DynaSent: A Dynamic Benchmark for Sentiment Analysis

Christopher Potts*
Stanford University
cgpotts@stanford.edu

Zhengxuan Wu*
Stanford University
wuzhengx@stanford.edu

Atticus Geiger
Stanford University
atticusg@stanford.edu

Douwe Kiela
Facebook AI Research
dkiela@fb.com

Abstract

We introduce DynaSent ('Dynamic Sentiment'), a new English-language benchmark task for ternary (positive/negative/neutral) sentiment analysis. DynaSent combines naturally occurring sentences with sentences created using the open-source Dynabench Platform, which facilities human-and-model-in-the-loop dataset creation. DynaSent has a total of 121,634 sentences, each validated by five crowdworkers, and its development and test splits are designed to produce chance performance for even the best models we have been able to develop; when future models solve this task, we will use them to create DynaSent version 2, continuing the dynamic evolution of this benchmark. Here, we report on the dataset creation effort, focusing on the steps we took to increase quality and reduce artifacts. We also present evidence that DynaSent's Neutral category is more coherent than the comparable category in other benchmarks, and we motivate training models from scratch for each round over successive fine-tuning.

1 Introduction

Sentiment analysis is an early success story for NLP, in both a technical and an industrial sense. It has, however, entered into a more challenging phase for research and technology development: while present-day models achieve outstanding results on all available benchmark tasks, they still fall short when deployed as part of real-world systems (Burn-Murdoch, 2013; Grimes, 2014, 2017; Gossett, 2020) and display a range of clear shortcomings (Kiritchenko and Mohammad, 2018; Hanwen Shen et al., 2018; Wallace et al., 2019; Tsai et al., 2019; Jin et al., 2019; Zhang et al., 2020).

In this paper, we seek to address the gap between benchmark results and actual utility by introducing version 1 of the DynaSent dataset for English-language ternary (positive/negative/neutral) sentiment analysis.[1] DynaSent is intended to be a *dynamic* benchmark that expands in response to new models, new modeling goals, and new adversarial attacks. We present the first two rounds here and motivate some specific data collection and modeling choices, and we propose that, when future models solve these rounds, we use those models to create additional DynaSent rounds. This is an instance of "the 'moving post' dynamic target" for NLP that Nie et al. (2020) envision.

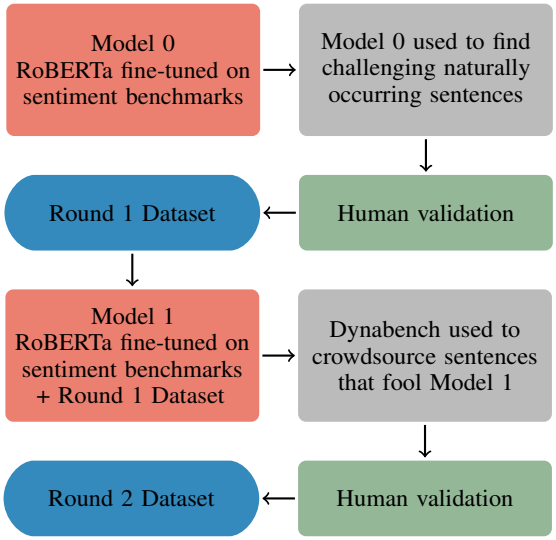


Figure 1: The DynaSent dataset creation process. The human validation task is the same for both rounds; five responses are obtained for each sentence. On Dynabench, we explore conditions with and without prompt sentences that workers can edit to achieve their goal.

Figure 1 summarizes our method, which incorporates both naturally occurring sentences and sentences created by crowdworkers with the goal of fooling a top-performing model. The starting point is Model 0, which is trained on standard sentiment

*Equal contribution.

[1] https://github.com/cgpotts/dynasent

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 2388–2404
August 1–6, 2021.

benchmarks and used to find challenging sentences in existing data. These sentences are fed into a human validation task, leading to the Round 1 Dataset. Next, we train Model 1 on Round 1 in addition to publicly available datasets. In Round 2, this model runs live on the Dynabench Platform for human-and-model-in-the-loop dataset creation;[2] crowdworkers try to construct examples that fool Model 1. These examples are human-validated, which results in the Round 2 Dataset. Taken together, Rounds 1 and 2 have 121,634 sentences, each with five human validation labels. Thus, with only two rounds collected, DynaSent is already a substantial new resource for sentiment analysis.

In addition to contributing DynaSent, we seek to address a pressing concern for any dataset collection method in which workers are asked to construct original sentences: human creativity has intrinsic limits. Individual workers will happen upon specific strategies and repeat them, and this will lead to dataset artifacts. These artifacts will certainly reduce the value of the dataset, and they are likely to perpetuate and amplify social biases.

We explore two methods for mitigating these dangers. First, by harvesting naturally occurring examples for Round 1, we tap into a wider population than we can via crowdsourcing, and we bring in sentences that were created for naturalistic reasons, rather than the more artificial goals present during crowdsourcing. Second, for the Dynabench cases created in Round 2, we employ a 'Prompt' setting, in which crowdworkers are asked to modify a naturally occurring example rather than writing one from scratch. We compare these sentences with those created without a prompt, and we find that the prompt-derived sentences are more like naturally occurring sentences in length and lexical diversity. Of course, fundamental sources of bias remain – we seek to identify these in the Datasheet (Gebru et al., 2018) distributed with our dataset – but we argue that these steps help, and can inform crowdsourcing efforts in general.

As noted above, DynaSent presently uses the labels Positive, Negative, and Neutral. This is a minimal expansion of the usual binary (Positive/Negative) sentiment task, but a crucial one, as it avoids the false presupposition that all texts convey binary sentiment. We chose this version of the problem to show that even basic sentiment analysis poses substantial challenges for our field. We find that the Neutral category is especially difficult. While it is common to synthesize such a category from middle-scale product and service reviews, we use an independent validation of the Stanford Sentiment Treebank (Socher et al., 2013) dev set to argue that this tends to blur neutrality together with mixed sentiment and uncertain sentiment (Section 5.2). DynaSent can help tease these phenomena apart, since it already has a large number of Neutral examples and a large number of examples displaying substantial variation in validation. Finally, we argue that the variable nature of the Neutral category is an obstacle to fine-tuning (Section 5.3), which favors our strategy of training models from scratch for each round.

[2] https://dynabench.org/

2 Related Work

Sentiment analysis was one of the first natural language understanding tasks to be revolutionized by data-driven methods. Rather than trying to survey the field (see Pang and Lee 2008; Liu 2012; Grimes 2014), we focus on the benchmark tasks that have emerged in this space, and then seek to situate these benchmarks with respect to challenge (adversarial) datasets and crowdsourcing methods.

2.1 Sentiment Benchmarks

Many sentiment datasets are derived from customer reviews of products and services (Pang and Lee, 2004, 2005; Socher et al., 2013; Maas et al., 2011; Jindal and Liu, 2008; Ni et al., 2019; McAuley et al., 2012; Zhang et al., 2015). This is an appealing source of data, since such texts are accessible and abundant in many languages and regions of the world, and they tend to come with their own author-provided labels (star ratings). On the other hand, over-reliance on such texts is likely also limiting progress; DynaSent begins moving away from such texts, though it remains rooted in this domain.

Not all sentiment benchmarks are based in review texts. The MPQA Opinion Corpus of Wiebe et al. (2005) contains news articles labeled at the phrase-level for a variety of subjective states; it presents an exciting vision for how sentiment analysis might become more multidimensional. SemEval 2016 and 2017 (Nakov et al., 2016; Rosenthal et al., 2017) offered Twitter-based sentiment datasets. And of course there are numerous additional datasets for specific languages, domains, and emotional dimensions; Google's Dataset Search currently reports over 100 datasets for sentiment.

2.2 Challenge and Adversarial Datasets

Challenge and adversarial datasets (Winograd, 1972; Levesque, 2013) have risen to prominence in response to the sense that benchmark results are over-stating the quality of the models we are developing (Linzen, 2020). These efforts seek to determine whether models have met specific learning targets (Alzantot et al., 2018; Glockner et al., 2018; Naik et al., 2018; Nie et al., 2019), exploit relatively superficial properties of their training data, (Jia and Liang, 2017; Kaushik and Lipton, 2018; Zhang et al., 2020), or inherit social biases in the data they were trained on (Kiritchenko and Mohammad, 2018; Rudinger et al., 2017, 2018; Sap et al., 2019; Schuster et al., 2019).

For the most part, challenge and adversarial datasets are meant to be used primarily for evaluation (though Liu et al. (2019a) show that even small amounts of training on them can be fruitful in some scenarios). However, there are existing adversarial datasets that are large enough to support full-scale training efforts (Zellers et al., 2018, 2019; Chen et al., 2019; Dua et al., 2019; Bartolo et al., 2020). DynaSent falls into this class; it has large train sets that can support from-scratch training as well as fine-tuning. Our approach is closest to, and directly inspired by, the Adversarial NLI (ANLI) project, which is reported on by Nie et al. (2020) and which continues on Dynabench. In ANLI, human annotators construct new examples that fool a top-performing model but make sense to other human annotators. This is an iterative process that allows the annotation project itself to organically find phenomena that fool current models. The resulting dataset has, by far, the largest gap between estimated human performance and model accuracy of any benchmark in the field right now. We hope DynaSent follows a similar pattern, and that its naturally occurring sentences and prompt-derived sentences bring beneficial diversity.

2.3 Crowdsourcing Methods

Within NLP, Snow et al. (2008) helped establish crowdsourcing as a viable method for collecting data for at least some core language tasks. Since then, it has become the dominant mode for dataset creation throughout all of AI, and the scientific study of these methods has in turn grown rapidly. For our purposes, a few core findings from research into crowdsourcing are centrally important.

First, crowdworkers are not fully representative of the general population (Hube et al., 2019), and any crowdsourcing project will reach only a small population of workers (Gadiraju et al., 2017). This narrowness seems to be an underlying cause of many of the artifacts that have been identified in prominent NLU benchmarks (Poliak et al., 2018; Gururangan et al., 2018; Tsuchiya, 2018; Belinkov et al., 2019). DynaSent's naturally occurring sentences and prompt sentences can help, but we acknowledge that those texts come from people who write online reviews, which is also a special group.

Second, as with all work, quality varies across workers and examples, which raises the question of how best to infer individual labels from response distributions. Dawid and Skene (1979) is an early contribution to this problem leveraging Expectation Maximization (Dempster et al., 1977). Much subsequent work has pursued similar strategies; for a full review, see Zheng et al. 2017. Our corpus release uses the true majority (3/5 labels) as the gold label where such a majority exists, leaving examples unlabeled otherwise, but we include the full response distributions in our corpus release and make use of those distributions when training Model 1. For additional details, see Section 3.3.

3 Round 1: Naturally Occurring Sentences

We now begin to describe our method for constructing DynaSent (Figure 1). The current section focuses on Model 0 and Round 1, and Section 4 explains how these feed into Model 1 and Round 2.

3.1 Model 0

Our Model 0 begins with the RoBERTa-base parameters (Liu et al., 2019b) and adds a three-way sentiment classifier head. The model was trained on a number of publicly-available datasets, as summarized in Table 2. See Appendix A for details on these datasets and how we processed them for our ternary task. We evaluate this and subsequent models on three datasets (Table 1): SST-3 dev and test, and the assessment portion of the Yelp and Amazon datasets from Zhang et al. 2015. For Yelp and Amazon, the original distribution contained only (very large) test files. We split them in half (by line number) to create dev and test splits.

In Table 3, we summarize our Model 0 assessments on these datasets. Across the board, our model does extremely well on the Positive and Negative categories, and less well on Neutral. We trace

	SST-3		Yelp		Amazon	
	Dev	Test	Dev	Test	Dev	Test
Pos	444	909	9,577	10,423	130,631	129,369
Neg	428	912	10,222	9,778	129,108	130,892
Neu	228	389	5,201	4,799	65,261	64,739
Total	1,100	2,210	25,000	25,000	325,000	325,000

Table 1: External assessment datasets.

	CR	IMDB	SST-3	Yelp	Amazon
Pos	2,405	12,500	42,672	260K	1.2M
Neg	1,366	12,500	34,944	260K	1.2M
Neu	0	0	81,658	130K	600K
Total	3,771	25,000	159,274	650K	3M

Table 2: Model 0 training data.

this to the fact that the Neutral categories for all these corpora were derived from three-star reviews, which actually mix a lot of different phenomena: neutrality, mixed sentiment, and (in the case of the reader judgments in SST) uncertainty about the author's intentions. We return to this issue in Section 5.2, arguing that DynaSent marks progress on creating a more coherent Neutral category.

Finally, Table 3 includes results for our Round 1 dataset, as we are defining it. Performance is at-chance across the board by construction (see Section 3.4 below). We include these columns to help with tracking the progress we make with Model 1. We also report performance of this model on our Round 2 dataset (described below in Section 4), again to help with tracking progress and understanding the two rounds.

3.2 Harvesting Sentences

Our first round of data collection focused on finding naturally occurring sentences that would challenge our Model 0. To do this, we harvested sentences from the Yelp Academic Dataset, using the version of the dataset that contains 8,021,122 reviews.[3] The sampling process was designed so that 50% of the sentences fell into two groups: those that occurred in 1-star reviews but were predicted by Model 0 to be Positive, and those that occurred in 5-star reviews but were predicted by Model 0 to be Negative. The intuition here is that these would likely be examples that fooled our model. Of course, negative reviews can (and often do) contain positive sentences, and vice-versa. This motivates the validation stage that we describe next.

[3] https://www.yelp.com/dataset

3.3 Validation

Our validation task was conducted on Mechanical Turk. Workers were shown ten sentences and asked to label them according to the categories **Positive**, **Negative**, **Neutral**, and **Mixed**. See Appendix B for the full interface, including glosses for the categories and the task instructions.

For this round, 1,978 workers participated in the validation process. In the final version of the corpus, each sentence is validated by five different workers. To obtain these ratings, we employed an iterative strategy. Sentences were uploaded in batches of 3–5K and, after each round, we measured each worker's rate of agreement with the majority. We then removed from the potential pool those workers who disagreed more than 80% of the time with their co-annotators, using a method of 'unqualifying' workers that does not involve rejecting their work or blocking them (Turk, 2017). We then obtained additional labels for examples that those 'unqualified' workers annotated. The final version of DynaSent keeps only the responses from the highest-rated workers. This led to a substantial increase in dataset quality by removing a lot of labels that seemed to us to be randomly assigned. Appendix B describes the process in more detail, and our Datasheet enumerates the known unwanted biases that this process can introduce.

3.4 Round 1 Dataset

The Round 1 dataset is summarized in Table 5, and Table 4 gives randomly selected short examples. Because each sentence has five ratings, there are two perspectives we can take on the dataset:

Distributional Labels We can repeat each example with each of its labels (de Marneffe et al., 2012; Pavlick and Kwiatkowski, 2019). For instance, the first sentence in Table 4 would be repeated three times with 'Mixed' as the label and twice with 'Negative'. For many classifier models, this reduces to labeling each example with its probability distribution over the labels. This is an appealing approach to creating training data, since it allows us to make use of all the examples,[4] even those that do not have a majority label, and it allows us to make maximal use of the labeling information. In our experiments, we found that training on the distributional labels consistently led to slightly better

[4] For 'Mixed' labels, we create two copies of the example, one labeled 'Positive', the other 'Negative'.

	SST-3		Yelp		Amazon		Round 1		Round 2	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Positive	85.1	89.0	88.3	90.5	89.1	89.4	33.3	33.3	58.4	63.0
Negative	84.1	84.1	88.8	89.1	86.6	86.6	33.3	33.3	61.0	63.1
Neutral	45.4	43.5	58.2	59.4	53.9	53.7	33.3	33.3	38.4	44.3
Macro avg	71.5	72.2	78.4	79.7	76.5	76.6	33.3	33.3	52.6	56.8

Table 3: Model 0 performance (F1 scores) on external assessment datasets (Table 1). We also report on our Round 1 dataset (Section 3.4), where performance is at chance by construction, and we report on our Round 2 dataset (Section 4) to further quantify the challenging nature of that dataset.

Sentence	Model 0	Responses
Good food nasty attitude by hostesses .	neg	**mix, mix, mix**, neg, neg
Not much of a cocktail menu that I saw.	neg	**neg, neg, neg, neg, neg**
I scheduled the work for 3 weeks later.	neg	**neu, neu, neu, neu**, pos
I was very mistaken, it was much more!	neg	neg, **pos, pos, pos, pos**
It is a gimmick, but when in Rome, I get it.	neu	**mix, mix, mix**, neu, neu
Probably a little pricey for lunch.	neu	mix, **neg, neg, neg, neg**
But this is strictly just my opinion.	neu	**neu, neu, neu, neu**, pos
The price was okay, not too pricey.	neu	mix, neu, **pos, pos, pos**
The only downside was service was a little slow.	pos	**mix, mix, mix**, neg, neg
However there is a 2 hr seating time limit.	pos	mix, **neg, neg, neg**, neu
With Alex, I never got that feeling.	pos	**neu, neu, neu, neu**, pos
Its ran very well by management.	pos	**pos, pos, pos, pos, pos**

Table 4: Round 1 train set examples, randomly selected from each combination of Model 0 prediction and majority label, but limited to examples with 30–50 characters.

	Dist	Majority Label		
	Train	Train	Dev	Test
Positive	130,045	21,391	1,200	1,200
Negative	86,486	14,021	1,200	1,200
Neutral	215,935	45,076	1,200	1,200
Mixed	39,829	3,900	0	0
No Majority	–	10,071	0	0
Total	472,295	94,459	3,600	3,600

Table 5: Round 1 Dataset.

models, suggesting that annotator disagreement is stable and informative.

Majority Label We can take a more traditional route and infer a label based on the distribution of labels. In Table 5, we show the labels inferred by assuming that an example has a label just in case at least three of the five annotators chose that label. This is a conservative approach that creates a fairly large 'No Majority' category. More sophisticated approaches might allow us to make fuller use of the examples and account for biases relating to annotator quality and example complexity (see Section 2.3). We set these options aside for now because our validation process placed more weight on the best workers we could recruit (Section 3.3).

The Majority Label splits given by Table 5 are designed to ensure five properties: (1) the classes are balanced, (2) Model 0 performs at chance, (3) the review-level rating associated with the sentence has no predictive value, (4) at least four of the five workers agreed, and (5) the majority label is Positive, Negative, or Neutral. (This excludes examples that received a Mixed majority and examples without a majority label at all.)

Over the entire round, 47% of cases are such that the validation majority label is Positive, Negative, or Neutral and Model 0 predicted a different label.

3.5 Estimating Human Performance

Table 6a provides a conservative estimate of human F1 in order to have a quantity that is comparable to our model assessments. To do this, we randomize the responses for each example to create five synthetic annotators, and we calculate the precision, recall, and F1 scores for each of these annotators with respect to the gold label. We average those scores. This heavily weights the single annotator who disagreed for the cases with 4/5 majorities. We

	Dev	Test
Pos	88.1	87.8
Neg	89.2	89.3
Neu	86.6	86.9
Avg	88.0	88.0

(a) Round 1. Fleiss κ: 0.62 dev, 0.62 test. 614 of 1,280 workers never disagreed with the gold label.

	Dev	Test
Pos	91.0	90.9
Neg	91.2	91.0
Neu	88.9	88.2
Avg	90.4	90.0

(b) Round 2. Fleiss κ: 0.68 dev, 0.67 test. 116 of 244 workers never disagreed with the gold label.

Table 6: Estimates of human performance (F1 scores) from comparing random synthesized human annotators against the gold labels using the response distributions. These are conservative estimates, offered as a way of tracking model performance to determine when the round is "solved" and a new round should begin.

	CR	IMDB	SST-3	Yelp	Amazon	Round 1
Pos	2,405	12,500	128,016	29,841	133,411	339,748
Neg	1,366	12,500	104,832	30,086	133,267	252,630
Neu	0	0	244,974	30,073	133,322	431,870
Total	3,771	25,000	477,822	90,000	400,000	1,024,248

Table 7: Model 1 training data. CR and IMDB are unchanged from Table 2. SST-3 is repeated 3 times. For Yelp and Amazon, we sample 1-, 3-, and 5-star reviews with the goal of down-weighting them and removing ambiguous reviews. Round 1 uses distributional labels and is copied twice.

can balance this against the fact that 614 of 1,280 workers *never* disagreed with the majority label (see Appendix B for the full distribution). However, it seems reasonable to say that a model has solved the round if it achieves comparable scores to our aggregate F1 – a signal to start a new round.

4 Round 2: Dynabench

In Round 2, we leverage Dynabench to begin creating a new dynamic sentiment benchmark.

4.1 Model 1

Model 1 was created using the same general methods as for Model 0 (Section 3.1): we begin with RoBERTa parameters and add a three-way sentiment classifier head. The differences between the two models lie in the data they were trained on. The train set is summarized in Table 7, and Appendix A provides additional details.

Table 8 summarizes the performance of our model on the same evaluation sets as are reported in Table 8 for Model 0. Overall, we see a small performance drop on the external datasets, but a huge jump in performance on our dataset (Round 1). While it is unfortunate to see a decline in performance on the external datasets, this is expected if we are shifting the label distribution with our new dataset – it might be an inevitable consequence of hill-climbing in our intended direction.

4.2 Dynabench Interface

Our data distribution provides the Dynabench interface we created for DynaSent as well the complete instructions and training items given to workers. The essence of the task is that the worker chooses a label y to target and then seeks to write an example that the model (currently, Model 1) assigns a label other than y but that other humans would label y. Workers can try repeatedly to fool the model, and they get feedback on the model's predictions as a guide for how to fool it.

4.3 Methods

We consider two conditions. In the **Prompt** condition, workers are shown a sentence and given the opportunity to modify it as part of achieving their goal. Prompts are sampled from parts of the Yelp Academic Dataset not used for Round 1. In the **No Prompt** condition, workers wrote sentences from scratch, with no guidance beyond their goal of fooling the model. We piloted both versions and compared the results. Our analyses are summarized in Section 5.1. The findings led us to drop the No Prompt condition and use the Prompt condition exclusively, as it clearly leads to examples that are more naturalistic and linguistically diverse.

For Round 2, our intention was for each prompt to be used only once, but prompts were repeated in a small number of cases. We have ensured that our dev and test sets contain only sentences derived from unique prompts (Section 4.5).

4.4 Validation

We used the identical validation process as described in Section 3.3, getting five responses for each example as before. This again opens up the possibility of using label distributions or inferring individual labels. 395 workers participated in this round. See Appendix B for additional details.

4.5 Round 2 Dataset

Table 10 summarizes our Round 2 dataset, and Table 9 provides train examples from Round 2 sampled using the same criteria we used for Table 4. Overall, workers' success rate in fooling Model 1

	SST-3		Yelp		Amazon		Round 1		Round 2	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Positive	84.6	88.6	80.0	83.1	83.3	83.3	81.0	80.4	33.3	33.3
Negative	82.7	84.4	79.5	79.6	78.7	78.8	80.5	80.2	33.3	33.3
Neutral	40.0	45.2	56.7	56.6	55.5	55.4	83.1	83.5	33.3	33.3
Macro avg	69.1	72.7	72.1	73.1	72.5	72.5	81.5	81.4	33.3	33.3

Table 8: Model 1 performance (F1 scores) on external assessment datasets (Table 1), as well as our Round 1 and Round 2 datasets. Chance performance for this model on Round 2 is by design (Section 4.5).

Sentence	Model 1	Responses
The place was somewhat good and not well	neg	**mix, mix, mix, mix**, neg
I bought a new car and met with an accident.	neg	**neg, neg, neg, neg, neg**
The retail store is closed for now at least.	neg	**neu, neu, neu, neu, neu**
Prices are basically like garage sale prices.	neg	neg, neu, **pos, pos, pos**
That book was good. I need to get rid of it.	neu	**mix, mix, mix**, neg, pos
I REALLY wanted to like this place	neu	mix, **neg, neg, neg**, pos
But I'm going to leave my money for the next vet.	neu	neg, **neu, neu, neu, neu**
once upon a time the model made a super decision.	neu	**pos, pos, pos, pos, pos**
I cook my caribbean food and it was okay	pos	**mix, mix, mix**, pos, pos
This concept is really cool in name only.	pos	mix, **neg, neg, neg**, neu
Wow, it'd be super cool if you could join us	pos	**neu, neu, neu, neu**, pos
Knife cut thru it like butter! It was great.	pos	**pos, pos, pos, pos, pos**

Table 9: Round 2 train set examples, randomly selected from each combination of Model 1 prediction and majority label, but limited to examples with 30–50 characters.

	Dist	Majority Label		
	Train	Train	Dev	Test
Positive	32,551	6,038	240	240
Negative	24,994	4,579	240	240
Neutral	16,365	2,448	240	240
Mixed	18,765	3,334	0	0
No Majority	–	2,136	0	0
Total	92,675	18,535	720	720

Table 10: Round 2 Dataset.

is about 19%, which is much lower than the comparable value for Round 1 (47%). There seem to be three central reasons for this. First, Model 1 is hard to fool, so many workers reach the maximum number of attempts. We retain the examples they enter, as many of them are interesting in their own right. Second, some workers seem to get confused about the true goal and enter sentences that the model in fact handles correctly. Some non-trivial rate of confusion here seems inevitable given the cognitive demands of the task, but we have taken steps to improve the interface to minimize this factor. Third, a common strategy is to create examples with mixed sentiment; the model does not predict this label, but it is chosen at a high rate in validation.

Despite these factors, we can construct splits that meet our core goals: (1) Model 1 performs at chance on the dev and test sets, and (2) the dev and test sets contain only examples where the majority label was chosen by at least four of the five workers. In addition, (3) our dev and test sets contain only examples from the Prompt condition (the No Prompt cases are in the train set, and flagged as such), and (4) all the dev and test sentences are derived from unique prompts to avoid leakage between train and assessment sets and reduce unwanted correlations within the assessment sets.

4.6 Estimating Human Performance

Table 6b provides estimates of human F1 for Round 2 using the same methods as described in Section 3.5. We again emphasize that these are conservative estimates. A large percentage of workers (116 of 244) never disagreed with the gold label on the examples they rated, suggesting that human performance can approach perfection. Nonetheless, the estimates we give here seem useful for helping us decide whether to continue hill-climbing on this round or begin creating new rounds.

5 Discussion

We now address a range of issues that our methods raise but that we have so far deferred in the interest of succinctly reporting on the methods themselves.

5.1 The Role of Prompts

As discussed in Section 4, we explored two methods for collecting original sentences on Dynabench: with and without a prompt sentence that workers could edit to achieve their goal. We did small pilot rounds in each condition and assessed the results. This led us to use the Prompt condition exclusively. This section explains our reasoning more fully.

First, we note that workers did in fact make use of the prompts. In Figure 2a, we plot the Levenshtein edit distance between the prompts provided to annotators and the examples the annotators produced, normalized by the length of the prompt or the example, whichever is longer. There is a roughly bimodal distribution in this plot, where the peak on the right represents examples generated by the annotator tweaking the prompt slightly and the peak on the left represents examples where they deviated significantly from the prompt. Essentially no examples fall at the extreme ends (literal reuse of the prompt; complete disregard for the prompt).

Second, we observe that examples generated in the Prompt condition are generally longer than those in the No Prompt condition, and more like our Round 1 examples. Figure 2b summarizes for string lengths; the picture is essentially the same for tokenized word counts. In addition, the Prompt examples have a more diverse vocabulary overall. Figure 2c provides evidence for this: we sampled 100 examples from each condition 500 times, sampled five words from each example, and calculated the vocabulary size (unique token count) for each sample. (These measures are intended to control for the known correlation between token counts and vocabulary sizes; Baayen 2001.) The Prompt-condition vocabularies are much larger, and again more similar to our Round 1 examples.

Third, a qualitative analysis further substantiates the above picture. For example, many workers realized that they could fool the model by attributing a sentiment to another group and then denying it, as in "They said it would be great, but they were wrong". As a result, there are dozens of examples in the No Prompt condition that employ this strategy. Individual workers hit upon more idiosyncratic strategies and repeatedly used them. This is just the sort of behavior that we know can create persistent dataset artifacts. For this reason, we include No Prompt examples in the training data only, and we make it easy to identify them in case one wants to handle them specially.

5.2 The Neutral Category

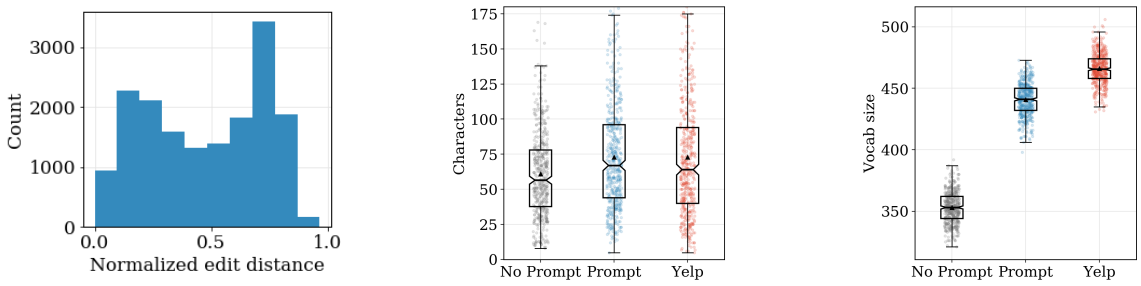
For both Model 0 and Model 1, there is consistently a large gap between performance on the Neutral category and performance on the other categories, but only for the external datasets we use for evaluation. For our dataset, performance across all three categories is fairly consistent. We hypothesized that this traces to semantic diversity in the Neutral categories for these external datasets. In review corpora, three-star reviews can signal neutrality, but they are also likely to signal mixed sentiment or uncertain overall assessments. Similarly, where the ratings are assigned by readers, as in the SST, it seems likely that the middle of the scale will also be used to register mixed and uncertain sentiment, along with a real lack of sentiment.

To further support this hypothesis, we ran the SST dev set through our validation pipeline. This leads to a completely relabeled dataset (distributed with DynaSent) with five ratings for each example and a richer array of categories. The new labels are closely aligned with SST's for Positive and Negative, but the SST-3 Neutral category has a large percentage of cases falling into Mixed and No Majority. Appendix D provides the full comparison matrix and gives a random sample of cases where the two label sets differ with regard to the Neutral category. It also provides all seven cases of sentiment confusion. We think these comparisons favor our labels over SST's original labels.

5.3 Fine-Tuning

Our Model 1 was trained from scratch (beginning with RoBERTa parameters)d. An appealing alternative would be to begin with Model 0 and fine-tune it on our Round 1 data. This would be more efficient, and it might naturally lead to the Round 1 data receiving the desired overall weight relative to the other datasets. Unfortunately, our attempts at this led to worse models, and the problems traced to very low performance on the Neutral category.

To study the effect of our dataset on Model 1 performance, we employ the "fine-tuning by inoculation" method of Liu et al. (2019a). We first divide our Round 1 train set into small subsets via random sampling. Then, we fine-tune our Model 0



(a) Normalized edit distances between the prompt and the example.

(b) String lengths. The picture is essentially the same for tokenized word counts.

(c) Vocabulary sizes in samples of 100 examples (500 samples with replacement).

Figure 2: The 'Prompt' and 'No Prompt' conditions.

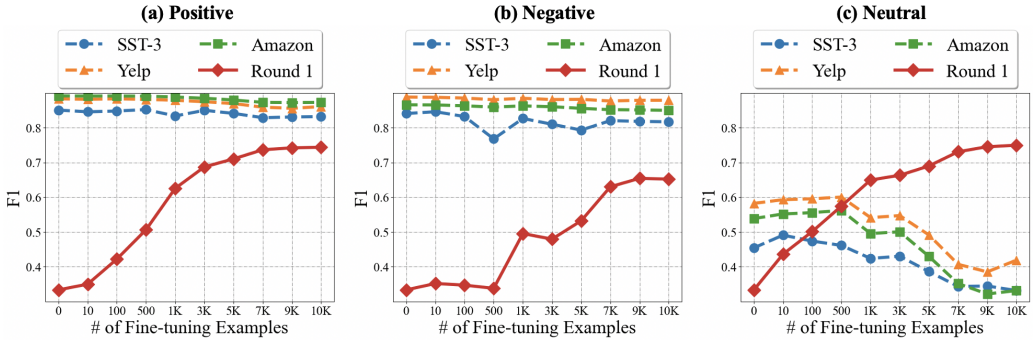


Figure 3: Inoculation by fine-tuning results with different number of fine-tuning examples.

using these subsets of Round 1 train with non-distributional labels. We early-stop our fine-tuning process if performance on the Round 0 dev set of Model 0 (SST-3 dev) has not improved for five epochs. Lastly, we measure model performance with Round 1 dev (SST-3 dev plus Round 1 dev) and our external evaluation sets (Table 1).

Figure 3 presents F1 scores for our three class labels using this method. Model performance on Round 1 dev increases for all three labels given more training examples. The F1 scores for the Positive and Negative classes remain high, but they begin to drop slightly with larger samples. The F1 scores on SST-3 dev show larger perturbations. The most striking trends are for the Neutral category, where the F1 score on Round 1 dev increases steadily while the F1 scores on the three original development sets for Model 0 decrease drastically. This is the pattern that Liu et al. (2019a) associate with dataset artifacts or label distribution shifts.

Our current hypothesis is that the pattern we observe can be attributed, at least in large part, to label shift – specifically, to the difference between our Neutral category and the other Neutral categories, as discussed in the preceding section. Our strategy of training from scratch seems less susceptible to these issues, though the label shift is still arguably a factor in the lower performance we see on this category with our external validation sets.

6 Conclusion

We presented DynaSent, as the first stage in an ongoing effort to create a dynamic benchmark for sentiment analysis. To date, the best future-looking Model 2 we have developed achieves 83.1 F1 on Round 1 and 70.8 F1 on Round 2 while maintaining good performance on our external benchmarks. Appendix E provides details on this model and others, and the Dynabench platform offers a detailed and up-to-date leaderboard. We hope and expect that the community will find models that solve both rounds. That will be our cue to launch another round of data collection to fool those models and push the field of sentiment forward by another step.

Acknowledgements

Our thanks to the developers of the Dynabench Platform, and special thanks to our Amazon Mechanical Turk workers for their essential contributions to this project. This research is supported in part by faculty research grants from Facebook and Google.

Impact Statement

DynaSent is distributed with a detailed Datasheet (Gebru et al., 2018) that describes the data collection process and its motivations, and seeks to articulate known limitations of the resource. The data distribution also includes a Model card (Mitchell et al., 2019) that seeks to provide similar disclosures concerning Model 0 and Model 1. Taken together, these documents further articulate our central goals for these resources and provide guidance on responsible use. These documents will be upated appropropriately as DynaSent and our associated models evolve.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

R. Harald Baayen. 2001. *Word Frequency Distributions*. Kluwer Academic Publishers, Dordrecht.

Max Bartolo, Alastair Roberts, Johannes Welbl, Sebastian Riedel, and Pontus Stenetorp. 2020. Beat the AI: Investigating adversarial human annotation for reading comprehension. *Transactions of the Association for Computational Linguistics*, 8:662–678.

Yonatan Belinkov, Adam Poliak, Stuart Shieber, Benjamin Van Durme, and Alexander Rush. 2019. Don't take the premise for granted: Mitigating artifacts in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 877–891, Florence, Italy. Association for Computational Linguistics.

John Burn-Murdoch. 2013. Social media analytics: Are we nearly there yet? *The Guardian*.

Michael Chen, Mike D'Arcy, Alisa Liu, Jared Fernandez, and Doug Downey. 2019. CODAH: An adversarially-authored question answering dataset for common sense. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*, pages 63–69, Minneapolis, USA. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2019. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

A. P. Dawid and A. M. Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):20–28.

Marie-Catherine de Marneffe, Christopher D. Manning, and Christopher Potts. 2012. Did it happen? the pragmatic complexity of veridicality assessment. *Computational Linguistics*, 38(2):301–333.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Ujwal Gadiraju, Besnik Fetahu, Ricardo Kawase, Patrick Siehndel, and Stefan Dietze. 2017. Using worker self-assessments for competence-based preselection in crowdsourcing microtasks. *ACM Transactions of Computer–Human Interaction*, 24(4).

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumeé III, and Kate Crawford. 2018. Datasheets for datasets. *arXiv preprint arXiv:1803.09010*.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655, Melbourne, Australia. Association for Computational Linguistics.

Stephen Gossett. 2020. Emotion AI has great promise (when used responsibly). *Built In Blog*.

Seth Grimes. 2014. Text analytics 2014: User perspectives on solutions and providers. Technical report, Alta Plana.

Seth Grimes. 2017. Data frontiers: Subjectivity, sentiment, and sense. *Brandwatch*.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Judy Hanwen Shen, Lauren Fratamico, Iyad Rahwan, and Alexander M. Rush. 2018. Darling or babygirl? Investigating stylistic bias in sentiment analysis. In *Fairness, Accountability, and Transparency in Machine Learning*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 168–177. ACL.

Christoph Hube, Besnik Fetahu, and Ujwal Gadiraju. 2019. Understanding and mitigating worker biases in the crowdsourced collection of subjective judgments. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, pages 1–12. Association for Computing Machinery.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is BERT really robust? Natural language attack on text classification and entailment. *arXiv preprint arXiv:1907.11932*, 2.

Nitin Jindal and Bing Liu. 2008. Opinion spam and analysis. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, pages 219–230, New York, NY, USA. Association for Computing Machinery.

Divyansh Kaushik and Zachary C. Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5010–5015, Brussels, Belgium. Association for Computational Linguistics.

Svetlana Kiritchenko and Saif Mohammad. 2018. Examining gender and race bias in two hundred sentiment analysis systems. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 43–53, New Orleans, Louisiana. Association for Computational Linguistics.

Hector J. Levesque. 2013. On our best behaviour. In *Proceedings of the Twenty-third International Conference on Artificial Intelligence*, Beijing.

Tal Linzen. 2020. How can we accelerate progress towards human-like linguistic generalization? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5210–5217, Online. Association for Computational Linguistics.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool.

Nelson F. Liu, Roy Schwartz, and Noah A. Smith. 2019a. Inoculation by fine-tuning: A method for analyzing challenge datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2171–2179, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. ROBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Julian McAuley, Jure Leskovec, and Dan Jurafsky. 2012. Learning attitudes and attributes from multi-aspect reviews. In *12th International Conference on Data Mining*, pages 1020–1025, Washington, D.C. IEEE Computer Society.

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. 2019. Model cards for model reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, pages 220–229, New York, NY, USA. Association for Computing Machinery.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani, and Veselin Stoyanov. 2016. SemEval-2016 task 4: Sentiment analysis in Twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1–18, San Diego, California. Association for Computational Linguistics.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197, Hong Kong, China. Association for Computational Linguistics.

Yixin Nie, Yicheng Wang, and Mohit Bansal. 2019. Analyzing compositionality-sensitivity of NLI models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6867–6874.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278, Barcelona, Spain.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124, Ann Arbor, Michigan. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1):1–135.

Ellie Pavlick and Tom Kwiatkowski. 2019. Inherent disagreements in human textual inferences. *Transactions of the Association for Computational Linguistics*, 7:677–694.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 task 4: Sentiment analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 502–518, Vancouver, Canada. Association for Computational Linguistics.

Rachel Rudinger, Chandler May, and Benjamin Van Durme. 2017. Social bias in elicited natural language inferences. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 74–79, Valencia, Spain. Association for Computational Linguistics.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14, New Orleans, Louisiana. Association for Computational Linguistics.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

Tal Schuster, Darsh Shah, Yun Jie Serene Yeo, Daniel Roberto Filizzola Ortiz, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3419–3425, Hong Kong, China. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Ng. 2008. Cheap and fast – but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Honolulu, Hawaii. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Yi-Ting Tsai, Min-Chu Yang, and Han-Yu Chen. 2019. Adversarial attack on sentiment classification. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 233–240, Florence, Italy. Association for Computational Linguistics.

Masatoshi Tsuchiya. 2018. Performance impact caused by hidden bias of training data for recognizing textual entailment. In *Proceedings of the*

Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan. European Language Resources Association (ELRA).

Amazon Mechanical Turk. 2017. Tutorial: Best practices for managing workers in follow-up surveys or longitudinal studies. Amazon Mechanical Turk Blog.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2153–2162, Hong Kong, China. Association for Computational Linguistics.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2–3):165–210.

Terry Winograd. 1972. Understanding natural language. *Cognitive Psychology*, 3(1):1–191.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32, pages 5753–5763. Curran Associates, Inc.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Brussels, Belgium. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transactions on Intelligent Systems and Technology*, 11(3).

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 649–657. Curran Associates, Inc.

Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. 2017. Truth inference in crowdsourcing: Is the problem solved? *Proceedings of VLDB Endowment*, 10(5):541–552.

Appendix

A Model training

To train our Model 0, we import weights from the pretrained RoBERTa-base model.[5] As in the original RoBERTa-base model (Liu et al., 2019b), our models have 12 heads and 12 layers, with hidden layer size 768. They use byte-pair encoding as the tokenizer (Sennrich et al., 2016), with a maximum sequence length of 128. The initial learning rate is $2e{-}5$ for all trainable parameters, with a batch size of 8 per device (GPU). We fine-tune for 3 epochs with a dropout probability of 0.1 for both attention weights and hidden states. To foster reproducibility, our training pipeline is adapted from the Hugging Face library (Wolf et al., 2020).[6] We used 6 $\times$ GeForce RTX 2080 Ti GPU each with 11GB memory. The training process takes about 15 hours.

To train Model 0, we pooled a number of public sentiment benchmarks, as summarized in Table 2. The Customer Reviews (CR; Hu and Liu 2004) and IMDB (Maas et al., 2011) datasets have only binary labels. The other datasets have five star-rating categories. We bin these ratings by taking the lowest two ratings to be negative, the middle rating to be neutral, and the highest two ratings to be positive. The Yelp and Amazon datasets are those used in Zhang et al. 2015; the first is derived from an earlier version of the Yelp Academic Dataset, and the second is derived from the dataset used by McAuley et al. (2012). SST-3 is the ternary version of the Stanford Sentiment Treebank (Socher et al., 2013) (labels 0–1 = Neg; 2 = Neu; 3–4 = Pos). We train on the phrase-level version of the dataset (and always evaluate only on its sentence-level labels).

To train Model 1, we used the same external datasets as we use for Model 0, but with a few crucial changes, as seen in Table 7. First, we subsample the large Yelp and Amazon datasets to ensure that they do not dominate the dataset, and we include only 1-star, 3-star, and 5-star reviews to try to reduce the number of ambiguous examples. Second, we upsample SST-3 by a factor of 3 and our own dataset by a factor of 2, using the distributional labels for our dataset (Section 3.4). This gives roughly equal weight, by example, to our dataset as to all the others combined. This makes sense given our general goal of doing well on our dataset and, especially, of shifting the nature of the Neutral category to something more semantically coherent than what the other corpora provide.

[5] https://dl.fbaipublicfiles.com/fairseq/models/roberta.base.tar.gz

[6] https://github.com/huggingface/transformers

B Additional Details on Validation

B.1 Validation Interface

Figure 4 shows the interface for the validation task used for both Round 1 and Round 2. The top provides the instructions, and then one item is shown. The full task had ten items per Human Interface Task (HIT). Workers were paid US$0.25 per HIT, and all workers were paid for all their work, regardless of whether we retained their labels.

B.2 Worker Selection

Examples were uploaded to Amazon's Mechanical Turk in batches of 3–5K examples. After each round, we assessed workers by the percentage of examples they labeled for which they agreed with the majority. For example, a worker who selects Negative where three of the other workers chose Positive disagrees with the majority for that example. If a worker disagreed with the majority more than 80% of the time, we removed that worker from the annotator pool and revalidated the examples they labeled. This process was repeated iteratively over the course of the entire validation process for both rounds. Thus, many examples received more than 5 labels; we collected a total of 808,289 responses, of which 608,170 (75%) are used in the final dataset, as we keep only those by the top-ranked workers according to agreement with the majority. We observed that this iterative process led to substantial improvements to the validation labels according to our own intuitions.

To remove workers from our pool, we used a method of 'unqualifying', as described in Turk 2017. This method does no reputational damage to workers and is often used in situations where the requester must limit responses to one per worker (e.g., surveys). We do not know precisely why workers tend to disagree with the majority. The reasons are likely diverse. Possible causes include inattentiveness, poor reading comprehension, a lack of understanding of the task, and a genuinely different perspective on what examples convey. While we think our method mainly increased label quality, we recognize that it can introduce unwanted biases. We acknowledge this in our Datasheet, which is distributed with the dataset.

B.3 Worker Distribution

Figure 5 show the distribution of workers for the validation task for both rounds. In the final version of Round 1, the median number of examples per worker was 45 and the mode was 11. For Round 2, the median was 20 and the mode was 1.

B.4 Worker Agreement with Gold Labels

Figure 6 summarizes the rates at which individual workers agree with the gold label. Across the dev and test sets for both rounds, substantial numbers of workers agreed with the gold label on all of the cases they labeled, and more than half were above 95% for this agreement rate for both rounds.

C Additional Details on Dynabench Task

C.1 Interface for the Prompt Condition

Figure 7 shows an example of the Dynabench interface in the Prompt condition.

C.2 Instructions

Our data distribution includes the complete instructions for the Dynabench task, and the list of comprehension questions we required workers to answer correctly before starting.

C.3 Data Collection Pipeline

For each task, a worker has ten attempts in total to find an example that fools the model. A worker can immediately claim their payment after submitting a single fooling example, or running out of attempts. The average number of attempts per task is two before the worker generates an example that they claim fools the model. Workers are paid US$0.30 per task. A confirmation step is required if the model predicts incorrectly: we explicitly ask workers to confirm the examples they come up with are truly fooling examples.

To incentivize workers, we pay a bonus of US$0.30 for each truly fooling example according to our separate validation phase. We temporarily disallow a worker to do our task if they fail to correctly answer all our onboarding questions within five attempts. We also temporarily disallow a worker to do our task if they consistently cannot come up with truly fooling examples according to our validation task.

A worker must meet the following qualifications before accepting our tasks. First, a worker must reside in the U.S. and speak English. Second, a worker must have completed at least 1,000 tasks on Amazon Mechanical Turk with an approval rating of 98%. Lastly, a worker must not be in any of our temporarily disallowing worker pools.

We adapt the open-source software package Mephisto as our data collection tool.[7]

D SST-3 Validation Examples

Table 11 compares the SST-3 labels with the labels from our separate validation task. There are just seven cases of polarity (Positive/Negative and Negative/Positive) disagreement. These are included in Table 12. The rate of disagreement is much higher where the SST-3 Neutral category is involved, which we trace (in Section 5.2) to the nature of the SST-3 category. Table 12 gives a random selection of cases involving the Neutral category to support these claims qualitatively.

	SST-3		
	Positive	Negative	Neutral
Positive	367	2	64
Negative	5	359	57
Neutral	23	8	44
Mixed	34	35	39
No Majority	15	24	25

Table 11: Comparison of the SST-3 labels (dev set) with labels derived from our separate validation.

E A Future-Looking Model 2

As we say in Section 6, we hope that DynaSent continues to grow. A future Round 3 would use a future Model 2 (or set of such models), either to harvest naturally occurring examples or to drive another round of adversarial example creation on Dynabench. We have explored a variety of Transformer-based architectures (Vaswani et al., 2017) for Model 2, designed and optimized according to the protocols given in Appendix A: RoBERTa (Liu et al., 2019b), BERT (Devlin et al., 2019), XLNet (Yang et al., 2019), and ELECTRA (Clark et al., 2019). ELECTRA has yielded the best results so far, with 83.1 F1 on Round 1 and 70.8 on Round 2. We do not think these are the best possible models; we offer these very preliminary results in the hope that they provide some useful guidance.

[7] https://github.com/facebookresearch/Mephisto

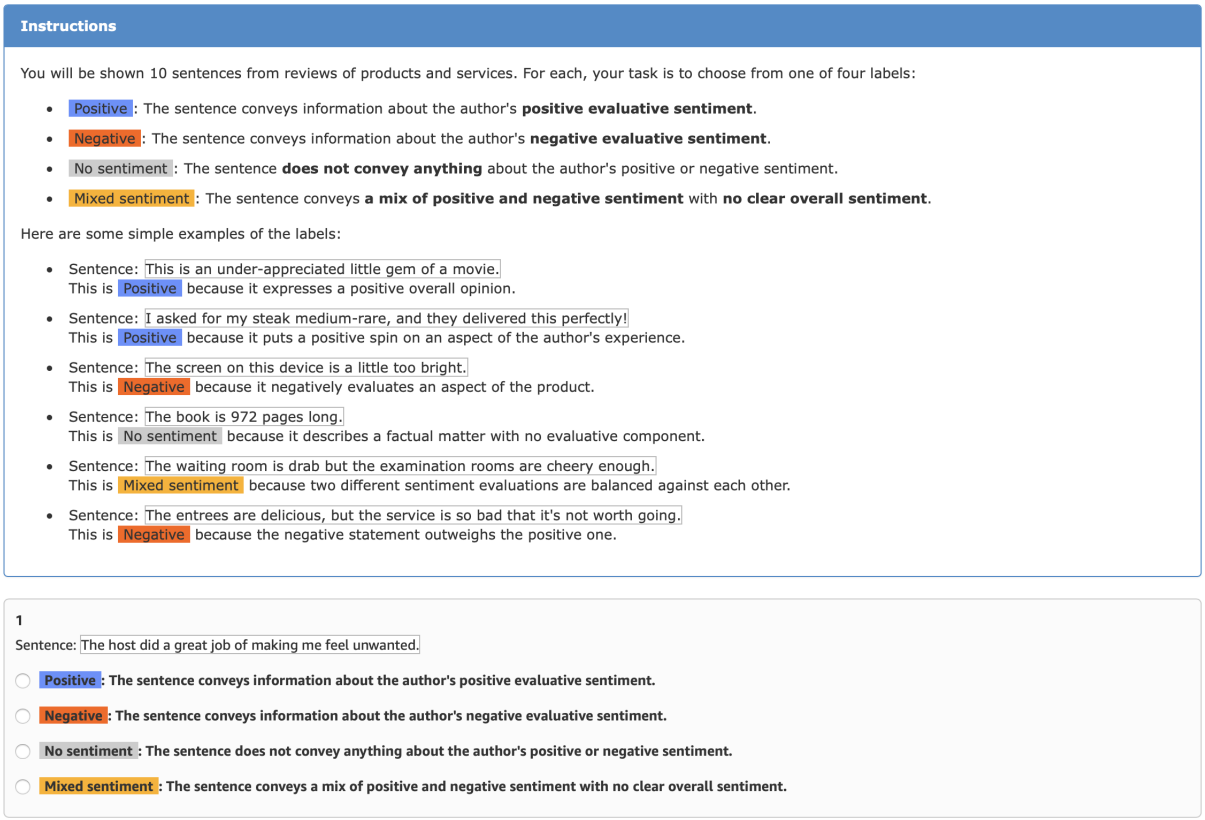
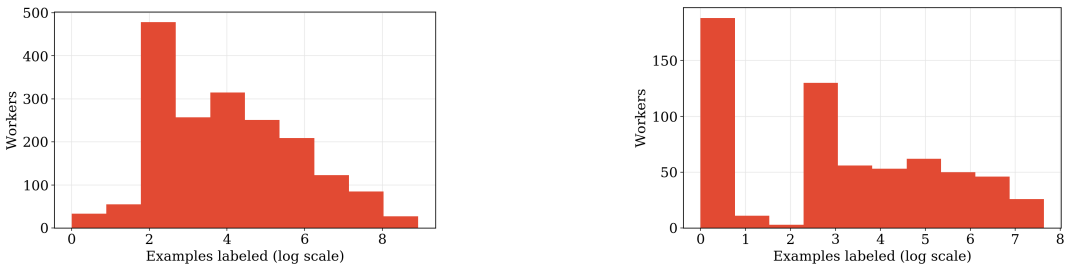
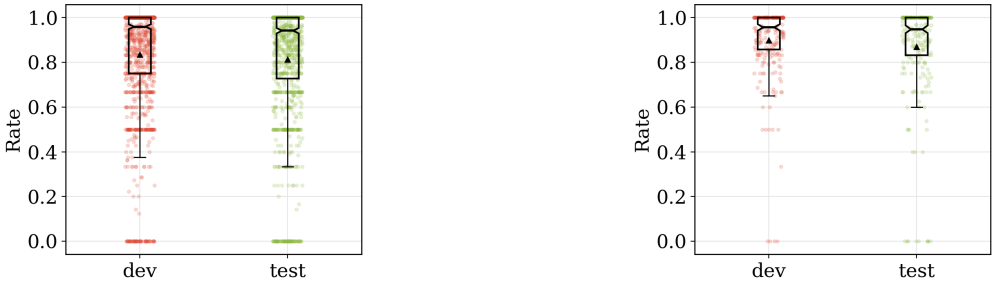


Figure 4: Validation interface.



(a) Round 1. (b) Round 2.

Figure 5: Worker distribution for the validation task.



(a) Round 1. (b) Round 2.

Figure 6: Rates at which individual worker agree with the majority label. The y-axis gives, for each worker, the total number of examples for which they chose the majority label divided by the total number of cases they labeled over all.

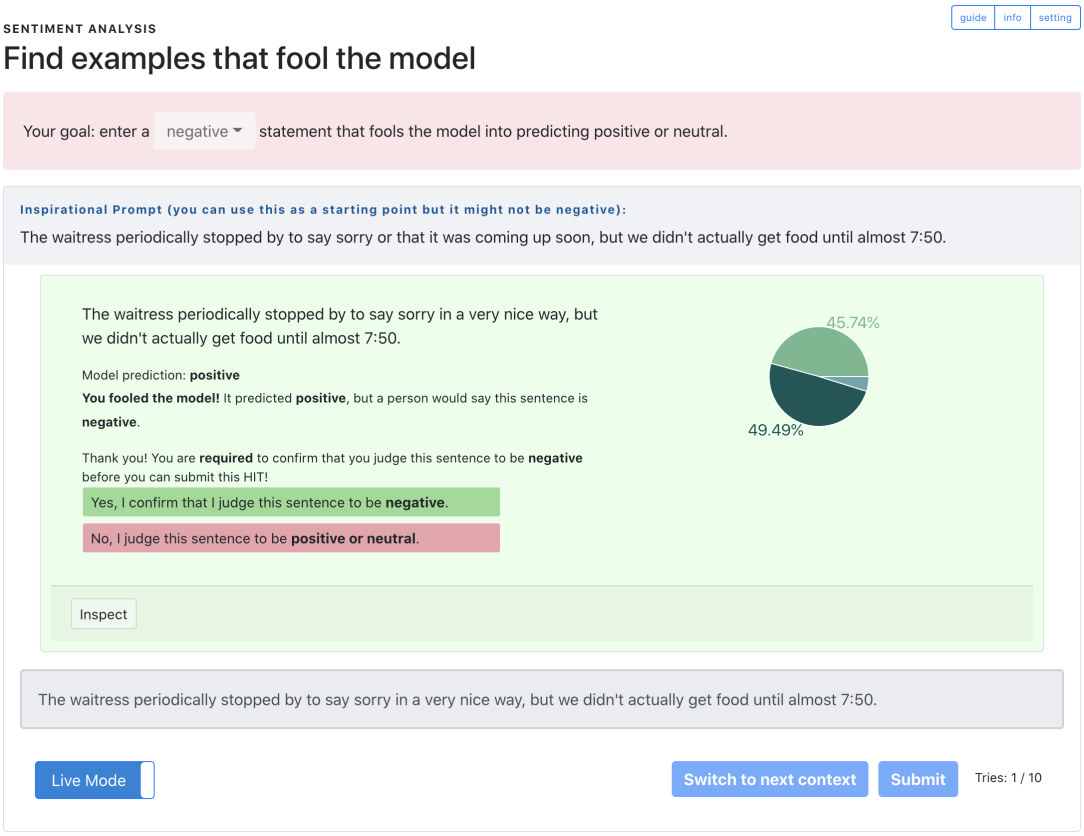


Figure 7: Dynabench interface.

	SST-3	Responses
Moretti 's compelling anatomy of grief and the difficult process of adapting to loss.	neg	neu, **pos, pos, pos, pos**
Nothing is sacred in this gut-buster.	neg	neg, neg, **pos, pos, pos**

(a) All examples for which the SST-3 label is Negative and our majority label is Positive.

	SST-3	Responses
... routine , harmless diversion and little else.	pos	mix, mix, **neg, neg, neg**
Hilariously inept and ridiculous.	pos	mix, **neg, neg, neg, neg**
Reign of Fire looks as if it was made without much thought – and is best watched that way.	pos	mix, **neg, neg, neg, neg**
So much facile technique, such cute ideas, so little movie.	pos	mix, mix, **neg, neg, neg**
While there 's something intrinsically funny about Sir Anthony Hopkins saying 'get in the car, bitch,' this Jerry Bruckheimer production has little else to offer	pos	mix, **neg, neg, neg, neg**

(b) All examples for which the SST-3 label is Positive and our majority label is Negative.

	SST-3	Responses
should be seen at the very least for its spasms of absurdist humor.	neu	**pos, pos, pos, pos, pos**
Van Wilder brings a whole new meaning to the phrase ' comedy gag . '	neu	mix, neu, **pos, pos, pos**
' They' begins and ends with scenes so terrifying I'm still stunned.	neu	neu, neu, **pos, pos, pos**
Barely gets off the ground.	neu	**neg, neg, neg, neg, neg**
As a tolerable diversion, the film suffices; a Triumph, however, it is not.	neu	**mix, mix, mix, mix**, neg

(c) A random selection of examples for which SST-3 label is Neutral and our validation label is not.

Table 12: Comparisons between the SST-3 labels and our new validation labels.