How Robust are Model Rankings : A Leaderboard Customization Approach for Equitable Evaluation

Swaroop Mishra, Anjana Arunkumar
School of Computing, Informatics, and Decision Systems Engineering
Arizona State University, Tempe, AZ 85281, USA
{srmishr1, aarunku5}@asu.edu

Abstract

Models that top leaderboards often perform unsatisfactorily when deployed in real world applications; this has necessitated rigorous and expensive pre-deployment model testing. A hitherto unexplored facet of model performance is: *Are our leaderboards doing equitable evaluation?* In this paper, we introduce a task-agnostic method to probe leaderboards by weighting samples based on their 'difficulty' level. We find that leaderboards can be adversarially attacked and top performing models may not always be the best models. We subsequently propose alternate evaluation metrics. Our experiments on 10 models show changes in model ranking and an overall reduction in previously reported performance– thus rectifying the overestimation of AI systems' capabilities. Inspired by behavioral testing principles, we further develop a prototype of a visual analytics tool that enables leaderboard revamping through customization, based on an end user's focus area. This helps users analyze models' strengths and weaknesses, and guides them in the selection of a model best suited for their application scenario. In a user study, members of various commercial product development teams, covering 5 focus areas, find that our prototype reduces pre-deployment development and testing effort by 41 % on average.

Introduction

Machine Learning (ML) models have achieved super-human performance on several popular benchmarks such as Imagenet (Russakovsky et al. 2015), SNLI (Bowman et al. 2015) and SQUAD (Rajpurkar et al. 2016). Models selected based on their success in leaderboards of existing benchmarks however, often fail when deployed in real life applications. A hitherto unexplored question that might answer why our 'optimal' model selection is sometimes found to be 'sub-optimal' is: *Are our leaderboards doing equitable evaluation?* There are two aspects associated with model evaluation– performance scores and ranking. Existing evaluation metrics have been shown to inflate model performance, and overestimate the capabilities of AI systems (Recht et al. 2019; Sakaguchi et al. 2019; Bras et al. 2020). Ranking on the other hand, has remained unexamined.

Consider that Model *X* is ranked higher than Model *Y* in a leaderboard. *X* could be solving all 'easy' questions and fail on all 'hard' questions, while *Y* solves some 'hard' questions but a lesser number of questions overall. Should *Y* be ranked higher than *X* then? The ranking depends on the application scenario– does the user want a model to compulsorily answer the hardest questions or answer more 'easy' questions?

Question 'difficulty' demarcation is application dependent. For example, in sentiment analysis for movie reviews, if a model learns to associate 'not' with negative sentiment, a review containing the phrase 'not bad' might be incorrectly labeled. Such spurious bias– unintended correlation between model input and output (Torralba and Efros 2011; Bras et al. 2020)– makes samples 'easier' for models to solve. Similarly, in conversational AI, models are often trained on clear questions ('where is the nearest restaurant'), but user queries are seen to often diverge from this training set (Kamath, Jia, and Liang 2020) ('I'm hungry'). Such atypical queries have higher out-of-distribution (OOD) characteristics, and are consequently 'harder' for models to solve.

How do we quantify and acknowledge 'difficulty' in model evaluation? Recently, semantic-textual-similarity (STS) with respect to the training set has been identified as an indicator of OOD characteristics (Mishra et al. 2020a). Similarly, we propose 2 algorithms to quantify spurious bias in a sample. We also propose *WSBias* (Weighted Spurious Bias), an equitable evaluation metric that assigns higher weights to 'harder' samples, extending the design of *WOOD* (Weighted OOD) Score (Mishra et al. 2020a).

A model's perception of sample 'difficulty' however, is based on the confidence it associates with its answer for each question. Models' tendency to silently answer incorrectly, with high confidence, hinders their deployment. For example, if Model *X* incorrectly answers questions with high confidence, but has higher accuracy than Model *Y* which has low confidence for incorrect answers, then Model *Y* is preferable, especially in safety critical domains. We quantify a model's view of difficulty using maximum softmax probability– a strong baseline which has been used as a threshold in selective answering (Hendrycks and Gimpel 2016a)– to construct *WMProb* (Weighted Maximum Probability), a metric that assigns higher weights to samples solved with higher confidence.

Based on these considerations, we make the following novel contributions to enable better model selection for deployment in real life applications:

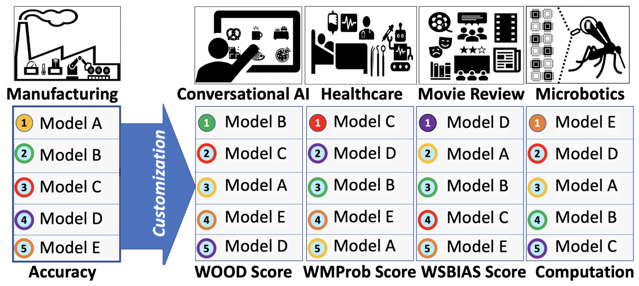

Figure 1: Leveraging various evaluation metrics for leaderboard customization, depending on application requirements. A robot in a manufacturing industry (Heyer 2010)– which is a less risky environment, compared to say healthcare– is unlikely to experience OOD data, computational constraints, and spurious bias. Thus accuracy can be used for evaluation.

- We present a methodology to probe leaderboards and check if 'better' models are ranked higher.
- We propose alternate evaluation metrics for leaderboard customization based on application specific requirements, as shown in Figure 1. To the best of our knowledge, we are the first to propose leaderboard customization.
- Our automatic sample annotation approach based on 'difficulty' indicators speeds up conventional evaluation processes in all focus areas. For example, current OOD generalization evaluation has the overhead of identifying OOD datasets corresponding to each dataset (without a consistent guideline distinguishing IID from OOD); OOD datasets might also have their own bias. Automatic annotation with STS allows for the identification of sample subsets that exhibit greater OOD characteristics, thus enabling the use of in-house data splits for evaluation.
- In order to help users select models best suited to their applications we leverage ideas from software engineering (Beizer 1995; Ribeiro et al. 2020) and develop a tool prototype that interactively visualizes how model rankings change, based on flexible weight assignment to different splits of data.
- We perform a user study with experts from industries with different focus areas, and show that our prototype helps in calibrating the proposed metrics by reducing development and testing effort on average by 41%.
- We show how the metrics reduce model performance inflation, minimizing the overestimation of capability of AI.
- Our analysis yields some preliminary observations of the strengths and weaknesses of 10 models; we recommend the use of appropriate evaluation metrics to do fair model evaluation, thus minimizing the gap between research and the real world.

Experimental Setup

We experiment with ten different models– Bag-of-Words Sum (BOW-SUM) (Harris 1954), Word2Vec Sum (W2V SUM) (Mikolov et al. 2013), GloVe Sum (GLOVE SUM) (Pennington, Socher, and Manning 2014), Word2Vec LSTM (W2V LSTM) (Hochreiter and Schmidhuber 1997), GloVe LSTM (GLOVE LSTM), Word2Vec CNN (W2V CNN) (LeCun, Bengio et al. 1995), GloVe CNN (GLOVE CNN), BERT Base (BERT BASE) (Devlin et al. 2018), BERT Large (BERT LARGE) with GELU (Hendrycks and Gimpel 2016b), and RoBERTa Large (ROBERTA LARGE) (Liu et al. 2019), following a recent work on OOD Robustness (Hendrycks et al. 2020). We analyze these models over two movie review datasets: (i) SST-2 (Socher et al. 2013)– contains short movie reviews written by experts , and (ii) IMDb (Maas et al. 2011)– has full-length lay movie reviews. We train models on SST-2 and evaluate on both SST-2 and IMDb. This setup ensures both IID (SST-2 test set) and OOD (IMDb) evaluation.

Metric Formulation

Based on our three-pronged quantification of difficulty, we adapt *WOOD* Score, and propose *WSBias* and *WMProb*.

Ranking Samples:

WSBias: Samples are ranked in decreasing order of the amount of spurious bias present. Algorithms 1 and 2 elaborate our approach to quantify the presence of spurious bias across samples. Algorithm 1 is inspired from Curriculum Learning (Xu et al. 2020) and AFLite (Sakaguchi et al. 2019; Bras et al. 2020)– a recent technique for adversarial filtering of dataset biases. Algorithm 2 incorporates bias related to train and test set overlap (Lewis, Stenetorp, and Riedel 2020; Gorman and Bedrick 2019). Spurious bias here represents the ease (% of times) with which a sample is correctly predicted by linear models on top of RoBERTA features, irrespective of inter-model agreement used in measuring sample uncertainty (Lung et al. 2013; Chen, Sun, and Chen 2014).
WOOD: First, STS is calculated for each test set value with respect to all train set samples. The STS values for the varying percentages of the train set data– ranging from the top 1%-100% of the train set, in turn obtained by sorting the train set samples in descending order of STS against each test set sample– with respect to each test set sample are averaged. The test samples are then sorted in descending order based on this averaged STS value. This is done as train-test similarity is a task dependent hyper-parameter, which can lead to either inductive or spurious bias (Mishra et al. 2020b; Gorman and Bedrick 2019).
WMProb: Test samples are ranked in increasing order, based on the confidence of model prediction for that sample (i.e., the maximum softmax probability). WMProb differs from the previous two metrics in that it operates on a model-dependent parameter, i.e., prediction confidence, rather than data-dependent (and model-independent) parameters.

Split Formation and Weight Assignment:

The dataset is divided into splits, based either on user defined thresholds of the sample 'difficulty', or such that the splits are equally populated (Figure 2). Weight assignment for samples can either be done continuously, or split-wise

Algorithm 1: Bias within Test Set

```
Result: Input: Testset $T$, Hyper-Parameters $m$, $t$,
        Models $M$-[Logistic Regression, SVM],
        Output: Spurious bias values $B$ for each
        sample $S$
Fine-tune RoBERTA on 10% of $T$ and discard this
  10% to get $R$ and find $R$'s embeddings $e$;
Evaluation Score $E(S) = 0$ for all $S$ in $R$ ;
Correct Evaluation Score $C(S) = 0$ for all $S$ in $R$ ;
forall $i \in m$ do
    Randomly select trainset of size $t$ from $R$ ;
    while $y < 2$ do
        Train $M[y]$ on $t$ using $e$ and evaluate on rest
          of $R$ i.e. $V$ ;
        forall $S \in V$ do
            $E(S) = E(S) + 1$;
            if model prediction is correct then
                $C(S) = C(S) + 1$
            end
        end
    end
end
forall $S \in T$ do
    $B(S) = C(S)/E(S)$
end
```

(Table 1). We also assign penalties to incorrectly answered samples by multiplying the assigned sample weights with penalty scores– as defined below.

Formalization:

Let D represent a dataset where T is the test set and Tr is the train set. B is the degree of bias / averaged STS value that a sample has, p is the model prediction, g is the gold annotation, a (= 1 in our experiments) is a hyper-parameter used in continuous weight calculation, b_1 and b_2 are the weights assigned per split, th_1 is the splitting parameter, either based on threshold value or equally populated splits. d and e are the reward and penalty scores respectively, which are each multiplied with both b_1 and b_2 across splits. Here, $W_{continuous}$ and $W_{two-split}$ refer to continuous and two-split weighting schemes respectively.

$$W_{continuous} = \frac{a}{B} \tag{1}$$

$$W_{two-split} = \begin{cases} b_1, & \text{if } B > th_1 \text{ (Split 1)} \\ b_2, & \text{otherwise (Split 2)} \end{cases} \tag{2}$$

$$K = \begin{cases} d, & \text{if } p = g \\ e, & \text{otherwise} \end{cases} \tag{3}$$

$$Metric = \frac{\sum_T K * W}{\sum_T d * W} * 100 \tag{4}$$

When B is the maximum softmax prediction probability, the value of $W_{continuous}$ must be reciprocated in equation 1, and the split assignment must be swapped in equation 2.

Algorithm 2: Bias across Train and Test Set

```
Result: Input: Trainset $Tr$, Testset $T$, $Z$={Logistic
        Regression, SVM Linear, SVM RBF, Naive
        Bayes} and Output: Spurious bias values $B$
        for each sample $S$
Correct Evaluation Score $C(S) = 0$ for all $S$ in $T$ ;
forall $i \in Z$ do
    Train Model $Z$ on $Tr$ and Evaluate on $T$ ;
    forall $S \in T$ do
        if model prediction is correct then
            $C(S) = C(S) + 1$
        end
    end
end
forall $S \in T$ do
    $B(S) = C(S)/4$
end
```

Reward and Penalty Scores:

For each sample, d and e are respectively used as reward and penalty multipliers of the assigned weights. This is done in order to flexibly score each sample's contribution to the overall performance. For example, in Table 1, $b_1 = 1$ for Split 1, $b_2 = 2$ for Split 2. In Case 1, $d = 1$ and $e = -1$, meaning that in Split 1, correct and incorrect samples are assigned scores of $b_1 * d = 1$ and $b_1 * e = -1$ respectively; in Split 2, this follows as $b_2 * d = 2$ and $b_2 * e = -2$. d and e can vary, as shown across the other cases. We also use quantified spurious bias, average STS, and maximum softmax probability values for d and e, as shown in Cases 6-9.

Leaderboard Probing Results

We use *WSBias*, *WOOD*, and *WMProb* Scores to probe accuracy-based model performance/ranking. Based on the metric selected, we gain insights of models' behavior over the two datasets considered, for different aspects of sample 'difficulty', in terms of overall/split-wise performance.

Range of Hyper-Parameters

We vary split numbers from 2 (as in Equation 2) to 7, which are either equally populated, or formed based on equally spaced thresholds. We use the splits in combination with all weighting schemes (as in Table 1) to calculate the metric values. Higher split numbers have additional weight and threshold hyper-parameters. For example, in the case of 4 splits, we have weight hyper-parameters of $b_1 - b_4$ and threshold hyper-parameters of th_1-th_3. We multiply each weight by d or e to assign sample scores. For example, the respective correct/incorrect sample scores per split when $b_1 - b_4$ take values from $1 - 4$, and $d = 1$, $e = -0.5$ are: +1/-0.5,+2/-1,+3/-1.5,+4/-2. We also vary the b_n hyper-parameter values along linear (additive and subtractive), logarithmic, and square scales. Results for SST-2 show lesser extents of performance inflation and ranking change– though these are still significant– which we attribute to SST-2 being the IID dataset. We report results for 7 splits over the IMDb Dataset

Case	Description	b_1	b_2	d	e	Split 1		Split 2	
						Correct	Incorrect	Correct	Incorrect
1	Reward = Penalty	1	2	1	-1	1	-1	2	-2
2	Reward Only	1	2	1	0	1	0	2	0
3	Penalty Only	1	2	0	-1	0	-1	0	-2
4	Reward $>$ Penalty	1	2	1	-0.5	1	-0.5	2	-1
5	Penalty $>$ Reward	1	2	0.5	-1	0.5	-1	1	-2
6	Continuous Weights	$1/B$	$1/B$	1	-1	$1/B$	$-1/B$	$1/B$	$-1/B$
7	Continuous Weights (*)	B	B	1	-1	B	$-B$	B	$-B$
8	Reward = Penalty = B	1	2	$1/B$	$-1/B$	$1/B$	$-1/B$	$2/B$	$-2/B$
9	Reward = Penalty = B (*)	1	2	B	$-B$	B	$-B$	$2B$	$-2B$

Table 1: Weighting Schemes Tested for *WSBias*, *WOOD*, and *WMProb*. Here, examples of weighting schemes for 2 splits are shown, with Split 1 containing the 'easy' samples, and Split 2 containing the 'hard' samples. In Cases 6 and 8, $1/B$ refers to quantified sample 'difficulty' and applies to *WSBias* and *WOOD* Score. In Cases 7 and 9, B is the 'difficulty' for *WMProb*.

below; we note that similar results are observed over different hyper-parameter considerations for both datasets.

WSBias

GLOVE-LSTM Outperforms ROBERTA-LARGE in 'Easy' Questions for Certain Splits: We see that when data is divided into 2-5 splits, ROBERTA-LARGE, despite having higher accuracy, has greater, equal, and lesser errors in Splits 1 (easiest) - 3 respectively, than GLOVE-LSTM. This indicates that ROBERTA-LARGE is unable to solve easy samples efficiently.

Transformers Fail to Answer 'Easy' Questions: In Figure 2, in the parallel coordinates plot (PCP) model performance values vary to a greater extent in Splits 1-3, and converge till they are near identical in Splits 6-7. Transformer models (BERT-BASE, BERT-LARGE, ROBERTA-LARGE) display poor performance on Split 1, with only ROBERTA-LARGE having a marginally positive value. This may be attributed to catastrophic forgetting, which occurs due to the larger quantities of data transformer models are trained on to improve generalization.

Frequent Ranking Changes Across Splits for Word Averaging and GLOVE Embedding Models: In the PCP, the occurrence of crossed/overlapping lines indicates frequent ranking changes for the BOW-SUM, GLOVE-LSTM, and GLOVE-CNN models over all the splits. This result is replicated for all weighting schemes.

Significant Changes in Model Ranking: In the multi-line chart (MLC) (Figure 3), the scores do not monotonically increase unlike the behavior seen in accuracy. There is a significant change in model ranking, with 8/10 models changing positions, as indicated by the red dots. The concurrence of significant ranking changes using both algorithms indicates that irrespective of model training, accuracy does not appropriately rank models based on spurious bias considerations.

Significant Reduction in Model Performance Inflation: The overall *WSBias* score values decrease by: (i) 25%-63% for Algorithm 1, and (ii) 11%-40% for Algorithm 2, with respect to accuracy. This indicates that *WSBias* solves accuracy's inflated model performance reporting.

Transformer Model Performance is the Least Inflated: We observe in the MLC that the performance drops seen with *WSBias* are the least for transformer models, and the most for word-averaging models. This follows from the insight that transformer models predominantly fail on 'easier' samples, i.e., those which contain greater amounts of spurious bias. As 'hard' samples– with low spurious bias– are predominantly negatively scored for simpler models, the overall *WSBias* Score of the model decreases. This is because the absolute weights assigned for those samples are higher. Our findings thus indicate that simpler models are over-reliant on spurious bias.

WOOD

Model Performance Inflation Increases as % of Training Set Used to Calculate STS Increases: We note that overall model performance increases when larger percentages of the training set are used in the calculation of averaged STS values for test samples. This is due to the corresponding decrease in the variance of the averaged STS values (values tend towards 0.5 at 100%).

Split-wise Ranking Varies More for Lesser %s of Training Set Used to Calculate STS: Due to the decreased variation effect of averaging STS values for higher percentages (> 50), there is less fluctuation in the split-wise distributions of incorrect predictions, especially in thresholded splits. As a result, the split-wise model ranking changes more frequently when lower percentages are used for STS calculation. In Figure 2 (25%), this is indicated by clear and frequent line crossings.

Overall Ranking Changes Significantly for All %s of Training Set Considered: There is significant ranking change irrespective of STS %, to a greater extent than with *WSBias*. For example, in Figure 3, we see that (i) 9/10 models change their ranking position based on WOOD (calculated with 25% STS), and (ii) there is a decrease of 15% in WOOD score values, with respect to accuracy. These respectively indicate that accuracy does not rank appropriately in the context of model generalization, and also inflates model performance on OOD datasets. These results are replicated over all weight and STS % considerations.

W2V-CNN Beats Transformers in Most Splits: The transformer models are consistently surpassed by W2V-CNN in

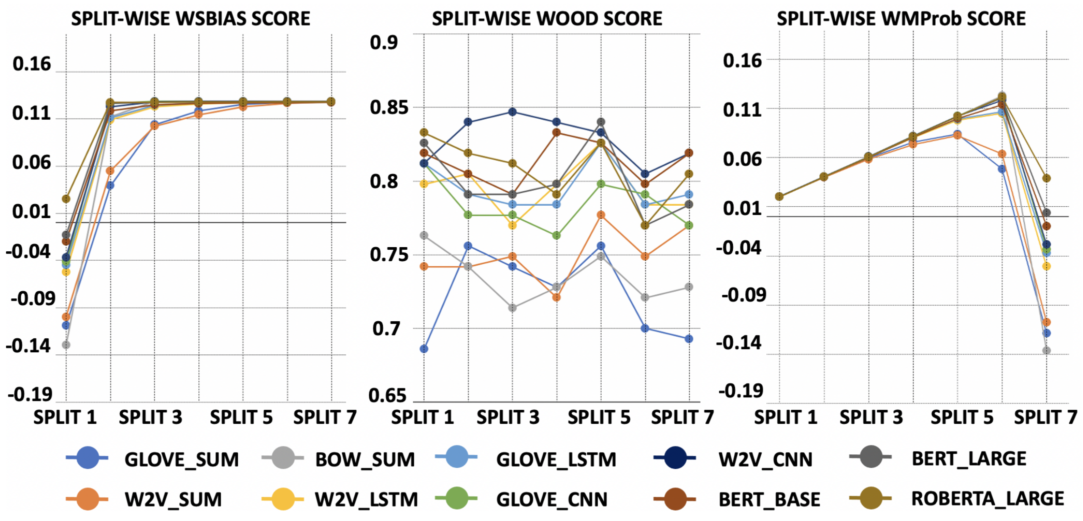

Figure 2: Split-wise results for each model using *WSBias* (based on Algorithm 1 – left), *WOOD* (based on top 25% STS averaging – middle), and *WMProb* (right) are shown in the PCPs. Each vertical line indicates the data split considered (Split 1 – Easiest/Lowest Confidence, Split 7 – Hardest/Highest Confidence). Weighting scheme considered is Case 1, for 7 equally populated splits on the IMDb dataset

Splits 2,3,4,6,7. W2V-CNN also beats transformers in overall *WOOD* Score; this is because transformers fail to solve samples with low STS (i.e., higher OOD characteristics, absolute weight assignments),and are thus heavily penalized.

WMProb

Some Models are Better Posed to Abstain than Others: When splits are formed based on both threshold and equal population constraints, the number of incorrect predictions is approximately equally distributed across splits for all models. In particular, ROBERTA-LARGE, BERT-LARGE, and GLOVE-SUM display near-equal numbers of incorrect samples at the 'easiest' (high confidence) and 'hardest' (low confidence) splits. On the other hand, BERT-BASE, GLOVE-LSTM, and W2V-LSTM show near-monotonic decrease in the number of samples correctly answered from high to low confidence. For safety critical applications, models are made to abstain from answering if the maximum softmax probability is below a specified threshold. If a lesser number of incorrect predictions (i.e., higher accuracy) is associated with high model confidence, that model is preferred for deployment.
Least Change in Ranking: The MLC (Figure 3) shows that out of the 3 metrics, accuracy most closely models *WMProb*, with 6/10 models changing ranking position; these changes are local, with models moving up or down a single rank in three swapped pairs.
All Models Perform Poorly While Answering With High Confidence: Unlike the other metrics, here we see that all models exhibit a sharp decrease in split-wise performance in the last split, where prediction confidence is highest.
Accuracy Least Inflates Model Performance in the Context of Prediction Confidence: There is only a 5%-16% decrease in *WMProb* values when compared to accuracy, indicating that while accuracy inflates performance measurement by ignoring model confidence, the influence of model confidence is lesser compared to OOD characteristics and bias used in the other metrics. Similar results are observed in other weight/split considerations.

Discussion

We find that GLOVE-LSTM efficiently utilizes spurious bias to solve 'easy' questions. Also, W2V-CNN has the highest capability for OOD generalization. While ROBERTA-LARGE has the highest number of correct predictions per split, it more or less uniformly distributes incorrect predictions across all splits (i.e., varying levels of confidence). Therefore, if ROBERTA-LARGE is augmented with the capability seen in BERT-BASE, GLOVE-LSTM and W2V-LSTM of better correlation of prediction correctness with prediction confidence, then it may be better suited for safety critical applications. Additionally, our results over both datasets show that transformers, in general, fail to solve 'easy' questions; the fact that simpler models solve easy questions more efficiently can be utilized to create ensembles, and subsequently, better architectures. Conducting further experiments on different datasets and tasks is a potential future work to generalize these findings.

Prototype for Leaderboard Probing and Customization

We develop an interactive tool prototype[1] – leveraging *WSBias*, *WOOD*, and *WMProb*– to help industry experts probe and customize leaderboards, as per their deployment scenario requirements. The provision of a platform for such structured analysis is aimed at helping experts select better

[1]https://github.com/swarooprm/Leaderboard-Customization

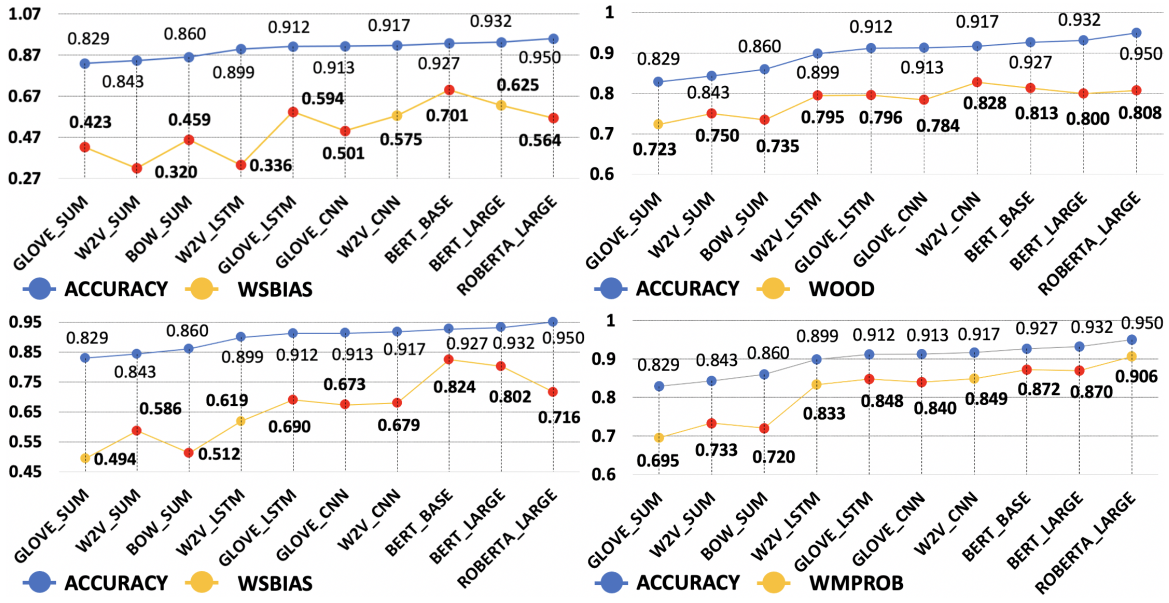

Figure 3: The MLCs shows the model performance based on Accuracy and *WSBias* (from Algorithms 1, 2) / *WOOD* / *WMProb*. Here, the yellow dots indicate that the model's ranking position is the same as that of accuracy, red dots indicate that there is a change in ranking position. Weighting scheme considered is Case 1, for 7 equally populated splits on the IMDb dataset.

models, as well as reducing their pre-deployment development and testing effort. We evaluate this prototype in two stages– (i) Expert Review, and (ii) Detailed User Study.

Expert Review

We present an initial schematic, covering ten models over the SST-2 and IMDb datasets, to five software testing experts (6+ years of work experience), each working in one of the focus areas illustrated in Figure 1. There is a shared belief among experts that *"the use of this tool will reduce testing time/effort by reducing the number of models initially vetted"* **(P1, P4)**. Additionally, *"... selecting good base models reduces testing times downstream the pipeline"* **(P3)**. Based on this initial feedback, we add a feature to custom-load model results and data (as present in conventional software testing platforms). This extends the utility of our prototype from the initial phase of probing and model selection, to extensive, iterative behavioral testing that occurs in later testing phases, prior to deployment.

User Study

We approach software developers and testing managers of several commercial product development teams (belonging to 9 companies) across all five focus areas. We first outline the basic intuitions behind the formulation of the three metrics, and then provide users with our prototype. In the visual interface, users can interactively test various splits (manual/random/automatic methods– i.e., equally populated/thresholded splits) and weighting schemes (with flexible penalty assignment, and discrete/continuous weighting) of data for different metrics. The users are then asked to fill a Google Form, juxtaposing their prototype usage with conventional model selection– refers to picking models from leaderboards (with accuracy as the metric, i.e., uniform weighting, no penalty). In conventional selection, users have to iteratively pick some models from the leaderboard, then test those models in their specific application (through test cases and deployment), until they find one that works efficiently. Figure 4 summarizes user study results.

Feedback

A total of 32 experts participated in the evaluation of our prototype. On average, **development and testing effort decrease of 41%** is reported using our customized leaderboard approach, as opposed to conventional model selection and testing approaches. This decrease is attributed to the prototype's *"... improvement for data collection time frames and downstream testing since we know exactly where the model fails."***(P7)**. **(P6)** believes our prototype *"can help in creating model ensembles"*, and **(P8)** that it will *"help with the expedited development and testing of AI models... auto-suggested model edits through the visual interface would be interesting"*. **(P14, P31)** say our prototype ... *helps compare in-house and commercial model strengths and weaknesses*. **(P20)** further attributes this facilitated comparison towards enabling *"...testing of various possible scenarios in deployment without writing test cases... we do not have to write separate test cases for stress testing"*, and **(P26)** adds that *"A big part of testing involves creating new datasets in the sandbox that mimic deployment scenarios. Dataset creation, annotation, controlling quality, and evaluation are costly. This tool's unique method of annotating existing datasets helps us significantly reduce previously spent effort."*. **(P22)** suggests *"... exploring the potential expansion of this tool to assist in customized functional testing of products"*.

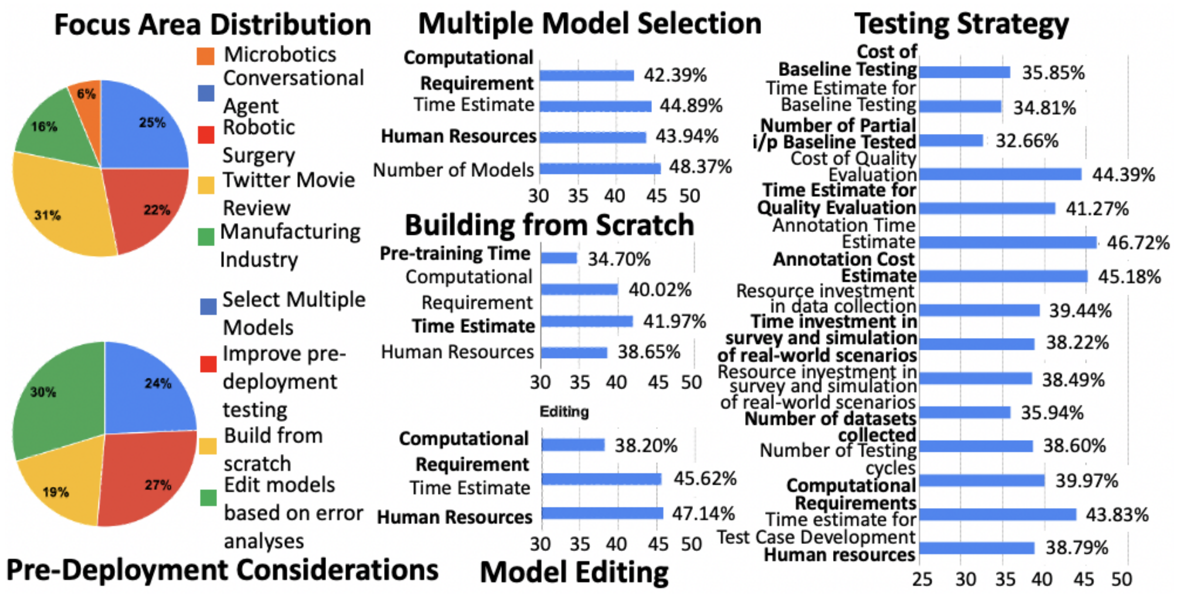

Figure 4: Reported **Decreases** in pre-deployment development and testing. Average : ***40.791%***

Related Works

Beyond Accuracy: CheckList, a matrix of general linguistic capabilities and test types, has been introduced to enable better model evaluation and address the performance overestimation associated with accuracy (Ribeiro et al. 2020). Alternate evaluation approaches such as model robustness to noise (Belinkov and Bisk 2017), Perturbation Sensitivity Analysis–a generic evaluation framework detects unintended model biases related to named entities– (Prabhakaran, Hutchinson, and Mitchell 2019), and energy usage in models (Henderson et al. 2020) (promoting Green AI (Schwartz et al. 2019; Mishra and Sachdeva 2020)) have also been proposed. Our proposed idea of weighting samples based on 'hardness', is metric-independent, and can be used to revamp other metrics beyond accuracy such as F1 Score.

Evaluation Metrics and Recommendations: The misalignment of automatic metrics with the gold standard of human evaluation for Text Generation has been studied in several works (Mathur, Baldwin, and Cohn 2020; Sellam, Das, and Parikh 2020). Our work is orthogonal to these, as we have no reference human evaluation, and our idea of weighted metrics is task-independent. Evaluation metric analysis has been used to provide recommendations (Peyrard 2019) for gathering human annotations in specified scoring ranges (Radev et al. 2003). 'Best practices' for reporting experimental results (Dodge et al. 2019), the use of randomly generated splits (Gorman and Bedrick 2019), and comparing score distributions on multiple executions (Reimers and Gurevych 2017) have been recommended. Our work recommends the reporting of performance in terms of the weighted metrics, along with accuracy, and their integration as part of leaderboard customization.

Adversarial Evaluation: Adversarial evaluation tests the capability of systems to not get distracted by text added to samples in several different forms (Jia and Liang 2017; Jin et al. 2019; Iyyer et al. 2018), intended to mislead models (Jia et al. 2019). Our work, in contrast, deals with the adversarial attack of leaderboards. Ladder (Blum and Hardt 2015) has been proposed as a reliable leaderboard for machine learning competitions, by preventing participants from overfitting to holdout sets. Our objective is orthogonal to this as it focuses on customizing leaderboards based on application specific requirements in industry.

Conclusion

We present a method to adversarially attack leaderboards, and utilize this to probe model rankings. We find that 'better' models are not always ranked higher, and therefore propose new evaluation metrics that enable equitable model evaluation, by weighting samples based on their hardness from perspectives of data and model dependencies. We use these metrics to develop a tool prototype for leaderboard customization by adapting software engineering concepts. We perform a user study with experts from nine industries, having five different focus areas, and find that our prototype helps in calibrating the proposed metrics by reducing user development and testing effort on average by 41%. Our metrics reduce inflation in model performance, thus rectifying overestimated capabilities of AI systems. Our results also give preliminary indications of the strengths and weaknesses of 10 models. We recommend the use of appropriate evaluation metrics to do fair model evaluation, thus minimizing the gap between research and the real world. In the future, we will expand our analysis to model ensembles, as they are seen to dominate leaderboards more often. Our equitable evaluation metrics can also be useful in competitions, and we plan to expand our user study to competitions in future. We hope the community expands the usage of such evaluation metrics to other domains, such as vision and speech.

Ethics Statement

The broad impact of our work is as follows:

Environmental Impact: Competitions and leaderboards in general can have a negative impact on climate change, as increasingly complex models are trained and retrained, requiring more computation time. Our proposed method could help teams reduce the exploration space to find a good model, and thus help reduce that team's overall carbon emissions.

Towards a Fair Leaderboard: Our definition of 'difficulty' can be extended based on several different perspectives. A viable application of this is a 'fair' customized leaderboard, where models having higher gender/racial bias will be heavily penalized, preventing them from dominating leaderboards.

Selecting the Best Natural Language Understanding (NLU) Systems: NLU requires a variety of linguistic capabilities and reasoning skills. Incorporating these requirements systematically as 'difficulties' in our framework will enable better selection of top NLU systems.

References

Beizer, B. 1995. *Black-box testing: techniques for functional testing of software and systems*. John Wiley & Sons, Inc.

Belinkov, Y.; and Bisk, Y. 2017. Synthetic and natural noise both break neural machine translation. *arXiv preprint arXiv:1711.02173* .

Blum, A.; and Hardt, M. 2015. The ladder: A reliable leaderboard for machine learning competitions. *arXiv preprint arXiv:1502.04585* .

Bowman, S. R.; Angeli, G.; Potts, C.; and Manning, C. D. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326* .

Bras, R. L.; Swayamdipta, S.; Bhagavatula, C.; Zellers, R.; Peters, M. E.; Sabharwal, A.; and Choi, Y. 2020. Adversarial Filters of Dataset Biases. *arXiv preprint arXiv:2002.04108* .

Chen, H.; Sun, J.; and Chen, X. 2014. Projection and uncertainty analysis of global precipitation-related extremes using CMIP5 models. *International journal of climatology* 34(8): 2730–2748.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* .

Dodge, J.; Gururangan, S.; Card, D.; Schwartz, R.; and Smith, N. A. 2019. Show Your Work: Improved Reporting of Experimental Results. In *EMNLP/IJCNLP*.

Gorman, K.; and Bedrick, S. 2019. We need to talk about standard splits. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, 2786–2791.

Harris, Z. S. 1954. Distributional structure. *Word* 10(2-3): 146–162.

Henderson, P.; Hu, J.; Romoff, J.; Brunskill, E.; Jurafsky, D.; and Pineau, J. 2020. Towards the Systematic Reporting of the Energy and Carbon Footprints of Machine Learning. *arXiv preprint arXiv:2002.05651* .

Hendrycks, D.; and Gimpel, K. 2016a. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136* .

Hendrycks, D.; and Gimpel, K. 2016b. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415* .

Hendrycks, D.; Liu, X.; Wallace, E.; Dziedzic, A.; Krishnan, R.; and Song, D. 2020. Pretrained Transformers Improve Out-of-Distribution Robustness. *arXiv preprint arXiv:2004.06100* .

Heyer, C. 2010. Human-robot interaction and future industrial robotics applications. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4749–4754. IEEE.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8): 1735–1780.

Iyyer, M.; Wieting, J.; Gimpel, K.; and Zettlemoyer, L. 2018. Adversarial example generation with syntactically controlled paraphrase networks. *arXiv preprint arXiv:1804.06059* .

Jia, R.; and Liang, P. 2017. Adversarial Examples for Evaluating Reading Comprehension Systems. *ArXiv* abs/1707.07328.

Jia, R.; Raghunathan, A.; Göksel, K.; and Liang, P. 2019. Certified Robustness to Adversarial Word Substitutions. In *EMNLP/IJCNLP*.

Jin, D.; Jin, Z.; Tianyi Zhou, J.; and Szolovits, P. 2019. Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment. *arXiv* arXiv–1907.

Kamath, A.; Jia, R.; and Liang, P. 2020. Selective Question Answering under Domain Shift. *arXiv preprint arXiv:2006.09462* .

LeCun, Y.; Bengio, Y.; et al. 1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks* 3361(10): 1995.

Lewis, P.; Stenetorp, P.; and Riedel, S. 2020. Question and Answer Test-Train Overlap in Open-Domain Question Answering Datasets. *arXiv preprint arXiv:2008.02637* .

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* .

Lung, T.; Dosio, A.; Becker, W.; Lavalle, C.; and Bouwer, L. M. 2013. Assessing the influence of climate model uncertainty on EU-wide climate change impact indicators. *Climatic change* 120(1-2): 211–227.

Maas, A. L.; Daly, R. E.; Pham, P. T.; Huang, D.; Ng, A. Y.; and Potts, C. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, 142–150. Association for Computational Linguistics.

Mathur, N.; Baldwin, T.; and Cohn, T. 2020. Tangled up in BLEU: Reevaluating the Evaluation of Automatic Machine Translation Evaluation Metrics. In *ACL*.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, 3111–3119.

Mishra, S.; Arunkumar, A.; Bryan, C.; and Baral, C. 2020a. Our Evaluation Metric Needs an Update to Encourage Generalization. *ArXiv* abs/2007.06898.

Mishra, S.; Arunkumar, A.; Sachdeva, B. S.; Bryan, C.; and Baral, C. 2020b. DQI: Measuring Data Quality in NLP. *ArXiv* abs/2005.00816.

Mishra, S.; and Sachdeva, B. S. 2020. Do We Need to Create Big Datasets to Learn a Task? In *SUSTAINLP*.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Peyrard, M. 2019. Studying Summarization Evaluation Metrics in the Appropriate Scoring Range. In *ACL*.

Prabhakaran, V.; Hutchinson, B.; and Mitchell, M. 2019. Perturbation sensitivity analysis to detect unintended model biases. *arXiv preprint arXiv:1910.04210* .

Radev, D.; Teufel, S.; Saggion, H.; Lam, W.; Blitzer, J.; Qi, H.; Celebi, A.; Liu, D.; and Drabek, E. F. 2003. Evaluation challenges in large-scale document summarization. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, 375–382.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250* .

Recht, B.; Roelofs, R.; Schmidt, L.; and Shankar, V. 2019. Do imagenet classifiers generalize to imagenet? *arXiv preprint arXiv:1902.10811* .

Reimers, N.; and Gurevych, I. 2017. Reporting Score Distributions Makes a Difference: Performance Study of LSTM-networks for Sequence Tagging. *ArXiv* abs/1707.09861.

Ribeiro, M. T.; Wu, T.; Guestrin, C.; and Singh, S. 2020. Beyond Accuracy: Behavioral Testing of NLP models with CheckList. In *ACL*.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision* 115(3): 211–252.

Sakaguchi, K.; Bras, R. L.; Bhagavatula, C.; and Choi, Y. 2019. Winogrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641* .

Schwartz, R.; Dodge, J.; Smith, N. A.; and Etzioni, O. 2019. Green ai. *arXiv preprint arXiv:1907.10597* .

Sellam, T.; Das, D.; and Parikh, A. P. 2020. BLEURT: Learning Robust Metrics for Text Generation. In *ACL*.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A. Y.; and Potts, C. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, 1631–1642.

Torralba, A.; and Efros, A. A. 2011. Unbiased look at dataset bias. In *CVPR 2011*, 1521–1528. IEEE.

Xu, B.; Zhang, L.; Mao, Z.; Wang, Q.; Xie, H.; and Zhang, Y. 2020. Curriculum Learning for Natural Language Understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 6095–6104.