many researchers have tried to perform the task of sarcasm detection and achieve different accuracy or F1 score depending upon their experiment setups. They have tried different feature extraction techniques and applied those features on different classification algorithms. Most of the work has happened in English language and results are not consistent because results varies due to quality of text in dataset, domain, classification techniques used, features used, data source used etc. Some work has been done for Hindi language and other Indian languages. We did not find any work in Hinglish language which is beyond Twitter dataset. If we observe the table in Appendix B we cannot say with certainty that there is any significant improvement in sarcasm detection results when we use transformers or CNN. We want to experiment with different features and classification algorithms and understand what best results we can achieve when want to detect sarcasm in Hinglish language text.

APPENDIX A: Summary Sarcasm Detection Papers

Table 1: Summary of Important Papers on Sarcasm Detection using NLP

#	Paper	Language, Text Type	Features	Model	Metrics
1	Irony Detection in Twitter: The Role of Affective Content (Fafias et al., 2016)	English, Twitter	colon, exclamation, question, punctuation mark, words in tweet, character length of tweet, verbs, nouns, adjectives, adverbs, number of uppercase letters in tweet, number of emoticons in tweet, degree of similarity in the tweet, number of hashtags, number of mentions.	SVM, DT, NB	Acc: 73-96% depends upon datasets and classifier.
2	Natural Language Processing Based Features for Sarcasm Detection: An Investigation Using Bilingual Social Media Texts (Suhaimin et al., 2017)	English, Twitter	Lexical features – Unigram, Pragmatic features- Punctuation marks, Hashtag, Prosodic features – Interjection, Syntactic features -Part of Speech, Idiosyncratic features - Idiosyncratic	Non Linear SVM	Acc: 82.5%
3	Semantics-aware BERT for Language Understanding (Zhang, Sun, Galley, Chen, Brockett, Gao, Gao, Liu and Dolan, 2020a)	English, Normal Text	Semantic Role Labeller	CNN	Acc: 94.6% on Large dataset of SST2
4	Multi-Rule Based Ensemble Feature Selection Model for Sarcasm Type Detection in Twitter (Sundararajan and Palanisamy, 2020)	English, Twitter	20 features. Noun and verb count, positive intensifier, negative intensifier, bigram, trigram, skip gram, unigram, emoji sentiment, sentiment score, interjections, punctuators, exclamations, question mark, uppercase, repeat words count, positive word frequency, negative word frequency, polarity flip, and parts of speech tagging. These are grouped in 3 categories Linguistic Features, Sentiment Features, Contradictory Features	Random Forest, Naive Bayes, Support Vector Machine, K-Nearest Neighbour, Gradient Boosting, AdaBoost, Logistic Regression, and Decision Tree.	Acc: 86.61% to 99.79% Depending upon the type of sarcasm. Final classifier is RF

#	Paper	Language, Text Type	Features	Model	Metrics
5	Sarcasm Detection in Typo-graphic Memes (Kumar et al., 2019)	English, Instagram Images	Number of negative words, number of positive words, POS tag, hashtag	SVM, Logistic Regression, GBoost, Random Forest, Decision Tree, RNN	Acc: 73.25% to 87.95% depending upon the classifier used.
6	Sentiment Analysis in a Resource Scarce Language: Hindi (Jha, N, Shenoy and R, 2016)	Hindi, Movie Reviews	POS Adjective	Naive Bayes, Multinomial NB, SVM, Maximum Entropy	Acc: 92.2% to 100% depending upon unigram or bigram feature and classifier
7	Sarcasm detection on twitter : A Behavioral Modeling Approach (Rajadesingan et al., 2015)	English, Tweet	Created 335 SCUBA features in following categories, Sentiment Score, Sentiment Transition between past and present, Sarcasm as a complex form of expression, emotion (mood, frustration, affects and sentiments), language familiarity, sarcasm familiarity, environment familiarity, written expression related, structural variation	Logistic Regression	Acc: 83.46%
8	Lexicon-Based Sentiment Analysis in the Social Web (Asghar et al., 2014)	English, Tweet	Emoticon score, Lexicon score, SentiWordNet Score, Slang Score	Rule based	Acc: 95.24%
9	Harnessing Context Incongruity for Sarcasm Detection (Joshi, Sharma and Bhattacharyya, 2015)	English, Tweet	a) Lexical features- unigram using chi-square test, (b) Pragmatic- emoticons, punctuation marks, capital words, (c) Explicit congruity-related to polarity changed, and (d) Implicit incongruity features.	LibSVM with RBF kernel	F1: 61%
10	Contextualized Sarcasm Detection on Twitter (Bamman and Smith, 2015)	English, Tweet	Author Features, Addressee Feature, Audience features: Historical data of author and addressee, Response Feature, Tweet Features, Environment Features	Logistic regression	Acc: 85.1%
11	Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews (Turney, 2002)	English, Opinion Survey of Products	POS	Rule Based PMI calculator	Acc: 74.39%

#	Paper	Language, Text Type	Features	Model	Metrics
12	Sentiment Analysis of Hindi Review based on Negation and Discourse Relation (Mittal and Agarwal, 2013)	Hindi, Movie Reviews	Semantic orientation & polarity values	Rule Based	Acc: 80.21%
13	BHAAV- A Text Corpus for Emotion Analysis from Hindi Stories (Kumar et al., 2019)	Hindi, Short stories	Tokens using Classical language tool kit, unigram, bigram, FastText embedding, TF-IDF	SVM linear kernel, LR, RF, Shallow CNN + Bi-Directional LSTM	Acc: 62%
14	Towards Multimodal Sarcasm Detection: An Obviously Perfect Paper (Castro, Hazarika, Pérez-Rosas, Zimmermann, Mihalcea and Poria, 2020)	English, Clips of YouTube, TV Shows, Transcription	Text features: lexical and pragmatic features, stylistic feature, incongruity, situational disparity, hashtag Speech features: Acoustic patterns that are related to sarcastic behavior Speaker related feature	SVM RBF kernel and a scaled gamma	F1: 71.8%
15	Context-based Sarcasm Detection in Hindi Tweets. (Bharti et al., 2018)	Hindi, Tweets	POS, Used SentiWordNet features, Word Polarity Score for tweet and context	Rule Based	Acc: 87%
16	A Sentiment Analyzer for Hindi Using Hindi Senti Lexicon (Sharma et al., 2014)	Hindi, Movie Reviews, Product Reviews	POS, Hindi SentiWordNet, Word Polarity Score	Rule Based	Acc: 85 to 89.5%
17	A Transformer-based approach to Irony and Sarcasm detection (Potamias et al., 2020)	English, Irony/SemVal-2018-Task, Reddit SARC2.0 politics, Riloff Sarcastic Dataset	Embedding: ELMo, USE, NB-SVM, FastText, XLnet, BERT base cased, BERT base uncased, RoBERTa base model, UPF, ClaC, DESC	RCNN-RoBERTa base model	Acc: 85% to 94% depending upon dataset
18	Detecting Sarcasm is Extremely Easy ;-) (Parde and Nielsen, 2018)	English, Tweet, Amazon product reviews	Contains twitter indicator, Twitter-based predicates and situations, Star rating, Laughter and interjections, Specific characters, Polarity, Subjectivity PMI, Consecutive characters, All-caps bag of words	Naïve Bayes	F1: 59% (Twitter) F1: 78% (Amazon)

#	Paper	Language, Text Type	Features	Model	Metrics
19	CARER: Contextualized Affect Representations for Emotion Recognition (Saravia et al., 2018)	English, Tweets	BoW, char n-gram, TF-IDF, Word2Vec, fastText(ch), word-cluster, enriched patterns, Twitter-based pre-trained word embeddings and reweight them via a sentiment corpus through distant supervision	CNN	Acc: 81% with CARER
20	A Corpus of English-Hindi Code-Mixed Tweets for Sarcasm Detection (Swami, Khandelwal, Singh, Akhtar and Shrivastava, 2018)	Hindi-English, Tweets	Word N-gram, Char N-gram, Sarcasm score of each token, emoticons, chi-square to select feature	SVM with RBF, SVM with Linear, RF	Acc: 78.4% with RF
21	Harnessing Online News for Sarcasm Detection in Hindi Tweets (Bharti et al., 2017)	Hindi, Tweets	POS	Rule Based	Acc: 79.4%
22	The perfect solution for detecting sarcasm in tweets #not (Liebrecht et al., 2013)	Dutch, Tweets	POS (Adjective, Adverb) - Intensifier	Ruled Based	AUC: 77%
23	A2Text-net: A novel deep neural network for sarcasm detection (Liu et al., 2019)	English, Tweet, News Headlines, Reddit	Punctuation, POS, chi-square test to selected variables.	DNN, LSTM, SVM, RF, LR, GRU, A2Text	F1: 71% - 90% depending upon dataset with A2Text classifier
24	Sarcasm as contrast between a positive sentiment and negative situation (Riloff, Qadir, Surve, Silva, Gilbert and Huang, 2013)	English, Tweet	Word N-Gram, POS	LibSVM with RBF	F1: 51%
25	Sarcasm Detection in Hindi sentences using Support Vector (Desai and Dave, 2016)	Hindi, various online sources (using polarity levelled corpora)	Unigram, Positive Score, Negative Score, Hashtag, Emoticons, Polarity, TFIDF	LibSVM with RBF	Acc: 84%

#	Paper	Language, Text Type	Features	Model	Metrics
26	Twitter as a Corpus for Sentiment Analysis and Opinion Mining (Pak and Paroubek, 2010)	English, Twitter	N-Gram, POS	Multinomial Naive Bayes, SVM, CRF	Not Mentioned
27	Exploring the fine-grained analysis and automatic detection of irony on Twitter (Hee et al., 2018)	English, Tweet	POS, Word unigram, bigram, Character tri and fourgram, number of character, number of punctuation, presence of punctuation, number of hashtags, interjections, tweet length, number of emoticons, hastags/word ratio, number of NE, tweet overall polarity, difference of highest positive word polarity, highest negative word polarity, cluster wise word2vec. Overall 4 group of features- Lexical, Sentiment, Semantic, Syntactic	SVM, LSTM	Acc: 67.54% (SVM) Acc: 68.27% (LSTM)
28	Exploiting Emojis for Sarcasm Detection (Subramanian et al., 2019)	English, Twitter, Facebook	Word vector + emoticon vector	GRU	F1: 89.36% (Twitter) F1: 97.97% (facebook)
29	A novel automatic satire and irony detection using ensembled feature selection and data mining. (Ravi and Ravi, 2017)	English, Newswire, Satire news articles, Amazon	(Linguistic, Semantic, Psychological, unigram) in named LIWC features (L), TAALES Features(T), Unigram Features(D), feature subset ensemble, feature selection (IG, GR, Chi, CORR, TSTAT)	SVM (Liner, RBF, Sigmoid, Polynomial), LMT, LR, RF, NB, BN, MLP	F1: 96.58% (L+T+D features) + GR feature selector + SVM RBF Classifier
30	Automatic Satire Detection: Are You Having a Laugh? (Burfoot and Baldwin, 2009)	English, Newswire and Satire news articles	Headline Features, Profanity Features, Slang Features, Binary Unigram Features, Unigram,	SVM	F1: 79.8%
31	Semi-supervised recognition of sarcastic sentences in twitter and Amazon (Davidov et al., 2010)	English, Twitter, Amazon	Meta Tag (User, Company, Product, Title, Author, Link, HashTags, Meta Tag based content Matching, Punctuation,	KNN	F1: 78% Amazon F1: 83% Twitter

#	Paper	Language, Text Type	Features	Model	Metrics
32	Identifying Sarcasm in Twitter: A Closer Look. In (González-Ibáñez et al., 2011)	English, Twitter	Lexical Features: unigram, dictionary based (Linguistic Processes (e.g., adverbs, pronouns), Psychological Processes (e.g., positive and negative emotions), Personal Concerns (e.g, work, achievement), and Spoken Categories (e.g., assent, non-fluencies)) + WordNet Affect + interjection + punctuation Pragmatic Features: positive emotions like smily, negative emotions like frowning face. ToUser like @Name χ^2 test to select features	SVM	Acc: 55.59% to 75.78% depending upon tweet format.

References

- Asghar, M.Z., Kundi, F.M., Khan, A., Ahmad, S., 2014. Lexicon-based sentiment analysis in the social web. J. Basic. Appl. Sci. Res 4, 238–248.
- Bamman, D., Smith, N.A., 2015. Contextualized sarcasm detection on twitter. Proceedings of the 9th International Conference on Web and Social Media, ICWSM 2015 , 574–577.
- Bharti, S.K., Babu, K.S., Jena, S.K., 2017. Harnessing online news for sarcasm detection in hindi tweets. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 10597 LNCS, 679–686. doi:10.1007/978-3-319-69900-4_86.
- Bharti, S.K., Babu, K.S., Raman, R., 2018. Context-based sarcasm detection in hindi tweets. 2017 9th International Conference on Advances in Pattern Recognition, ICAPR 2017 , 410–415doi:10.1109/ICAPR.2017.8593198.
- Brunt, J.V., 1987. A closer look at fermentors and bioreactors. Nature Biotechnology 5, 1133–1138. URL: <http://www.nature.com/doi-finder/10.1038/nbt1187-1133>, doi:10.1038/nbt1187-1133.
- Burfoot, C., Baldwin, T., 2009. Automatic satire detection: Are you having a laugh? ACL-IJCNLP 2009 - Joint Conf. of the 47th Annual Meeting of the Association for Computational Linguistics and 4th Int. Joint Conf. on Natural Language Processing of the AFNLP, Proceedings of the Conf. , 161–164URL: <http://search.cpan.org/perldoc?>
- Carvalho, P., Sarmiento, L., Silva, M.J., Oliveira, E.D., 2009. Clues for detecting irony in user-generated contents: Oh...!! it's "so easy";-). International Conference on Information and Knowledge Management, Proceedings , 53–56doi:10.1145/1651461.1651471.
- Castro, S., Hazarika, D., Pérez-Rosas, V., Zimmermann, R., Mihalcea, R., Poria, S., 2020. Towards multimodal sarcasm detection (an obviously perfect paper). ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference , 4619–4629.
- Clark, K., Luong, M.T., Le, Q.V., Manning, C.D., 2020. Electra: Pre-training text encoders as discriminators rather than generators. URL: <https://arxiv.org/abs/2003.10555><https://doi.org/10.48550/arxiv.2003.10555>, doi:10.48550/ARXIV.2003.10555.
- Davidov, D., Tsur, O., Rappoport, A., 2010. Semi-supervised recognition of sarcastic sentences in twitter and amazon. CoNLL 2010 - Fourteenth Conference on Computational Natural Language Learning, Proceedings of the Conference , 107–116.
- Desai, N.P., Dave, A.D., 2016. Sarcasm detection in hindi sentences using support vector machine. International Journal of Advance Research in Computer Science and Management Studies 4, 8–15. URL: www.ijarcsms.com.
- Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. URL: <https://arxiv.org/abs/1810.04805>, doi:10.48550/ARXIV.1810.04805.
- Farias, D.I.H., Patti, V., Rosso, P., 2016. Irony detection in twitter: The role of affective content. ACM Transactions on Internet Technology 16, 1–24. URL: <http://dx.doi.org/10.1145/2930663><https://dl.acm.org/doi/10.1145/2930663>, doi:10.1145/2930663.
- Gaikwad, V., Haribhakta, Y., 2020. Adaptive glove and fasttext model for hindi word embeddings. ACM International Conference Proceeding Series , 175–179doi:10.1145/3371158.3371179.
- González-Ibáñez, R., Muresan, S., Wacholder, N., 2011. Identifying sarcasm in twitter: A closer look. ACL-HLT 2011 - Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies 2, 581–586.
- Hee, C.V., Lefever, E., Hoste, V., 2018. Exploring the fine-grained analysis and automatic detection of irony on twitter. Language Resources and Evaluation 52, 707–731. doi:10.1007/s10579-018-9414-2.
- Jha, V., N, M., Shenoy, P.D., R, V.K., 2016. Sentiment analysis in a resource scarce language:hindi. International Journal of Scientific & Engineering Research 7, 968–980. doi:10.14299/ijser.2016.09.005.
- Joshi, A., Bhattacharyya, P., Carman, M.J., 2018. Investigations in Computational Sarcasm. 1st ed., Springer Publishing Company, Incorporated.
- Joshi, A., Sharma, V., Bhattacharyya, P., 2015. Harnessing context incongruity for sarcasm detection. ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference 2, 757–762. doi:10.3115/v1/p15-2124.
- Kumar, A., Singh, S., Kaur, G., 2019. Fake news detection of indian and united states election data using machine learning algorithm. International Journal of Innovative Technology and Exploring Engineering 8, 1559–1563. doi:10.35940/ijitee.K1829.0981119.
- Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R., 2019. Albert: A lite bert for self-supervised learning of language

- representations. URL: <https://github.com/google-research/ALBERT>. <http://arxiv.org/abs/1909.11942>.
- Liebrecht, C., Kunneman, F., Van den Bosch, A., 2013. The perfect solution for detecting sarcasm in tweets #not, pp. 29–37.
- Ling, J., Klinger, R., 2016. An empirical, quantitative analysis of the differences between sarcasm and irony, pp. 203–216. URL: http://dx.doi.org/10.1007/978-3-319-47602-5_39 http://link.springer.com/10.1007/978-3-319-47602-5_39, doi:10.1007/978-3-319-47602-5_39.
- Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V., 2019. Roberta: A robustly optimized bert pretraining approach. URL: <https://github.com/pytorch/fairseq> <http://arxiv.org/abs/1907.11692>.
- Mittal, N., Agarwal, B., 2013. Sentiment analysis of hindi review based on negation and discourse relation. Sixth International Joint Conference on Natural Language Processing, 57–62. URL: http://www.aclweb.org/website/old_anthology/W/W13/W13-43.pdf page=57.
- Nozza, D., Fersini, E., Messina, E., 2016. Unsupervised irony detection: A probabilistic model with word embeddings. URL: <http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0006052000680076>, doi:10.5220/0006052000680076.
- Pak, A., Paroubek, P., 2010. Twitter as a corpus for sentiment analysis and opinion mining. Proceedings of the 7th International Conference on Language Resources and Evaluation, LREC 2010, 1320–1326. doi:10.17148/ijarcce.2016.51274.
- Parde, N., Nielsen, R., 2018. Detecting sarcasm is extremely easy ;-), pp. 21–26. doi:10.18653/v1/W18-1303.
- Potamias, R.A., Siolas, G., Stafylopatis, A.G., 2020. A transformer-based approach to irony and sarcasm detection. Neural Computing and Applications doi:10.1007/s00521-020-05102-3.
- Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J., 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research 21, 1–67. URL: <http://jmlr.org/papers/v21/20-074.html>. <http://arxiv.org/abs/1910.10683>.
- Rajadesingan, A., Zafarani, R., Liu, H., 2015. Sarcasm detection on twitter: a behavioral modeling approach. WSDM 2015 - Proceedings of the 8th ACM International Conference on Web Search and Data Mining, 97–106. doi:10.1145/2684822.2685316.
- Ramos, J., 2003. Using tf-idf to determine word relevance in document queries. URL: <https://sites.google.com/site/caonmsu/ir/UsingTFIDFtoDetermineWordRelevanceinDocumentQueries.pdf>.
- Ravi, K., Ravi, V., 2017. A novel automatic satire and irony detection using ensembled feature selection and data mining. Knowledge-Based Systems 120, 15–33. URL: <http://dx.doi.org/10.1016/j.knosys.2016.12.018>, doi:10.1016/j.knosys.2016.12.018.
- Riloff, E., Qadir, A., Surve, P., Silva, L.D., Gilbert, N., Huang, R., 2013. Sarcasm as contrast between a positive sentiment and negative situation. EMNLP 2013 - 2013 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, 704–714.
- Romano, S., . Multilingual transformers - towards data science. URL: <https://towardsdatascience.com/multilingual-transformers-ae917b36034d>.
- Sanh, V., Debut, L., Chaumond, J., Wolf, T., 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. URL: <https://github.com/huggingface/transformers> <http://arxiv.org/abs/1910.01108>.
- Saravia, E., Liu, H.C., Huang, Y.H., Wu, J., Chen, Y.S., 2018. Carer: Contextualized affect representations for emotion recognition, pp. 3687–3697. doi:10.18653/v1/D18-1404.
- Sharma, D.S., Sangal, R., Pawar, J.D., Sharma, R., Bhattacharyya, P., 2014. A sentiment analyzer for hindi using hindi senti lexicon. Subramanian, J., Sridharan, V., Shu, K., Liu, H., 2019. Exploiting emojis for sarcasm detection. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 11549 LNCS, 70–80. doi:10.1007/978-3-030-21741-9_8.
- Suhaimin, M.S.M., Hijazi, M.H.A., Alfred, R., Coenen, F., 2017. Natural language processing based features for sarcasm detection: An investigation using bilingual social media texts. ICIT 2017 - 8th International Conference on Information Technology, Proceedings, 703–709. doi:10.1109/ICITECH.2017.8079931.
- Sulis, E., Farias, D.I.H., Rosso, P., Patti, V., Ruffo, G., 2016. Figurative messages and affect in twitter: Differences between #irony, #sarcasm and #not. Knowledge-Based Systems 108, 132–143. doi:10.1016/j.knosys.2016.05.035.
- Sundararajan, K., Palanisamy, A., 2020. Multi-rule based ensemble feature selection model for sarcasm type detection in twitter. Computational Intelligence and Neuroscience 2020. doi:10.1155/2020/2860479.
- Swami, S., Khandelwal, A., Singh, V., Akhtar, S.S., Shrivastava, M., 2018. A corpus of english-hindi code-mixed tweets for sarcasm detection.
- Turney, P.D., 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews, 417–424. URL: <http://arxiv.org/abs/cs/0212032>.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. URL: <https://arxiv.org/abs/1706.03762>, doi:10.48550/ARXIV.1706.03762.
- Wallace, B.C., Choe, D.K., Charniak, E., 2015. Sparse, contextually informed models for irony detection: Exploiting user communities, entities and sentiment, pp. 1035–1044. URL: <http://www.reddit.com>, doi:10.3115/v1/p15-1100.
- Wang, Q., Xu, J., Chen, H., He, B., 2017. Two improved continuous bag-of-word models. URL: <http://ieeexplore.ieee.org/document/7966208/>, doi:10.1109/IJCNN.2017.7966208.
- Wang, S., Manning, C.D., 2012. Baselines and bigrams: Simple, good sentiment and topic classification. 50th Annual Meeting of the Association for Computational Linguistics, ACL 2012 - Proceedings of the Conference 2, 90–94.
- WikipediaA, . Demographics of india - wikipedia. URL: https://en.wikipedia.org/wiki/Demographics_of_India.
- Yang, Dai, Z., Yang, Z., Carbonell, Y., Jaime Salakhutdinov, R.L., V, Q., 2019. Xlnet: Generalized autoregressive pretraining for language understanding. URL: <https://github.com/zihangdai/xlnet> <http://arxiv.org/abs/1906.08237>.
- Zhang, Y., Sun, S., Galley, M., Chen, Y.C., Brockett, C., Gao, X., Gao, J., Liu, J., Dolan, B., 2020a. Dialogpt : Large-scale generative pre-training for conversational response generation, pp. 270–278. URL: <https://github.com/mgalley/>, doi:10.18653/v1/2020.acl-demos.30.
- Zhang, Z., Wu, Y., Zhao, H., Li, Z., Zhang, S., Zhou, X., Zhou, X., 2020b. Semantics-aware bert for language understanding. Proceedings of the AAAI Conference on Artificial Intelligence 34, 9628–9635. doi:10.1609/aaai.v34i05.6510.
- Zhang, Z., Yang, J., Zhao, H., 2020c. Retrospective reader for machine reading comprehension. URL: <http://arxiv.org/abs/2001.09694>.



Hari Thapliyal is a Data Science and Project Management professional. He is a mentor, trainer, coach, consultant, mediator, philosopher and blogger. In his 28+ years professional career in software development, project management, training and consulting he has been deeply involved in all kind of roles from software design, development, quality assurance, training, mentoring, PMO

head and many others. He has deep interests in diverse subjects like BFSI Sector, Artificial Intelligence, Data Mining, Data Analytics, Deep Learning, ML Modelling, NLP, Economics, Physics, Sanskrit, Vedic Chanting, Vedanta, Healing, History, Culture, Project Management, Meditation and Spirituality. This helps him to understand that how and where to discover

new equilibrium among many variables like religion, culture, ethics, morality, societies, business, process automation, and new age technologies like AI, NLP, Deep Learning, GAN, Robotics, Cryptocurrency etc. He is Xpert Coach and Mentor for various AI, ML courses of upGrad and he is founder of dasarpAI an AI Training, Consulting startup.