Detecting Sarcasm in Reddit Comments

Shalini Kunapuli[1], Lavanya Vijayan[2]

Abstract— **The use of sarcasm and irony on social media allows us to study and model them in depth. We look at the SARC (Self Annotated Reddit Corpus) dataset, which contains 1.3 million comments from Reddit. Using this corpus, we approach sarcasm from both supervised and unsupervised directions. We build transformer models to classify Reddit comments as sarcastic or non-sarcastic, evaluate the models for accuracy, and compare to baseline and prior work done in this area. We also exploit the concept of cloze questions to create language models, which are then used to assign soft labels to our examples. We discuss complexities of identifying sarcasm due to thoughts being multi-faceted, words having various connotations, and the context of Internet slang.**

Keywords—**Sarcasm detection, Classification, BERT**

I. Introduction

SArcasm is a form of ironic speech intended to make fun of or criticize someone or something. Sarcasm detection, an interesting area of natural language processing (NLP), aims to determine the true meaning behind the words a person says. There are many applications of sarcasm detection in advertising, social media strategy, sentiment analysis, and more. Detecting sarcasm is a challenging task, as there are several factors that shape the meaning of a phrase, including tone, context, and history between two people conversing.

II. Background

There have been various approaches to sarcasm detection previously. Lydia Xu and Vera Xu from Stanford worked with both SARC and Discussion Forum datasets, aiming to improve the accuracy of sarcasm detection by focusing on contextual information. They used various LSTM models and BERT, finding that contextual information improves model performance and that the BERT classifier produced the best results.

Using their findings, we will be expanding on the BERT model architecture for sarcasm and developing various models to detect sarcasm. Moreover, we will add in additional context into the model to improve model accuracy. Previous models have analyzed comments along with their respective labels. We will take this a step further by adding information about the parent comment, on top of the comment and label. We hope this will provide useful context to the transformer models in classifying the comment as sarcastic or not.

Exploiting cloze questions for text classification is another method that Timo Schick and Hinrich Schutze address in their paper. They discuss this idea to help language models understand a given task, specifically question answering. They define pairs of cloze question patterns that help to leverage the knowledge and context within language models. They note that for several tasks, pattern exploiting training outperforms supervised training, and also gives large improvements with little training data. We will use the idea of cloze questions for few shot text classification for sarcasm detection, which are two natural language processing concepts that have not been combined before. We will utilize the context in the comments and parent comments from SARC to create soft labels for each example, which we will then run through a transformer model.

III. Approach and Methods

A. Data

We use the SARC (Self Annotated Reddit Corpus) dataset, which contains 1.3 million comments from Reddit (the annotation and data collection process is explained in detail by the authors Mikhail Khodak et al in their paper). Reddit is a social media platform where users comment on submission posts and other comments. The site has "subreddits" which allows for topic-specific conversations. The data includes: the comment, the parent comment (the post that the comment is in response to), which subreddit the post/comment belongs to, number of upvotes, number of downvotes, date, and the label (1 if the comment is sarcastic and 0 if it is non-sarcastic).

B. Evaluation Method

For evaluation of the models, we will primarily be using accuracy. We will also look at the precision, recall, and F1 scores in later models to see whether the models have more propensity towards sarcastic or non-sarcastic labels.

C. Baseline Model

For our baseline model, we ran a convolutional neural network model, using only the comment and label to train. Our model had two dense layers (ReLu and Sigmoid). Our resulting training accuracy was 0.7319 and the testing accuracy was 0.7068.

D. Modeling

D.1 BERT

We used `TFBertForSequenceClassification` to build a BERT model with a binary classification layer on top and `TFTrainer` to train, validate, evaluate, and predict.

We worked with different amounts of the data to do this. Initially, we took 10% of the data and per-

[1]UC Berkeley MIDS, e-mail: `shalinikunapuli@berkeley.edu`.
[2]UC Berkeley MIDS, e-mail: `lvijayan1@berkeley.edu`.

formed a 70%-15%-15% split, yielding 7,000 training examples, 1,500 validation examples, and 1,500 test examples. Afterwards, we took all of the data. In some model versions, we performed a 10%-10%-80% split, yielding 101,077 training examples, 101,077 validation examples, and 808,618 test examples. In other model versions, we performed a 20%-10%-70% split, yielding 202,154 training examples, 101,077 validation examples, and 808,618 test examples. Due to disk and memory constraints, we did not train on a larger subset of the data.

We experimented with the values of the following hyperparameters:

- `learning_rate = 3e-5, 4e-5, 5e-5`
- `batch_size = 16, 32`
- `num_train_epochs = 2, 3`
- `max_length = 30, 50`
- `weight_decay = 0, 0.01, 0.1`

Overall, we found that increasing the learning rate helped improve accuracy and loss significantly. While working with `TFBertForSequenceClassification` and `TFTrainer`, we observed that there were limitations on how much we could customize the model architecture such as adding more layers and specifics within layers. So this informed our next step in modeling — building a BERT model in such a way that we can control more of the model architecture.

Results from this modeling approach are documented in the respective section of this paper.

D.2 DistilBERT

After iterations of the `TFBertForSequenceClassification` model, we decided to try DistilBERT on our data. Since we had a large corpus we were working with, DistilBERT seemed like it would be more efficient, since it is a smaller, faster, and lighter Transformer model. DistilBERT, according to HuggingFace's documentation, "runs 60% faster while preserving over 95% of BERT's performances". We once again tested on just the comments as well as both the comments and respective parent comments. In preprocessing, we tokenized using `DistilBertTokenizerFast` to extract the word tokens and sentences, and add the `CLS` tokens, `SEP` token between the comment and parent comment, and `PAD` for padding each context based on the size of the max length. With this model, we tried out two different ways of tokenizing and padding: (1) padding based on a dynamic max length and (2) padding based on a specific max length. After tokenizing the comments and the parent comment + comment in these two different ways, we ran the tokenized data through `TFDistilBertModel`. Some iterations were trained and evaluated on the freezed model while others were trained and evaluated on the unfreezed models.

Most of the models were run on 100% of the data (1,010,771 rows total), so the resulting training, validation, and test sets sizes were a 0.7, 0.15, 0.15 split (707,541 rows in the training set and 151,615 rows each in the validation and test data sets).

Hyperparameter tuning and trying different combinations to get the best model results were also critical. The metrics below show the values we experimented with:

- `learning_rate = 5e-5`
- `batch_size = 32, 64`
- `optimizer = Adam`
- `num_train_epochs = 6, 10`
- `num_train_unfreeze_epochs = 1, 4, 6, 10`
- `max_length = 30, 40, 50`
- `dropout = 0.2, 0.4`
- `l2_regularization = 0.01`

In addition to hyperparameter tuning, implementing training on freezing and unfreezing models led to a large difference in the accuracies and loss. While unfreezing took more time, it tended to lead to higher accuracies (with slight overfitting) due to more training happening at deeper layers. The model results are outlined in the Results section.

D.3 Cloze + BERT

Our approach for unsupervised learning with sarcasm detection was to use the idea of cloze questions for few shot text classification. We came up with a list of test phrases to try out:

- `" I am [MASK]!"`
- `" I am being [MASK]!"`
- `" OP is [MASK]!"`
- `" OP is [MASK]."`

Using these test phrases, the parent comment, and the actual comment, we tokenized using BertTokenizer. To see what fit in the `[MASK]`, we tried out combinations of words relating to sarcastic and non-sarcastic, listed in Table I.

Sarcastic Related Words	Non-Sarcastic Related Words
joking	serious
ironic	honest
satirical	blunt
exaggerating	transparent
sarcastic	unsarcastic
kidding	

Table I: Potential Sarcastic and Non-Sarcastic Mask Words

Using `TFBertForMaskedLM`, we were able to get the logit values for each of the pairs of words above, for each respective parent comment + comment phrase. As part of preprocessing, we chose samples of comments that were not too long (the `TFBertForMaskedLM` model had memory issues for some outlier long comments that were greater than 40 words long). We also spot checked and ensured that the given labels made sense to be sarcastic or non-sarcastic.

We decided to try out smaller samples of data, building off of the work Schick and Schutze outlined in their paper with "few shot text classification". We sampled balanced datasets with 100, 200, 500, and 1000 examples total. For each sample, after finding the test phrase and pairs of words that most accurately classified the comments, we took an average of the logits and created soft labels for each comment. The comparison of our soft labels to our true labels is shown below in Table II, III, IV, and V.

	Precision	Recall	F1 score	Accuracy
not sarc	0.52	0.72	0.61	0.53
sarc	0.55	0.34	0.42	0.53

Table II: Soft vs Actual Labels for 100 examples (50 per class)

	Precision	Recall	F1 score	Accuracy
not sarc	0.50	0.64	0.56	0.51
sarc	0.51	0.37	0.43	0.51

Table III: Soft vs Actual Labels for 200 examples (100 per class)

	Precision	Recall	F1 score	Accuracy
not sarc	0.50	0.68	0.57	0.50
sarc	0.49	0.31	0.38	0.50

Table IV: Soft vs Actual Labels for 500 examples (250 per class)

	Precision	Recall	F1 score	Accuracy
not sarc	0.50	0.69	0.58	0.50
sarc	0.49	0.30	0.37	0.50

Table V: Soft vs Actual Labels for 1000 examples (500 per class)

We also want to note that these soft labels metrics are compared to the actual labels, which were originally self-annotated as detailed in Khodak, Saunshi, and Vodrahalli's paper. There may be some limitation in how we assess the accuracy of the soft label due to the fact that the actual labels are conditional on what the annotator thinks is sarcastic or non-sarcastic.

Once we had these soft labels, created using this cloze question method, we were able to use `TFBertForSequenceClassification` to train our model using the tokenized parent + comment phrases on the soft labels that we generated using cloze questions.

IV. Results

A. BERT

The results of running our BERT model on just the comment data are shown below in Table VI. We use the hyperparameters as a way to differentiate each model from each other and we report the accuracy and loss for each one.

Next, the results of running our BERT model on both the parent comment + comment data are shown below in Table VII. Similar to the previous table, we report the accuracy and loss for each one.

B. DistilBERT

The results of running DistilBERT on only the comment data are shown below. We report the accuracy and the loss for each of the models in Table VIII.

In addition to running DistilBERT on comment data, we ran it on parent and comment data. The best results after hyperparameter tuning are outlined in Table IX.

C. Cloze + BERT

We go through the Cloze + BERT model process on four different sizes of data: 100, 200, 500, and 1000 examples. Each of these samples has a balanced amount of sarcastic and non-sarcastic comments. The results of this model are detailed in Table X.

V. Error Analysis

Due to the nature of sarcasm, we have classification errors in our models. Sarcasm is a very subjective part of language. There are many elements to sarcasm: what words someone uses, the context it is used in, what tone is portrayed, and more.

While exploring and analyzing the data, we also noticed how some of the true labels did not make sense to us. There were some comments that were labeled as sarcastic but did not seem funny or sarcastic to us. This also emphasized how sarcasm is subjective and changes from person to person. Since the original SARC dataset was self annotated, it is possible that some of these labels would be classified differently if others annotated the same data. Therefore, our analysis could be based upon labels that are not entirely accurate, due to the subjectivity of sarcasm.

Some examples of misclassifications (false positives and false negatives) for each model are shown in Table XI.

In this table, we see some examples of misclassified comments. Many examples in the data contain slang words and Internet words, which can be used in different contexts and have various meanings. For example, the second misclassified `DistilBERT` example contains "lol", which is one of the things that makes the model predict it as sarcastic. In addition, the second `Cloze + BERT` model also uses "ugh". However for both of these examples, both the parent comment and comment are sincere responses addressing the topic. For some of the examples additionally, sarcastic phrases use the same language as non-sarcastic examples. It is hard for the model to interpret them as sarcastic when syntactically and context-wise, they look like non-sarcastic phrases. The sarcasm comes in the way someone says something, which is not always detected through the literal words that appear on a screen.

The first misclassified `Cloze + BERT` example contains "wow", which is intended sarcastically, but the model interprets it as non-sarcastic. "Wow" has several different connotations, depending on the con-

Hyperparameters	No. Train Examples	No. Val Examples	No. Test Examples	Accuracy	Loss
Training batch size 16, Evaluation batch size 64, Learning rate 5e-5, No. of Epochs 2, Weight decay 0.01, Hidden dropout prob 0.1	7,000	1,500	1,500	0.7910	0.4483
Training batch size 16, Evaluation batch size 64, Learning rate 5e-5, No. of Epochs 2, Weight decay 0.01, Hidden dropout prob 0.1	101,077	101,077	808,618	0.7613	0.4892

Table VI: BERT Model Results for only Comment Data

Hyperparameters	No. Train Examples	No. Val Examples	No. Test Examples	Accuracy	Loss
Training batch size 16, Evaluation batch size 64, Learning rate 5e-5, No. of Epochs 2, Weight decay 0.01, Hidden dropout prob 0.1	101,077	101,077	808,618	0.76217	0.48771

Table VII: BERT Model Results for Parent Comment + Comment Data

Model Architecture	No. Train Examples	No. Val Examples	No. Test Examples	Accuracy	Loss
Default Model	707,541	151,615	151,615	0.6537	0.6214
Additional Model Layers	707,541	151,615	151,615	0.8423	0.3594

Table VIII: DistilBERT Model Results for only Comment Data

Training Method	No. Train Examples	No. Val Examples	No. Test Examples	Accuracy	Loss
Freeze Layers for Training	707,541	151,615	151,615	0.6218	0.6489
Unfreeze Layers for Training	707,541	151,615	151,615	0.9299	0.1752

Table IX: DistilBERT Model Results for Parent Comment + Comment Data

text. It has a duality; it can be used by someone who is genuinely surprised, unsurprised, impressed, or unimpressed, to name a few. Understanding these nuances is something people develop over time through diverse social interactions, and it is difficult to capture these nuances into systematic rules that can be fed into a model.

The first misclassified `BERT` example contains the phrase "Don't count on it" which is a phrase that can be said jokingly or seriously. However, in this context, with the topic of "bird poop," the phrase is meant as a joke; the sarcasm is delivered through the tone rather than the literal meaning of the phrase. The model might have interpreted it as non-sarcastic based on "count" and "cleaned," which have empirical connotations and could be found in many serious responses.

In the second misclassified `Bert` example, the author of the parent comment seems to be genuinely asking fellow Redditors for advice, and the author of the reply seems to provide a strategy that helps them face social situations in an honest and humorous way. However, the model interprets the advice as sarcastic, which is likely due to the presence of the word "game," which has a playful connotation. This example highlights how humor and honesty are not mutually exclusive and if sarcasm is associated with strictly one or the other, the resulting interpretation is narrow and not reflective of linguistic and contextual complexities.

VI. Conclusion

In this paper, we experimented with supervised and unsupervised learning methods along with different BERT models on sarcasm detection using both parent comments and comments from Reddit. The results show that our best `BERT` model got an accuracy of `0.7622`, our best `DistilBERT` model got

Data Size	Tokenizing	Test Phrase	Word Pairs	Accuracy	Loss
100 examples (50 per class)	Comment + Parent Comment	" I am being [MASK]!"	(('joking', 'transparent'), ('kidding', 'transparent'), ('joking', 'blunt'), ('ironic', 'transparent'), ('ironic', 'blunt'))	0.6406	0.6901
200 total (100 per class)	Comment + Parent Comment	" I am being [MASK]!"	(('joking', 'transparent'), ('kidding', 'transparent'), ('joking', 'blunt'), ('ironic', 'transparent'), ('ironic', 'blunt'))	0.5156	0.7348
500 total (250 per class)	Comment + Parent Comment	" I am being [MASK]!"	(('joking', 'transparent'), ('kidding', 'transparent'), ('sarcastic', 'serious'), ('ironic', 'transparent'), ('joking', 'blunt'))	0.6250	0.6815
1000 total (500 per class)	Comment + Parent Comment	" I am being [MASK]!"	(('joking', 'transparent'), ('kidding', 'transparent'), ('joking', 'blunt'), ('sarcastic', 'serious'), ('kidding', 'blunt'))	0.7266	0.5999

Table X: Cloze + BERT Model Results for Different Data Sizes

Model	Parent Comment	Comment	Actual Label	Predicted Label
BERT	Man that's alot of bird poop	Don't count on it being cleaned up either!	sarcastic	non- sarcastic
BERT	What are some tips to help out a socially awkward person?	When I make eye contact with someone, I feel like it's a just a game to see who looks away first.	non- sarcastic	sarcastic
Distil BERT	i'd say avatar was one of the few exceptions, but that's probably just me	yep, it's only you.	sarcastic	non- sarcastic
Distil BERT	why are there records so cheap? lol	herb alpert sold something like 70 million records so you're bound to find em	non- sarcastic	sarcastic
Cloze + BERT	Local paper does piece on Grandma Shirley	Wow, that is awesome.	sarcastic	non- sarcastic
Cloze + BERT	Bernicrat. Ugh	Y'know,the people who registered as a Democrat two weeks ago and call everyone who disagrees with them DINOs.	non- sarcastic	sarcastic

Table XI: Example False Positives and False Negatives per Model

an accuracy of `0.9299`, and our best `Cloze + BERT` model got an accuracy of `0.7266`. Our supervised learning BERT training methods generate better results compared to using cloze questions for unsupervised learning. We learned that sarcasm is highly subjective, making it hard for language models to infer if a phrase is sarcastic or not. Our BERT and DistilBERT supervised models performed better because the labels were determined by human annotators. These BERT models outperformed our baseline CNN model (`0.7068` accuracy) by around `2-20%`. For future work, we plan to improve our current models by adding more context related to the subreddit a post/comment was put in. We think that sarcasm language may vary based on the topic (for example politics may have different sarcastic phrases as compared to sports). Gaining a better understanding of sarcasm with the context of subreddit, we would also like to explore how this affects the cloze questions approach. Additionally, we are also interested in response generation based on a previous phrase, similar to question answering. More details about our models and code can be found in our Github repository: `https://github.com/shalziekay/w266_nlp_final_project`.

References

[1] Nikunj Saunshi Mikhail Khodak and Kiran Vodrahalli, *A large self-annotated corpus for sarcasm*, 2018.

[2] Lena Reed Ernesto Hernandez Ellen Riloff Shereen Oraby, Vrindavan Harrison and Marilyn Walker, *Creating and characterizing a diverse corpus of sarcasm in dialogue*, Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue.

[3] Lydia Xu and Vera Xu, *Project Report: Sarcasm Detection.*

[4] Hinrich Schutze Timo Schick, *Exploiting Cloze Questions for Few Shot Text Classification and Natural Language Inference.*

VII. Appendix

Below are the full result tables from running all our models:

Data Size	Tokenizing	Hyperparameters	No. Train Examples	No. Val Examples	No. Test Examples	Accuracy	Loss
1%	Comment	Training batch size 16, Evaluation batch size 64, Learning rate 5e-5, No. of Epochs 2, Weight decay 0.01, Hidden dropout prob 0.1	7,000	1,500	1,500	0.79101	0.44835
100%	Comment	Training batch size 16, Evaluation batch size 64, Learning rate 5e-5, No. of Epochs 2, Weight decay 0.01, Hidden dropout prob 0.1	101,077	101,077	808,618	0.76131	0.48920
100%	Comment	Training batch size 16, Evaluation batch size 64, Learning rate 3e-5, No. of Epochs 2, Weight decay 0.01, Hidden dropout prob 0.1	101,077	101,077	808,618	0.49971	0.69347
100%	Comment	Training batch size 32, Evaluation batch size 64, Learning rate 5e-5, No. of Epochs 3, Weight decay 0, Hidden dropout prob 0.5	202,154	101,077	707,541	0.72614	0.60109
100%	Comment	Training batch size 32, Evaluation batch size 64, Learning rate 5e-5, No. of Epochs 3, Weight decay 0.01, Hidden dropout prob 0.6	202,154	101,077	707,541	0.50038	0.69315

100%	Comment	Training batch size 16, Evaluation batch size 64, Learning rate 5e-5, No. of Epochs 2, Weight decay 0.01, Hidden dropout prob 0.7	101,077	101,077	808,618	0.50010	0.69313
100%	Comment	Training batch size 16, Evaluation batch size 64, Learning rate 5e-5, No. of Epochs 2, Weight decay 0.1, Hidden dropout prob 0.4	101,077	101,077	808,618	0.73798	0.54111
100%	Comment	Training batch size 16, Evaluation batch size 64, Learning rate 5e-5, No. of Epochs 2, Weight decay 0.1, Hidden dropout prob 0.5	101,077	101,077	808,618	0.73686	0.54515
100%	Comment	Training batch size 16, Evaluation batch size 64, Learning rate 5e-5, No. of Epochs 2, Weight decay 0.1, Hidden dropout prob 0.1	101,077	101,077	808,618	0.73892	0.53588
100%	Comment	Training batch size 16, Evaluation batch size 64, Learning rate 4e-5, No. of Epochs 2, Weight decay 0.1, Hidden dropout prob 0.1	101,077	101,077	808,618	0.73905	0.53472
100%	Comment	Training batch size 16, Evaluation batch size 64, Learning rate 4e-5, No. of Epochs 2, Weight decay 0.01, Hidden dropout prob 0.1	101,077	101,077	808,618	0.73904	0.53419
100%	Parent Comment + Comment	Training batch size 16, Evaluation batch size 64, Learning rate 5e-5, No. of Epochs 2, Weight decay 0.01, Hidden dropout prob 0.1	101,077	101,077	808,618	0.76217	0.48771

Table XII: Full BERT Model Results

Data Size	Tokenizing	Hyperparameters	No. Train Examples	No. Val Examples	No. Test Examples	Accuracy	Loss
0.01%	Comment (dynamic padding)	Max length = 30, distilbert dropout = 0.2, distilbert attention dropout - 0.2, layer dropout - 0.2, learning rate = 5e-5, epochs = 6, batch size = 64	7,000	1,500	1,500	0.5593	0.6811
100%	Comment (dynamic padding) and unfreeze layers	Max length = 30, distilbert dropout = 0.2, distilbert attention dropout - 0.2, layer dropout - 0.2,learning rate = 5e-5, epochs = 6, batch size = 64, ft epochs = 4	707,541	151,615	151,615	0.7783	0.4618
100%	Comment (dynamic padding) and unfreeze layers	Max length = 30, distilbert dropout = 0.2, distilbert attention dropout - 0.2, layer dropout - 0.2,learning rate = 5e-5, epochs = 4, ft epochs = 10, batch size = 64, l2 reg = 0.01, additional layers added to previous model (dense(512) relu + dropout layer)	707,541	151,615	151,615	0.8692, 0.9355	0.3076, 0.1641
100%	Comment (max length padding)	Max length = 30, distilbert dropout = 0.2, distilbert attention dropout - 0.2, layer dropout - 0.2,learning rate = 5e-5, epochs = 6, batch size = 64	707,541	151,615	151,615	0.6537	0.6214
100%	Comment (max length padding)	Max length = 30, distilbert dropout = 0.2, distilbert attention dropout - 0.2, layer dropout - 0.2,learning rate = 5e-5, epochs = 3, batch size = 64, l2 reg = 0.01, additional layers added to previous model (dense(512) relu + dropout layer)	707,541	151,615	151,615	0.8423	0.3594

100%	Comment + Parent (max length padding)+ freeze/un-freeze	Max length = 50, distilbert dropout = 0.2, distilbert attention dropout - 0.2, layer dropout - 0.2,learning rate = 5e-5, epochs = 6, ft epochs = 3, batch size = 64	707,541	151,615	151,615	0.6218, 0.9299	0.6489, 0.1752
100%	Comment + Parent (max length padding)+ freeze/un-freeze	Max length = 30, distilbert dropout = 0.4, distilbert attention dropout - 0.4, layer dropout - 0.4,learning rate = 5e-5, epochs = 6, ft$_{e}pochs$ = $6, batchsize = 64$	707,541	151,615	151,615	0.6030, 0.9098	0.6615, 0.2187

Table XIII: Full DistilBERT Model Results

Data Size	Tokenizing	Test Phrase	Word Pairs	BERT Hyperparameters	Accuracy	Loss
100 examples (50 per class)	Comment + Parent Comment	” I am being [MASK]!”	((’joking’, ’transparent’), (’kidding’, ’transparent’), (’joking’, ’blunt’), (’ironic’, ’transparent’), (’ironic’, ’blunt’))	Num epochs = 2, batch size = 16, eval batch size = 64, max length = 50, train size = 0.5, val size = 0.25, test size = 0.25, num hidden layers = 12, hidden size = 768, hidden dropout prob = 0.5	0.6406	0.6901
200 total (100 per class)	Comment + Parent Comment	” I am being [MASK]!”	((’joking’, ’transparent’), (’kidding’, ’transparent’), (’joking’, ’blunt’), (’ironic’, ’transparent’), (’ironic’, ’blunt’))	Num epochs = 2, batch size = 16, eval batch size = 64, max length = 50, train size = 0.5, val size = 0.25, test size = 0.25, num hidden layers = 12, hidden size = 768, hidden dropout prob = 0.5	0.5156	0.7348
500 total (250 per class)	Comment + Parent Comment	” I am being [MASK]!”	((’joking’, ’transparent’), (’kidding’, ’transparent’), (’sarcastic’, ’serious’), (’ironic’, ’transparent’), (’joking’, ’blunt’))	Num epochs = 2, batch size = 16, eval batch size = 64, max length = 50, train size = 0.5, val size = 0.25, test size = 0.25, num hidden layers = 12, hidden size = 768, hidden dropout prob = 0.5	0.6250	0.6815

	1000 total (500 per class)	Comment + Parent Comment	" I am being [MASK]!"	(('joking', 'transparent'), ('kidding', 'transparent'), ('joking', 'blunt'), ('sarcastic', 'serious'), ('kidding', 'blunt'))

Table XIV: Full Cloze + BERT Model Data Sizes