Is This Sarcastic? A Reddit Bot for Detecting Sarcasm

by

Christopher McVeigh

A thesis submitted to the University of Birmingham for the degree of BSc Computer Science

School of Computer Science
University of Birmingham
March 2020

Abstract

Humans struggle to recognise sarcasm in written statements as easily as we can in person. However, our ability to discern written sarcasm improves considerably when we know more contextual details of the statement. This project presents /u/IsThisSarcastic, a Reddit bot for detecting sarcasm in comments using a combination of content and context. /u/IsThisSarcastic extracts contextual information just as a human user would by having access to the score awarded to a comment by other Reddit users; the post originally commented on; and the subreddit (community) being posted to. When combining this data with syntactic and semantic feature extraction, /u/IsThisSarcastic outperforms similar web-based sarcasm detectors, and approaches the performance enjoyed by state-of-the-art deep learning models.

Contents

1 Introduction 1

2 Literature Review 2
2.1 Sarcasm Detection 2
2.2 Reddit Bots 3

3 Design 4
3.1 Task Summary 4
3.2 Dataset 4
3.3 Feature extraction 5
3.4 Training process 8
3.5 Reddit implementation 8

4 Results 10
4.1 Survey Responses 10
4.2 Classifier Experiments 11
4.3 Final Results 13

5 Discussion 14

6 Conclusion 14

7 References 14

8 Appendices 15

1 Introduction

Sarcasm is defined as "a sharp, bitter, or cutting expression or remark; a bitter gibe or taunt"[1], typically involving irony on the part of the speaker. As such, sarcasm detection remains one of the biggest challenges in NLP (Cambria et al., 2017). We as humans are more likely to misunderstand a sarcastic statement in writing than if we hear it in person, making this problem even more challenging. In a face-to-face conversation, sarcasm can be signalled using an altered tone of voice or exaggerated gestures, but these characteristics of conversation are lost in written discourse such as social media.

One factor of sarcasm that remains present in online discussions is context. A sarcastic statement presumes that the listener has a certain level of familiarity with the topic of discussion, and that applies to written comments just as much as verbal statements. In practice, our ability to recognise sarcasm in writing involves a level of background knowledge, common sense, anaphora resolution, and logical reasoning that it is not feasible to emulate (Poria et al., 2017). However, when dealing with online comments, we are able to obtain some empirical context using the comment itself.

To illustrate how context can aid our ability to recognise sarcasm, I conducted a survey where respondents were shown a series of comments from Reddit[2]. They were initially presented with the comment's text and no further details, before being shown the subreddit[3] the comment was posted in; the comment's score as voted by other users; and the title of the original post being commented on. At each step, respondents would indicate whether they believed the given comment was sarcastic or not. One example was a comment reading *"I'm certain this will be of particular use to people studying mechanical engineering."* 80.6% of respondents believed this comment was non-sarcastic, but when told that the original post was titled *"University of Winnipeg makes indigenous course a requirement"*, 95.9% now believed that it was sarcastic. Nothing had changed about the original comment, but with the knowledge of the comment's original context, the majority ended up changing their minds.

For automated sarcasm detection, much influence has been placed on lexical, syntactic, and semantic analysis of text. I felt that continued emphasis on these aspects could only do so much, as it is functionally equivalent to answering the survey before receiving any context. I therefore opted to develop

[1] Oxford English Dictionary

[2] www.reddit.com

[3] A community on Reddit for a dedicated subject

a classifier that takes context into account by including contextual data as part of its feature extraction. To this end, I propose /u/IsThisSarcastic[4]; a Reddit-based sarcasm detector taking the form of a bot account. Reddit users may summon the bot by entering its username in reply to a comment, or providing a link to another. /u/IsThisSarcastic then retrieves the appropriate comment and extracts lexical, syntactic, semantic, and contextual features, applying the result to a trained machine learning model to generate a classification for the comment. Finally, /u/IsThisSarcastic replies to its summon with its judgement and confidence score. Experiments on a Reddit corpus show improved performance over other solutions that use similar approaches, with overall performance close to that of state-of-the-art deep learning models.

2 Literature Review

2.1 Sarcasm Detection

Automated sarcasm detection is still a relatively young topic of research, with a recent surge in interest due to the rise of social media allowing easier access to representative data.

One early work was González-Ibáñez et al. (2011), who confirmed the difficulty of detecting sarcasm in text by running a support vector machine and a logistic regression machine over a corpus of Twitter posts (tweets). Their solution returns a maximum accuracy of 65.44% for the problem of classifying sarcastic vs. non-sarcastic. They also conducted a similar study using human judges, whose maximum accuracy was 73%. This inspired my decision to run a survey for human results.

Twitter is often used for sarcasm detection corpora, such as in Bamman & Smith (2015), which uses a logistic regression machine with context modelling based on the tweet's author as well as responses to the tweet. These features produce a very high accuracy of 85.1%. Another high-accuracy result from a Twitter corpus comes from Poria et al. (2017), who used a pre-trained convolutional neural network for sentiment, emotion, and author personality. On a balanced corpus of sarcastic/non-sarcastic tweets, they attained an F_1 score of 0.97. Both of these high-scoring solutions use contextual features in their modelling, which helped to inform my position that using context would be beneficial.

[4] Usernames on Reddit are traditionally preceded by "/u/"

The research that I found myself most "in competition" with was The Sarcasm Detector by Cliche (2014). This project uses lexical, syntactic, and semantic features including sentiment analysis and topic modelling to achieve an F_1 score of 0.60 on a support vector machine. This provided me with a realistic benchmark for my project, and presented my first goal - my classifier should out-score this one.

Automated sarcasm detectors based on Reddit corpora do exist, but are less common than Twitter-based solutions. Tay et al. (2018) produced the Multi-dimensional Intra-Attention Recurrent Network (MIARN), which focuses on the semantics between each word in the input sentence, but not any contextual information. Experiments on two subreddits - /r/movies and /r/technology[5] - produce F_1 scores of 0.695 and 0.691 respectively.

Ghosh et al. (2018) opted to model contextual data using conversations on a variety of platforms, with Reddit being one of them. They introduce a pair of Long Short-Term Memory (LSTM) networks, where one analyses the current turn (i.e. comment) and one analyses the prior turn (i.e. parent comment). On a Reddit corpus with sentence-level attentiveness, they report an F_1 score of 0.77 on the sarcastic class and 0.74 on the non-sarcastic class.

Finally, Hazarika et al. (2018) introduced CASCADE, a contextual sarcasm detector trained exclusively on a Reddit corpus. Their implementation of context involves building a personality profile for comment authors to establish an individual's writing style, as well as their "big five" personality traits of openness, conscientiousness, extraversion, agreeableness, and neuroticism. Using a convolutional neural network with this information, they report an F_1 score of 0.77 on a balanced corpus.

2.2 Reddit Bots

There are several examples of Reddit bots, though they are not traditionally published and often closed-source. A prominent example of a Reddit bot is /u/AutoModerator[6], an automated moderator developed by Reddit administrators that can perform basic moderator duties on any subreddit where it is enabled. For example, it can automatically remove a post if enough users report it, or comment on any posts with certain keywords in the title.

An example of an open-source Reddit bot is /u/User_Simulator[7]. This bot can be summoned by any Reddit user by commenting its username, which will cause the bot to read the user's last 1000 comments and generate

[5] Subreddits are preceded by "/r/"

[6] https://www.reddit.com/wiki/automoderator

[7] https://github.com/trambelus/UserSim

a Markov chain to construct a new comment in the style of the summoning user.

3 Design

3.1 Task Summary

When summoned to act on a given comment, /u/IsThisSarcastic extracts lexical, syntactic, and semantic information from the comment's text body, as well as contextual information relating to the comment. Figure 1 shows the overall pipeline of the classifier.

The rest of this section will explain the design of each section of the pipeline in turn, before explaining the design of the Reddit implementation.

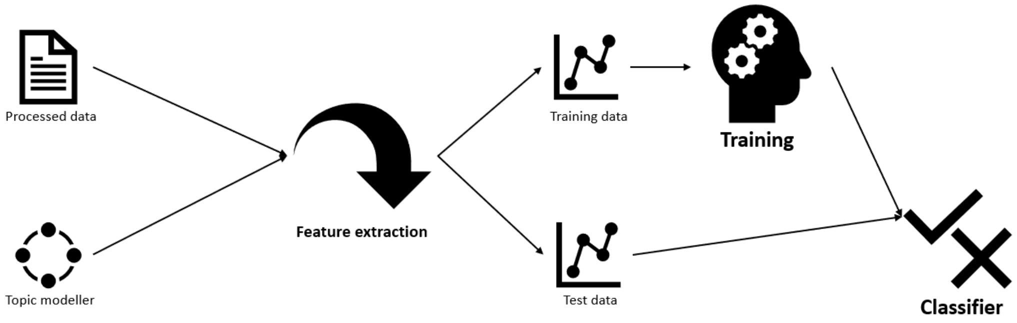


Figure 1: The machine learning pipeline for the classifier.

3.2 Dataset

The dataset used for this project is the Self-Annotated Reddit Corpus (SARC) (Khodak et al., 2018). SARC provides large-scale sets of Reddit comments with markers for sarcasm representing ground truths. A comment is marked as sarcastic if the original comment included the annotation "/s". This is a conventional sarcasm marker for Reddit submissions, intended specifically to mitigate the ambiguity of sarcasm in text. For this project, I used a subset of 50,000 comments from SARC's `main-balanced` set. This set may include comments from any subreddit and has a balanced selection of sarcastic and non-sarcastic statements. SARC also offers an unbalanced set, where sarcastic statements make up <1% of the data, a more realistic proportion, but

I felt this would severely impact the classifier's ability to detect sarcastic statements.

The unprocessed data from SARC contains the following information about a comment:

- Reddit ID
- Text body
- Author
- Score
- Date created
- Subreddit

SARC provides a CSV file linking comments to their parent posts by ID, as well as their self-annotated sarcasm status. By combining the information made available, I was able to create a local database of comments linked to their parent post and the subreddit they were posted to. The size of the full dataset made it impractical to index parent posts using the original file, so I further altered my local file by adding the title of each parent post and the description of the subreddit using the Reddit API through the Python Reddit API Wrapper (PRAW)[8].

Preprocessing on this data is intended to remove features in the text that would confuse the classifier during training. Empty comments, as well as those containing links, non-Unicode characters, and those that are replies to other comments rather than top-level comments are removed from the final dataset. Subreddits (preceded by ”/r/”), username mentions (preceded by ”/u/”), sarcasm markers (”/s”), hashtags, and any mention of the words ”sarcasm” or ”sarcastic” are removed from comment text, though the comments themselves are still included. Further processing is performed during training, where emoticons are replaced with textual descriptions (e.g. *:)* is replaced with *happy*). In addition, abbreviations are extended (e.g. *doesn't* is replaced with *does not*).

3.3 Feature extraction

Feature extraction is one of the key components of the training process. During training, features are extracted into a dictionary for each comment, which

[8] https://github.com/praw-dev/praw

is appended to a list that becomes the target for the vectoriser that informs the classifier. The same feature extraction is performed when /u/IsThisSarcastic is summoned on Reddit, with the dictionary of features informing the classifier's decision function. Features extracted during this process are:

- N-grams
- Sentiment
- Parts of speech
- Capitalisation
- Topic modelling
- Reddit context

3.3.1 N-grams

Unigrams are formed by tokenising the comment's text using NLTK[9]. Bigrams are formed using these tokens, and all n-grams are added to the feature dictionary.

3.3.2 Sentiment analysis

Sentiment is one of the major features extracted during this process. Two methods for scoring sentiment of a document are used in order to limit the impact of one scoring incorrectly. The first is SentiWordNet 3.0 (Baccianella et al., 2010), a project that assigns a positive and negative sentiment score to every word in WordNet[10]. The second is VADER (Hutto & Gilbert, 2014), a lexicon and rule-based tool designed for sentiment expression on social media, making it ideal for use with Reddit comments. VADER returns a value for positive, negative, neutral, and compound sentiment. For SentiWordNet, I record the difference between positive and negative sentiment for the input text, and for VADER I record all sentiment values. I also use TextBlob[11] to record the subjectivity of the text. This process is performed first over the comment as a whole, before splitting the comment in half and working on both halves, and finally again with the comment split into thirds. At the stages where the comment is split into parts, the contrast in sentiment between each part is also recorded.

[9] https://www.nltk.org/

[10] https://wordnet.princeton.edu/

[11] https://textblob.readthedocs.io/en/dev/

3.3.3 Parts of speech

Using NLTK, the comment text is tagged with its parts of speech. The feature dictionary is then updated with the number of occurrences of nouns, adjectives, verbs, and adverbs.

3.3.4 Capitalisation

The number of words in ALL CAPS in the comment text is totalled. If there are at least 4 words in capitals, the feature dictionary is updated to note heavy capitalisation. Otherwise, it is updated to show normal capitalisation.

3.3.5 Topic modelling

Topic modelling for this project takes the form of an LDA model from Gensim[12]. The topic modeller uses a symmetric distribution with 200 topics, with its corpus being formed by performing Porter stemming over the original comment data. During feature extraction, the comment is Porter stemmed and converted to a bag-of-words format, with the feature dictionary updated with the topics included in the given comment.

3.3.6 Reddit context

In order to model Reddit context, I give the classifier access to the same information the human respondents to my survey had; the comment's score, subreddit, and parent post. The score is directly added to the feature dictionary. The subreddit's description and parent post's title have their sentiments analysed in the same manner as the original comment, with their results added to the feature dictionary as well. Finally, the difference between the comment's sentiment and the subreddit's sentiment, as well as between the comment's sentiment and the post's sentiment, are both added to the feature dictionary. The idea behind this is that a sarcastic comment may be likely to have the opposite overall sentiment compared to their original post or subreddit. For example, a positive post about the weather being nice today may have a comment reading *"This is terrible! I don't know what to do when it's like this!"*, which may return a more negative sentiment.

[12] https://radimrehurek.com/gensim/index.html

3.4 Training process

Following feature extraction, the vectoriser is generated from the list of feature dictionaries. The vectors are shuffled and split 70:30 to form a training and test set. The classifier is fitted to the training set and written to file. I conducted a series of experiments to establish the type of classifier I would use for this project, the details of which are in the next section.

3.5 Reddit implementation

This section also details the use of PRAW; although I had used it previously to get information about my dataset, this was its primary use in the project.

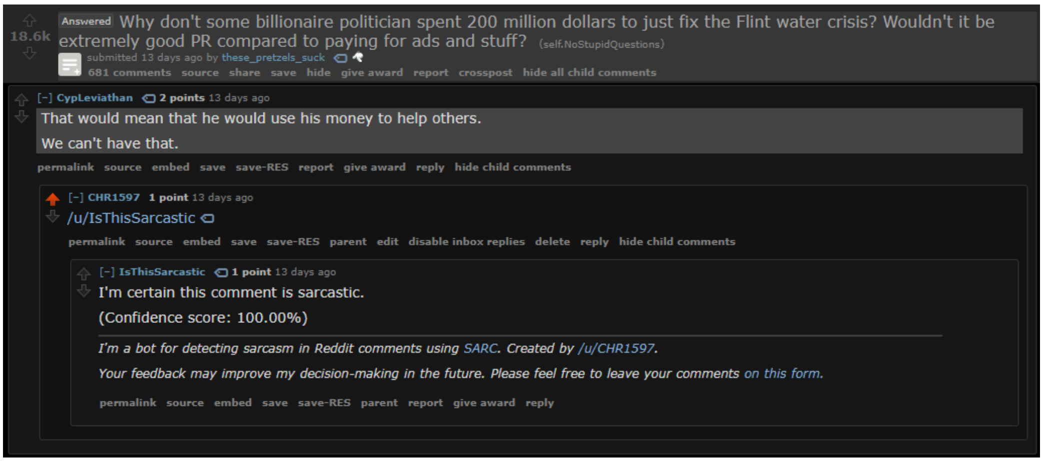
Reddit requires that any applications using their API authenticate with OAuth2[13]. As such, running the program requires access to a Reddit account as well as an OAuth2 client secret. In the interest of privacy, this information is not present in this submission. References to `project1` in the code refer to my personal Reddit account which I used for early testing as well as generating the master file of comment data. References to `project2` refer to the account /u/IsThisSarcastic, which is used when the bot is active.

When the bot is run, it generates a stream of its mentions; inbox messages that users receive when others mention their username. When a mention is received, the mentioning comment is checked for one of two correct formats. The first is simply the bot's username as a reply to another comment, and the second is the username followed by a link to another comment. In the first case, the targeted comment is the parent of the mentioning comment; in the second, the targeted comment is the one linked to. The required information (text, score, subreddit description, and parent post title) is placed into a dictionary, and the comment is then scored by the classifier. The classifier returns a value between -1 and 1, where a positive score indicates that the comment is predicted as sarcastic. The bot then begins to build its reply to the user who summoned it. Table 1 shows how the bot communicates different scores in its reply. Figure 2 shows a usage example where the bot is tagged in an existing comment thread. Figure 3 shows a usage example where the bot is provided a link to another comment, with that comment being shown in figure 4.

[13] https://oauth.net/

Score (absolute value × 100)	Response
score ≤ 10	”It’s too close to call!”
$10 <$ score ≤ 25	”It’s a very difficult decision”
$25 <$ score ≤ 50	”It’s quite a difficult decision”
$50 <$ score ≤ 75	”I’m fairly sure”
$75 <$ score ≤ 90	”I’m sure”
score > 90	”I’m certain”

Table 1: How the bot’s reply to its summoner changes according to the score generated by the classifier.



Answered Why don't some billionaire politician spent 200 million dollars to just fix the Flint water crisis? Wouldn't it be extremely good PR compared to paying for ads and stuff? (self.NoStupidQuestions)

submitted 13 days ago by these_pretzels_suck

[–] CypLeviathan 2 points 13 days ago

That would mean that he would use his money to help others.

We can't have that.

[–] CHR1597 1 point 13 days ago

/u/IsThisSarcastic

[–] IsThisSarcastic 1 point 13 days ago

I'm certain this comment is sarcastic.

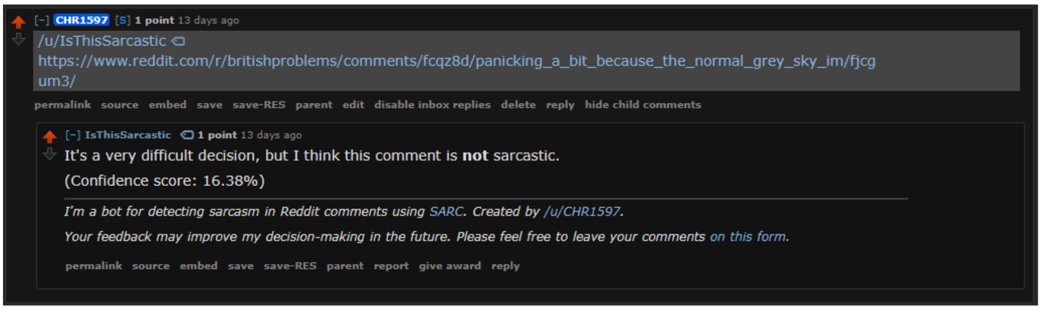
(Confidence score: 100.00%)

I'm a bot for detecting sarcasm in Reddit comments using SARC. Created by /u/CHR1597.

Your feedback may improve my decision-making in the future. Please feel free to leave your comments on this form.

18.6k
681 comments source share save hide give award report crosspost hide all child comments
permalink source embed save save-RES report give award reply hide child comments
permalink source embed save save-RES parent edit disable inbox replies delete reply hide child comments
permalink source embed save save-RES parent report give award reply

Figure 2: The bot responding to a username mention in a comment thread.



[–] CHR1597 [S] 1 point 13 days ago

/u/IsThisSarcastic
https://www.reddit.com/r/britishproblems/comments/fcqz8d/panicking_a_bit_because_the_normal_grey_sky_im/fjcgum3/

[–] IsThisSarcastic 1 point 13 days ago

It's a very difficult decision, but I think this comment is **not** sarcastic.

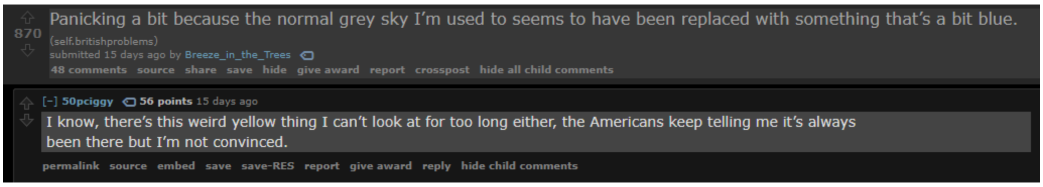
(Confidence score: 16.38%)

I'm a bot for detecting sarcasm in Reddit comments using SARC. Created by /u/CHR1597.

Your feedback may improve my decision-making in the future. Please feel free to leave your comments on this form.

permalink source embed save save-RES parent edit disable inbox replies delete reply hide child comments
permalink source embed save save-RES parent report give award reply

Figure 3: The bot responding to a username mention with a link to another comment to judge.



Panicking a bit because the normal grey sky I'm used to seems to have been replaced with something that's a bit blue.

870

(self.britishproblems)

submitted 15 days ago by Breeze_in_the_Trees

48 comments source share save hide give award report crosspost hide all child comments

[–] 50pciggy 56 points 15 days ago

I know, there's this weird yellow thing I can't look at for too long either, the Americans keep telling me it's always been there but I'm not convinced.

permalink source embed save save-RES report give award reply hide child comments

Figure 4: The comment linked in figure 3.

The bot also includes a link to a feedback form in its reply. The intention here is that people who read the bot's reply may provide their opinion on whether the original comment was sarcastic. This would allow the original comment dataset to be augmented by more real-world data based on comments the bot has already judged. This takes the form of a Google Forms survey[14] where respondents provide a link to the comment being judged, their personal judgement of sarcasm, and optionally what made them make that decision.

4 Results

This section shows the results of all parts of this project. First is an analysis of the human respondents to the survey I conducted before starting development. Next is an overview of the experiments I conducted to find the optimal classifier for this problem. Finally, I examine the results I obtained from the final version of the classifier, comparing it to the earlier work I have mentioned and analysing potential error cases that prevent the classifier from scoring higher.

4.1 Survey Responses

As previously mentioned, I conducted a survey before starting development work to see how context affected our own ability to detect sarcasm[15]. From 98 respondents, I was able to generate precision, recall, and F_1 scores to find the average human performance on this task. Table 2 shows the results for human respondents with and without context.

From this data, we can see that the human respondents were better at detecting sarcasm when they had the context, and that they are better at detecting sarcasm than non-sarcasm. However, it is worth bearing in mind

[14] https://forms.gle/VsvWNwCrU36YbxQp8

[15] https://forms.gle/YAkc9tRhMVeVYJVt6

Sarcastic						**Non-Sarcastic**					
No context			**All context**			**No context**			**All context**		
P	**R**	**$\mathbf{F_1}$**	**P**	**R**	**$\mathbf{F_1}$**	**P**	**R**	**$\mathbf{F_1}$**	**P**	**R**	**$\mathbf{F_1}$**
0.83	0.49	0.61	0.92	0.73	0.81	0.39	0.77	0.52	0.58	0.86	0.69

Table 2: Details of human respondents to the sarcasm survey.

that the respondents may have been more inclined to consider a comment sarcastic than they normally would, due to having to make a decision either way.

4.2 Classifier Experiments

The decision of which classifier to use was a major part of the development process. Very early in development, I decided the best suite of options would come from `scikit-learn` (Pedregosa et al., 2011). This was because it provided several classification models, including the support vector and logistic regression models I saw used in my prior research.

My early tests of the library were guided by Singh (2019), where I attempted to use a limited corpus to train classifiers of the following types:

- Linear Support Vector Classifier (Linear SVC)
- Gaussian Naive Bayes
- Logistic Regression
- Random Forest Classifier

The Gaussian Naive Bayes option was eliminated immediately, as its implementation in `scikit-learn` requires its input vectors to take the form of a dense matrix rather than the sparse matrix generated after feature extraction. Conversion of a sparse matrix to a dense matrix uses more memory than I had access to during development. In addition, this classifier performs the worst of the four in Singh (2019), and a Gaussian Naive Bayes approach is not mentioned in any of the other papers I rersearched. As such, I decided it would not be worth trying to make this approach work.

I also opted to eliminate the Random Forest approach. It performed second-worst in Singh (2019), and again was not featured in any of the other research I performed. That left me with a choice between Linear SVC and Logistic Regression. These two classification models both have access to a `coef_` attribute, which provides the coefficient of features in the decision function. This allows me to view the most and least important attributes in judging a comment as sarcastic. In addition, these two models are frequently used in other works, with Cliche (2014) using a Linear SVC model. With my goal for this project being to out-perform that solution, it made sense to use a similar model.

The experiments I performed used two different corpora based on subsets of my final corpus. The first set of tests used 1000 comments, and the second used 10,000. The Linear SVC model had a regularisation (C) of 0.1 and a maximum iteration value of 10,000. The Logistic Regression model uses a limited-memory BFGS solver with a C of 0.1 and 2000 maximum iterations. Both models were trained on each corpus, initially without extracting Reddit context. This gave a baseline result for how each classifier performed, before the Reddit context was introduced. Figure 5 shows the results of this testing with 5-fold cross-validation.

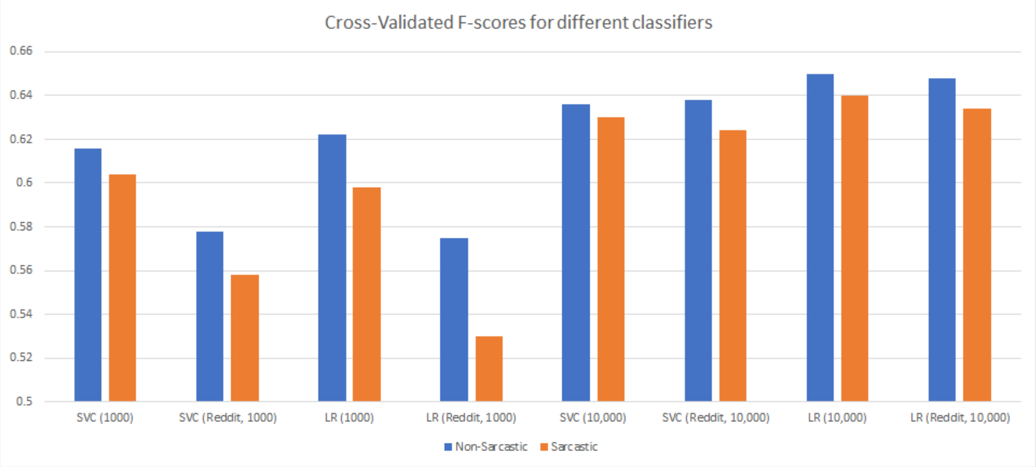


Figure 5: F_1 scores for each classifier during testing. Parenthesised is the size of the corpus and whether Reddit context was included.

Perhaps unsurprisingly, all classifiers perform significantly better with a larger training set. However, what surprised me initially was that the addition of Reddit context actually made the performance worse, with the lowest-performing classifier being the Logistic Regression machine on 1000 comments with Reddit context. This classifier returned an F_1 score of 0.53 for the sarcastic category and 0.575 for non-sarcastic.

With a larger corpus of 10,000 comments, the situation is quite different. The classifier versions that use Reddit context performed almost identically to those that do not. On average, a classifier with 10,000 comments is equally adept at detecting non-sarcastic comments with or without context, and is worse at detecting sarcasm by a margin of 0.006 F_1 score. This negligible difference convinced me that a larger corpus made context more useful. In addition, a 10,000-comment corpus makes the Logistic Regression model outperform the Linear SVC model. At this point, the Linear SVC model often failed to fully converge despite having 5 times the maximum iterations of the Logistic Regression model. This would suggest that the Linear SVC model was unable to find the ”most similar” comments in opposing classes in 10,000 attempts. The Logistic Regression model draws its decision boundary by taking every data point into account, so it did not have this problem. This may explain why the Logistic Regression model scored higher at this stage of testing, but I felt that the problem would only be exacerbated at the full scope of the corpus of 50,000 comments.

4.3 Final Results

The final classifier uses a Logistic Regression model with a limited-memory BFGS solver, a C of 0.1, and 2000 maximum iterations. Table 3 presents the performance results of /u/IsThisSarcastic alongside human results, similar sarcasm detection approaches, and state-of-the-art deep learning approaches.

Classifier	**Precision**	**Recall**	**F_1**	**Accuracy**
Human	0.92	0.73	0.81	-
González-Ibáñez et al. (2011)	-	-	-	0.65
Cliche (2014)	-	-	0.60	-
Tay et al. (2018)	0.697	0.694	0.695	0.699
Hazarika et al. (2018)	-	-	0.77	0.77
/u/IsThisSarcastic	0.64	0.68	0.66	0.66

Table 3: Comparison of the performance between /u/IsThisSarcastic, human respondents to the survey, and various examples of previous work.

While I do not have full results for all of the previous solutions, what we can see from these results is that /u/IsThisSarcastic out-performs Cliche (2014) in F_1 score by 6 percentage points. It also has better accuracy than the seminal work from González-Ibáñez et al. (2011). In addition, /u/IsThisSarcastic scores within 4 percentage points of the state-of-the-art deep learning solution proposed by Tay et al. (2018). Perhaps unsurprisingly, the human respondents to my survey perform better than any automated solutions.

5 Discussion

6 Conclusion

7 References

8 Appendices