Dynatask: A Framework for Creating Dynamic AI Benchmark Tasks

Tristan Thrush[‡,*], **Kushal Tirumala**[†], **Anmol Gupta**[§], **Max Bartolo**[¶], **Pedro Rodriguez**[†], **Tariq Kane**[††], **William Gaviria Rojas**[††], **Peter Mattson**[⋆], **Adina Williams**[†], **Douwe Kiela**[‡]

[‡] Hugging Face; [†] Facebook AI Research; [§] The University of Hong Kong; [¶] University College London; [⋆] Google; [††] Coactive AI;

dynabench@fb.com

Abstract

We introduce Dynatask: an open source system for setting up custom NLP tasks that aims to greatly lower the technical knowledge and effort required for hosting and evaluating state-of-the-art NLP models, as well as for conducting model in the loop data collection with crowdworkers. Dynatask is integrated with Dynabench, a research platform for rethinking benchmarking in AI that facilitates human and model in the loop data collection and evaluation. To create a task, users only need to write a short task configuration file from which the relevant web interfaces and model hosting infrastructure are automatically generated. The system is available at https://dynabench.org/ and the full library can be found at https://github.com/facebookresearch/dynabench.

1 Introduction

Data is the backbone of NLP research. One of the most fruitful approaches for making progress on NLP tasks has historically been *benchmarking*. Benchmarking is where the community adopts a high quality dataset for a particular task and tests various models against it to determine which is best. The process of benchmarking requires the effort of a large number of researchers, who collect and clean data, train and evaluate models, and work to understand model weaknesses. This process is iterative: once models perform very highly on the currently accepted community benchmark, another is created to push progress further. Taken as a whole, the benchmarking process is both notoriously difficult and expensive. This is due to a variety of facts: the community is a loose conglomeration of researchers with different areas of expertise, there is ever increasing need for larger datasets (Halevy et al., 2009), and the AI community has historically under-valued (Wagstaff, 2012) and under-invested in data collection and best practices (Kiela et al., 2021; Sambasivan et al., 2021; Mattson et al., 2022).

To make matters worse, in recent years, benchmarks have been saturating with increasing speed. Taking the trends from the greater AI community into account, it took MNIST (LeCun et al., 1998), Switchboard (Godfrey et al., 1992), and ImageNet (Deng et al., 2009) several years to saturate, and newer benchmarks such as SQuAD (Rajpurkar et al., 2016), GLUE (Wang et al., 2018), and SuperGLUE (Wang et al., 2019) about a year. Because of this, data-centric approaches are gaining more attention (Ng et al., 2021; Mattson et al., 2022; Lhoest et al., 2021; Paullada et al., 2021; Luccioni et al., 2021). This trend is clear evidence of the urgency of finding a sustainable and data-centric way to support the full benchmarking ecosystem, from end-to-end, in a way that causes the least amount of friction for anyone who wants to use it.

In this paper, we introduce our answer to these issues: an easy-to-use, open source system that integrates the creation of benchmark datasets for any task, the selection of appropriate metrics, and the evaluation of models while natively supporting revisions to the benchmark as models saturate the original version. We share a unified library that enables these functionalities for the Dynabench platform (Kiela et al., 2021).

2 Background

Dynabench was proposed as an open-source and community-driven platform to host dynamic benchmarks. The existing Dynabench tasks avoid saturation by leveraging crowdworkers who continually interact with state-of-the-art models. Crowdworkers either write examples that fool existing models (Nie et al., 2020), or collaborate with generative models to increase example diversity (Bartolo et al., 2021b). Each task is administered by one or more *task owners* from the research community who col-

[*] TT and DK conducted most of the work for this paper when they were at Facebook AI Research.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
System Demonstrations, pages 174 - 181

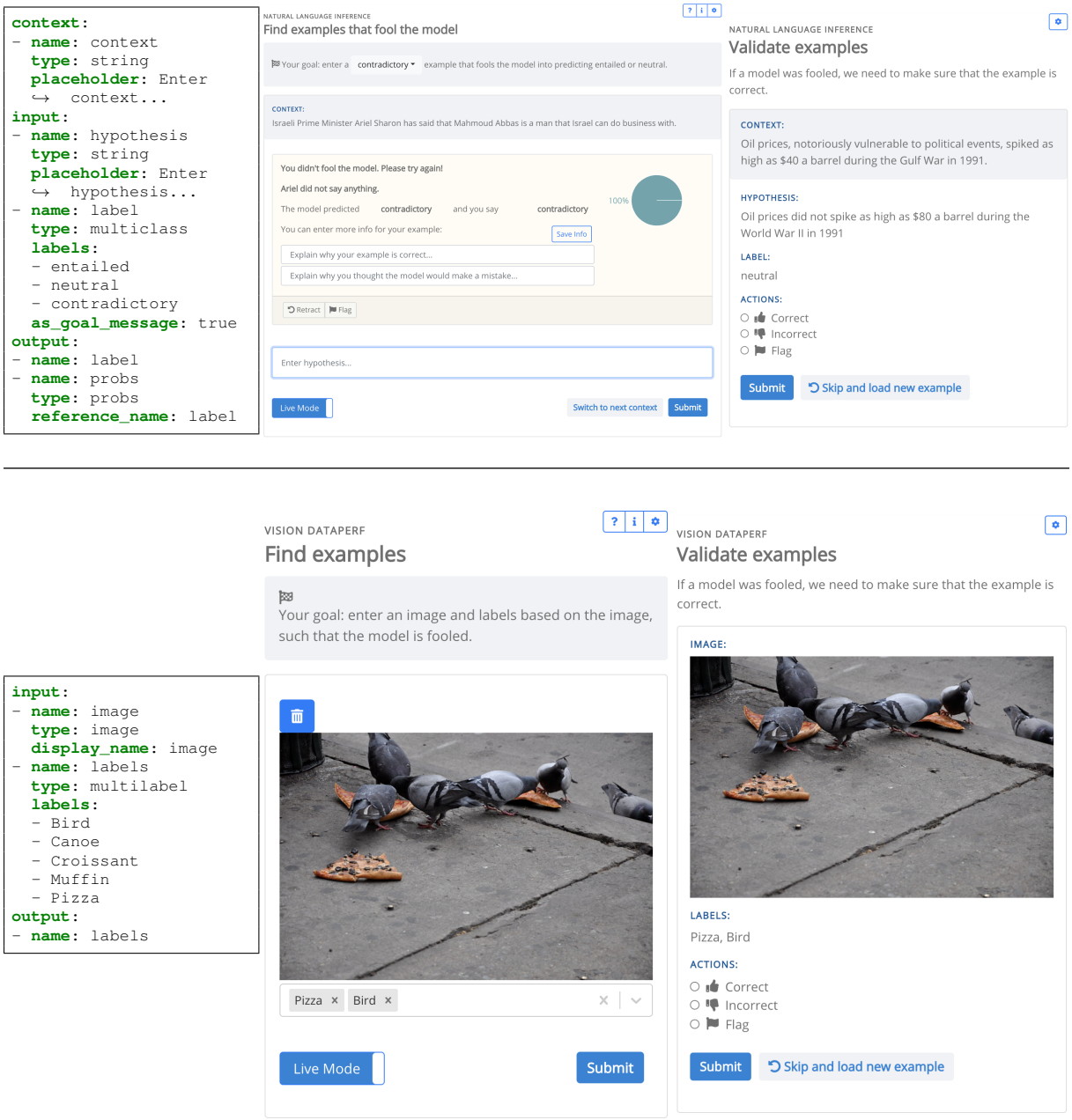

Figure 1: Two example config files and the data collection and validation interfaces they generate. Only config fields that impact the data collection interfaces are shown (e.g. metrics for model ranking are not shown). The Context and Input fields define the type of data that humans can enter. The Output field defines what models will output, given the Context and the Input. Crowdworkers are typically expected to provide the gold truth annotations for a task. In this case, Output will contain some of the object names from Input and Context. These gold truth annotations are removed from the Context and the Input before they are sent to models to get a model-in-the-loop output.

(**Top**) The config implements a natural language inference task. The first image is the collection interface, after a crowdworker submits their example and gets a model-in-the-loop response. The second image is the validation interface. For brevity, the metadata field in the config is omitted. This field is used to define the UI components for additional information, such as the "Explain why your example is correct..." input field.

(**Bottom**) The config implements an image labelling task. The first image is the collection interface, before a crowdworker submits their example. The second image is the validation interface with the same example.

lect data, make the competition's design decisions, select metrics, and configure the task's leaderboard.

Kiela et al. (2021) introduced Dynabench with four English language NLP tasks: Natural Language Inference (Nie et al., 2020), Extractive QA (Bartolo et al., 2020), Sentiment Analysis (Potts et al., 2020) and Hate Speech Detection (Vidgen et al., 2021). In follow-up work, Ma et al. (2021) updated Dynabench with additional leaderboard functionalities that allow task owners to upload task-specific models which are evaluated on each of the task's datasets, and can subsequently be included in model-ensembles that crowdworkers interact with. As the platform kept expanding, it became clear that Dynabench needed a scalable and configurable system for adding new tasks.

A *task* is an essential concept in understanding our work. On Dynabench, a distinct task is a particular relationship between inputs and outputs.[1] Inputs and outputs are framed within some prespecified format. For example, Natural Language Inference is a task on Dynabench. The input format is two strings and the output format is a classification label. The relationship between the inputs and outputs is defined by what humans would do when loosely instructed to treat the input strings as a context (sometimes called the "premise") and a hypothesis, and return a label for whether they think the hypothesis is entailed by the context. MNLI (Williams et al., 2017), SNLI (Bowman et al., 2015), and ANLI (Nie et al., 2020) can be viewed as different datasets that instantiate the same task. Schlangen (2021) takes a similar view.

3 Dynatask

Before the introduction of Dynatask, adding a new task required close collaboration between task owners and the Dynabench team, and extensive software contributions to the Dynabench codebase. This paper presents a system that enables Dynabench to scale up to more tasks, including into multimodal and multilingual domains, without such requirements. Now, a task owner can create their own task page on Dynabench with a short task config file. The config file is used to automatically generate crowdworker data collection interfaces, as well as the model and dataset hosting/evaluating infrastructure. The data collection interfaces and hosting overlay existing services such as Amazon Mechanical Turk,[2] which provide a workforce and payment mechanisms, but do not provide crowdworker interfaces for dynamic model-in-the-loop data collection or their corresponding backends. In fact, local installations of Dynabench can be run on Mechanical Turk. Overall, a Dynabench task owner can set up and host:

Crowdworker data collection: Task owners can configure interfaces for data collection. Models-in-the-loop can be optionally added, so crowdworkers can receive real-time model responses from their data (Figure 1).

Crowdworker data validation: Task owners can configure interfaces for crowdworkers to label collected examples as correct or incorrect. (Figure 1).

Dynamic dataset metrics: Metrics on the crowdworker data are computed, such as verified model error rate (vMER) (Nie et al., 2020). Crowdworker example leaderboards are displayed.

A train file leaderboard: Task owners can enable users to upload training data files for the automatic creation, training, and evaluation of models in our evaluation cloud.

A dynamic and interactive model leaderboard (Ma et al., 2021): Task owners can configure a leaderboard, selecting from a variety of metrics to determine model performance. Owners can also upload new datasets, which triggers automatic evaluation for all of the user-uploaded models. Every leaderboard model can be interacted with in real-time. See Figure 2 for an example.

A model upload pipeline: Once a new task goes live on Dynabench, our command line tool[3] allows anyone to create a handler script and upload models by following a few command line instructions. After models are uploaded, they are dockerized and deployed automatically. Models can be viewed on the leaderboard and put in-the-loop with crowdworkers for data collection.

3.1 Task Configuration

To become task owners, Dynabench users submit a short written proposal for their task which requires approval by an administrator. We are still developing procedures for how Dynabench accepts tasks;

[1] Although, any user can set up a new task that is a duplicate of an existing one, with a duplicate config file.

[2] https://www.mturk.com

[3] https://github.com/facebookresearch/dynalab

```yaml
aggregation_metric:
  type: dynascore
perf_metric:
  type: squad_f1
  reference_name:
    ↪ answer
delta_metrics:
- type: fairness
- type: robustness
```

MODEL LEADERBOARD

Model	QA F1 %	Throughput examples/second	Memory GiB	Fairness %	Robustness %	Dynascore
ELECTRA-Large SynQA (maxbartolo)	82.66	2.27	14.25	92.30	92.62	49.74
DeBERTa default params (dynateam)	76.25	4.42	6.97	88.33	90.06	46.49
ELECTRA-large default params (dynateam)	76.07	2.37	25.30	93.13	91.64	46.30
RoBERTa default params (dynateam)	69.67	6.88	6.17	88.32	86.10	43.11
ALBERT default params (dynateam)	68.63	6.85	2.54	87.44	80.90	42.28
BERT default params (dynateam)	57.14	6.70	5.55	91.45	80.81	36.58
BiDAF default params (dynateam)	53.48	10.71	3.60	80.79	77.03	34.53
bertstyleqa (fzchriha)	52.75	11.19	3.77	92.17	77.97	34.52
Unrestricted T5 default params (dynateam)	28.80	4.51	10.69	92.32	88.41	22.72
Always Return the Context (dynateam)	5.99	89.80	1.10	95.97	91.61	17.07

Previous Next

Figure 2: An example of a task config next to the generated model leaderboard. Only config fields that impact the leaderboard are shown. Throughput and memory do not need to be in the config; they are computed by default.

so far, we have reached out to have a discussion with the proposer before accepting their proposal and all non-spam proposals have been slated for acceptance. After approval, the task owner submits a task config file, which can be written in minutes. Once complete, the task is actively hosted on Dynabench; data collection, data validation, model hosting, and model evaluation starts immediately. A complete config file is the combination of a snippet in Figure 1 with that in Figure 2.

The task config is a YAML file which allows someone to encode the specifications for their task—it can be viewed as a lightweight declarative programming language. Task owners can specify:

The datatypes of the task's inputs and outputs. There are a variety to choose from, including String, String Selection, Multiclass, Multilabel, Probabilities, and Image. The datatype definition enables Dynatask to automatically construct the UIs for data collection, the dataset uploading and downloading infrastructure, and the model uploading and hosting infrastructure.

A variety of metrics to understand the task's datasets and models. Several metrics can currently be computed for the leaderboard: Macro F1, F1 for Visual Question Answering, F1 for Question Answering, Accuracy, BLEU, robustness and fairness (Ma et al., 2021), memory usage, and example throughput. Task owners select or propose an aggregation metric, which combines results across multiple datasets and metrics to arrive at a ranking for the leaderboard. Currently, the only supported aggregation metric is the Dynascore (Ma et al., 2021), which combines metrics across datasets based on microeconomic utility (Ethayarajh and Jurafsky, 2020) of user provided weights. Metrics can also be specified for model-in-the-loop data collection to judge whether a model's output matches

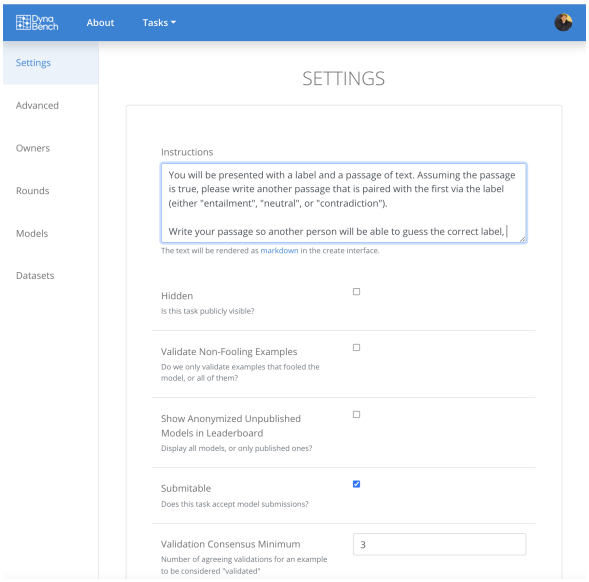

Figure 3: The task owner interface for ANLI.

that of a crowdworker (i.e., whether the model is "correct"). Dynatask supports a variety of such metrics, including a string F1 threshold (for outputs that are strings), exact match, and simply asking the crowdworker whether the model was correct.

Other optional items, such as messages and instructions that appear in crowdworker interfaces, and options for train-file leaderboards.

3.2 Options After Task Configuration

Crowdworker Interfaces, Data Generation, and Data Evaluation: Data collection interfaces are automatically hosted at `dynabench.org`. In order for Dynabench to scale, task owners source and pay crowdworkers themselves. If crowdworker management, compensation, and sourcing features are needed, an owner can clone Dynabench and run it on Mechanical Turk by hosting the data col-

Figure 4: A cross-section from a model card. The task owner has enabled downloading of model evaluation logs for each dataset via the buttons on the right.

lection frontend using Mephisto.[4] Task owners can upload context data for crowdworkers to use and download data collected from crowdworkers directly from the Dynabench web interface. Task owners can also initiate new rounds of data collection where they are free to upload entirely new contexts and models. As part of the data collection process, vMER, number of total collected examples, and number of validated examples are computed. Finally, task owners can alter instructions to crowdworkers at any time. They can also specify whether crowdworkers should validate non-model fooling examples, and provide a validation consensus threshold above which examples are considered fully validated. Figure 3 shows an example of the interface that task owners use to adjust settings.

Model Submission, Interaction, and Evaluation: Task owners can decide whether their task accepts model submissions, they can upload datasets for model evaluation, and they can download model evaluation logs for any dataset and model. Task owners can optionally allow users to download these logs to debug their models; see the example in Figure 4.

4 Decentralized Evaluation-As-A-Service

Most task owners currently use the centralized Dynabench evaluation and model deployment server. With Dynatask, however, we offer a decentralized evaluation feature that will increase the platform's flexibility even further. With this feature, task owners can set up a Dynabench model deployment and evaluation server or select an existing one. To set up a new server, an owner only needs to follow our documentation, creating an AWS account and installing some Dynabench code along the way. Distributed hosting of model building and evaluation enables Dynatask to scale: no one organization needs to fund hosting for all of the models on Dynabench, and every owner of a model deployment and evaluation server can flexibly upload or take down models to suit their budget. It is also designed with re-usability in mind: several tasks can share the same evaluation servers. Task owners do not need to do any setup if they have permission to use an existing evaluation server.

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

Statistic	Count
Datasets Hosted	191
Unique Crowdworkers	5,595
Model Uploads	589
Data Collection Rounds	38
Tasks (incl. private)	24
Examples Collected	559,229
Example Validations	436,922

Table 1: Current Dynabench statistics.

5 Case Studies of Tasks Enabled so Far

Tables 1 and 2 provide an overview of Dynabench so far. In this section, we report on some use cases. Most of the following projects (besides Image Labelling and Open Domain QA) were added to Dynabench before the introduction of Dynatask, which took months of coding in every case. With Dynatask, they can all be implemented in minutes.

Hate Speech Detection: There are a number of hate speech detection projects on Dynabench, where a model must label strings as hateful or not. Groups from Oxford, The Alan Turing Institute, The University of Sheffield, and Facebook AI own task pages that focus on collecting adversarial data (Vidgen et al., 2021), collecting emoji-based hate (Kirk et al., 2021), and evaluating models on a large number of hate speech data perturbations.

Visual Question Answering: To combat saturating datasets for the VQA task, which is about answering a question based on an image, Facebook AI and Tecnológico de Monterrey introduced AdVQA (Sheng et al., 2021) using Dynabench. The task's model leaderboard has an additional adversarial VQA dataset from Microsoft and Tsinghua (Li et al., 2021).

Extractive Question Answering: Groups from UCL and Facebook AI run SQuAD-style (Rajpurkar et al., 2016) extractive QA projects on Dynabench. The Adversarial QA (Bartolo et al., 2020) project resulted in a popular dataset on the Hugging Face hub (Lhoest et al., 2021). Follow-up

Selected Dynabench Tasks	Context and Input Types	Output Types
Hate Speech Detection https://dynabench.org/tasks/hs	String, String, Multiclass	Multiclass, Probs
Visual QA https://dynabench.org/tasks/vqa	Image, String	String
Extractive QA https://dynabench.org/tasks/qa	String, String, String Select	String Select, Probs
Open Domain QA https://dynabench.org/tasks/qb	String, String, String	String, Probs
Natural Language Inference https://dynabench.org/tasks/nli	String, String, Multiclass	Multiclass, Probs
Sentiment Analysis https://dynabench.org/tasks/sentiment	String, String, Multiclass	Multiclass, Probs
Machine Translation https://dynabench.org/tasks/flores	String, String, String, String	String
Image Labelling https://dynabench.org/tasks/vision-dataperf	Image, Multilabel	Multilabel

Table 2: IO types from the task config, for some tasks on Dynabench. Tasks share the same building blocks.

projects explored the generation of synthetic adversarial QA data (Bartolo et al., 2021a), generative assistants in the loop to help annotators create examples (Bartolo et al., 2021b), and a study of how adversarial model-in-the-loop training data affects generalization out of domain (Kaushik et al., 2021).

Open-Domain Question Answering: A team at Facebook AI and The University of Maryland has started a model-in-the-loop data collection effort for the Quizbowl task (Rodriguez et al., 2019; Wallace et al., 2019), as well as a model leaderboard. The task is open domain question answering, where both the question and answer are strings.

Natural Language Inference: The NLI dataset ANLI (Nie et al., 2020) is currently a popular dataset on Hugging Face datasets (Lhoest et al., 2021) and an ongoing Dynabench project. Groups from Facebook AI, UC Berkeley, and UNC have set up additional NLI projects on distinct Dynabench task pages. These projects have ranged from an analysis of the contents of adversarially collected development sets (Williams et al., 2022), to an explication of the benefits of dynamic adversarial data collection over multiple rounds (Wallace et al., 2021), to model and leaderboard hosting for a large number of robustness-perturbed NLI datasets.

Sentiment Analysis: In later rounds of their work, a team at Stanford used Dynabench to create a new adversarial sentiment analysis dataset, called Dynasent (Potts et al., 2020). They added prompts to their data collection interfaces to encourage crowdworkers to generate naturalistic and diverse data.

Large-Scale Machine Translation: The Workshop on Machine Translation (Wenzek et al., 2021) organizers created a Dynabench task page and hosted the FLORES benchmark competition (Goyal et al., 2021) of over 10,000 language pairs. It featured competitors from Microsoft, Huawei, Tencent, and Facebook, and individual competitors. The result of the competition was a BLEU increase of over 10 points on the full task. The owners used Dynabench for its leaderboard, model upload, and evaluation-as-a-service feature, without collecting data on the platform yet.

Image Labelling: DataPerf (Mattson et al., 2022) is a working group of the non-profit ML Commons, which focuses on dataset benchmarking for general AI. For their image labelling task hosted on Dynabench, they configured their task via the task config to accept training data file uploads. Users upload train files and models are automatically trained against them and evaluated in the evaluation cloud.

6 Conclusion

We introduced Dynatask, a collection of open source features in the Dynabench platform that empowers anyone to create and own a task on Dynabench with only a short config file. Dynabench started as an NLP project with only four English-only tasks. Since then, Dynatask has helped researchers produce several datasets and host competitions, expanding scalably into multimodal and multilingual domains with owners from various corners of the AI community. Dynatask offers the functionalities of Dynabench to the broader research community by allowing them to easily create and host new AI tasks on the platform: it provides a one-stop shop for constructing datasets with or without models in the loop, hosting challenges and competitions, investigating the effects of models in the loop, characterizing distributional shift and continual learning, exploring annotator efficiency and expertise, and improving model robustness through collaboration with humans.

Finally, Dynabench is an open source, community-driven effort. Anyone who wants to add a new input/output type, a new metric, or any other new feature, need only submit a pull request. We hope that our our work can help enable new exciting scientific progress in data-centric AI research in general and dynamic (adversarial) data collection in particular.

References

Max Bartolo, Alastair Roberts, Johannes Welbl, Sebastian Riedel, and Pontus Stenetorp. 2020. Beat the AI: Investigating adversarial human annotation for reading comprehension. *TACL*.

Max Bartolo, Tristan Thrush, Robin Jia, Sebastian Riedel, Pontus Stenetorp, and Douwe Kiela. 2021a. Improving question answering model robustness with synthetic adversarial data generation. In *EMNLP*.

Max Bartolo, Tristan Thrush, Sebastian Riedel, Pontus Stenetorp, Robin Jia, and Douwe Kiela. 2021b. Models in the loop: Aiding crowdworkers with generative annotation assistants. In *arXiv preprint arXiv:2112.09062*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*.

Kawin Ethayarajh and Dan Jurafsky. 2020. Utility is in the eye of the user: A critique of NLP leaderboards. In *EMNLP*.

John Godfrey, Edward Holliman, and Jane McDaniel. 1992. Switchboard: telephone speech corpus for research and development. In *ICASSP*.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzman, and Angela Fan. 2021. The flores-101 evaluation benchmark for low-resource and multilingual machine translation. In *arXiv preprint arXiv:2106.03193*.

Alon Halevy, Peter Norvig, and Fernando Pereira. 2009. The unreasonable effectiveness of data. *IEEE Intelligent Systems*.

Divyansh Kaushik, Douwe Kiela, Zachary C Lipton, and Wen tau Yih. 2021. On the efficacy of adversarial data collection for question answering: Results from a large-scale randomized study. In *ACL-IJCNLP*.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. 2021. Dynabench: Rethinking Benchmarking in NLP. In *NAACL*.

Hannah Rose Kirk, Bertram Vidgen, Paul Röttger, Tristan Thrush, and Scott A Hale. 2021. Hatemoji: A test suite and adversarially-generated dataset for benchmarking and detecting emoji-based hate. In *arXiv preprint arXiv:2108.05921*.

Yann LeCun, Corinna Cortes, and Christopher Burges. 1998. The mnist database of handwritten digits.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *EMNLP: System Demonstrations*.

Linjie Li, Jie Lei, Zhe Gan, and Jingjing Liu. 2021. Adversarial VQA: A new benchmark for evaluating the robustness of VQA models. In *ICCV*.

Sasha Luccioni, Yacine Jernite, and Meg Mitchell. 2021. Hugging face data measurements tool.

Zhiyi Ma, Kawin Ethayarajh, Tristan Thrush, Somya Jain, Ledell Wu, Robin Jia, Christopher Potts, Adina Williams, and Douwe Kiela. 2021. Dynaboard: An evaluation-as-a-service platform for holistic next-generation benchmarking. In *NeurIPS*.

Peter Mattson, Cody Coleman, Ce Zhang, Praveen Paritosh, Vijay Janapa Reddi, Douwe Kiela, Greg Diamos, Carole-Jean Wu, Sabri Eyuboglu, Joaquin Vanschoren, William Gaviria Rojas, Tariq Kane, Bojan Karlas, Lynn He, Lora Aroyo, Colby Banbury, Mark Mazumder, Bilge Acun, Newsha Ardalani, Tristan Thrush, Adina Williams, Amirata Ghorbani, Emmett Goodman, and Serena Yeung. 2022. Dataperf: Benchmarking data for better ml. In *Forthcoming*.

Andrew Ng, Lora Aroyo, Greg Diamos, Cody Coleman, Vijay Janapa Reddi, Joaquin Vanschoren, Carole-Jean Wu, Sharon Zhou, and Lynn He. 2021. [link].

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *ACL*.

Amandalynne Paullada, Inioluwa Deborah Raji, Emily M Bender, Emily Denton, and Alex Hanna. 2021. Data and its (dis) contents: A survey of dataset development and use in machine learning research. *Patterns*.

Christopher Potts, Zhengxuan Wu, Atticus Geiger, and Douwe Kiela. 2020. DynaSent: A dynamic benchmark for sentiment analysis. *arXiv preprint arXiv:2012.15349*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *EMNLP*.

Pedro Rodriguez, Shi Feng, Mohit Iyyer, He He, and Jordan Boyd-Graber. 2019. Quizbowl: The case for incremental question answering. In *arXiv preprint arXiv:1904.04792*.

Nithya Sambasivan, Shivani Kapania, Hannah Highfill, Diana Akrong, Praveen Paritosh, and Lora M. Aroyo. 2021. "everyone wants to do the model work, not the data work": Data cascades in high-stakes AI. In *CHI Conference on Human Factors in Computing Systems*.

David Schlangen. 2021. Targeting the benchmark: On methodology in current natural language processing research. In *ACL-IJCNLP: Short Papers*.

Sasha Sheng, Amanpreet Singh, Vedanuj Goswami, Jose Alberto Lopez Magana, Tristan Thrush, Wojciech Galuba, Devi Parikh, and Douwe Kiela. 2021. Human-adversarial visual question answering. In *NeurIPS*.

Bertie Vidgen, Tristan Thrush, Zeerak Waseem, and Douwe Kiela. 2021. Learning from the worst: Dynamically generated datasets to improve online hate detection. In *ACL*.

Kiri Wagstaff. 2012. Machine learning that matters. In *ICML*.

Eric Wallace, Pedro Rodriguez, Shi Feng, Ikuya Yamada, and Jordan Boyd-Graber. 2019. Trick me if you can: Human-in-the-loop generation of adversarial examples for question answering. In *TACL*.

Eric Wallace, Adina Williams, Robin Jia, and Douwe Kiela. 2021. Analyzing dynamic adversarial training data in the limit. In *arXiv preprint arXiv:2110.08514*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *NeurIPS*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *BlackboxNLP at EMNLP*.

Guillaume Wenzek, Vishrav Chaudhary, Angela Fan, Sahir Gomez, Naman Goyal, Somya Jain, Douwe Kiela, Tristan Thrush, and Francisco Guzmán. 2021. Findings of the wmt 2021 shared task on large-scale multilingual machine translation. In *WMT*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. In *arXiv preprint arXiv:1704.05426*.

Adina Williams, Tristan Thrush, and Douwe Kiela. 2022. ANLIzing the adversarial natural language inference dataset. In *SCiL*.