

Appendices for WILDS: A Benchmark of in-the-Wild Distribution Shifts

We encourage readers to view the up-to-date version of the paper hosted at <https://wilds.stanford.edu>, which combines the main text and appendices and is formatted to be more readable.

Contents

A	Dataset realism	1
B	Prior work on ML benchmarks for distribution shifts	2
C	Distribution shifts in other application areas	3
D	Potential extensions to other problem settings	6
E	Empirical trends	8
F	Using the WILDS package	9
G	Additional experimental details	9
H	Additional dataset details and results	10
H.1	iWILDCAM2020-WILDS	11
H.2	CAMELYON17-WILDS	14
H.3	RxRx1-WILDS	18
H.4	OGB-MOLPCBA	22
H.5	GLOBALWHEAT-WILDS	24
H.6	CIVILCOMMENTS-WILDS	27
H.7	FMoW-WILDS	32
H.8	POVERTYMAP-WILDS	36
H.9	AMAZON-WILDS	40
H.10	PY150-WILDS	43
I	Datasets with distribution shifts that do not cause performance drops	45
I.1	SQF: Criminal possession of weapons across race and locations	45
I.2	BDD100K: Object recognition in autonomous driving across locations	47
I.3	Amazon: Sentiment classification across different categories and time	48
I.4	Yelp: Sentiment classification across different users and time	51

A. Dataset realism

In this section, we discuss the framework we use to assess the realism of a benchmark dataset. Realism is subtle to pin down and highly contextual, and assessing realism often requires consulting with domain experts and practitioners. As a general framework, we can view a benchmark dataset as comprising the data, a task and associated evaluation metric, and a train/test split that potentially reflects a distribution shift. Each of these components can independently be more or less realistic:

1. The **data**—which includes not just the inputs x but also any associated metadata (e.g., the domain that each data point came from)—is realistic if it accurately reflects what would plausibly be collected and available for a model to use in a real application. The realism of data also depends on the application context; for example, using medical images captured with state-of-the-art equipment might be realistic for well-equipped hospitals, but not necessarily for clinics that use older generations of the technology, or vice versa. Extreme examples of unrealistic data include the Gaussian distributions that are often used to cleanly illustrate the theoretical properties of various algorithms.
2. The **task and evaluation metric** is realistic if the task is relevant to a real application and if the metric measures how successful a model would be in that application. Here and with the other components, realism lies on a spectrum. For example, in a wildlife conservation application where the inputs are images from camera traps, the real task might be to estimate species populations (Parham et al., 2017), i.e., the number of distinct individual animals of each species seen in the overall collection of images; a task that is less realistic but still relevant and useful for ecologists might be to classify what species of animal is seen in each image (Tabak et al., 2019). The choice of evaluation metric is also important. In the wildlife example, conservationists might care more about rare species than common species, so measuring average classification accuracy would be less realistic than a metric that prioritizes classifying the rare species correctly.
3. The **distribution shift (train/test split)** is realistic if it reflects training and test distributions that might arise in deployment for that dataset and task. For example, if a medical algorithm is trained on data from a few hospitals and then expected to be deployed more widely, then it would be realistic to test it on hospitals that are not in the training set. On the other hand, an example of a less realistic shift is to, for instance, train a pedestrian classifier entirely on daytime photos and then test it only on nighttime photos; in practice, any reasonable dataset

for pedestrian detection that is used in a real application would include both daytime and nighttime photos.

Through the lens of this framework, existing ML benchmarks tend to focus on object recognition tasks with realistic data (e.g., photos) but not necessarily with realistic distribution shifts. With WILDS, we seek to address this gap by selecting datasets that represent a wide variety of tasks (with realistic evaluation metrics and data) and that reflect realistic distribution shifts, i.e., train/test splits that are likely to arise in real-world deployments.

To elaborate on the realism of the distribution shift, we associate each dataset in WILDS with the distribution shift (i.e., problem setting) that we believe best reflects the real-world challenges in the corresponding application area. For example, domain generalization is a realistic setting for the CAMELYON17-WILDS dataset as medical models are typically trained on data collected from a handful of hospitals, but with the goal of general deployment across different hospitals. On the other hand, subpopulation shift is appropriate for the CIVILCOMMENTS-WILDS dataset, as the real-world challenge is that some demographic subpopulations (domains) are underrepresented, rather than completely unseen, in the training data. The appropriate problem setting depends on many dataset-specific factors, but some common considerations include:

- **Domain type.** Certain types of domains are generally more appropriate for a particular setting. For example, if the domains represent time, as in FMOW-WILDS, then domain generalization is suitable as a common challenge is to generalize from past data to future data. On the other hand, if the domains represent demographics and the goal is to improve performance on minority subpopulations, as in CIVILCOMMENTS-WILDS, then subpopulation shift is typically more appropriate.
- **Data collection challenges.** When collecting data from a new domain is expensive, domain generalization is often appropriate, as we might want to train on data from a limited number of domains but still generalize to unseen domains. For example, it is difficult to collect patient data from multiple hospitals, as in CAMELYON17-WILDS, or survey data from new countries, as in POVERTYMAP-WILDS.
- **Continuous addition of new domains.** A special case of the above is when new domains are continuously created. For example, in AMAZON-WILDS, where domains correspond to users, new users are constantly signing up for the platform; and in iWILDCAM2020-WILDS, where domains correspond to camera traps, new cameras are constantly being deployed. These are natural domain generalization settings.

B. Prior work on ML benchmarks for distribution shifts

In this section, we discuss existing ML distribution shift benchmarks in more detail, categorizing them by how they induce their respective distribution shifts. We focus here on work that has appeared in ML conferences and journals; we discuss related work from other research communities in Section C and Appendix H. We also restrict our attention to publicly-available datasets. While others have studied some proprietary datasets with realistic distribution shifts, such as the StreetView StoreFronts dataset (Hendrycks et al., 2020b) or diabetic retinopathy datasets (D’Amour et al., 2020a), these datasets are not publicly available due to privacy and other commercial reasons.

Distribution shifts from transformations. Some of the most widely-adopted benchmarks induce distribution shifts by synthetically transforming the data. Examples include rotated and translated versions of MNIST and CIFAR (Worrall et al., 2017; Gulrajani & Lopez-Paz, 2020); surface variations such as texture, color, and corruptions like blur in Colored MNIST (Gulrajani & Lopez-Paz, 2020), Stylized ImageNet (Geirhos et al., 2018a), ImageNet-C (Hendrycks & Dietterich, 2019), and similar ImageNet variants (Geirhos et al., 2018b); and datasets that crop out objects and replace their backgrounds, as in the Backgrounds Challenge (Xiao et al., 2020) and other similar datasets (Sagawa et al., 2020a; Koh et al., 2020). Benchmarks for adversarial robustness also fall in this category of distribution shifts from transformations (Goodfellow et al., 2015; Croce et al., 2020). Though adversarial robustness is not a focus of this work, we note that recent work on temporal perturbations with the ImageNet-Vid-Robust and YTBB-Robust datasets (Shankar et al., 2019) represents a different form of distribution shift that also impacts real-world applications. Outside of visual object recognition, other work has used synthetic datasets and transformations to explore compositional generalization, e.g., SCAN (Lake & Baroni, 2018). We discuss this more in Section C.

Synthetic-to-real transfers. Fully synthetic datasets such as SYNTHIA (Ros et al., 2016) and StreetHazards (Hendrycks et al., 2020a) have been adopted for out-of-distribution detection as well as domain adaptation and generalization, e.g., by testing robustness to transformations in the seasons, weather, time, or architectural style (Hoffman et al., 2018; Volpi et al., 2018). While the data is synthetic, it can still look realistic if a high-fidelity simulator is used. In particular, synthetic benchmarks that study transfers from synthetic to real data (Ganin & Lempitsky, 2015; Richter et al., 2016; Peng et al., 2018) can be important tools for tackling real-world problems: even though the data is synthesized and by definition, not real, the synthetic-to-real

distribution shift can still be realistic in contexts where real data is much harder to acquire than synthetic data (Belle-mare et al., 2020). In this work, we do not study these types of synthetic distribution shifts; instead, we focus on distribution shifts that occur in the wild between real data distributions.

Distribution shifts from constrained splits. Other benchmarks do not rely on transformations but instead split the data in a way that induces particular distribution shifts. These benchmarks have realistic data, e.g., the data points are derived from real-world photos, but they do not necessarily reflect distribution shifts that would arise in the wild. For example, BREEDS (Santurkar et al., 2020) and a related dataset (Hendrycks & Dietterich, 2019) test generalization to unseen ImageNet subclasses by holding out subclasses specified by several controllable parameters; similarly, NICO (He et al., 2020) considers subclasses that are defined by their context, such as dogs at home versus dogs on the beach; DeepFashion-Remixed (Hendrycks et al., 2020b) constrains the training set to include only photos from a single camera viewpoint and tests generalization to unseen camera viewpoints; BDD-Anomaly (Hendrycks et al., 2020a) uses a driving dataset but with all motorcycles, trains, and bicycles removed from the training set only; and ObjectNet (Barbu et al., 2019) comprises images taken from a few pre-specified viewpoints, allowing for systematic evaluation for robustness to camera angle changes but deviating from natural camera angles.

Distribution shifts across datasets. A well-studied special case of the above category is the class of distribution shifts obtained by combining several disparate datasets (Torralba & Efros, 2011), training on one or more of them and then testing on the remaining datasets. A recent influential example is the ImageNetV2 dataset (Recht et al., 2019), which was constructed to be similar to the original ImageNet dataset. Unlike ImageNetV2, however, many of these distribution shifts were constructed to be more drastic than might arise in the wild. For example, standard domain adaptation benchmarks include training on MNIST but testing on SVHN street signs (LeCun et al., 1998; Yuval et al., 2011; Tzeng et al., 2017; Hoffman et al., 2018), as well as transfers across datasets containing different renditions (e.g., photos, clipart, sketches) in DomainNet (Peng et al., 2019) and the Office-Home dataset (Venkateswara et al., 2017).

The main difference between domain adaptation and domain generalization is that in the latter, we do not assume access to unlabeled data from the test distribution. This makes it straightforward to use domain adaptation benchmarks for domain generalization, e.g., in DomainBed (Gulrajani & Lopez-Paz, 2020); we focus on domain generalization in this work, but further discuss unsupervised domain adapta-

tion in Section D. Other similar benchmarks that have been proposed for domain generalization include VLCS (Fang et al., 2013), which tests generalization across similar visual object recognition datasets; PACS (Li et al., 2017a), which (like DomainNet) tests generalization across datasets with different renditions; and ImageNet-R (Hendrycks et al., 2020b) and ImageNet-Sketch (Wang et al., 2019c), which also test generalization across different renditions by collecting separate datasets from Flickr and Google Image queries.

C. Distribution shifts in other application areas

Beyond the datasets currently included in WILDS, there are many other applications where it is critical for models to be robust to distribution shifts. In this section, we discuss some of these applications and the challenges of finding appropriate benchmark datasets in those areas. We also highlight examples of datasets with distribution shifts that we considered but did not include in WILDS, because their distribution shifts did not lead to a significant performance drop. Constructing realistic benchmarks that reflect distribution shifts in these application areas is an important avenue of future work, and we would highly welcome community contributions of benchmark datasets in these areas.

C.1. Algorithmic fairness

Distribution shifts which degrade model performance on minority subpopulations are frequently discussed in the algorithmic fairness literature. Geographic inequities are one concern (Shankar et al., 2017; Atwood et al., 2020): e.g., publicly available image datasets overrepresent images from the US and Europe, degrading performance in the developing world (Shankar et al., 2017) and prompting the creation of more geographically diverse datasets (Atwood et al., 2020). Racial disparities are another concern: e.g., commercial gender classifiers are more likely to misclassify the gender of darker-skinned women, likely in part because training datasets overrepresent lighter-skinned subjects (Buolamwini & Gebru, 2018), and pedestrian detection systems fare worse on darker-skinned pedestrians (Wilson et al., 2019). As in Appendix H.6, NLP models can also show racial bias.

Unfortunately, publicly available algorithmic fairness benchmarks (Mehrabi et al., 2019)—e.g., the COMPAS recidivism dataset (Larson et al., 2016)—suffer from several limitations. First, the datasets are often quite small by the standards of modern ML: the COMPAS dataset has only a few thousand rows (Larson et al., 2016). Second, they tend to have relatively few features, and disparities in subgroup performance are not always large (Larrazabal et al., 2020), limiting the benefit of more sophisticated approaches: on COMPAS, logistic regression performs comparably to a

black-box commercial algorithm (Jung et al., 2020; Dressel & Farid, 2018). Third, the datasets sometimes represent “toy” problems: e.g., the UCI Adult Income dataset (Asuncion & Newman, 2007) is widely used as a fairness benchmark, but its task—classifying whether a person will have an income above \$50,000—does not represent a real-world application. Finally, because many of the domains in which algorithmic fairness is of most concern—e.g., criminal justice and healthcare—are high-stakes and disparities are politically sensitive, it can be difficult to make datasets publicly available.

Creating algorithmic fairness benchmarks which do not suffer from these limitations represents a promising direction for future work. In particular, such datasets would ideally have: 1) information about a sensitive attribute like race or gender; 2) a prediction task which is of immediate real-world interest; 3) enough samples, a rich enough feature set, and large enough disparities in group performance that more sophisticated machine learning approaches would plausibly produce improvement over naive approaches.

Dataset: New York stop-and-frisk. Predictive policing is a prominent example of a real-world application where fairness considerations are paramount: algorithms are increasingly being used in contexts such as predicting crime hotspots (Lum & Isaac, 2016) or a defendant’s risk of reoffending (Larson et al., 2016; Corbett-Davies et al., 2016; 2017; Lum & Shah, 2019). There are numerous concerns about these applications (Larson et al., 2016; Corbett-Davies et al., 2016; 2017; Lum & Shah, 2019), one of which is that these ML models might not generalize beyond the distributions that they were trained on (Corbett-Davies & Goel, 2018; Slack et al., 2019). These distribution shifts include shifts over locations—e.g., a criminal risk assessment trained on several hundred defendants in Ohio was eventually used throughout the United States (Latessa et al., 2010)—and shifts over time, as sentencing and other criminal justice policies evolve (Corbett-Davies & Goel, 2018). There are, of course, also subpopulation shift concerns around whether models are biased against particular demographic groups.

We investigated these shifts using a dataset of pedestrian stops made by the New York City Police Department under its “stop-and-frisk” policy, where the task is to predict whether a pedestrian who was stopped on suspicion of weapon possession would in fact possess a weapon (Goel et al., 2016). This policy had a pronounced racial bias: Black people stopped by the police on suspicion of possessing a weapon were 5× less likely to actually possess one than their White counterparts (Goel et al., 2016). We emphasize that we oppose stop-and-frisk (and any “improved” ML-powered stop-and-frisk) since there is overwhelming evidence that the policy was racially discriminatory (Gel-

man et al., 2007; Goel et al., 2016; Pierson et al., 2018) and such massive inequities require more than algorithmic fixes. Rather, we use the dataset as a realistic example of the phenomena that arise in real policing contexts, including 1) substantial heterogeneity across locations and racial groups and 2) distributions that arise in part because of biased policing practices.

Overall, we find large performance disparities across race groups and locations. Interestingly, however, we also find that these disparities cannot be attributed to the distribution shift, as the disparities were not reduced when we trained models specifically on the race groups or locations that suffer the worst performance. Indeed, the groups that see the worst performance—Black and Hispanic pedestrians—comprise large *majorities* of the dataset, making up more than 90% of the stops. This contrasts with the typical setting in algorithmic fairness where models perform worse on *minority* groups in the training data. Our results suggest the disparities are due to the dataset being noisier for some race and location groups, potentially as a result of the biased policing practices underlying the dataset. We provide further details in Appendix I.1.

C.2. Medicine and healthcare

Substantial evidence indicates the potential for distribution shifts in medical settings. One concern is *demographic* subpopulation shifts (e.g., across race, gender, or socioeconomic status), since historically-disadvantaged populations are underrepresented in many medical datasets (Chen et al., 2020). Another concern is heterogeneity *across hospitals*; this might include differences in imaging, as in Appendix H.2, and other operational protocols such as lab tests (D’Amour et al., 2020a; Subbaswamy et al., 2020). Finally, changes *over time* can also produce distribution shifts: for example, Nestor et al. (2019) showed that switching between two electronic health record (EHR) systems produced a drop in performance, and the COVID-19 epidemic has affected the distribution of chest radiographs (Wong et al., 2020).

Creating medical distribution shift benchmarks thus represents a promising direction for future work, if several challenges can be overcome. First, while there are large demographic disparities in healthcare outcomes (e.g., by race or socioeconomic status), many of them are not due to distribution shifts, but to disparities in non-algorithmic factors (e.g., access to care or prevalence of comorbidities (Chen et al., 2020)) or to algorithmic problems unrelated to distribution shift (e.g., choice of a biased outcome variable (Obermeyer et al., 2019)). Indeed, several previous investigations have found relatively small disparities in algorithmic performance (as opposed to healthcare outcomes) across demographic groups (Chen et al., 2019a; Larrazabal

et al., 2020); Seyyed-Kalantari et al. (2020) finds larger disparities in true positive rates across demographic groups, but this might reflect the different underlying label distributions between groups.

Second, many distribution shifts in medicine arise from concept drifts, in which the relationship between the input and the label changes, for example due to changes in clinical procedures and the definition of the label (Widmer & Kubat, 1996; Beyene et al., 2015; Futoma et al., 2020). It can be difficult to ensure that a potential benchmark has sufficient leverage for models to learn how to handle, e.g., an abrupt change in the way a particular clinical procedure is carried out.

A last challenge is data availability, as stringent medical privacy laws often preclude data sharing (Price & Cohen, 2019). For example, EHR datasets are fundamental to medical decision-making, but there are few widely adopted EHR benchmarks—with the MIMIC database being a prominent exception (Johnson et al., 2016)—and relatively little progress in predictive performance has been made on them (Bellamy et al., 2020).

C.3. Genomics

Advances in high-throughput genomic and molecular profiling platforms have enabled systematic mapping of biochemical activity of genomes across diverse cellular contexts, populations, and species (Consortium et al., 2012; Ho et al., 2014; Kundaje et al., 2015; Aviv et al., 2017; Consortium et al., 2019; Moore et al., 2020; Consortium et al., 2020). These datasets have powered ML models that aim to learn predictive representations of functional DNA and predict genome-wide biochemical profiles in contexts for which experimental data is unavailable (Ching et al., 2018; Eraslan et al., 2019; Libbrecht & Noble, 2015). These models have been fairly successful at deciphering functional DNA sequence patterns from learned representations and predicting the consequences of genetic perturbations in contexts in which the models are trained (Avsec et al., 2019; Zhou & Troyanskaya, 2015; Kelley et al., 2016; Jaganathan et al., 2019). However, distribution shifts pose a significant challenge to generalizing predictions to new contexts.

One such challenge involves the prediction of genome-wide profiles of regulatory protein-DNA interactions across cellular contexts (Srivastava & Mahony, 2020). Regulatory proteins bind regulatory DNA elements in a sequence-specific manner to orchestrate gene expression programs. However, these proteins often form different cooperative complexes with each other in different cellular contexts. These context-specific complexes can recognize distinct combinatorial regulatory sequence syntax, thereby exhibiting dynamic genomic binding landscapes across diverse cell states and cell types that all contain the same genomic sequence. This phe-

nomenon essentially induces a concept shift across cellular contexts, as the same DNA sequence can be associated with different binding labels for a protein across contexts. Hence, ML models that aim to predict protein-DNA binding landscapes across cell types typically integrate DNA sequence and additional context-specific input data modalities that provide auxiliary information about the regulatory state of DNA in each cellular context (Srivastava & Mahony, 2020).

Recently, an open community challenge was introduced to systematically benchmark state-of-the-art predictive models that integrate DNA sequence and cell context-specific experiments of genome-wide regulatory state for cross cell-type prediction of protein-DNA binding maps.¹ Prospective evaluation of top-performing models highlighted a significant drop in prediction performance across cellular contexts relative to cross-validation performance within training cell types (Keilwagen et al., 2019; Quang & Xie, 2019; Li et al., 2019a; Li & Guan, 2019). We expect that novel approaches that explicitly correct for the concept shifts across cellular contexts are likely to provide significant improvements in cross-context generalization.

Such approaches could also benefit other prediction tasks in genomics that suffer from distribution shifts, such as generalizing animal models of human disease; adapting models of immortalized cell-lines to primary human cells; predicting the influence of different cellular micro-environments; and predicting cell fate in longitudinal time courses of cellular differentiation, reprogramming and exposure to stimuli.

C.4. Natural language and speech processing

Subpopulation shifts are an issue in automated speech recognition (ASR) systems, which have been shown to have higher error rates for Black speakers than for White speakers (Koenecke et al., 2020) and for speakers of some dialects (Tatman, 2017). These disparities were demonstrated using commercial ASR systems, and therefore do not have any accompanying training datasets that are publicly available. There are many public speech datasets with speaker metadata that could potentially be used to construct a benchmark, e.g., LibriSpeech (Panayotov et al., 2015), the Speech Accent Archive (Weinberger, 2015), VoxCeleb2 (Chung et al., 2018), the Spoken Wikipedia Corpus (Baumann et al., 2019), and Common Voice (Ardila et al., 2020). However, these datasets have their own challenges: some do not have a sufficiently diverse sample of speaker backgrounds and accents, and others focus on read speech (e.g., audiobooks) instead of more natural speech.

In natural language processing (NLP), a current focus is on challenge datasets that are crafted to test particular aspects of

¹ENCODE-DREAM in vivo Transcription Factor Binding Site Prediction Challenge, <http://synapse.org/encode>.

models, e.g., HANS (McCoy et al., 2019b), PAWS (Zhang et al., 2019), and CheckList (Ribeiro et al., 2020). These challenge datasets are drawn from test distributions that are often (deliberately) quite different from the data distributions that models are typically trained on. Counterfactually-augmented datasets (Kaushik et al., 2019) are a related type of challenge dataset where the training data is modified to make spurious correlates independent of the target, which can result in more robust models. Others have studied train/test sets that are drawn from different sources, e.g., Wikipedia, Reddit, news articles, travel reviews, and so on (Oren et al., 2019; Miller et al., 2020; Kamath et al., 2020).

Several synthetic datasets have also been designed to test compositional generalization, such as CLEVR (Johnson et al., 2017), SCAN (Lake & Baroni, 2018), and COGS (Kim & Linzen, 2020). The test sets in these datasets are chosen such that models need to generalize to novel combinations of parts of training examples, e.g., familiar primitives and grammatical roles (Kim & Linzen, 2020). CLEVR is a visual question-answering (VQA) dataset; other examples of VQA datasets that are formulated as challenge datasets are the VQA-CP v1 and v2 datasets (Agrawal et al., 2018), which create subpopulation shifts by intentionally altering the distribution of answers per question type between the train and test splits.

These NLP examples involve English-language models; other languages typically have fewer and smaller datasets available for training and benchmarking models. Multilingual models and benchmarks (Conneau et al., 2018; Conneau & Lample, 2019; Hu et al., 2020a; Clark et al., 2020) are another source of subpopulation shifts with corresponding disparities in performance: training sets might contain fewer examples in low-resource languages (Nekoto et al., 2020), but we would still hope for high model performance on these minority groups.

Datasets: Other distribution shifts in Amazon and Yelp reviews. In addition to user shifts on the Amazon Reviews dataset (Ni et al., 2019), we also looked at category and time shifts on the same dataset, as well as user and time shifts on the Yelp Open Dataset². However, for many of those shifts, we only found modest performance drops. We provide additional details on Amazon in Appendix I.3 and on Yelp in Appendix I.4.

C.5. Education

ML models can help in educational settings in a variety of ways: e.g., assisting in grading (Piech et al., 2013; Shermis, 2014; Kulkarni et al., 2014; Taghipour & Ng, 2016), estimating student knowledge (Desmarais & Baker, 2012; Wu et al., 2020), identifying students who need help (Ahadi et al.,

²<https://www.yelp.com/dataset>

2015), or automatically generating explanations (Williams et al., 2016; Wu et al., 2019a). However, there are substantial distribution shifts in these settings as well. For example, automatic essay scoring has been found to be affected by rater bias (Amorim et al., 2018) and spurious correlations like essay length (Perelman, 2014), leading to problems with subpopulation shift. Ideally, these systems would also generalize across different contexts, e.g., a model for scoring grammar should work well across multiple different essay prompts. Recent attempts at predicting grades algorithmically (BBC, 2020; Broussard, 2020) have also been found to be biased against certain subpopulations.

Unfortunately, there is a general lack of standardized education datasets, in part due to student privacy concerns and the proprietary nature of large-scale standardized tests. Datasets from massive open online courses are a potential source of large-scale data (Kulkarni et al., 2015). In general, dataset construction for ML in education is an active area—e.g., the NeurIPS 2020 workshop on Machine Learning for Education³ has a segment devoted to finding “ImageNets for education”—and we hope to be able to include one in the future.

C.6. Robotics

Robot learning has emerged as a strong paradigm for automatically acquiring complex and skilled behaviors such as locomotion (Yang et al., 2019; Peng et al., 2020), navigation (Mirowski et al., 2017; Kahn et al., 2020), and manipulation (Gu et al., 2017; et al., 2019). However, the advent of learning-based techniques for robotics has not convincingly addressed, and has perhaps even exasperated, problems stemming from distribution shift. These problems have manifested in many ways, including shifts induced by weather and lighting changes (Wulfmeier et al., 2018), location changes (Gupta et al., 2018), and the simulation-to-real-world gap (Sadeghi & Levine, 2017; Tobin et al., 2017). Dealing with these challenging scenarios is critical to deploying robots in the real world, especially in high-stakes decision-making scenarios.

For example, to safely deploy autonomous driving vehicles, it is critical that these systems work reliably and robustly across the huge variety of conditions that exist in the real world, such as locations, lighting and weather conditions, and sensor intrinsics. This is a challenging requirement, as many of these conditions may be underrepresented, or not represented at all, by the available training data. Indeed, prior work has shown that naively trained models can suffer at segmenting nighttime driving scenes (Dai & Van Gool, 2018), detecting relevant objects in new or challenging locations and settings (Yu et al., 2020; Sun et al., 2020a), and, as discussed earlier, detecting pedestrians with darker skin

tones (Wilson et al., 2019).

Creating a benchmark for distribution shifts in robotics applications, such as autonomous driving, represents a promising direction for future work. Here, we briefly summarize our initial findings on distribution shifts in the BDD100K driving dataset (Yu et al., 2020), which is publicly available and widely used, including in some of the works listed above.

Dataset: BDD100K. We investigated the task of multi-label binary classification of the presence of each object category in each image. In general, we found no substantial performance drops across a wide range of different test scenarios, including user shifts, weather and time shifts, and location shifts. We provide additional details in Section I.2.

Our findings contrast with previous findings that other tasks, such as object detection and segmentation, can suffer under the same types of shifts on the same dataset (Yu et al., 2020; Dai & Van Gool, 2018). Currently, WILDS consists of datasets involving classification and regression tasks. However, most tasks of interest in autonomous driving, and robotics in general, are difficult to formulate as classification or regression. For example, autonomous driving applications may require models for object detection or lane and scene segmentation. These tasks are often more challenging than classification tasks, and we speculate that they may suffer more severely from distribution shift.

C.7. Feedback loops

Finally, we have restricted our attention to settings where the data distribution is independent of the model. When the data distribution does depend on the model, distribution shifts can arise from feedback loops between the data and the model. Examples include recommendation systems and other consumer products (Bottou et al., 2013; Hashimoto et al., 2018); dialogue agents (Li et al., 2017b); molecular compound optimization (Cuccarese et al., 2020; Reker, 2020); decision systems (Liu et al., 2018; D’Amour et al., 2020b); and adversarial settings like fraud or malware detection (Rigaki & Garcia, 2018). While these adaptive settings are outside the scope of our benchmark, dealing with these types of distribution shifts is an important area of ongoing work.

D. Potential extensions to other problem settings

In this paper, we have focused on two problem settings involving domain shifts: domain generalization and subpopulation shifts. Here, we discuss other problem settings within the framework of domain shifts that could also apply to WILDS datasets. Using WILDS to benchmark and

³<https://www.ml4ed.org/>

develop algorithms for these settings is an important avenue for future work, and we welcome community contributions towards this effort.

D.1. Problem settings in domain shifts

Within the general framework of domain shifts, specific problem settings can differ along the following axes of variation:

1. **Seen versus unseen test domains.** Test domains may be seen during training time ($\mathcal{D}^{\text{test}} \subseteq \mathcal{D}^{\text{train}}$), as in subpopulation shift, or unseen ($\mathcal{D}^{\text{train}} \cap \mathcal{D}^{\text{test}} = \emptyset$), as in domain generalization. The domain generalization and subpopulation shift settings mainly differ on this factor.
2. **Train-time domain annotations.** The domain identity d may be observed for none, some, or all of the training examples. Train-time domain annotations are straightforward to obtain in some settings, e.g., we should know which patients in the training sets came from which hospitals, but can be harder to obtain in some settings, e.g., we might only have demographic information on a subset of training users. In our domain generalization and subpopulation shift settings, d is always observed at training time.
3. **Test-time domain annotations.** The domain identity d may be observed for none, some, or all of the test examples. Test-time domain annotations allow models to be domain-specific, e.g., by treating domain identity as a feature if the train and test domains overlap. For example, if the domains correspond to continents and the data to satellite images from a continent, we would presumably know what continent each image was taken from. On the other hand, if the domains correspond to demographic information, this might be hard to obtain at test time (as well as training time, as mentioned above). In domain generalization, d may be observed at test time, but it is not helpful by itself as all of the test domains are unseen at training time. However, when combined with test-time unlabeled data, observing the domain d at test time could help with adaptation. In subpopulation shift, we typically assume that d is unobserved at test time, though this need not always be true.
4. **Test-time unlabeled data.** Varying amounts of unlabeled test data—samples of x drawn from the test distribution P^{test} —may be available, from none to a small batch to a large pool. This affects the degree to which models can adapt to test distributions. For example, if the domains correspond to locations and the data points to photos taken at those locations, we might assume access to some unlabeled photos taken at the test locations.

Each combination of the above four factors corresponds to

a specific problem setting with a different set of applicable methods. In the current version of the WILDS benchmark, we focus on domain generalization and subpopulation shifts, which represent specific configurations of these factors. We briefly discuss a few other problem settings in the remainder of this section.

D.2. Unsupervised domain adaptation

In the presence of distribution shift, a potential source of leverage is observing unlabeled test points from the test distribution. In the unsupervised domain adaptation setting, we assume that at training time, we have access to a large amount of unlabeled data from each test distribution of interest, as well as the resources to train a separate model for each test distribution. For example, in a satellite imagery setting like FMOW-WILDS, it might be appropriate to assume that we have access to a large set of unlabeled recent satellite images from each continent and the wherewithal to train a separate model for each continent.

Many of the methods for domain generalization discussed in Section 6 were originally methods for domain adaptation, since methods for both settings share the common goal of learning models that can transfer between domains. For example, methods that learn features that have similar distributions across domains are equally applicable to both settings (Ben-David et al., 2006; Long et al., 2015; Sun et al., 2016; Ganin et al., 2016; Tzeng et al., 2017; Shen et al., 2018; Wu et al., 2019b). In fact, the CORAL algorithm that we use as a baseline in this work was originally developed for, and successfully applied in, unsupervised domain adaptation (Sun & Saenko, 2016). Other methods rely on knowing the test distribution and are thus specific to domain adaptation, e.g., learning to map data points from source to target domains (Hoffman et al., 2018), or estimating the test label distribution from unlabeled test data (Saerens et al., 2002; Zhang et al., 2013; Lipton et al., 2018; Azizzadenesheli et al., 2019; Alexandari et al., 2020; Garg et al., 2020).

D.3. Test-time adaptation

A closely related setting to unsupervised domain adaptation is test-time adaptation, which also assumes the availability of unlabeled test data. For datasets where there are many potential test domains (e.g., in iWILDCAM2020-WILDS, we want a model that can ideally generalize to any camera trap), it might be infeasible to train a separate model for each test domain, as unsupervised domain adaptation would require. In the test-time adaptation setting, we assume that a model is allowed to adapt to a small amount of unlabeled test data in a way that is computationally much less intensive than typical domain adaptation methods. This is a difference of degree and not of kind, but it can have significant practical

implications. For example, domain adaptation approaches typically require access to the training set and a large unlabeled test set at the same time, whereas test-time adaptation methods typically only require the learned model (which can be much smaller than the original training set) as well as a smaller amount of unlabeled test data.

A number of test-time adaptation methods have been recently proposed (Li et al., 2017c; Sun et al., 2020b; Wang et al., 2020a). For example, adaptive risk minimization (ARM) is a meta-learning approach that adapts models to each batch of test examples under the assumption that all data points in a batch come from the same domain (Zhang et al., 2020). Many datasets in WILDS are suitable for the test-time adaptation setting. For example, in iWILDCAM2020-WILDS, images from the same domain are highly similar, sharing the same location, background, and camera angle, and prior work has shown inferring these shared features can improve performance considerably (Beery et al., 2020b).

D.4. Selective prediction

A different problem setting that is orthogonal to the settings described above is selective prediction. In the selective prediction setting, models are allowed to abstain on points where their confidence is below a certain threshold. This is appropriate when, for example, abstentions can be handled by backing off to human experts, such as pathologists for CAMELYON17-WILDS, content moderators for CIVILCOMMENTS-WILDS, wildlife experts for iWILDCAM2020-WILDS, etc. Many methods for selective prediction have been developed, from simply using softmax probabilities as a proxy for confidence (Cordella et al., 1995; Geifman & El-Yaniv, 2017), to methods involving ensembles of models (Gal & Ghahramani, 2016; Lakshminarayanan et al., 2017; Geifman et al., 2018) or jointly learning to abstain and classify (Bartlett & Wegkamp, 2008; Geifman & El-Yaniv, 2019; Feng et al., 2019).

Intuitively, even if a model is not robust to a distribution shift, it might at least be able to maintain high accuracies on some subset of points that are close to the training distribution, while abstaining on the other points. Indeed, prior work has shown that selective prediction can improve model accuracy under distribution shifts (Pimentel et al., 2014; Hendrycks & Gimpel, 2017; Liang et al., 2018; Ovadia et al., 2019; Feng et al., 2019; Kamath et al., 2020). However, distribution shifts still pose a problem for selective prediction methods; for instance, it is difficult to maintain desired abstention rates under distribution shifts (Kompa et al., 2020), and confidence estimates have been found to drift over time (e.g., Davis et al. (2017)).

E. Empirical trends

We end our discussion of experimental results by briefly reporting on several trends that we observed across multiple datasets.

E.1. Underspecification

Prior work has shown that there is often insufficient information at training time to distinguish models that would generalize well under distribution shift; many models that perform similarly in-distribution (ID) can vary substantially out-of-distribution (OOD) (McCoy et al., 2019a; Zhou et al., 2020; D’Amour et al., 2020a). In WILDS, we attempt to alleviate this issue by providing multiple training domains in each dataset as well as an OOD validation set for model selection. Perhaps as a result, we do not observe significantly higher variance in OOD performance than ID performance in Table 1, with the exception of AMAZON-WILDS and CIVILCOMMENTS-WILDS, where the OOD performance is measured on a smaller subpopulation and is therefore naturally more variable. Excluding those datasets, the average standard deviation from Table 1 is 2.6% for OOD performance and 2.0% for ID performance, which is comparable. These results raise the question of when underspecification, as reported in prior work, could be more of an issue.

E.2. Model selection with in-distribution versus out-of-distribution validation sets

All of the baseline results reported in this paper use an OOD validation set for model selection, as discussed in Appendix G.2. To facilitate research into comparisons of ID versus OOD performance, most WILDS datasets also provide an ID validation and/or test set. For example, in iWILDCAM2020-WILDS, the ID validation set comprises photos from the same set of camera traps used for the training set. These ID sets are not used for model selection nor official evaluation.

Gulrajani & Lopez-Paz (2020) showed that on the DomainBed domain generalization datasets, selecting models with an ID validation set leads to higher OOD performance than using an OOD validation set. This contrasts with our approach of using OOD validation sets, which we find to generally provide a good estimate of OOD test performance. Specifically, in Appendix G.2, we show that for our baseline models, model selection using an OOD validation set results in comparable or higher OOD performance than model selection using an ID validation set. This difference could stem from many factors: for example, WILDS datasets tend to have many more domains, whereas DomainBed datasets tend to have fewer domains that can be quite different from each other (e.g., cartoons vs. photos); and there are some differences in the exact procedures for comparing perfor-

mance using ID versus OOD validation sets. Further study of the effects of these different model selection procedures and choices of validation sets would be a useful direction for future work.

E.3. The compounding effects of multiple distribution shifts

Several WILDS datasets consider hybrid settings, where the goal is to simultaneously generalize to unseen domains as well as to certain subpopulations. We observe that combining these types of shifts can exacerbate performance drops. For example, in POVERTYMAP-WILDS and FMoW-WILDS, the shift to unseen domains exacerbates the gap in subpopulation performance (and vice versa). Notably, in FMoW-WILDS, the difference in subpopulation performance (across regions) is not even manifested until also considering another shift (across time). While we do not always observe the compounding effect of distribution shifts—e.g., in AMAZON-WILDS, subpopulation performance is similar whether we consider shifts to unseen users or not—these observations underscore the importance of evaluating models on the combination of distribution shifts that would occur in practice, instead of considering each shift in isolation.

F. Using the WILDS package

Finally, we discuss our open-source PyTorch-based package that exposes a simple interface to our datasets and automatically handles data downloads, allowing users to get started on a WILDS dataset in just a few lines of code. In addition, the package provides various data loaders and utilities surrounding domain annotations and other metadata, which supports training algorithms that need access to these metadata. The package also provides standardized evaluations for each dataset. More documentation and installation information can be found at <https://wilds.stanford.edu>.

Datasets and data loading. The WILDS package provides a simple, standardized interface for all datasets in the benchmark as well as their data loaders, as summarized in Figure 3. This short code snippet covers all of the steps of getting started with a WILDS dataset, including dataset download and initialization, accessing various splits, and initializing the data loader. We also provide multiple data loaders in order to accommodate a wide array of algorithms, which often require specific data loading schemes.

Domain information. To allow algorithms to leverage domain annotations as well as other groupings over the available metadata, the WILDS package provides Grouper objects. Grouper objects (e.g., `grouper` in Figure 4) extract group annotations from metadata, allowing users to specify the grouping scheme in a flexible fashion.

```
>>> from wilds import get_dataset
>>> from wilds.common.data_loaders import get_train_loader
>>> import torchvision.transforms as transforms
# Load the full dataset
>>> dataset = get_dataset(dataset="iwildcam", download=True)
# Get the training set
>>> train_data = dataset.get_subset("train",
...                                 transform=transforms.ToTensor())
# Prepare the "standard" data loader
>>> train_loader = get_train_loader("standard", train_data,
...                                 batch_size=16)
# Train loop
>>> for x, y_true, metadata in train_loader:
...     ...
```

Figure 3: Dataset initialization and data loading.

```
>>> from wilds.common.grouper import CombinatorialGrouper
# Initialize grouper, which extracts domain (location) information
>>> grouper = CombinatorialGrouper(dataset, ["location"])
# Train loop
>>> for x, y_true, metadata in train_loader:
...     z = grouper.metadata_to_group(metadata)
...     ...
```

Figure 4: Accessing domain and other group information via a Grouper object.

Evaluation. Finally, the WILDS package standardizes and automates the evaluation for each dataset. As summarized in Figure 5, invoking the `eval` method of each dataset yields all metrics reported in the paper and on the leaderboard.

```
>>> from wilds.common.data_loaders import get_eval_loader
# Get the test set
>>> test_data = dataset.get_subset("test",
...                                 transform=transforms.ToTensor())
# Prepare the data loader
>>> test_loader = get_eval_loader("standard", test_data,
...                                 batch_size=16)
... # Get predictions for the full test set
>>> for x, y_true, metadata in test_loader:
...     y_pred = model(x)
...     [ accumulate(y_true, y_pred, metadata)
# Evaluate
>>> dataset.eval(all_y_pred, all_y_true, all_metadata)
{"macro_recall": 0.66, ...}
```

Figure 5: Evaluation.

G. Additional experimental details

G.1. Model architectures

We used standard model architectures for each dataset: ResNet and DenseNet for images (He et al., 2016; Huang et al., 2017), DistilBERT for text (Sanh et al., 2019), a Graph Isomorphism Network (GIN) for graphs (Xu et al., 2018), and Faster-RCNN (Ren et al., 2015) for detection.

G.2. Model hyperparameters

As our goal is high OOD performance, we use a separate OOD validation set for early stopping and hyperparameter selection. Relative to the training set, this OOD validation set reflects a distribution shift similar to, but distinct from, the test set. For example, in iWILDCAM2020-WILDS, the training, validation, and test sets each comprise photos from

distinct sets of camera traps.

For each hyperparameter setting, we used early stopping to pick the epoch with the best OOD validation performance (as measured by the specified metrics for each dataset described in Section 4), and then picked the model hyperparameters with the best early-stopped validation performance. We found that this gave similar or slightly better OOD test performance than selecting hyperparameters using the ID validation set (Table 3). For the ID comparisons in Section 5, we use the same hyperparameters optimized on the OOD validation set, which generally means that the reported ID results are slightly lower than they would have been if they were optimized; in other words, the ID-OOD gaps in Table 1 are slightly underestimated (Appendix G).

Using the OOD validation set for early stopping means that even if the training procedure does not explicitly use additional metadata, as in ERM, the metadata might still be implicitly (but mildly) used for model selection in one of two related ways. First, the metric might use the metadata directly (e.g., by computing the accuracy over different subpopulations defined in the metadata). Second, the OOD validation set is generally selected according to this metadata (e.g., comprising data from a disjoint set of domains as the training set). We expect that implicitly using the metadata in these ways should increase the OOD performance of each model. Nevertheless, as Sections 5 and 6 show, there are still large gaps between OOD and ID performance.

In general, we selected model hyperparameters with ERM and used the same hyperparameters for the other algorithm baselines (e.g., CORAL, IRM, or Group DRO). For CORAL and IRM, we did a subsequent grid search over the weight of the penalty term, using the defaults from Gulrajani & Lopez-Paz (2020). Specifically, we tried penalty weights of $\{0.1, 1, 10\}$ for CORAL and penalty weights of $\{1, 10, 100, 1000\}$ for IRM. We fixed the step size hyperparameter for Group DRO to its default value of 0.01 (Sagawa et al., 2020a).

G.3. Replicates

We typically use a fixed train/validation/test split and report results averaged across 3 replicates (random seeds for model initialization and minibatch order), as well as the unbiased standard deviation over those replicates. There are three exceptions to this. For POVERTYMAP-WILDS, we report results averaged over 5-fold cross validation, as model training is relatively fast on this dataset. For CAMELYON17-WILDS, results vary substantially between replicates, so we report results averaged over 10 replicates instead. Similarly, for CIVILCOMMENTS-WILDS, we report results averaged over 5 replicates.

G.4. Baseline algorithms

For all classification datasets, we train models against the cross-entropy loss. For the POVERTYMAP-WILDS regression dataset, we use the mean-squared-error loss.

We adapted the implementations of CORAL from Gulrajani & Lopez-Paz (2020); IRM from Arjovsky et al. (2019); and Group DRO from Sagawa et al. (2020a). We note that CORAL was originally proposed in the context of domain adaptation (Sun & Saenko, 2016), where it was shown to substantially improve performance on standard domain adaptation benchmarks, and it was subsequently adapted for domain generalization (Gulrajani & Lopez-Paz, 2020).

Following these implementations, we use minibatch stochastic optimizers to train models under each algorithm, and we sample uniformly from each domain regardless of the number of training examples in it. This means that the CORAL and IRM algorithms optimize for their respective penalty terms plus a reweighted ERM objective that weights each domain equally (i.e., effectively upweighting minority domains). The Group DRO objective is unchanged, as it still optimizes for the domain with the worst loss, but the uniform sampling improves optimization stability.

Both CORAL and IRM are designed for models with featurizers, i.e., models that first map each input to a feature representation and then predict based on the representation. To estimate the feature distribution for a domain, these algorithms need to see a sufficient number of examples from that domain in a minibatch. However, some of our datasets have large numbers of domains, making it infeasible for each minibatch to contain examples from all domains. For these algorithms, our data loaders form a minibatch by first sampling a few domains, and then sampling examples from those domains. For consistency in our experiments, we used the same total batch size for these algorithms and for ERM and Group DRO, with a default of 8 examples per domain in each minibatch (e.g., if the batch size was 32, then in each minibatch we would have $8 \text{ examples} \times 4 \text{ domains}$).

For Group DRO, as in Sagawa et al. (2020a), each example in the minibatch is sampled independently with uniform probabilities across domains, and therefore each minibatch does not need to only comprise a small number of domains. We note that reweighting methods like Group DRO are effective only when the training loss is non-vanishing, which we achieve through early stopping (Byrd & Lipton, 2019; Sagawa et al., 2020a;b).

H. Additional dataset details and results

In this section, we discuss each WILDS dataset in more detail. For completeness, we start by repeating the motivation behind each dataset from Section 4. We then describe the

Table 3: The performance of models trained with empirical risk minimization with hyperparameters tuned using the out-of-distribution (OOD) vs. in-distribution (ID) validation set. We excluded OGB-MOLPCBA, RXRX1-WILDS, and GLOBALWHEAT-WILDS, as they do not have separate ID validation sets, and CIVILCOMMENTS-WILDS, which is a subpopulation shift setting where we measure worst-group accuracy on a validation set that is already identically distributed to the training set.

Dataset	Metric	ID performance		OOD performance	
		Tuned on ID val	Tuned on OOD val	Tuned on ID val	Tuned on OOD val
iWILDCAM2020-WILDS	Macro F1	47.2 (2.0)	47.0 (1.4)	29.8 (0.6)	31.0 (1.3)
CAMELYON17-WILDS	Average acc	98.7 (0.1)	93.2 (5.2)	65.8 (4.9)	70.3 (6.4)
FMOW-WILDS	Worst-region acc	58.0 (0.5)	57.4 (0.2)	31.9 (0.8)	32.8 (0.5)
POVERTYMAP-WILDS	Worst-U/R Pearson R	0.65 (0.03)	0.62 (0.04)	0.46 (0.06)	0.46 (0.07)
AMAZON-WILDS	10th percentile acc	72.0 (0.0)	71.9 (0.1)	53.8 (0.8)	53.8 (0.8)
PY150-WILDS	Method/class acc	75.6 (0.0)	75.4 (0.4)	67.9 (0.1)	67.9 (0.1)

task, the distribution shift, and the evaluation criteria, and present baseline results that elaborate upon those in Sections 5 and 6. We also discuss the broader context behind each dataset and how it connects with other distribution shifts in similar applications. Finally, we describe how each dataset was modified from its original version in terms of the evaluation, splits, and data. Unless otherwise specified, all experiments follow the protocol laid out in Appendix G.

H.1. iWILDCAM2020-WILDS

Animal populations have declined 68% on average since 1970 (Grooten et al., 2020). To better understand and monitor wildlife biodiversity loss, ecologists commonly deploy camera traps—heat or motion-activated static cameras placed in the wild (Wearn & Glover-Kapfer, 2017)—and then use ML models to process the data collected (Weinstein, 2018; Norouzzadeh et al., 2019; Tabak et al., 2019; Beery et al., 2019; Ahumada et al., 2020). Typically, these models would be trained on photos from some existing camera traps and then used across new camera trap deployments. However, across different camera traps, there is drastic variation in illumination, color, camera angle, background, vegetation, and relative animal frequencies, which results in models generalizing poorly to new camera trap deployments (Beery et al., 2018).

We study this shift on a variant of the iWildCam 2020 dataset (Beery et al., 2020a).

H.1.1. SETUP

Problem setting. We consider the domain generalization setting, where the domains are camera traps, and we seek to learn models that generalize to photos taken from new camera deployments (Figure 6). The task is multi-class species classification. Concretely, the input x is a photo taken by a camera trap, the label y is one of 182 different animal species, and the domain d is an integer that identifies the camera trap that took the photo.

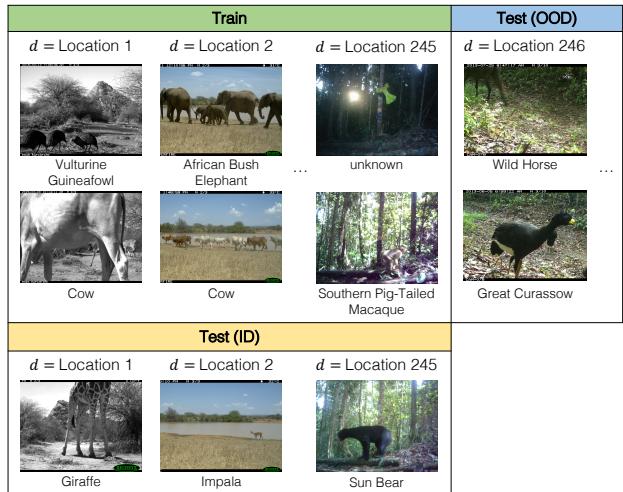


Figure 6: The iWILDCAM2020-WILDS dataset comprises photos of wildlife taken by a variety of camera traps. The goal is to learn models that generalize to photos from new camera traps that are not in the training set. Each WILDS dataset contains both in-distribution (ID) and out-of-distribution (OOD) evaluation sets; for brevity, we omit the ID sets from the subsequent dataset figures.

Data. The dataset comprises 203,029 images from 323 different camera traps spread across multiple countries in different parts of the world. The original camera trap data comes from the Wildlife Conservation Society (<http://lila.science/datasets/wcscameratraps>). These images tend to be taken in short bursts following motion-activation of a camera trap, so the images can be additionally grouped into sequences of images from the same burst, though our baseline models do not exploit this information and our evaluation metric treats each image individually. Each image is associated with the following metadata: camera trap ID, sequence ID, and datetime.

As is typical for camera trap data, approximately 35% of the total number of images are empty (i.e., do not contain any animal species); this corresponds to one of the 182 class labels. The ten most common classes across the full dataset

are “empty” (34%), ocellated turkey (8%), great curassow (6%), impala (4%), black-fronted duiker (4%), white-lipped peccary (3%), Central American agouti (3%), ocelot (3%), gray fox (2%) and cow (2%).

We split the dataset by randomly partitioning the data by camera traps:

1. **Training:** 129,809 images taken by 243 camera traps.
2. **Validation (OOD):** 14,961 images taken by 32 different camera traps.
3. **Test (OOD):** 42,791 images taken by 48 different camera traps.
4. **Validation (ID):** 7,314 images taken by the same camera traps as the training set, but on different days from the training and test (ID) images.
5. **Test (ID):** 8,154 images taken by the same camera traps as the training set, but on different days from the training and validation (ID) images.

The camera traps are randomly distributed across the training, validation, validation (OOD), and test (OOD) sets. The number of examples per location vary widely from 1 to 8494, with a median of 194 images (Figure 7). All images from the same sequence (i.e., all images taken in the same burst) are placed together in the same split. See Appendix H.1.4 for more details.

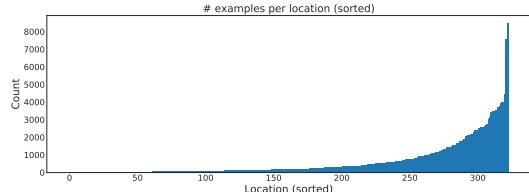


Figure 7: Number of examples per location in the iWILDCAM2020-WILDS dataset. The locations are sorted such that locations with the least amount of examples are to the left on the x-axis.

Evaluation. We evaluate models by their macro F1 score (i.e., we compute the F1 score for each class separately, then average those scores). We also report the average accuracy of each model across all test images, but primarily use the macro F1 score to better capture model performance on rare species. In the natural world, protected and endangered species are rare by definition, and are often the most important to accurately monitor. However, common species are much more likely to be captured in camera trap images; this imbalance can make metrics like average accuracy an inaccurate picture of model effectiveness.

Potential leverage. Though the problem is challenging for existing ML algorithms, adapting to photos from different camera traps is simple and intuitive for humans. Repeated backgrounds and habitual animals, which cause each sensor to have a unique class distribution, provide a strong implicit signal across data from any one location. We anticipate that approaches that utilize the provided camera trap annotations can learn to factor out these common features and avoid learning spurious correlations between particular backgrounds and animal species.

H.1.2. BASELINE RESULTS

Model. For all experiments, we use ResNet-50 models (He et al., 2016) that were pretrained on ImageNet, using a learning rate of 3×10^{-5} and no L_2 -regularization. As input, these models take in images resized to 448 by 448. We trained these models with the Adam optimizer and a batch size of 16 for 12 epochs. To pick hyperparameters, we did a grid search over learning rates $\{1 \times 10^{-5}, 3 \times 10^{-5}, 1 \times 10^{-4}\}$ and L_2 regularization strengths $\{0, 1 \times 10^{-3}, 1 \times 10^{-2}\}$. We report results aggregated over 3 random seeds.

ERM results and performance drops. Model performance dropped substantially and consistently going from in-distribution (ID) to out-of-distribution (OOD) camera traps (Table 4), with a macro F1 score of 47.0 on the ID test set but only 31.0 on the OOD test set. We note that macro F1 and average accuracy both differ between the OOD validation and test sets: this is in part due to the difference in class balance between them, which in turn is due to differences in the proportion of classes across camera traps. In particular, macro F1 can vary between splits because we take the average F1 score across all classes that are present in the evaluation split, and not all splits have the same classes present (e.g., a rare species might be in the OOD validation set but not OOD test set, or vice versa). In addition, average accuracy can differ between splits due in part to variation in the fraction of empty images per location (e.g., the camera traps that were randomly assigned to the OOD validation set have a smaller proportion of empty images).

Additional baseline methods. We trained models with CORAL, IRM, and Group DRO, treating each camera trap as a domain, and using the same model hyperparameters as ERM. These did not improve upon the ERM baseline (Table 4). The IRM models performed especially poorly on this dataset; we suspect that this is because the default estimator of the IRM penalty term can be negatively biased when examples are sampled without replacement from small domains, but further investigation is needed.

Discussion. Across locations, there is drastic variation in illumination, camera angle, background, vegetation, and

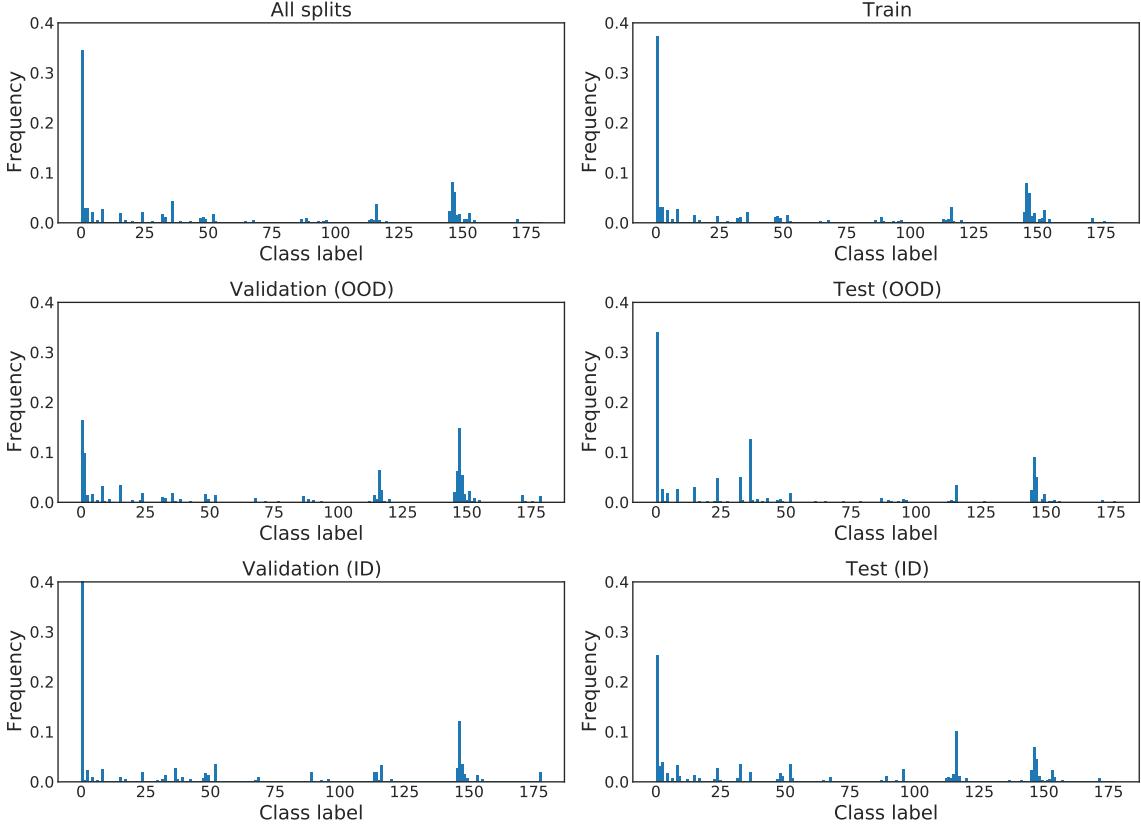


Figure 8: Label distribution for each iWILDCAM2020-WILDS split.

color. This variation, coupled with considerable differences in the distribution of animals between camera traps, likely encourages the model to overfit to specific animal species appearing in specific locations, which may account for the performance drop.

The original iWildCam 2020 competition allows users to use MegaDetector (Beery et al., 2019), which is an animal detector trained on a large set of data beyond what is provided in the training set. Using an animal detection model like MegaDetector typically improves classification performance on camera traps (Beery et al., 2018). However, we intentionally do not use MegaDetector in our baselines for iWILDCAM2020-WILDS for two reasons. First, though the trained MegaDetector model is publicly available, the MegaDetector training set is not, which makes it difficult to build on top of it and run controlled experiments. Second, bounding box annotations are costly and harder to obtain, and there is much more data with image-level species label, so it would be useful to be able to train models that do not have to rely on bounding box annotations.

We still welcome leaderboard submissions that use MegaDetector, as it is useful to see how much better models can perform when they use MegaDetector or other similar an-

imal detectors, but we will distinguish these submissions from others that only use what is provided in the training set.

A different source of leverage comes from the temporal signal in the camera trap images, which are organized into sequences that each correspond to a burst of images from a single motion trigger. Using this sequence information (e.g., by taking the median prediction across a sequence) can also improve model performance (Beery et al., 2018), and we welcome submissions that explore this direction.

H.1.3. BROADER CONTEXT

Differences across data distributions at different sensor locations is a common challenge in automated wildlife monitoring applications, including using audio sensors to monitor animals that are easier heard than seen such as primates, birds, and marine mammals (Crunchant et al., 2020; Stowell et al., 2019; Shiu et al., 2020), and using static sonar to count fish underwater to help maintain sustainable fishing industries (Pipal et al., 2012; Vatnehol et al., 2018; Schneider & Zhuang, 2020). As with camera traps, each static audio sensor has a specific species distribution as well as a sensor specific background noise signature, making general-

Table 4: Baseline results on iWILDCAM2020-WILDS.

Algorithm	Val (ID)		Val (OOD)		Test (ID)		Test (OOD)	
	Macro F1	Avg acc	Macro F1	Avg acc	Macro F1	Avg acc	Macro F1	Avg acc
ERM	48.8 (2.5)	82.5 (0.8)	37.4 (1.7)	62.7 (2.4)	47.0 (1.4)	75.7 (0.3)	31.0 (1.3)	71.6 (2.5)
CORAL	46.7 (2.8)	81.8 (0.4)	37.0 (1.2)	60.3 (2.8)	43.5 (3.5)	73.7 (0.4)	32.8 (0.1)	73.3 (4.3)
IRM	24.4 (8.4)	66.9 (9.4)	20.2 (7.6)	47.2 (9.8)	22.4 (7.7)	59.9 (8.1)	15.1 (4.9)	59.8 (3.7)
Group DRO	42.3 (2.1)	79.3 (3.9)	26.3 (0.2)	60.0 (0.7)	37.5 (1.7)	71.6 (2.7)	23.9 (2.1)	72.7 (2.0)
Reweighted (label)	42.5 (0.5)	77.5 (1.6)	30.9 (0.3)	57.8 (2.8)	42.2 (1.4)	70.8 (1.5)	26.2 (1.4)	68.8 (1.6)

ization to new sensors challenging. Similarly, static sonar used to measure fish escapement have sensor-specific background reflectance based on the shape of the river bottom. Moreover, since species are distributed in a non-uniform and long-tailed fashion across the globe, it is incredibly challenging to collect sufficient samples for rare species to escape the low-data regime. Implicitly representing camera-specific distributions and background features in per-camera memory banks and extracting relevant information from these via attention has been shown to help overcome some of these challenges for static cameras (Beery et al., 2020b).

More broadly, shifts in background, image illumination and viewpoint have been studied in computer vision research. First, several works have shown that object classifiers often rely on the background rather than the object to make its classification (Rosenfeld et al., 2018; Shetty et al., 2019; Xiao et al., 2020). Second, common perturbations such as blurriness or shifts in illumination, tend to reduce performance (Dodge & Karam, 2017; Temel et al., 2018; Hendrycks & Dietterich, 2019). Finally, shifts in rotation and viewpoint of the object has been shown to degrade performance (Barbu et al., 2019).

H.1.4. ADDITIONAL DETAILS

Data processing. We generate the data splits in three steps. First, to generate the OOD splits, we randomly split all locations into three groups: Validation (OOD), Test (OOD), and Others. Then, to generate the ID splits, we split the Others group uniformly by date at random into three sets: Training, Validation (ID), and Test (ID).

When doing the ID split, some locations only ended up in some of but not all of Training, Validation (ID), and Test (ID). For instance, if there were very few dates for a specific location (camera trap), it may be that no examples from that location ended up in the train split. This defeats the purpose of the ID split, which is to test performance on locations that were seen during training. We therefore put these locations in the train split. Finally, any images in the test set with classes not present in the train set were removed.

Modifications to the original dataset. The original iWildCam 2020 Kaggle competition similarly split the dataset by camera trap, though the competition focused on average accuracy. We consider a smaller subset of the data here. Specifically, the Kaggle competition uses a held-out test set that we are not utilizing, as the test set is intended to be reused in a future competition and is not yet public. Instead, we constructed our own test set by splitting the Kaggle competition training data into our own splits: train, validation (ID), validation (OOD), test (ID), test (OOD).

Images are organized into sequences, but we treat each image separately. In the iWildCam 2020 competition, the top participants utilized the sequence data and also used a pre-trained MegaDetector animal detection model that outputs bounding boxes over the animals. These images are cropped using the bounding boxes and then fed into a classification network. As we discuss above, we intentionally do not use MegaDetector in our experiments.

In addition, compared to the iWildCam 2020 competition, the iWildCam 2021 competition changed several class definitions (such as removing the “unknown” class) and removed some images that were taken indoors or had humans in the background. We have applied these updates to iWILDCAM2020-WILDS as well.

H.2. CAMELYON17-WILDS

Models for medical applications are often trained on data from a small number of hospitals, but with the goal of being deployed more generally across other hospitals. However, variations in data collection and processing can degrade model accuracy on data from new hospital deployments (Zech et al., 2018; AlBadawy et al., 2018). In histopathology applications—studying tissue slides under a microscope—this variation can arise from sources like differences in the patient population or in slide staining and image acquisition (Veta et al., 2016; Komura & Ishikawa, 2018; Tellez et al., 2019).

We study this shift on a patch-based variant of the Camelyon17 dataset (Bandi et al., 2018).

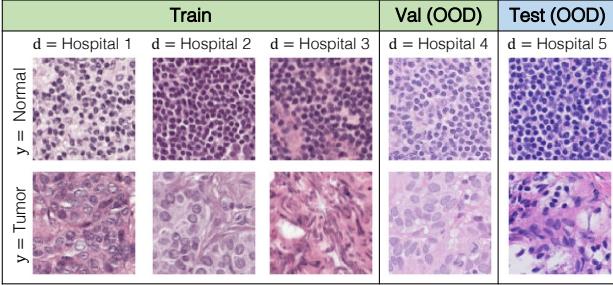


Figure 9: The CAMELYON17-WILDS dataset comprises tissue patches from different hospitals. The goal is to accurately predict the presence of tumor tissue in patches taken from hospitals that are not in the training set. In this figure, each column contains two patches, one of normal tissue and the other of tumor tissue, from the same slide.

H.2.1. SETUP

Problem setting. We consider the domain generalization setting, where the domains are hospitals, and our goal is to learn models that generalize to data from a hospital that is not in the training set (Figure 9). The task is to predict if a given region of tissue contains any tumor tissue, which we model as binary classification. Concretely, the input x is a 96×96 histopathological image, the label y is a binary indicator of whether the central 32×32 region contains any tumor tissue, and the domain d is an integer that identifies the hospital that the patch was taken from.

Data. The dataset comprises 450,000 patches extracted from 50 whole-slide images (WSIs) of breast cancer metastases in lymph node sections, with 10 WSIs from each of 5 hospitals in the Netherlands. Each WSI was manually annotated with tumor regions by pathologists, and the resulting segmentation masks were used to determine the labels for each patch. We also provide metadata on which slide (WSI) each patch was taken from, though our baseline algorithms do not use this metadata.

We split the dataset by domain (i.e., which hospital the patches were taken from):

1. **Training:** 302,436 patches taken from 30 WSIs, with 10 WSIs from each of the 3 hospitals in the training set.
2. **Validation (OOD):** 34,904 patches taken from 10 WSIs from the 4th hospital. These WSIs are distinct from those in the other splits.
3. **Test (OOD):** 85,054 patches taken from 10 WSIs from the 5th hospital, which was chosen because its patches were the most visually distinctive. These WSIs are also distinct from those in the other splits.
4. **Validation (ID):** 33,560 patches taken from the same 30 WSIs from the training hospitals.

We do not provide a Test (ID) set, as there is no practical setting in which we would have labels on a uniformly randomly sampled set of patches from a WSI, but no labels on the other patches from the same WSI.

Evaluation. We evaluate models by their average test accuracy across patches. Histopathology datasets can be unwieldy for ML models, as individual images can be several gigabytes large; extracting patches involves many design choices; the classes are typically very unbalanced; and evaluation often relies on more complex slide-level measures such as the free-response receiver operating characteristic (FROC) (Gurcan et al., 2009). To improve accessibility, we pre-process the slides into patches and balance the dataset so that each split has a 50/50 class balance, making average accuracy is a reasonable measure of performance (Veeling et al., 2018; Tellez et al., 2019).

Potential leverage. Prior work has shown that differences in staining between hospitals are the primary source of variation in this dataset, and that specialized stain augmentation methods can close the in- and out-of-distribution accuracy gap on a variant of the dataset based on the same underlying slides (Tellez et al., 2019). However, the general task of learning histopathological models that are robust to variation across hospitals (from staining and other sources) is still an open research question. In this way, the CAMELYON17-WILDS dataset is a controlled testbed for general-purpose methods that can learn to be robust to stain variation between hospitals, given a training set from multiple hospitals.

H.2.2. BASELINE RESULTS

Model. For all experiments, we use DenseNet-121 models (Huang et al., 2017) models trained from scratch on the 96×96 patches, following prior work (Veeling et al., 2018). These models used a learning rate of 10^{-3} , L_2 -regularization strength of 10^{-2} , a batch size of 32, and SGD with momentum (set to 0.9), trained for 5 epochs with early stopping. We selected hyperparameters by a grid search over learning rates $\{10^{-4}, 10^{-3}, 10^{-2}\}$, and L_2 -regularization strengths $\{0, 10^{-3}, 10^{-2}\}$. We report results aggregated over 10 random seeds.

ERM results and performance drops. Table 5 shows that the model was consistently accurate on the in-distribution (ID) validation set and to a lesser extent on the out-of-distribution (OOD) validation set, which was from a held-out hospital. However, it was wildly inconsistent on the test set, which was from a different held-out hospital, with a standard deviation of 6.4% in accuracies across 10 random seeds. There is a large gap between ID validation and OOD validation accuracy, and between OOD validation and OOD test accuracy (in part because we early stop on the highest

Table 5: Baseline results on CAMELYON17-WILDS. Parentheses show standard deviation across 10 replicates.

Algorithm	Validation (ID) accuracy	Validation (OOD) accuracy	Test (OOD) accuracy
ERM	93.2 (5.2)	84.9 (3.1)	70.3 (6.4)
CORAL	95.4 (3.6)	86.2 (1.4)	59.5 (7.7)
IRM	91.6 (7.7)	86.2 (1.4)	64.2 (8.1)
Group DRO	93.7 (5.2)	85.5 (2.2)	68.4 (7.3)

Table 6: Performance drops for ERM models on CAMELYON17-WILDS. In the standard split, we train on data from three hospitals and evaluate on a different test hospital, whereas in the mixed split, we add data from one extra slide from the test hospital to the training set. The original test (OOD) set has data from 10 slides; here, we report performance for both splits on 9 slides (without the slide that was moved to the training set). This makes the numbers (71.0 vs. 70.3) for the standard split slightly different from Table 5. Parentheses show standard deviation across 10 replicates. As in the rest of the paper, note that the standard error of the mean is smaller (by a factor of $\sqrt{10}$).

	Algorithm	Test (OOD) accuracy
Standard split (train on ID examples)	ERM	71.0 (6.3)
Mixed split (train on ID + OOD examples)	ERM	82.9 (9.8)

OOD validation accuracy). Nevertheless, we found that using the OOD validation set gave better results than using the ID validation set; see Appendix G.2 for more discussion.

We ran an additional experiment on a mixed split, where we moved 1 of the 10 slides⁴ from the test hospital to the training set and tested on the patches from the remaining 9 slides. By training on this mixed split, the resulting oracle model consistently gets significantly higher accuracy on the reduced test set (Table 6), further suggesting that the observed performance drop is due to the distribution shift, as opposed to the intrinsic difficulty of the examples from the test hospital.

Additional baseline methods. We trained models with CORAL, IRM, and Group DRO, treating each hospital as a domain. However, they performed comparably or worse than the ERM baseline. For the CORAL and IRM models, our grid search selected the lowest values of their penalty weights (0.1 and 1, respectively) based on OOD validation accuracy.

Discussion. These results demonstrate a subtle failure mode when considering out-of-distribution accuracy: there are models (i.e., choices of hyperparameters and random seeds) that do well both in- and out-of-distribution, but we cannot reliably choose these models from just the training/validation set. Due to the substantial variability in test accuracy on CAMELYON17-WILDS (see Figure 10), we ask researchers to submit leaderboard submissions with results from 10 random seeds, instead of the 3 random seeds required for other datasets.

⁴This slide was randomly chosen and corresponded to about 6% of the test patches; some slides contribute more patches than others because they contain larger tumor regions.

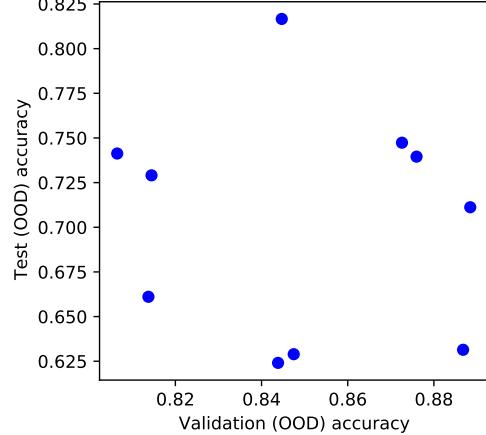


Figure 10: Test (OOD) accuracy versus validation (OOD) accuracy for different random seeds on CAMELYON17-WILDS, using the same hyperparameters. The test accuracy is far more variable than the validation accuracy (note the differences in the axes), in part because we early stop on the highest OOD validation accuracy.

Many specialized methods have been developed to handle stain variation in the context of digital histopathology. These typically fall into one of two categories: data augmentation methods that perturb the colors in the training images (e.g., Liu et al. (2017); Bug et al. (2017); Tellez et al. (2018)) or stain normalization methods that seek to standardize colors across training images (e.g., Macenko et al. (2009); BenTaieb & Hamarneh (2017)). These methods are reasonably effective at mitigating stain variation, at least in some contexts (Tellez et al., 2019), though the general problem of learning digital histopathology models that can be effectively deployed across multiple hospitals/sites is still an open challenge.

To facilitate more controlled experiments, we will have two leaderboard tracks for CAMELYON17-WILDS. For the first track, which focuses on general-purpose algorithms, submissions should not use color-specific techniques (e.g., color augmentation) and should also train their models from scratch, instead of fine-tuning models that are pre-trained from ImageNet or other datasets. For the second track, submissions can use any of those techniques, including specialized methods for dealing with stain variation. These separate tracks will help to disentangle the contributions of more general-purpose learning algorithms and model architectures from the contributions of specialized augmentation techniques or additional training data.

H.2.3. BROADER CONTEXT

Other than stain variation, there are many other distribution shifts that might occur in histopathology applications. For example, patient demographics might differ from hospital to hospital: some hospitals might tend to see patients who are older or more sick, and patients from different backgrounds and countries vary in terms of cancer susceptibility (Henderson et al., 2012). Some cancer subtypes and tissues of origin are also more common than others, leading to potential subpopulation shift issues, e.g., a rare cancer subtype in one context might be more common in another; or even if it remains rare, we would seek to leverage the greater quantity of data from other subtypes to improve model accuracy on the rare subtype (Weinstein et al., 2013).

Beyond histopathology, variation between different hospitals and deployment sites has also been shown to degrade model accuracy in other medical applications such as diabetic retinopathy (Beede et al., 2020) and chest radiographs (Zech et al., 2018; Phillips et al., 2020), including recent work on COVID-19 detection (DeGrave et al., 2020). Even within the same hospital, process variables like which scanner/technician took the image can significantly affect models (Badgeley et al., 2019).

In these medical applications, the gold standard is to evaluate models on an independent test set collected from a different hospital (e.g., Beck et al. (2011); Liu et al. (2017); Courtiol et al. (2019); McKinney et al. (2020)) or at least with a different scanner within the same hospital (e.g., Campanella et al. (2019)). However, this practice has not been ubiquitous due to the difficulty of obtaining data spanning multiple hospitals (Esteva et al., 2017; Bejnordi et al., 2017; Codella et al., 2019; Veta et al., 2019). The baseline results reported above show that even evaluating on a single different hospital might be insufficient, as results can vary widely between different hospitals (e.g., between the validation and test OOD datasets). We hope that the CAMELYON17-WILDS dataset, which has multiple hospitals in the training set and independent hospitals in the validation and test sets,

will be useful for developing models that can generalize reliably to new hospitals and contexts (Chen et al., 2020).

H.2.4. ADDITIONAL DETAILS

Data processing. The CAMELYON17-WILDS dataset is adapted from whole-slide images (WSIs) of breast cancer metastases in lymph nodes sections, obtained from the CAMELYON17 challenge (Bandi et al., 2018). Each split is balanced to have an equal number of positive and negative examples. The varying number of patches per slide and hospital is due to this class balancing, as some slides have fewer tumor (positive) patches. We selected the test set hospital as the one whose patches were visually most distinct; the difference in test versus OOD validation performance shows that the choice of OOD hospital can significantly affect performance.

From these WSIs, we extracted patches in a standard manner, similar to Veeling et al. (2018). The WSIs were scanned at a resolution of $0.23\mu\text{m}$ – $0.25\mu\text{m}$ in the original dataset, and each WSI contains multiple resolution levels, with approximately $10,000 \times 20,000$ pixels at the highest resolution level (Bandi et al., 2018). We used the third-highest resolution level, corresponding to reducing the size of each dimension by a factor of 4. We then tiled each slide with overlapping 96×96 pixel patches with a step size of 32 pixels in each direction (such that none of the central 32×32 regions overlap), labeling them as the following:

- *Tumor* patches have at least one pixel of tumor tissue in the central 32×32 region. We used the pathologist-annotated tumor annotations provided with the WSIs.
- *Normal* patches have no tumor and have at least 20% normal tissue in the central 32×32 region. We used Otsu thresholding to distinguish normal tissue from background.

We discarded all patches that had no tumor and <20% normal tissue in the central 32×32 region.

To maintain an equal class balance, we then subsampled the extracted patches in the following way. First, for each WSI, we kept all tumor patches unless the WSI had fewer normal than tumor patches, which was the case for a single WSI; in that case, we randomly discarded tumor patches from that WSI until the numbers of tumor and normal patches were equal. Then, we randomly selected normal patches for inclusion such that for each hospital and split, there was an equal number of tumor and normal patches.

Modifications to the original dataset. The task in the original CAMELYON17 challenge (Bandi et al., 2018) was the patient-level classification task of determining the pathologic lymph node stage of the tumor present in all slides

from a patient. In contrast, our task is a lesion-level classification task. Patient-level, slide-level, and lesion-level tasks are all common in histopathology applications. As mentioned above, the original dataset provided WSIs and tumor annotations, but not a standardized set of patches, which we provide here. Moreover, it did not consider distribution shifts; both of the original training and test splits contained slides from all 5 hospitals.

The CAMELYON17-WILDS patch-based dataset is similar to one of the datasets used in Tellez et al. (2019), which was also derived from the CAMELYON17 challenge; there, only one hospital is used as the training set, and the other hospitals are all part of the test set. CAMELYON17-WILDS is also similar to PCam (Veeling et al., 2018), which is a patch-based dataset based on an earlier CAMELYON16 challenge; the data there is derived from only two hospitals.

Additional data sources. The full, original CAMELYON17 dataset contains 1000 WSIs from the same 5 hospitals, although only 50 of them (which we use here) have tumor annotations. The other 950 WSIs may be used as unlabeled data. Beyond the CAMELYON17 dataset, the largest source of unlabeled WSI data is the Cancer Genome Atlas (Weinstein et al., 2013), which typically has patient-level annotations (e.g., patient demographics and clinical outcomes).

H.3. RXRX1-WILDS

High-throughput screening techniques that can generate large amounts of data are now common in many fields of biology, including transcriptomics (Harrill et al., 2019), genomics (Echeverri & Perrimon, 2006; Zhou et al., 2014), proteomics and metabolomics (Taylor et al., 2021), and drug discovery (Broach et al., 1996; Macarron et al., 2011; Swinney & Anthony, 2011; Boutros et al., 2015). Such large volumes of data, however, need to be created in experimental batches, or groups of experiments executed at similar times under similar conditions. Despite attempts to carefully control experimental variables such as temperature, humidity, and reagent concentration, measurements from these screens are confounded by technical artifacts that arise from differences in the execution of each batch. These *batch effects* make it difficult to draw conclusions from data across experimental batches (Leek et al., 2010; Parker & Leek, 2012; Soneson et al., 2014; Nygaard et al., 2016; Caicedo et al., 2017).

We study the shift induced by batch effects on a variant of the RXRX1-WILDS dataset (Taylor et al., 2019). As illustrated in Figure 11, there are significant visual differences between experimental batches, making recognizing siRNA perturbations in OOD experiments in the RXRX1-WILDS dataset a particularly challenging task for existing ML algo-

rithms.

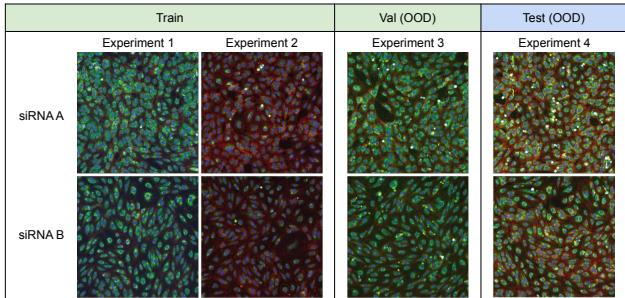


Figure 11: The RXRX1-WILDS dataset comprises images of cells that have been genetically perturbed by siRNA (Tuschl, 2001). The goal is to predict which siRNA the cells have been treated with, where the images come from experimental batches not in the training set. Here, we show sample images from different batches for two of the 1,139 possible classes.

H.3.1. SETUP

Problem setting. We consider the domain generalization setting, where the domains are experimental batches and we seek to generalize to images from unseen experimental batches. Concretely, the input x is a 3-channel image of cells obtained by fluorescent microscopy, the label y indicates which of the 1,139 genetic treatments (including no treatment) the cells received, and the domain d specifies the experimental batch of the image.

Data. RXRX1-WILDS was created by Recursion (recursion.com) in its automated high-throughput screening laboratory in Salt Lake City, Utah. It is comprised of fluorescent microscopy images of human cells in four different cell lines: HUVEC, RPE, HepG2, and U2OS. These were acquired via fluorescent microscopy using a 6-channel variant of the *Cell Painting* assay (Bray et al., 2016). Figure 12 shows an example of the cellular contents of each of these 6 channels: nuclei, endoplasmic reticuli, actin, nucleoli and cytoplasmic RNA, mitochondria, and Golgi. To make the dataset smaller and more accessible, we only included the first 3 channels in RXRX1-WILDS.

The images in RXRX1-WILDS are the result of executing the same experimental design 51 different times, each in a different batch of experiments. The design consists of four 384-well plates, where individual wells are used to isolate populations of cells on each plate (see Figure 13). The wells are laid out in a 16×24 grid, but only the wells in the inner 14×22 grid are used since the outer wells are most susceptible to environmental factors. Of these 308 usable wells, one is left untreated to provide a *negative control* phenotype, while the rest are treated with small interfering ribonucleic acid, or siRNA, at a fixed concentration. Each siRNA is designed to knockdown a single target gene via

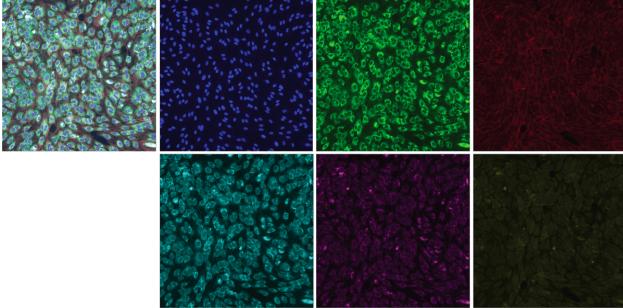


Figure 12: 6-channel composite image of HUVEC cells (left) and its individual channels (rest): nuclei (blue), endoplasmic reticuli (green), actin (red), nucleoli and cytoplasmic RNA (cyan), mitochondria (magenta), and Golgi (yellow). The overlap in channel content is due in part to the lack of complete spectral separation between fluorescent stains. Note that only the first 3 channels are included in RxRx1-WILDS.

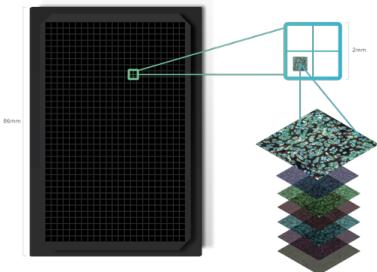


Figure 13: Schematic of a 384-well plate demonstrating imaging sites and 6-channel images. The 4-plate experiments in RxRx1-WILDS were run in the wells of such 384-well plates. RxRx1-WILDS contains two imaging sites per well.

the RNA interference pathway, reducing the expression of the gene and its associated protein (Tuschl, 2001). However, siRNAs are known to have significant but consistent off-target effects via the microRNA pathway, creating partial knockdown of many other genes as well. The overall effect of siRNA transfection is to perturb the morphology, count, and distribution of cells, creating a *phenotype* associated with each siRNA. The phenotype is sometimes visually recognizable, but often the effects are subtle and hard to detect.

In each plate, 30 wells are set aside for 30 *positive control* siRNAs. Each has a different gene as its primary target, which together with the single untreated well already mentioned, provides a set of reference phenotypes per plate. Each of the remaining 1,108 wells of the design ($277 \text{ wells} \times 4 \text{ plates}$) receives one of 1,108 *treatment* siRNA, respectively, so that there is at most one well of each treatment siRNA in each experiment. We say at most once because, although rare, it happens that either an siRNA is not correctly transferred into the designated destination well, resulting in an additional untreated well, or an operational error is

detected by quality control procedures that render the well unsuitable for inclusion in the dataset.

Each experiment was run in a single cell type, and of the 51 experiments in RxRx1-WILDS, 24 are in HUVEC, 11 in RPE, 11 in HepG2, and 5 in U2OS. Figure 14 shows the phenotype of the same siRNA in each of these four cell types.

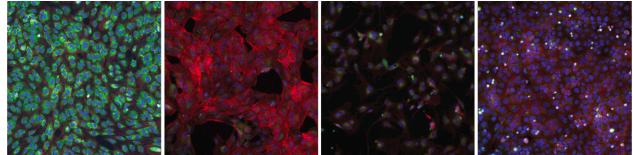


Figure 14: Images of the same siRNA in four cell types, from left to right: HUVEC, RPE, HepG2, U2OS.

We split the dataset by experimental batches, with the training and test splits having roughly the same composition of cell types:

- 1. Training:** 33 experiments (16 HUVEC, 7 RPE, 7 HepG2, 3 U2OS), site 1 only = 40,612 images.
- 2. Validation (OOD):** 4 experiments (1 HUVEC, 1 RPE, 1 HepG2, 3 U2OS), sites 1 and 2 = 9,854 images.
- 3. Test (OOD):** 14 experiments (7 HUVEC, 3 RPE, 3 HepG2, 1 U2OS), sites 1 and 2 = 34,432 images.
- 4. Test (ID):** same 33 experiments as in the training set, site 2 only = 40,612 images.

In addition to the class (siRNA), each image is associated with the following metadata: cell type, experiment, plate, well, and site. We emphasize that all the images of an experiment are found in exactly one of the training, validation (OOD) or test (OOD) splits. See Appendix H.3.4 for more data processing details.

Evaluation. We evaluate models by their average accuracy across test images. Note that there are two images per well in the test set, which we evaluate independently.

The cell types are not balanced in the training and test sets. Correspondingly, we observed higher performance on the HUVEC cell type, which is over-represented, and lower performance on the U2OS cell type, which is under-represented. While maintaining high performance on minority (or even unseen) cell types is an important problem, for RxRx1-WILDS, we opt to measure the average accuracy across all experiments instead of, for example, the worst accuracy across cell types. This is because the relatively small amount of training data available from the minority cell type (U2OS) makes it challenging to cast RxRx1-WILDS

as a tractable subpopulation shift problem. We also note that the difference in performance across cell types leads to the validation performance being significantly lower than the test performance, as there is a comparatively smaller fraction of HUVEC and a comparatively higher fraction of U2OS.

Potential leverage. By design, there is usually one sample per class per experiment in the training set, with the following exceptions: 1) there are usually four samples per positive control, though 2) samples may be missing, as described above. Moreover, while batch effects can manifest themselves in many complicated ways, it is the case that the training set consists of a large number of experiments selected randomly amongst all experiments in the dataset, hence we expect models to be able to learn what is common amongst all such samples per cell type, and for that ability to generalize to test batches. We emphasize that, whether in the training or test sets, the same cell types are perturbed with the same siRNA, and thus the phenotypic distributions for each batch share much of the same generative process.

We also note that, while not exploited here, there is quite a bit of structure in the RxRx1-WILDS dataset. For example, except in the case of errors, all treatment siRNA appear once in each experiment, and all control conditions appear once per plate, so four times per experiment. Also, due to the operational efficiencies gained, the 1,108 treatment siRNAs always appear in the same four groups of 277 per experiment. So while the particular well an siRNA appears in is randomized, it will always appear with the same group of 276 other siRNAs. This structure can be exploited for improving predictive accuracy via post-prediction methods such as linear sum assignment. However, such methods do not represent improved generalization to OOD samples, and should be avoided.

H.3.2. BASELINE RESULTS

Model. For all experiments, we train the standard ResNet-50 model (He et al., 2016) pretrained on ImageNet, using a learning rate of $1e - 4$ and L_2 -regularization strength of $1e - 5$. We trained these models with the Adam optimizer, using default parameter values $\beta_1 = 0.9$ and $\beta_2 = 0.999$, with a batch size of 75 for 90 epochs, linearly increasing the learning rate for 10 epochs, then decreasing it following a cosine learning rate schedule. We selected hyperparameters by a grid search over learning rates $\{10^{-5}, 10^{-4}, 10^{-3}\}$, L_2 -regularization strengths $\{10^{-5}, 10^{-3}\}$, and numbers of warmup epochs $\{5, 10\}$. We report results aggregated over 3 random seeds.

ERM results and performance drops. Model performance dropped significantly from the in-distribution (ID) to out-of-distribution (OOD) batches (Table 7), with an aver-

age accuracy of 35.9% on the ID test set but only 29.9% on the OOD test set for ERM.

We ran an additional experiment on a mixed split, where we moved half of the OOD test set into the training set, while keeping the overall amount of training data the same. Specifically, we moved one site per experiment from the OOD test set into the training set, and discarded an equivalent number of training sites, while leaving the validation set unchanged. While the test set in the mixed split is effectively half as large as in the standard split, we expect it to be distributed similarly, since the two test set versions comprise the same 14 experiments.

Table 8 shows that there is a large gap between the OOD test accuracies in the standard split (29.9%) versus in the mixed split (39.8%). We note that the latter is higher than the ID test accuracy of 35.9% reported for the standard split in Table 7. This difference mainly stems from the slight difference in cell type composition between the ID test set of the standard split and the OOD test set of the mixed split; in particular, the latter a slightly higher proportion of the minority cell type (U2OS), on which performance is worse, and a slightly lower proportion of the majority cell type (HUVEC), on which performance is better. In this sense, the mixed split result of 39.8% is a more accurate reflection of the ID performance on this dataset, and the results in Table 7 therefore *understate* the magnitude of the distribution shift. We use the results in Table 7 for Section 5 in the main text for consistency and to make it easier for future comparisons, since those results do not require training a separate model on the mixed split.

Additional baseline methods. We also trained models with CORAL, IRM, and group DRO, treating each experiment as a domain, and using the same model hyperparameters as ERM. However, the models trained using these methods all performed poorly compared to the ERM model (Table 7). One complication with these methods is that the experiments in the training set comprise different cell types, as mentioned above; this heterogeneity can pose a challenge to methods that treat each domain equivalently.

Discussion. An important observation about batch effects in biological experiments: it is often the case that batch effects are mediated via biological mechanisms. For example, an increase in cellular media concentration may lead to cell growth and proliferation, while the upregulation of proliferation genes will do the same. Thus the “nuisance” factors associated with batch effects are often correlated with the biological signal we are attempting to observe, and cannot be disentangled from the biological factors that explain the data. Correction algorithms should take account of such trade-offs and attempt to optimize for both correction and signal preservation.

Table 7: Baseline results on RxRx1-WILDS. Parentheses show standard deviation across 3 replicates.

Algorithm	Validation (OOD) accuracy	Test (ID) accuracy	Test (OOD) accuracy
ERM	19.4 (0.2)	35.9 (0.4)	29.9 (0.4)
CORAL	18.5 (0.4)	34.0 (0.3)	28.4 (0.3)
IRM	5.6 (0.4)	9.9 (1.4)	8.2 (1.1)
Group DRO	15.2 (0.1)	28.1 (0.3)	23.0 (0.3)

Table 8: Performance drops for ERM models on RxRx1-WILDS. In the standard split, we train on data from 33 experiments (1 site per experiment) and test on 14 different experiments (2 sites per experiment). In the mixed split, we replace 14 of the training experiments with 1 site from each of the test experiments, which keeps the training set size the same, but halves the test set size. Parentheses show standard deviation across 3 replicates.

	Algorithm	Test (OOD) accuracy
Standard split (train on ID examples)	ERM	29.9 (0.4)
Mixed split (train on ID + OOD examples)	ERM	39.8 (0.2)

H.3.3. BROADER CONTEXT

As previously mentioned, high-throughput screening techniques are used broadly across many areas of biology, and therefore batch effects are a common problem in fields such as genomics, transcriptomics, proteomics, metabolomics, etc., so a particular solution in one such area may prove to be applicable in many areas of biology (Goh et al., 2017).

There are other datasets that are used in studying batch effects. The one most comparable to RxRx1-WILDS is the BBBC021 dataset (Ljosa et al., 2012), which contains 13,200 3-channel fluorescent microscopy images of MCF7 cells acquired across 10 experimental batches. A subset of 103 treatments from 38 drug compounds belonging to 12 known mechanism of action (MoA) groups was first studied in (Ando et al., 2017), and has been the subject of subsequent studies (Caicedo et al., 2018; Godinez et al., 2018; Tabak et al., 2020). Note that this dataset differs dramatically from RxRx1, in that there are fewer images, treatments, batches, and cell types, and each batch contains only a small subset of the total treatments.

H.3.4. ADDITIONAL DETAILS

Data processing. RxRx1-WILDS contains two non-overlapping 256×256 fields of view per well. Therefore, there could be as many as 125,664 images in the dataset ($= 51 \text{ experiments} \times 4 \text{ plates/experiment} \times 308 \text{ wells/plate} \times 2 \text{ images/well}$). 154 images were removed based on data quality, leaving a total dataset of 125,510 images.

Modifications to the original dataset. The underlying raw dataset consists of 2048×2048 pixel, 6 channel, 16bpp images. To fit within the constraints of the WILDS benchmark, images for RxRx1-WILDS were first downsampled to 1024×1024 and 8bpp, cropped to the center 256×256

pixels, and only the first three channels (nuclei, endoplasmic reticuli, actin) were retained. The original RxRx1 dataset, available at rxrx.ai and described in Taylor et al. (2019), provides 512×512 center crops of the downsampled images with all 6 channels retained.

The original RxRx1 dataset was also used in a NeurIPS 2019 competition hosted on Kaggle. The validation (OOD) and test (OOD) splits in RXRX1-WILDS correspond to the public and private test sets from the Kaggle competition. The original RxRx1 dataset did not have an additional test (ID) split, and thus the original training split had both sites 1 and 2, for a total of 81,442 images. The Kaggle competition also aggregated predictions from both sites to form a single prediction per well, whereas in RXRX1-WILDS, we treat each site separately.

As described in Section H.3.1, each plate in both the training and test sets contains the same 31 control conditions (one untreated well, and 30 positive control siRNAs). The Kaggle competition provided the labels for these control conditions in the test set, expecting that competitors would use them for various domain alignment techniques such as CORAL. However, these labels were instead used by the top competitors to bootstrap pseudo-labeling techniques. For RxRx1-WILDS, for consistency with the other datasets and the typical domain generalization setting, we have opted not to release these control test labels for training.

The poor performance reported here on RxRx1-WILDS may seem surprising in light of the fact that the top finishers of the Kaggle competition achieved near perfect accuracy on the test (OOD) set. This difference is due to a number of factors, including:

1. Adjustments made to the original RxRx1 dataset for RxRx1-WILDS, as detailed in this subsection.

2. Differences in the network architectures used. To make training on RxRx1-WILDS more accessible, we used a less compute-intensive architecture than typical in the competition.
3. Differences in training techniques used like pseudo-labeling (using the test control labels, as described above) and batch-level dataset augmentations or ensembling.
4. Differences in the way accuracy is measured. In the Kaggle competition, accuracy was measured for each well, meaning site-level predictions were aggregated to well-level predictions, and only for treatment classes, whereas in RxRx1-WILDS, for convenience, accuracy is measured at each site and for both treatment and control classes.
5. The use of post-prediction methods like linear sum assignment that exploited the particular structure of the experiments in the RxRx1 dataset, as described under Potential Leverage in Section H.3.1.

H.4. OGB-MOLPCBA

Accurate prediction of the biochemical properties of small molecules can significantly accelerate drug discovery by reducing the need for expensive lab experiments (Shoichet, 2004; Hughes et al., 2011). However, the experimental data available for training such models is limited compared to the extremely diverse and combinatorially large universe of candidate molecules that we would want to make predictions on (Bohacek et al., 1996; Sterling & Irwin, 2015; Lyu et al., 2019; McCloskey et al., 2020). This means that models need to generalize to out-of-distribution molecules that are structurally different from those seen in the training set.

We study this issue through the OGB-MOLPCBA dataset, which is directly adopted from the Open Graph Benchmark (Hu et al., 2020b) and originally curated by MoleculeNet (Wu et al., 2018).

Train				Test
Scaffold 11	Scaffold 32	Scaffold 321	Scaffold 4413	Scaffold 54113
 (1,0,?,0,?,...)	 (?,0,0,0,?,...)	 (0,1,1,0,0,...)	 (?,0,0,0,?,...)	 (0,?,1,0,0,...)
 (?,0,0,0,?,...)	 (?,0,?,1,0,..)	 (?,0,0,0,1,...)	 (1,1,0,1,0,...)	...
				Scaffold 65912  (0,1,0,0,0,...)

Figure 15: The OGB-MOLPCBA dataset comprises molecules with many different structural scaffolds. The goal is to predict biochemical assay results in molecules with scaffolds that are not in the training set. Here, we show sample molecules from each scaffold, together with target labels: each molecule is associated with 128 binary labels and ‘?’ indicates that the label is not provided for the molecule.

H.4.1. SETUP

Problem setting. We consider the domain generalization setting, where the domains are molecular scaffolds, and our goal is to learn models that generalize to structurally-distinct molecules with scaffolds that are not in the training set (Figure 15). This is a multi-task classification problem: for each molecule, we predict the presence or absence of 128 different kinds of biological activities, such as binding to a particular enzyme. In addition, we cluster the molecules into different scaffold groups according to their two-dimensional structure, and annotate each molecule with the scaffold group that it belongs to. Concretely, the input x is a molecular graph, the label y is a 128-dimensional binary vector where each component corresponds to a biochemical assay result, and the domain d specifies the scaffold. Not all biological activities are measured for each molecule, so y can have missing values.

Data. OGB-MOLPCBA contains more than 400K small molecules with 128 kinds of prediction labels. Each small molecule is represented as a graph, where the nodes are atoms and the edges are chemical bonds. The molecules are pre-processed using RDKIT (Landrum et al., 2006). Input node features are 9-dimensional, including atomic number, chirality, whether the atom is in the ring. Input edge features are 3-dimensional, including bond type and bond stereochemistry.

We split the dataset by scaffold structure. This *scaffold split* (Wu et al., 2018) is also used in the Open Graph Benchmark (Hu et al., 2020b). By attempting to separate structurally different molecules into different subsets, it provides a realistic estimate of model performance in prospective experimental settings. We assign the largest scaffolds to the training set to make it easier for algorithms to leverage scaffold information, and the smallest scaffolds to the test set to ensure that it is maximally diverse in scaffold structure:

1. **Training:** The largest 44,930 scaffolds, with an average of 7.8 molecules per scaffold.
2. **Validation (OOD):** The next largest 31,361 scaffolds, with an average of 1.4 molecules per scaffold.
3. **Test (OOD):** The smallest 43,793 scaffolds, which are all singletons.

Evaluation. We evaluate models by their average Average Precision (AP) across tasks (i.e., we compute the average precision for each task separately, and then average those scores), following Hu et al. (2020b). This accounts for the extremely skewed class balance in OGB-MOLPCBA (only 1.4% of data is positive). Not all labels are available for each molecule; when calculating the AP for each task, we only consider the labeled molecules for the task.

Potential leverage. We provide the scaffold grouping of molecules for training algorithms to leverage. Finding generalizable representations of molecules across different scaffold groups is useful for models to make accurate extrapolation on unseen scaffold groups. In fact, very recent work (Jin et al., 2020) has leveraged scaffold information of molecules to improve the extrapolation performance of molecular property predictors.

One notable characteristic of the scaffold group is that the size of each group is rather small; on the training split, each scaffold contains only 7.8 molecules on average. This also results in many scaffold groups: 44,930 groups in the training split. In Appendix H.4.4, we show that these scaffold groups are well-behaved in the sense that the train/validation/test splits contain similar ratios of positive labels as well as missing labels.

H.4.2. BASELINE RESULTS

Model. For all experiments, we use Graph Isomorphism Networks (GIN) (Xu et al., 2018) combined with virtual nodes (Gilmer et al., 2017), as this is currently the model with the highest performance in the Open Graph Benchmark (Hu et al., 2020b). We follow the same hyperparameters as in the Open Graph Benchmark: 5 GNN layers with a dimensionality of 300; the Adam optimizer (Kingma & Ba, 2015) with a learning rate of 0.001; and training for 100 epochs with early stopping. For each of the baseline algorithms (ERM, CORAL, IRM, and Group DRO), we separately tune the dropout rate from {0, 0.5}; in addition, for CORAL and IRM, we tune the penalty weight as in Appendix G.

ERM results and performance drops. We first compare the generalization performance of ERM on the scaffold split against the conventional mixed split, in which the entire molecules are randomly split into train/validation/test sets with the same split ratio as the scaffold split (i.e., 80/10/10). Results are in Table 10. The test performance of ERM drops by 7.2 points AP when the scaffold split is used, suggesting that the scaffold split is indeed harder than the random split.

Additional baseline methods. Table 9 also shows that ERM performs better than CORAL, IRM, and Group DRO, all of which use scaffolds as the domains. For CORAL and IRM, we find that smaller penalties give better generalization performance, as larger penalty terms make the training insufficient. We use the 0.1 penalty for CORAL and $\lambda = 1$ for IRM.

The primary issue with these existing methods is that they make the model significantly underfit the training data even when dropout is turned off. For instance, the training AP of CORAL and IRM is 20.0% and 15.9%, respectively, which

are both lower than the 36.1% that ERM obtains even with 0.5 dropout. Also, these methods are primarily designed for the case when each group contains a decent number of examples, which is not the case for the OGB-MOLPCBA dataset.

H.4.3. BROADER CONTEXT

Because of the very nature of discovering *new* molecules, out-of-distribution prediction is prevalent in nearly all applications of machine learning to chemistry domains. Beyond drug discovery, a variety of tasks and their associated datasets have been proposed for molecules of different sizes.

For small organic molecules, the scaffold split has been widely adopted to stress-test models’ capability for out-of-distribution generalization. While OGB-MOLPCBA primarily focuses on predicting biophysical activity (e.g., protein binding), other datasets in MoleculeNet (Wu et al., 2018) include prediction of quantum mechanical properties (e.g., HOMO/LUMO), physical chemistry properties (e.g., water solubility), and physiological properties (e.g., toxicity prediction (Attene-Ramos et al., 2013)).

Besides small molecules, it is also of interest to apply machine learning over larger molecules such as catalysts and proteins. In the domain of catalysis, using machine learning to approximate expensive quantum chemistry simulation has gotten attention. The OC20 dataset has been recently introduced, containing 200+ million samples from quantum chemistry simulations relevant to the discovery of new catalysts for renewable energy storage and other energy applications (Becke, 2014; Chanusso et al., 2020; Zitnick et al., 2020). The OC20 dataset explicitly provides test sets with qualitatively different materials. In the domain of proteins, the recent trend is to use machine learning to predict 3D structure of proteins given their amino acid sequence information. This is known as the protein folding problem, and has sometimes been referred to as the Holy Grail of structural biology (Dill & MacCallum, 2012). CASP is a bi-annual competition to benchmark the progress of protein folding (Moult et al., 1995), and it evaluates predictions made on proteins whose 3D structures are identified very recently, presenting a natural temporal distribution shift. Recently, the AlphaFold2 deep learning model obtained breakthrough performance on the CASP challenge (Jumper et al., 2020), demonstrating exciting avenues of machine learning for structural biology.

H.4.4. ADDITIONAL DETAILS

Data processing and scaffold split analysis. The OGB-MOLPCBA dataset contains 437,929 molecules annotated with 128 kinds of labels, each representing a bioassay curated in the PubChem database (Kim et al., 2016b). More details are provided in the MoleculeNet (Wu et al., 2018)

Table 9: Baseline results on OGB-MOLPCBA. Parentheses show standard deviation across 3 replicates.

Algorithm	Validation AP (%)	Test AP (%)
ERM	27.8 (0.1)	27.2 (0.3)
CORAL	18.4 (0.2)	17.9 (0.5)
IRM	15.8 (0.2)	15.6 (0.3)
Group DRO	23.1 (0.6)	22.4 (0.6)

Table 10: Out-of-distribution vs. in-distribution performance for ERM models on OGB-MOLPCBA. In the standard split, we train on molecules from some scaffolds and evaluate on molecules from different scaffolds, whereas in the mixed split, we randomly divide molecules into training and test sets without using scaffold information.

	Algorithm	Test AP (%)
Standard split (split by scaffolds)	ERM	27.2 (0.3)
Mixed split (split i.i.d.)	ERM	34.4 (0.9)

and the Open Graph Benchmark (Hu et al., 2020b), from which the dataset is adopted.

In WILDS, we additionally provide the scaffold information that training algorithms can leverage in order to improve model’s extrapolation capability. In Figure 16 (A), we plot the statistics of the scaffold groups in terms of their sizes. We see that the scaffold sizes are highly skewed. The training split contains the largest 44,930 scaffolds with 7.8 molecules per scaffold, the validation split contains the next largest 31,361 scaffolds with 1.4 molecules per scaffold, and the test split contains the smallest all-singleton 43,793 scaffolds. This implies that test molecules are maximally diverse in their structure, making it suitable to evaluate model’s performance across diverse scaffold domains. How does the above data split affect the distribution of target labels? In Figures 16 (B) and (C), we quantify whether the scaffold split creates distribution shift in the prediction target labels. We see that the label statistics remain almost across train/validation/test splits, suggesting that the main distribution shift comes from the difference in the input molecular graph structure.

H.5. GLOBALWHEAT-WILDS

Models for automated, high-throughput plant phenotyping—measuring the physical characteristics of plants and crops, such as wheat head density and counts—are important tools for crop breeding (Thorp et al., 2018; Reynolds et al., 2020) and agricultural field management (Shi et al., 2016). These models are typically trained on data collected in a limited number of regions, even for crops grown worldwide such as wheat (Madec et al., 2019; Xiong et al., 2019; Ubbens et al., 2020; Ayalew et al., 2020). However, there can be substantial variation between regions, due to differences in crop varieties, growing conditions, and data collection protocols. Prior work on wheat head detection has shown that

this variation can significantly degrade model performance on regions unseen during training (David et al., 2020).

We study this shift in an expanded version of the Global Wheat Head Dataset (David et al., 2020; 2021), a large set of wheat images collected from 12 countries around the world.

H.5.1. SETUP

Problem setting. We consider the domain generalization setting, where the goal is to learn models that generalize to images taken from new countries and acquisition sessions (Figure 17). The task is wheat head detection, which is a single-class object detection task. Concretely, the input x is an overhead outdoor image of wheat plants, and the label y is a set of bounding box coordinates that enclose the wheat heads (the spike at the top of the wheat plant containing grain), excluding the hair-like awns that may extend from the head. The main challenge of this task is that the wheat head instances are densely packed and may overlap each other. The domain d specifies an “acquisition session,” which corresponds to a specific location, time, and sensor for which a set of images were collected. While one of the goals is to generalize to unseen acquisition sessions, we aim to generalize to unseen locations in particular; the dataset split captures shift in location, with training and test sets comprising images from disjoint countries as discussed below.

Data. The dataset comprises 6,515 images containing 275,187 wheat heads, spanning 16 research institutes across 12 countries. These images were collected over 47 acquisition sessions, whose metadata and statistics are described in Table 11.

While many factors contribute to the variation in wheat appearance across acquisition sessions, substantial variation exists due to cross-location differences in wheat genotypes,

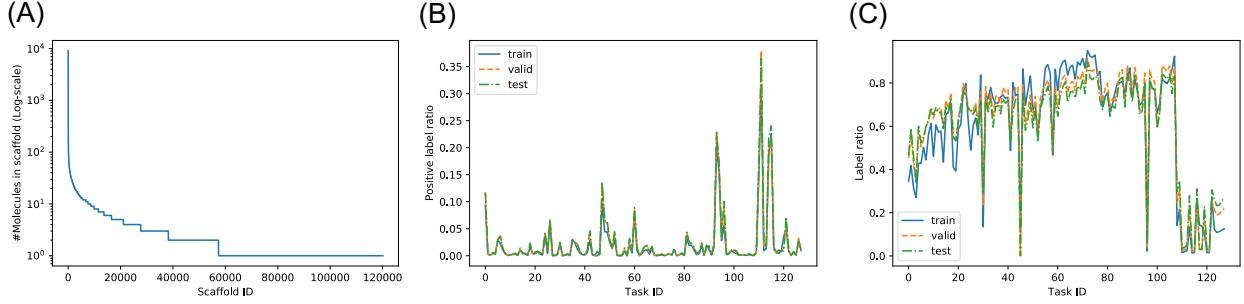


Figure 16: Analyses of scaffold groups in the OGB-MOLPCBA dataset. (A) shows the distribution of the scaffold sizes, (B) and (C) show how the ratios of positive molecules and labeled molecules for the 128 tasks vary across the train/validation /test splits.

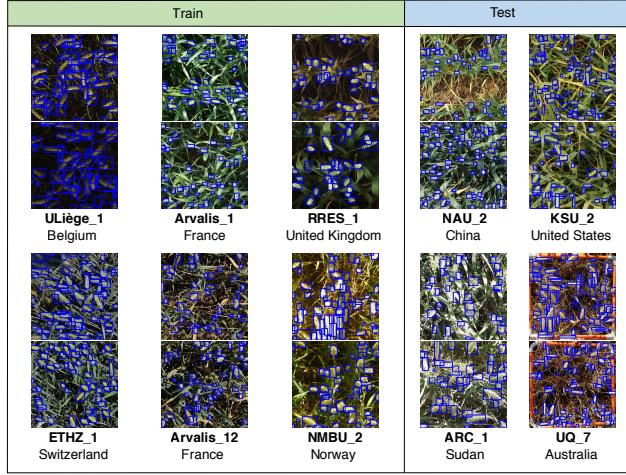


Figure 17: The GLOBALWHEAT-WILDS dataset consists of overhead images of wheat fields in 12 countries. The goal is to detect and predict the bounding boxes of wheat heads, where test images are taken from regions unseen during training time. In this figure, we show sample images with bounding boxes from various countries, organized by acquisition sessions; a set of images are taken in each acquisition session, each corresponding to a specific location, time, and sensor.

growing conditions (e.g., planting density), illumination protocols, and sensors. To this end, we split the dataset by country and study a shift in location. In particular, we assign disjoint continents to training and test splits in order to maximize the difference in wheat appearance:

- Training:** 18 domains from Europe ($\text{France} \times 13$, $\text{Norway} \times 2$, Switzerland, United Kingdom, Belgium) and containing 3,657 images and 163,690 wheat heads.
- Validation:** 11 domains from Australia ($\times 6$), Japan ($\times 3$), China, Canada containing 1,476 images and 44,347 wheat heads.
- Test:** 18 domains from North America ($\text{USA} \times 6$, Mexico ($\times 3$)), Asia (China $\times 2$, Japan $\times 1$), Sudan, Australia ($\times 5$) containing 1,287 images and 67,150 wheat heads.

Evaluation. We evaluate models by their average accuracy across acquisition sessions, where accuracy is measured at a fixed Intersection over Union (IoU) threshold of 0.5. We compute the accuracy as $\frac{TP}{TP+FN+FP}$, where TP is the number of true positives, which are ground-truth bounding boxes that can be matched with some predicted bounding box at IoU above the threshold; FN is the number of false negatives, which are ground-truth bounding boxes that cannot be matched as above; and FP is the number of false positives, which are predicted bounding boxes that cannot be matched with any ground-truth bounding box. We use accuracy rather than average precision, which is a common metric for object detection, because it was used in previous Global Wheat Challenges with the dataset (David et al., 2020; 2021). We use an IoU threshold of 0.5 because there is some uncertainty regarding the precise outline of wheat head instances due to the stem and awns extending from the head. We average the accuracy across acquisition sessions because the number of images significantly varies across acquisition sessions, from 30 to 747 images, and we use average performance rather than worst-case performance because the wheat images are more difficult for some acquisition sessions.

Potential leverage. We include images from 5 countries and 12 acquisition sessions in the training set as leverage, providing coverage over all growth stages, many different sensors, and different illumination conditions. As a result, some of the variation between training and test sets are captured to some (though often lesser) extent across the training domains; for example, illumination conditions vary across acquisition sessions within the training set due to differences in the time of day that images were acquired, but also vary between the training and testing domains due to differences in region latitude and time of year. In contrast, other variations are not captured within the training set. For example, wheat varieties used in Asia have different genotypes from the ones used in Europe and North America, which results in a different appearance of the wheat plants and wheat heads across splits. Likewise, wheat planting strategies and

Table 11: Acquisition sessions in GLOBALWHEAT-WILDS. Growth stages are abbreviated as, F: Filling, R: Ripening, PF: Post-flowering. Locations are abbreviated as, VLB: Villiers le Bâcle, VSC: Villers-Saint-Christophe. UTokyo_1 and UTokyo_2 are from the same location with different sensors and UTokyo_3 consists of image from a variety of farms in Hokkaido between 2016 and 2019.

Name	Owner	Country	Site	Date	Sensor	Stage	# Images	# Heads
Ethz_1	ETHZ	Switzerland	Eschikon	06/06/2018	Spidercam	F	747	49603
Rres_1	Rothamsted	UK	Rothamsted	13/07/2015	Gantry	F-R	432	19210
ULiège_1	Uliège	Belgium	Gembloux	28/07/2020	Cart	R	30	1847
NBUM_1	NMBU	Norway	NMBU	24/07/2020	Cart	F	82	7345
NBUM_2	NMBU	Norway	NMBU	07/08/2020	Cart	R	98	5211
Arvalis_1	Arvalis	France	Gréoux	02/06/2018	Handheld	PF	66	2935
Arvalis_2	Arvalis	France	Gréoux	16/06/2018	Handheld	F	401	21003
Arvalis_3	Arvalis	France	Gréoux	07/2018	Handheld	F-R	588	21893
Arvalis_4	Arvalis	France	Gréoux	27/05/2019	Handheld	F	204	4270
Arvalis_5	Arvalis	France	VLB*	06/06/2019	Handheld	F	448	8180
Arvalis_6	Arvalis	France	VSC*	26/06/2019	Handheld	F-R	160	8698
Arvalis_7	Arvalis	France	VLB*	06/2019	Handheld	F-R	24	1247
Arvalis_8	Arvalis	France	VLB*	06/2019	Handheld	F-R	20	1062
Arvalis_9	Arvalis	France	VLB*	06/2020	Handheld	R	32	1894
Arvalis_10	Arvalis	France	Mons	10/06/2020	Handheld	F	60	1563
Arvalis_11	Arvalis	France	VLB*	18/06/2020	Handheld	F	60	2818
Arvalis_12	Arvalis	France	Gréoux	15/06/2020	Handheld	F	29	1277
Inrae_1	INRAe	France	Toulouse	28/05/2019	Handheld	F-R	176	3634
Usask_1	USaskatchewan	Canada	Saskatoon	06/06/2018	Tractor	F-R	200	5985
KSU_1	KansasStateU	US	Manhattan, KS	19/05/2016	Tractor	PF	100	6435
KSU_2	KansasStateU	US	Manhattan, KS	12/05/2017	Tractor	PF	100	5302
KSU_3	KansasStateU	US	Manhattan, KS	25/05/2017	Tractor	F	95	5217
KSU_4	KansasStateU	US	Manhattan, KS	25/05/2017	Tractor	R	60	3285
Terraref_1	TERRA-REF	US	Maricopa	02/04/2020	Gantry	R	144	3360
Terraref_2	TERRA-REF	US	Maricopa	20/03/2020	Gantry	F	106	1274
CIMMYT_1	CIMMYT	Mexico	Ciudad Obregon	24/03/2020	Cart	PF	69	2843
CIMMYT_2	CIMMYT	Mexico	Ciudad Obregon	19/03/2020	Cart	PF	77	2771
CIMMYT_3	CIMMYT	Mexico	Ciudad Obregon	23/03/2020	Cart	PF	60	1561
Utokyo_1	UTokyo	Japan	NARO-Tsukuba	22/05/2018	Cart	R	538	14185
Utokyo_2	UTokyo	Japan	NARO-Tsukuba	22/05/2018	Cart	R	456	13010
Utokyo_3	UTokyo	Japan	NARO-Hokkaido	2016-19	Handheld	multiple	120	3085
Ukyoto_1	UKyoto	Japan	Kyoto	30/04/2020	Handheld	PF	60	2670
NAU_1	NAU	China	Baima	n/a	Handheld	PF	20	1240
NAU_2	NAU	China	Baima	02/05/2020	Cart	PF	100	4918
NAU_3	NAU	China	Baima	09/05/2020	Cart	F	100	4596
UQ_1	UQueensland	Australia	Gatton	12/08/2015	Tractor	PF	22	640
UQ_2	UQueensland	Australia	Gatton	08/09/2015	Tractor	PF	16	39
UQ_3	UQueensland	Australia	Gatton	15/09/2015	Tractor	F	14	297
UQ_4	UQueensland	Australia	Gatton	01/10/2015	Tractor	F	30	1039
UQ_5	UQueensland	Australia	Gatton	09/10/2015	Tractor	F-R	30	3680
UQ_6	UQueensland	Australia	Gatton	14/10/2015	Tractor	F-R	30	1147
UQ_7	UQueensland	Australia	Gatton	06/10/2020	Handheld	R	17	1335
UQ_8	UQueensland	Australia	McAllister	09/10/2020	Handheld	R	41	4835
UQ_9	UQueensland	Australia	Brookstead	16/10/2020	Handheld	F-R	33	2886
UQ_10	UQueensland	Australia	Gatton	22/09/2020	Handheld	F-R	53	8629
UQ_11	UQueensland	Australia	Gatton	31/08/2020	Handheld	PF	42	4345
ARC_1	ARC	Sudan	Wad Medani	03/2021	Handheld	F	30	888

growing conditions vary between regions and contribute to differences between the training and testing set, e.g. higher planting density may result in more closely packed plants and more occlusion between wheat head instances.

H.5.2. BASELINE RESULTS

Model. For all experiments, we use the Faster-RCNN detection model (Ren et al., 2015), which has been successfully applied to the wheat head localization problem

(David et al., 2020; Madec et al., 2019). To train, we fine-tune a model pre-trained with ImageNet, using a batch size of 4, a learning rate of 10^{-4} , and weight decay of 10^{-4} for 10 epochs with early stopping. The hyperparameters were chosen from a grid search over learning rates $\{10^{-6}, 10^{-5}, 10^{-4}\}$ and weight decays $\{0, 10^{-4}, 10^{-3}\}$. We report results aggregated over 3 random seeds.

ERM results and performance drops. To demonstrate a substantial performance drop due to the distribution shift,

Table 12: Baseline results on GLOBALWHEAT-WILDS. Parentheses show standard deviation across 3 replicates.

Algorithm	Validation acc (%)	Test acc (%)
ERM	65.5 (0.8)	49.2 (1.5)
Group DRO	62.9 (0.7)	46.1 (1.6)

Table 13: Out-of-distribution vs in-distribution performance for ERM models on GLOBALWHEAT-WILDS. The ID split corresponds to a model trained on 33% of the test set, while the OOD split is trained on the training dataset defined by the setup.

	Algorithm	Test acc (%)
Official	ERM	48.4 (1.8)
Fixed-test	ERM	64.8 (0.4)

we show that an ERM model trained on the official split performs significantly worse than an oracle ERM model trained on the target distribution. Concretely, we train ERM models on the official split as well as a fixed-test split, whose training data is 33% of each domain from the official test set, containing 425 images. We evaluate both models on the remaining 67% of the official test set, and we observe a substantial performance gap of 16.4%.

Additional baseline methods. We also trained models with group DRO, treating each acquisition session as a domain, and using the same model hyperparameters as ERM. However, the group DRO models perform poorly compared to the ERM model as reported in Table 12. We leave the investigation of CORAL and IRM for future work because it is not straightforward to apply these algorithms to detection tasks.

Discussion. Our baseline models were trained without any data augmentation, in contrast to baselines reported in the original dataset (David et al., 2020). Data augmentation could reduce the performance gap and warrants further investigation in future work, although David et al. (2020) observed performance gaps on models trained with data augmentation in the original version of the dataset. More generally, techniques for improving the general performance of the object detection model, rather than the domain gap specifically, are also a promising avenue for improving model performance on test domains. Lastly, while we evaluated models by their average performance across acquisition sessions, we noticed a large variability in performance across domains. It is possible that some domains are more challenging or suffer from larger performance drops than others, and characterizing and mitigating these variations is interesting future work.

H.5.3. BROADER CONTEXT

Wheat head localization, while being an important operational trait for wheat breeders and farmers, is not the only deep learning application in plant phenotyping that suffers from lack of generalization. Other architectural traits such as plant segmentation (Kuznichov et al., 2019; Sadeghi-Tehran et al., 2017), plant and plant organ detection (Fan et al., 2018; Madec et al., 2019), leaves and organ disease classification (Fuentes et al., 2017; Toda & Okura, 2019; Shakoor et al., 2017), and biomass and yield prediction (Aich et al., 2018; Drecer et al., 2019) would also benefit from plant phenotyping models that generalize to new deployments. In many of these applications, field images exhibit variations in illumination and sensors, and there has been work on mitigating biases across sensors (Ayalew et al., 2020; Gogoll et al., 2020). Finally, developing models that generalize across plant species would benefit the breeding and growing of specialized crops that are presently under-represented in plant phenotyping research worldwide (Ward & Moghadam, 2020). We hope that GLOBALWHEAT-WILDS can foster the development of general solutions to plant phenotyping problems, increase collaboration between plant scientists and computer vision scientists, and encourage the development of new multi-domain plant datasets to ensure that plant phenotyping results are generalizable to all crop growing regions of the world.

H.5.4. ADDITIONAL DETAILS

Modifications to the original dataset. The dataset is taken directly from the 2021 Global Wheat Challenge (David et al., 2021), which is an expanded version of the 2020 Global Wheat Challenge dataset (David et al., 2020). We note that images from North America were in the training set for the 2020 version, but have been moved to the validation set for GLOBALWHEAT-WILDS.

H.6. CIVILCOMMENTS-WILDS

Automatic review of user-generated text is an important tool for moderating the sheer volume of text written on the Internet. We focus here on the task of detecting toxic comments. Prior work has shown that toxicity classifiers can pick up on biases in the training data and spuriously associate toxicity with the mention of certain demographics (Park et al., 2018; Dixon et al., 2018). These types of spurious correlations can significantly degrade model performance on particular subpopulations (Sagawa et al., 2020a).

We study this issue through a modified variant of the CivilComments dataset (Borkan et al., 2019b).

Toxic	Comment Text	Male	Female	LGBTQ	White	Black	... Christian	
0	I applaud your father. He was a good man! We need more like him.	1	0	0	0	0	...	0
0	As a Christian, I will not be patronizing any of those businesses.	0	0	0	0	0	...	1
0	What do Black and LGBTQ people have to do with bicycle licensing?	0	0	1	0	1	...	0
0	Government agencies track down foreign baddies and protect law-abiding white citizens. How many shows does that describe?	0	0	0	1	0	...	0
1	Maybe you should learn to write a coherent sentence so we can understand WTF your point is.	0	0	0	0	0	...	0

Figure 18: The CIVILCOMMENTS-WILDS dataset involves classifying the toxicity of online comments. The goal is to learn models that avoid spuriously associating mentions of demographic identities (like male, female, etc.) with toxicity due to biases in the training data.

H.6.1. SETUP

Problem setting. We cast CIVILCOMMENTS-WILDS as a subpopulation shift problem, where the subpopulations correspond to different demographic identities, and our goal is to do well on all subpopulations (and not just on average across these subpopulations). Specifically, we focus on mitigating biases with respect to comments that mention particular demographic identities, and not comments written by members of those demographic identities; we discuss this distinction in the broader context section below.

The task is a binary classification task of determining if a comment is toxic. Concretely, the input x is a comment on an online article (comprising one or more sentences of text) and the label y is whether it is rated toxic or not. In CIVILCOMMENTS-WILDS, unlike in most of the other datasets we consider, the domain annotation d is a multi-dimensional binary vector, with the 8 dimensions corresponding to whether the comment mentions each of the 8 demographic identities *male*, *female*, *LGBTQ*, *Christian*, *Muslim*, *other religions*, *Black*, and *White*.

Data. CIVILCOMMENTS-WILDS comprises 450,000 comments, each annotated for toxicity and demographic mentions by multiple crowdworkers. We model toxicity classification as a binary task. Toxicity labels were obtained in the original dataset via crowdsourcing and majority vote, with each comment being reviewed by at least 10 crowdworkers. Annotations of demographic mentions were similarly obtained through crowdsourcing and majority vote.

Each comment was originally made on some online article. We randomly partitioned these articles into disjoint training, validation, and test splits, and then formed the corresponding datasets by taking all comments on the articles in those splits. This gives the following splits:

1. **Training:** 269,038 comments.
2. **Validation:** 45,180 comments.

3. Test: 133,782 comments.

Evaluation. We evaluate a model by its worst-group accuracy, i.e., its lowest accuracy over groups of the test data that we define below.

As mentioned above, toxicity classifiers can spuriously latch onto mentions of particular demographic identities, resulting in a biased tendency to flag comments that innocuously mention certain demographic groups as toxic (Park et al., 2018; Dixon et al., 2018). To measure the extent of this bias, we define subpopulations based on whether they mention a particular demographic identity, compute the sensitivity (a.k.a. recall, or true positive rate) and specificity (a.k.a. true negative rate) of the classifier on each subpopulation, and then report the worst of these two metrics over all subpopulations of interest. This is equivalent to further dividing each subpopulation into two groups according to the label, and then computing the accuracy on each of these two groups.

Specifically, for each of the 8 identities we study (e.g., “male”), we form 2 groups based on the toxicity label (e.g., one group of comments that mention the male gender and are toxic, and another group that mentions the male gender and are not toxic), for a total of 16 groups. These groups overlap (a comment might mention multiple identities) and are not a complete partition (a comment might not mention any identity).

We then measure a model’s performance by its worst-group accuracy, i.e., its lowest accuracy over these 16 groups. A high worst-group accuracy (relative to average accuracy) implies that the model is not spuriously associating a demographic identity with toxicity. We can view this subpopulation shift problem as testing on multiple test distributions (corresponding to different subsets of the test set, based on demographic identities and the label) and reporting the worst performance over these different test distributions. In Appendix H.6.4, we further discuss the motivation for this choice of evaluation metric as well as its limitations.

As variability in performance over replicates can be high due to the small sizes of some demographic groups (Table 16), we report results averaged over 5 random seeds, instead of the 3 seeds that we use for most other datasets.

Potential leverage. Since demographic identity annotations are provided at training time, we have an i.i.d. dataset available at training time for each of the test distributions of interest (corresponding to each group). Moreover, even though demographic identity annotations are unavailable at test time, they are relatively straightforward to predict.

H.6.2. BASELINE RESULTS

Model. For all experiments, we fine-tuned DistilBERT-base-uncased models (Sanh et al., 2019), using the imple-

Table 14: Baseline results on CIVILCOMMENTS-WILDS. The reweighted (label) algorithm samples equally from the positive and negative class; the group DRO (label) algorithm additionally weights these classes so as to minimize the maximum of the average positive training loss and average negative training loss. Similarly, the reweighted (label \times Black) and group DRO (label \times Black) algorithms sample equally from the four groups corresponding to all combinations of class and whether there is a mention of Black identity. The CORAL and IRM algorithms extend the reweighted algorithm by adding their respective penalty terms, so they also sample equally from each group. We show standard deviation across 5 random seeds in parentheses.

Algorithm	Avg val acc	Worst-group val acc	Avg test acc	Worst-group test acc
ERM	92.3 (0.2)	50.5 (1.9)	92.2 (0.1)	56.0 (3.6)
Reweighted (label)	90.1 (0.4)	65.9 (1.8)	89.8 (0.4)	69.2 (0.9)
Group DRO (label)	90.4 (0.4)	65.0 (3.8)	90.2 (0.3)	69.1 (1.8)
Reweighted (label \times Black)	89.5 (0.6)	66.6 (1.5)	89.2 (0.6)	66.2 (1.2)
CORAL (label \times Black)	88.9 (0.6)	64.7 (1.4)	88.7 (0.5)	65.6 (1.3)
IRM (label \times Black)	89.0 (0.7)	65.9 (2.8)	88.8 (0.7)	66.3 (2.1)
Group DRO (label \times Black)	90.1 (0.4)	67.7 (1.8)	89.9 (0.5)	70.0 (2.0)

Table 15: Accuracies on each subpopulation in CIVILCOMMENTS-WILDS, averaged over models trained by group DRO (label).

Demographic	Test accuracy on non-toxic comments	Test accuracy on toxic comments
Male	88.4 (0.7)	75.1 (2.1)
Female	90.0 (0.6)	73.7 (1.5)
LGBTQ	76.0 (3.6)	73.7 (4.0)
Christian	92.6 (0.6)	69.2 (2.0)
Muslim	80.7 (1.9)	72.1 (2.6)
Other religions	87.4 (0.9)	72.0 (2.5)
Black	72.2 (2.3)	79.6 (2.2)
White	73.4 (1.4)	78.8 (1.7)

mentation from [Wolf et al. \(2019\)](#) and with the following hyperparameter settings: batch size 16; learning rate 10^{-5} using the AdamW optimizer ([Loshchilov & Hutter, 2019](#)) for 5 epochs with early stopping; an L_2 -regularization strength of 10^{-2} ; and a maximum number of tokens of 300, since 99.95% of the input examples had ≤ 300 tokens. The learning rate was chosen through a grid search over $\{10^{-6}, 2 \times 10^{-6}, 10^{-5}, 2 \times 10^{-5}\}$, and all other hyperparameters were simply set to standard/default values.

ERM results and performance drops. The ERM model does well on average, with 92.2% average accuracy (Table 14). However, it does poorly on some subpopulations, e.g., with 57.4% accuracy on toxic comments that mention *other religions*. Overall, accuracy on toxic comments (which are a minority of the dataset) was lower than accuracy on non-toxic comments, so we also trained a reweighted model that balanced toxic and non-toxic comments by up-sampling the toxic comments. This reweighted model had a slightly worse average accuracy of 89.8% and a better worst-group accuracy of 69.2%, (Table 14, Reweighted (label)), but a significant gap remains between average and worst-group accuracy.

Additional baseline methods. The CORAL, IRM, and group DRO baselines involve partitioning the training data into disjoint domains. We study the following partitions, corresponding to different rows in Table 14:

1. *Label*: 2 domains, 1 for each class.
2. *Label \times Black*: 4 domains, 1 for each combination of class and *Black*.

On the *Label* partition, we used Group DRO to train a model that seeks to balance the losses on the positive and negative examples. This performs similarly to the standard reweighted models described above (Table 14, Group DRO (label)). We found that the worst-performing demographic for non-toxic comments was the Black demographic (Table 15), which motivated the *Label \times Black* partition. There, we used CORAL, IRM, and Group DRO to train models. However, these models did not perform significantly better (Table 14, label \times Black). While there were slight improvements on the Black groups, accuracy degraded on some other groups like non-toxic LGBTQ comments.

We note that our implementations of CORAL and IRM are built on top of the standard reweighting algorithm, i.e., they

sample equally from each group. As these two algorithms perform similarly to reweighting, it indicates that the additional penalty term is not significantly affecting performance. Indeed, our grid search for the penalty weights selected the lowest value of the penalties ($\lambda = 10.0$ for CORAL and $\lambda = 1.0$ for IRM).

Discussion. Adapting the baseline methods to handle multiple overlapping groups, which were not studied in their original settings, could be a potential approach to improving accuracy on this task. Another potential approach is using baselining to account for different groups having different intrinsic levels of difficulty (Oren et al., 2019). For example, comments mentioning different demographic groups might differ in terms of how subjective classifying them is. Others have also explored specialized data augmentation techniques for mitigating demographic biases in toxicity classifiers (Zhao et al., 2018).

Adragna et al. (2020) recently used a simplified variant of the CivilComments dataset, with artificially-constructed training and test environments, to show a proof-of-concept that IRM can improve performance on minority groups. Methods such as IRM rely heavily on the choice of groups/domains/environments; investigating the effect of different choices would be a useful direction for future work.

Toxicity classification is one application where human moderators can work together with an ML model to handle examples that the model is unsure about. However, Jones et al. (2021) found that using selective classifiers—where the model is allowed to abstain if it is unsure—can actually further worsen performance on minority subpopulations. This suggests that in addition to having low accuracy on minority subpopulations, standard models can be poorly calibrated on them.

Another important consideration for toxicity detection in practice is shifts over time, as online discourse changes quickly, and what is seen as toxic today might not have even appeared in the dataset from a few months ago. We do not study this distribution shift in this work. One limitation of the CIVILCOMMENTS-WILDS dataset is that it is fixed to a relatively short period in time, with most comments being written in the span of a year; this makes it harder to use as a dataset for studying temporal shifts.

Finally, we note that collecting “ground truth” human annotation of toxicity is itself a subjective and challenging process; recent work has studied ways of making it less biased and more efficient (Sap et al., 2019; Han & Tsvetkov, 2020).

H.6.3. BROADER CONTEXT

The CIVILCOMMENTS-WILDS dataset does not assume that user demographics are available; instead, it uses mentions of different demographic identities in the actual comment text. For example, we want models that do not associate comments that mention being Black with being toxic, regardless of whether a Black or non-Black person wrote the comment. This setting is particularly relevant when user demographics are unavailable, e.g., when considering anonymous online comments.

A related and important setting is subpopulation shifts with respect to user demographics (e.g., the demographics of the author of the comment, regardless of the content of the comment). Such demographic disparities have been widely documented in natural language and speech processing tasks (Hovy & Spruit, 2016), among other areas, and these disparities are instances of dataset biases that are common in many contemporary datasets (Gebru et al., 2018; Bender & Friedman, 2018). For example, NLP models have been shown to obtain worse performance on African-American Vernacular English compared to Standard American English on part-of-speech tagging (Jørgensen et al., 2015), dependency parsing (Blodgett et al., 2016), language identification (Blodgett & O’Connor, 2017), and auto-correct systems (Hashimoto et al., 2018). Similar disparities exist in speech, with state-of-the-art commercial systems obtaining higher word error rates on particular races (Koenecke et al., 2020) and dialects (Tatman, 2017).

These disparities are present not just in academic models, but in large-scale commercial systems that are already widely deployed, e.g., in speech-to-text systems from Amazon, Apple, Google, IBM, and Microsoft (Tatman, 2017; Koenecke et al., 2020) or language identification systems from IBM, Microsoft, and Twitter (Blodgett & O’Connor, 2017). Indeed, the original CivilComments dataset was developed by Google’s Conversation AI team, which is also behind a public toxicity classifier (Perspective API) that was developed in partnership with The New York Times (NYTimes, 2016).

H.6.4. ADDITIONAL DETAILS

Evaluation metrics. The evaluation metric used in the original competition was a complex weighted combination of various metrics, including subgroup AUCs for each demographic identity, and a new pinned AUC metric introduced by the original authors (Borkan et al., 2019b); conceptually, these metrics also measure the degree to which model accuracy is uniform across the different identities. After discussion with the original authors, we replace the composite metric with worst-group accuracy (i.e., worst TPR/FPR over identities) for simplicity. Measuring subgroup AUCs can be misleading in this context, because it assumes that the

classifier can set separate thresholds for different subgroups (Borkan et al., 2019b;a).

One downside is that measuring worst-group accuracy treats false positives and false negatives equally. In deployment systems, one might want to weight these differently, e.g., using cost-sensitive learning or by simply raising or lowering the classification threshold, especially since real data is highly imbalanced (with a lot more negatives than positives). One could also binarize the labels and identities differently: in this benchmark, we simply use majority voting from the annotators.

Perhaps more fundamentally, even if TPR and FPR were balanced across different identities, this need not imply unambiguously equitable performance, because different subpopulations might have different intrinsic levels of noise and difficulty. See Corbett-Davies & Goel (2018) for more discussion of this problem of infra-marginality.

In practice, models might also do poorly on intersections of groups (Kearns et al., 2018), e.g., on comments that mention multiple identities. Given the size of the dataset and comparative rarity of some identities and of toxic comments in general, accuracies on these intersections are difficult to estimate from this dataset. A potential avenue of future work is to develop methods for evaluating models on such subgroups, e.g., by generating data in particular groups through templates (Park et al., 2018; Ribeiro et al., 2020).

Data processing. The CIVILCOMMENTS-WILDS dataset comprises comments from a large set of articles from the Civil Comments platform, annotated for toxicity and demographic identities (Borkan et al., 2019b). We partitioned the articles into disjoint training, validation, and test splits, and then formed the corresponding datasets by taking all comments on the articles in those splits. In total, the training set comprised 269,038 comments (60% of the data); the validation set comprised 45,180 comments (10%); and the test set comprised 133,782 (30%).

Modifications to the original dataset. The original dataset⁵ also had a training and test split with disjoint articles. These splits are related to ours in the following way. Let the number of articles in the original test split be m . To form our validation split, we took m articles (sampled uniformly at random) from the original training split, and to form our test split, we took $2m$ articles (also sampled uniformly at random) from the original training split and added it to the existing test split. We added a fixed validation set to allow other researchers to be able to compare methods more consistently, and we tripled the size of the test set to allow for more accurate worst-group accuracy measurement.

⁵www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification/

Similarly, we combined some of the demographic identities in the original dataset to obtain larger groups (for which we could more accurately estimate accuracy). Specifically, we created an aggregate *LGBTQ* identity that combines the original *homosexual_gay_or_lesbian*, *bisexual*, *other_sexual_orientation*, *transgender*, and *other_gender* identities (e.g., it is 1 if any of those identities are 1), and an aggregate *other_religions* identity that combines the original *jewish*, *hindu*, *buddhist*, *atheist*, and *other_religion* identities. We also omitted the *psychiatric_or_mental_illness* identity, which was evaluated in the original Kaggle competition, because of a lack of sufficient data for accurate estimation; but we note that baseline group accuracies for that identity seemed higher than for the other groups, so it is unlikely to factor into worst-group accuracy. In our new split, each identity we evaluate on (*male*, *female*, *LGBTQ*, *Christian*, *Muslim*, *other_religions*, *Black*, and *White*) has at least 500 positive and 500 negative examples. In Table 16 we show the sizes of each subpopulation in the test set; the training and validation sets follow similar proportions.

For convenience, we also add an *identity_any* identity; this combines all of the identities in the original dataset, including *psychiatric_or_mental_illness* and related identities.

Additional baseline results. We also trained a group DRO model using $2^9 = 512$ domains, 1 for each combination of class and the 8 identities. This model performed similarly to the other group DRO models.

We note that the relatively small size of some of these subpopulations makes it infeasible to estimate how well a model could do on each subpopulation (corresponding to demographic identity) if it were trained on just that subpopulation. For example, Black comments comprise only <4% of the training data, and training just on those Black comments is insufficient to achieve high in-distribution accuracy.

Additional data sources. All of the data, including the data with identity annotations that we use and the data with just label annotations, are also annotated for additional toxicity subtype attributes, specifically *severe_toxicity*, *obscene*, *threat*, *insult*, *identity_attack*, and *sexual_explicit*. These annotations can be used to train models that are more aware of the different ways that a comment can be toxic; in particular, using the *identity_attack* attribute to learn which comments are toxic because of the use of identities might help the model learn how to avoid spurious associations between toxicity and identity. These additional annotations are included in the metadata provided through the WILDS package.

The original CivilComments dataset (Borkan et al., 2019b) also contains $\approx 1.5M$ training examples that have toxicity (label) annotations but not identity (group) annotations. For

Table 16: Group sizes in the test data for CIVILCOMMENTS-WILDS. The training and validation data follow similar proportions.

Demographic	Number of non-toxic comments	Number of toxic comments
Male	12092	2203
Female	14179	2270
LGBTQ	3210	1216
Christian	12101	1260
Muslim	5355	1627
Other religions	2980	520
Black	3335	1537
White	5723	2246

simplicity, we have omitted these from the current version of CIVILCOMMENTS-WILDS. These additional data points can be downloaded from the original data source and could be used, for example, by first inferring which group each additional point belongs to, and then running group DRO or a similar algorithm that uses group annotations at training time.

H.7. FMoW-WILDS

ML models for satellite imagery can enable global-scale monitoring of sustainability and economic challenges, aiding policy and humanitarian efforts in applications such as deforestation tracking (Hansen et al., 2013), population density mapping (Tiecke et al., 2017), crop yield prediction (Wang et al., 2020b), and other economic tracking applications (Katona et al., 2018). As satellite data constantly changes due to human activity and environmental processes, these models must be robust to distribution shifts over time. Moreover, as there can be disparities in the data available between regions, these models should ideally have uniformly high accuracies instead of only doing well on data-rich regions and countries.

We study this problem on a variant of the Functional Map of the World dataset (Christie et al., 2018).

Satellite Image (x)	Train		Test		
	2002 / Americas	2009 / Africa	2012 / Europe	2016 / Americas	2017 / Africa
Year / Region (d)	shopping mall	multi-unit residential	road bridge	recreational facility	educational institution

Figure 19: The FMoW-WILDS dataset contains satellite images taken in different geographical regions and at different times. The goal is to generalize to satellite imagery taken in the future, which may be shifted due to infrastructure development across time, and to do equally well across geographic regions.

H.7.1. SETUP

Problem setting. We consider a hybrid domain generalization and subpopulation shift problem, where the input x is a RGB satellite image (resized to 224×224 pixels), the label y is one of 62 building or land use categories, and the domain d represents both the year the image was taken as well as its geographical region (Africa, the Americas, Oceania, Asia, or Europe). We aim to solve both a domain generalization problem across time and improve subpopulation performance across regions.

Data. FMoW-WILDS is based on the Functional Map of the World dataset (Christie et al., 2018), which collected and categorized high-resolution satellite images from over 200 countries based on the functional purpose of the buildings or land in the image, over the years 2002–2018 (see Figure 19). We use a subset of this data and split it into three time range domains, 2002–2013, 2013–2016, and 2016–2018, as well as five geographical regions as subpopulations (Africa, Americas, Oceania, Asia, and Europe). For each example, we also provide the timestamp and location coordinates, though our baseline models only use the coarse time ranges and geographical regions instead of these additional metadata.

We use the following data splits:

1. **Training:** 76,863 images from the years 2002–2013.
2. **Validation (OOD):** 19,915 images from the years from 2013–2016.
3. **Test (OOD):** 22,108 images from the years from 2016–2018.
4. **Validation (ID):** 11,483 images from the years from 2002–2013.
5. **Test (ID):** 11,327 images from the years from 2002–2013.

The original dataset did not evaluate models under distribution shifts. Our training split is a subset of the original

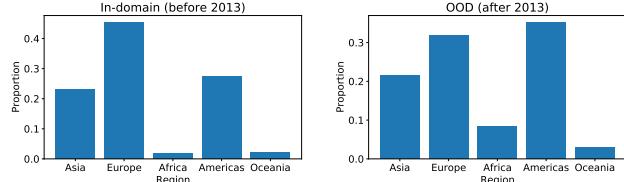


Figure 20: Number of examples from each region of the world in FMoW-WILDS on the ID vs. OOD splits of the data. There is much less data from Africa and Oceania than other regions.

training dataset, filtered for images in the appropriate time range; similarly, our OOD and ID validation splits are subsets of the original validation dataset, and our OOD and ID test splits are subsets of the original test dataset. See Appendix H.7.4 for more dataset details.

The train/val/test data splits contain images from disjoint location coordinates, and all splits contain data from all 5 geographic regions. The ID and OOD splits within the test and validation sets may have overlapping locations, but have non-overlapping time ranges. There is a disparity in the number of examples in each region, with Africa and Oceania having the least examples (Figure 20); this could be due to bias in sampling and/or a lack of infrastructure and land data in certain regions.

Evaluation. We evaluate models by their average and worst-region OOD accuracies. The former measures the ability of the model to generalize across time, while the latter additionally measures how well models do across different regions/subpopulations under a time shift.

Potential leverage. FMoW-WILDS considers both domain generalization across time and subpopulation shift across regions. As we provide both time and region annotations, models can leverage the structure across both space and time to improve robustness. For example, one hypothesis is that infrastructure development occurs smoothly over time. Utilizing this gradual shift structure with the timestamp metadata may enable adaptation across longer time periods (Kumar et al., 2020). The data distribution may also shift smoothly over spatial locations, and so enforcing some consistency with respect to spatial structure may improve predictions (Rolf et al., 2020; Jean et al., 2018). Furthermore, to mitigate the fact that some regions (e.g., Africa) have less labeled data, one could potentially transfer knowledge of other regions with similar economies and infrastructure. The location coordinate metadata allows for transfer learning across similar locations at any spatial scale.

H.7.2. BASELINE RESULTS

Model. For all experiments, we follow Christie et al. (2018) and use a DenseNet-121 model (Huang et al., 2017)

pretrained on ImageNet and with no L_2 regularization. We use the Adam optimizer (Kingma & Ba, 2015) with an initial learning rate of 10^{-4} that decays by 0.96 per epoch, and train for 50 epochs for with early stopping and with a batch size of 64. All reported results are averaged over 3 random seeds.

ERM results and performance drops. Table 18 shows that accuracy drops almost 7% when evaluated on the OOD test set (≥ 2016) vs. the ID test set, and that the accuracy drop is especially large (11.6%) on images from the last year of the dataset (2017), furthest in the future from the training set. In addition, there is a substantial 26.0% drop in worst-region accuracy, with the model performing much worse in Africa than other regions (Table 19).

We ran an additional experiment where we mixed in some data from the OOD period (2013–2018) into the training set, while keeping the overall training set size constant. A model trained on this mixed split had a much smaller drop in performance under the time and region shifts (Table 18). This comparison implies that the performance drop between the ID and OOD test sets is largely due to the distribution shift across time and region.

Additional baseline methods. We compare ERM against CORAL, IRM, and Group DRO, using examples from different years as distinct domains. Table 17 shows that many of these methods are comparable or worse than ERM in terms of both ID and OOD test performance. As with most other datasets, our grid search selected the lowest values of the penalty weights for CORAL ($\lambda = 0.1$) and IRM ($\lambda = 1$).

Discussion. Intriguingly, a large subpopulation shift across regions only occurs with a combination of time and region shift. This is corroborated by the mixed-split region shift results (Table 18), which do not have a time shift between training and test sets, and correspondingly do not display a large disparity in performance across regions. This drop in performance may be partially due to label shift: from Figure 21, we see that the label distributions between Africa and other regions are very different, e.g., with a large drop in recreational facilities and a sharp increase in single residential units. We do not find a similarly large label shift between < 2013 and ≥ 2013 splits of the dataset.

Despite having the smallest number of training examples (Figure 20), the baseline models do not suffer a drop in performance in Oceania on validation or test sets (Table 19). We hypothesize that infrastructure in Oceania is more similar to regions with a large amount of data than Africa. In contrast, Africa may be more distinct and may have changed more drastically over 2002–2018, the time extent of the dataset. This suggests that the subpopulation shift is not merely a function of the number of training examples.

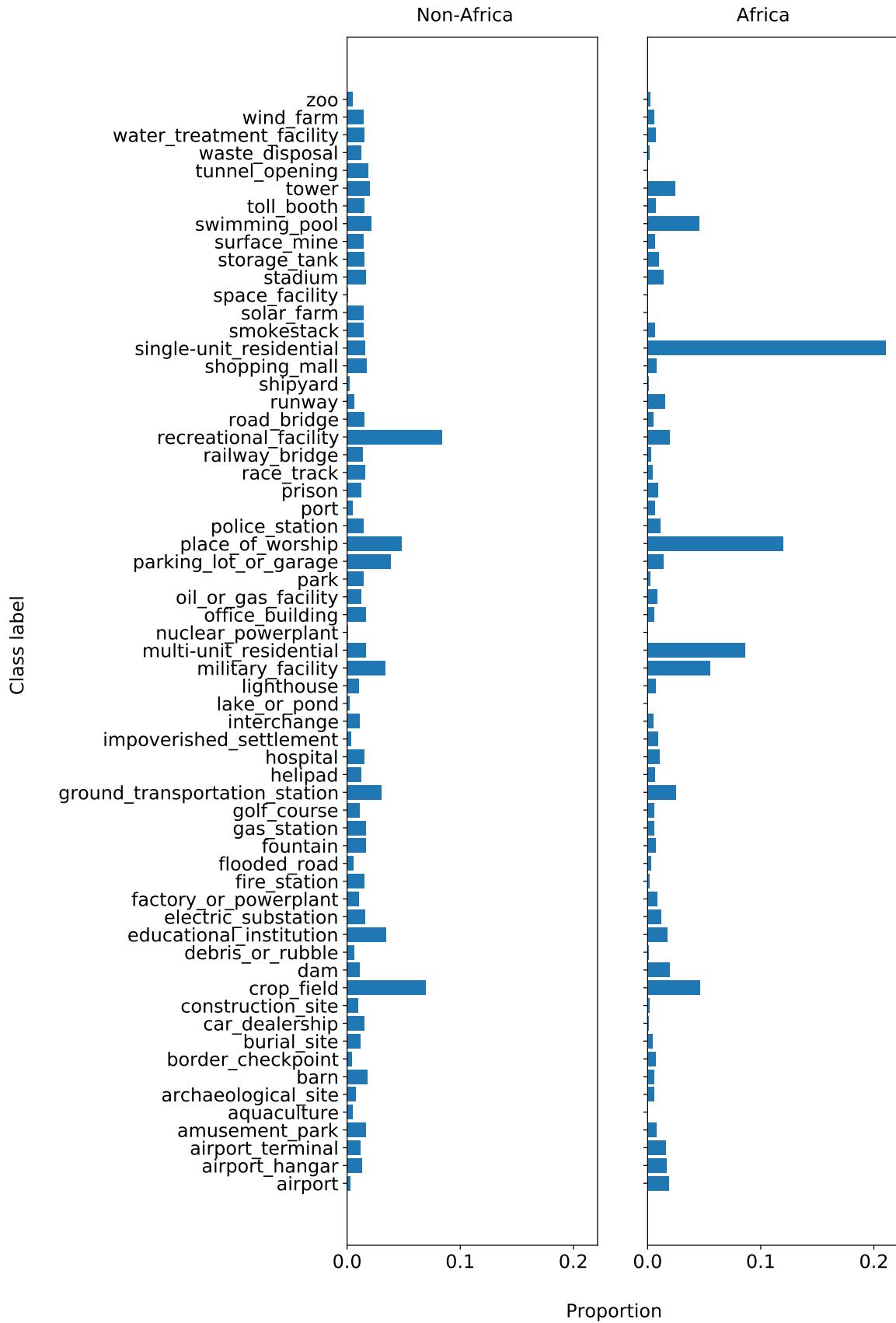


Figure 21: Number of examples from each category in FMoW-WILDS in non-African and African regions. There is a large label shift between non-African regions and Africa.

Table 17: Average and worst-region accuracies (%) under time shifts in FMOW-WILDS. Models are trained on data before 2013 and tested on held-out location coordinates from in-distribution (ID) or out-of-distribution (OOD) test sets. The models are early-stopped with respect to OOD validation accuracy. Standard deviations over 3 trials are in parentheses.

	Validation (ID)	Validation (OOD)	Test (ID)	Test (OOD)
Average				
ERM	61.2 (0.52)	59.5 (0.37)	59.7 (0.65)	53.0 (0.55)
CORAL	58.3 (0.28)	56.9 (0.25)	57.2 (0.90)	50.5 (0.36)
IRM	58.6 (0.07)	57.4 (0.37)	57.7 (0.10)	50.8 (0.13)
Group DRO	60.5 (0.36)	58.8 (0.19)	59.4 (0.11)	52.1 (0.50)
Worst-region				
ERM	59.2 (0.69)	48.9 (0.62)	58.3 (0.92)	32.3 (1.25)
CORAL	55.9 (0.50)	47.1 (0.43)	55.0 (1.02)	31.7 (1.24)
IRM	56.6 (0.59)	47.5 (1.57)	56.0 (0.34)	30.0 (1.37)
Group DRO	57.9 (0.62)	46.5 (0.25)	57.8 (0.60)	30.8 (0.81)

Table 18: Performance drops for ERM models on FMOW-WILDS. In the standard split, we train on ID examples (i.e., data from 2002–2013), whereas in the mixed split, we train on ID + OOD examples (i.e., the same amount of data but half from 2002–2013 and half from 2013–2018). In both cases, we test on data from 2016–2018. Models trained on the standard split degrade in performance under the time shift, especially on the last year (2017) of the test data, and also fare poorly on the subpopulation shift, with low worst-region accuracy. Models trained on the mixed split have higher OOD average and last year accuracy and much higher OOD worst-region accuracy. Standard deviations over 3 trials are in parentheses.

	Algorithm	Test (ID)		Test (OOD)	
		Average	Worst-region	Average	Last year
Standard split	ERM	59.7 (0.65)	58.3 (0.92)	53.0 (0.55)	48.1 (1.20)
Mixed split	ERM	59.0 (0.47)	56.9 (0.80)	57.4 (0.27)	54.3 (0.22)

Table 19: The regional accuracies of models trained on data before 2013 and tested on held-out locations from ID (< 2013) or OOD (\geq 2016) test sets in FMOW-WILDS. Standard deviations over 3 trials are in parentheses.

	Asia	Europe	Africa	Americas	Oceania	Worst region
OOD Test						
ERM	55.4 (0.95)	55.6 (0.53)	32.3 (1.25)	55.7 (0.48)	59.1 (0.85)	32.3 (1.25)
CORAL	52.4 (0.96)	52.6 (0.82)	31.7 (1.24)	53.3 (0.27)	56.0 (2.02)	31.7 (1.24)
IRM	52.9 (0.73)	53.9 (0.28)	30.0 (1.37)	53.7 (0.51)	55.0 (2.22)	30.0 (1.37)
Group DRO	54.7 (0.52)	55.1 (0.39)	30.8 (0.81)	54.6 (0.48)	58.5 (1.65)	30.8 (0.81)
ID Test						
ERM	58.9 (1.19)	58.4 (0.81)	69.1 (2.64)	61.4 (0.35)	69.9 (0.53)	58.3 (0.92)
CORAL	56.6 (1.35)	55.0 (1.02)	69.2 (2.92)	59.7 (0.83)	70.8 (2.53)	55.0 (1.02)
IRM	56.9 (0.62)	56.0 (0.34)	69.7 (2.16)	59.7 (0.49)	68.3 (2.00)	56.0 (0.34)
Group DRO	58.7 (0.33)	57.9 (0.74)	69.2 (0.28)	61.1 (0.57)	68.8 (2.38)	57.8 (0.60)

We note that our dataset splits can separate on particular factors such as the introduction of new sensors, which is natural with progression over time. For example, the WorldView-3 sensor came online in 2014. Future work should look into the role of auxiliary factors such as new sensors that are associated with time but may be controllable. We did not find a sharp difference in performance due to the introduction of WorldView-3; we found that the performance decays gradually over time, suggesting that the performance drop comes from other factors.

As with POVERTYMAP-WILDS, there are important ethical considerations associated with remote sensing applications, e.g., around surveillance and privacy issues, as well as the potential for systematic biases that negatively affect particular populations. As an example of the latter, the poor model performance on satellite images from Africa that we observe in FMOW-WILDS raises issues of bias and fairness. With regard to privacy, we note that the image resolution in FMOW-WILDS is lower than that of other public and easily-accessible satellite data such as that from Google Maps.

We refer interested readers to the UNICEF discussion paper by [Berman et al. \(2018\)](#) for a more in-depth discussion of the ethics of remote sensing especially as it pertains to development and humanitarian endeavors.

H.7.3. BROADER CONTEXT

Recognizing infrastructure and land features is crucial to many remote sensing applications. For example, in crop land prediction ([Wang et al., 2020b](#)), recognizing gridded plot lines, plot circles, farm houses, and other visible features are important in recognizing crop fields. However, farming practices and equipment evolve over time and vary widely across the world, requiring both robust object recognition and synthesis of their different usage patterns.

Although the data is typically limited, we desire generalization on a global scale without requiring frequent large-scale efforts to gather more ground-truth data. It is natural to have labeled data with limited temporal or spatial extent since ground truth generally must be verified on the ground or requires manual annotations from domain experts (i.e., they are often hard to be crowdsourced). A number of existing remote sensing datasets have limited spatial or temporal scope, including the UC Merced Land Use Dataset ([Yang & Newsam, 2010](#)), TorontoCity ([Wang et al., 2017](#)), and SpaceNet ([DigitalGlobe & Works, 2016](#)). However, works based on these datasets generally do not systematically study shifts in time or location.

H.7.4. ADDITIONAL DETAILS

Data processing and modifications to the original dataset. The FMOW-WILDS dataset is derived from [Christie et al. \(2018\)](#), which collected over 1 million satellite images from over 200 countries over 2002-2018. We use the RGB version of the original dataset, which contains 523,846 total examples, excluding the multispectral version of the images. Methods that can utilize a sequence of images can group the images from the same location across multiple years together as input, but we consider the simple formulation here for our baseline evaluation.

The original dataset from [Christie et al. \(2018\)](#) is provided as a set of hierarchical directories with JPEG images of varying sizes. To reduce download times and I/O usage, we resize these images to 224×224 pixels, and then store them as PNG images. We also collect all the metadata into CSV format for easy processing.

The original dataset is posed as a image time-series classification problem, where the model has access to a sequence of images at each location. For simplicity, we treat each image as a separate example, while making sure that the data splits all contain disjoint locations. We use the train/val/test splits from the original dataset, but separate out two OOD time

segments: we treat the original validation data from 2013-2016 as OOD val and the original test data from 2016-2018 as OOD test. We remove data from after 2013 from the training set, which reduces the size of the training set in comparison to the original dataset.

Additional challenges in high-resolution satellite datasets. Compared to POVERTYMAP-WILDS, FMOW-WILDS contains much higher resolution images (sub-meter resolution vs. 30m resolution) and contains a larger variety of viewpoints/tilts, both of which could present computational or algorithmic challenges. For computational purposes, we resized all images to 224×224 (following [Christie et al. \(2018\)](#)), but raw images can be thousands of pixels wide. Some recent works have tried to balance this trade-off between viewing overall context and the fine-grained detail ([Uzkent & Ermon, 2020](#); [Kim et al., 2016a](#)), but how best to do this is an open question. FMOW-WILDS also contains additional information on azimuth and cloud cover which could be used to correct for the variety in viewpoints and image quality.

H.8. POVERTYMAP-WILDS

A different application of satellite imagery is poverty estimation across different spatial regions, which is essential for targeted humanitarian efforts in poor regions ([Abelson et al., 2014](#); [Espey et al., 2015](#)). However, ground-truth measurements of poverty are lacking for much of the developing world, as field surveys are expensive ([Blumenstock et al., 2015](#); [Xie et al., 2016](#); [Jean et al., 2016](#)). For example, at least 4 years pass between nationally representative consumption or asset wealth surveys in the majority of African countries, with seven countries that had either never conducted a survey or had gaps of over a decade between surveys ([Yeh et al., 2020](#)). One approach to this problem is to train ML models on countries with ground truth labels and then deploy them to different countries where we have satellite data but no labels.

We study this problem through a variant of the poverty mapping dataset collected by [Yeh et al. \(2020\)](#).

H.8.1. SETUP

Problem setting. We consider a hybrid domain generalization and subpopulation shift problem, where the input x is a multispectral LandSat satellite image with 8 channels (resized to 224×224 pixels), the output y is a real-valued asset wealth index computed from Demographic and Health Surveys (DHS) data, and the domain d represents the country the image was taken in and whether the image is of an urban or rural area. We aim to solve both a domain generalization problem across country borders and improve subpopulation performance across urban and rural areas.

Satellite Image (x)	Train			Test	
	Angola / urban	Angola / rural	Angola / urban	Kenya / urban	Kenya / rural
Asset Index (y)	0.259	-1.106	2.347	0.827	0.130

Figure 22: The POVERTYMAP-WILDS dataset contains satellite images taken in different countries. The goal is to predict asset wealth in countries that are not present in the training set, while being accurate in both urban and rural areas. There may be significant economic and cultural differences across country borders that contribute to the spatial distribution shift.

Data. POVERTYMAP-WILDS is based on a dataset collected by Yeh et al. (2020), which assembles satellite imagery and survey data at 19,669 villages from 23 African countries between 2009 and 2016 (Figure 22). Each input image has 8 channels: 7 from the LandSat satellite and an 8th channel for nighttime light intensity from a separate satellite, as prior work has established that these night lights correlate with poverty measures (Noor et al., 2008; Elvidge et al., 2009).

There are $23 \times 2 = 46$ domains corresponding to the 23 countries and whether the location is urban or rural. Each example comes with metadata on its location coordinates, survey year, and its urban/rural classification.

In contrast to other datasets, which have a single fixed ID/OOD split, the relatively small size of POVERTYMAP-WILDS allows us to use 5 different folds, where each fold defines a different set of OOD countries. In each fold, we use the following splits of the data (the number of countries and images in each split varies slightly from fold to fold):

1. **Training:** ~10000 images from 13–14 countries.
2. **Validation (OOD):** ~4000 images from 4–5 different countries (distinct from training and test (OOD) countries).
3. **Test (OOD):** ~4000 images from 4–5 different countries (distinct from training and validation (OOD) countries).
4. **Validation (ID):** ~1000 images from the same 13–14 countries in the training set.
5. **Test (ID):** ~1000 images from the same 13–14 countries in the training set.

All splits contain images of both urban and rural locations, with the countries assigned randomly to each split in each fold.

The distribution of wealth may shift across countries due to differing levels economic development, agricultural practices, and other factors. For example, Abelson et al. (2014) use thatched vs. metal roofs to distinguish between poor and wealthy households, respectively in Kenya and Uganda. However, other countries may have a different mapping of roof type to wealth where metal roofs signify more poor households. Similar issues can arise when looking at the health of crops (related to vegetation indices such as NDVI that are simple functions of the multispectral channels in the satellite image) as a sign for wealth in rural areas, since crop health is related to climate and the choice of crops, which vary upon region.

Asset wealth may also shift dramatically between countries. Figure 23 shows the mean asset wealth per country, as well as urban vs. rural asset wealth per country. Mean asset wealth ranges from -0.4 to +0.8 depending on the country. There is a stark difference between mean asset wealth in urban and rural areas, with urban asset wealth being positive in all countries while rural mean asset wealth being mostly negative.

Evaluation. As is standard in the literature (Jean et al., 2016; Yeh et al., 2020), the models are evaluated on the Pearson correlation (r) between their predicted and actual asset wealth indices. We measure the average correlation, to test generalization under country shifts, and also the lower of the correlations on the urban and rural subpopulations, to test generalization between urban and rural subpopulations. We report the latter as previous works on poverty prediction from satellite imagery have noted that a significant part of model performance relies on distinguishing urban vs. rural areas, and improving performance within these subpopulations is an ongoing challenge, with rural areas generally faring worse under existing models (Jean et al., 2016; Yeh et al., 2020).

We average all correlations across the 5 different folds, using 1 random seed per fold. The resulting standard deviations reflect the fact that different folds have different levels of difficulty (e.g., depending on how similar the ID and OOD countries are). For the purposes of comparing different algorithms and models, we note that these standard deviations might make the comparisons appear noisier than they are, since a model might perform similarly across random seeds but still have a high standard deviation if it has different performances on different folds on the data. In contrast, other WILDS datasets report results on the same data split but averaged across different random seeds.

Potential leverage. Large socioeconomic differences between countries makes generalization across borders challenging. However, some indicators of wealth are known to be robust and are able to be seen from space. For example,

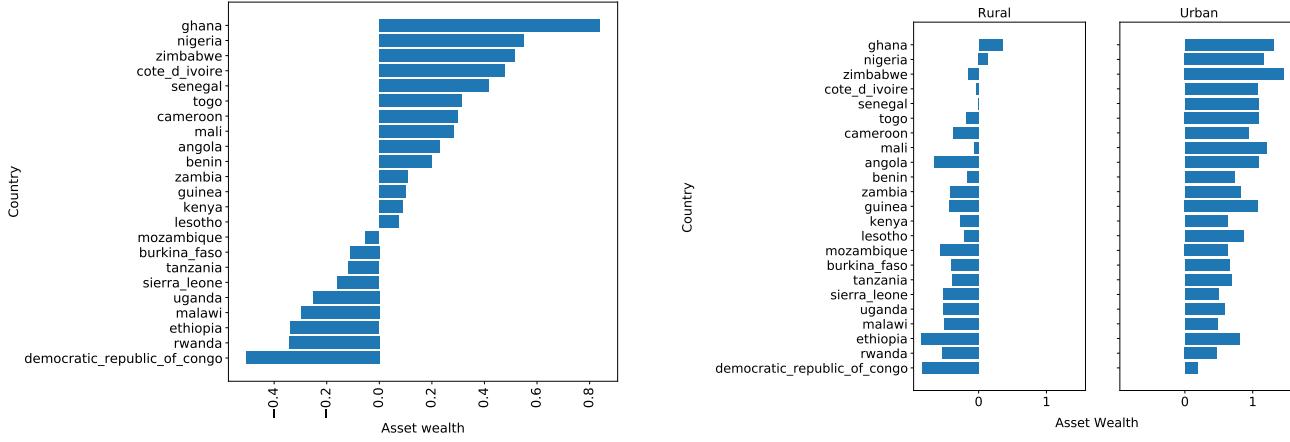


Figure 23: Mean asset wealth by country on aggregate as well as urban and rural splits for each country, computed on the full dataset.

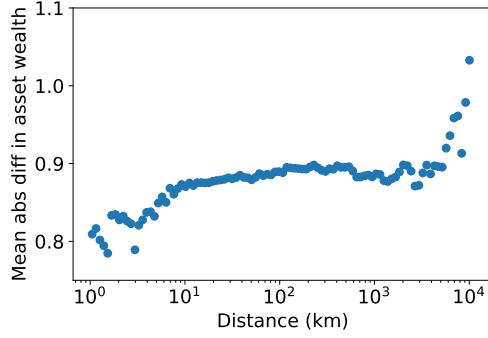


Figure 24: Mean absolute difference in asset wealth between two data points in the full dataset as a function of (great circle) distance between the two points. Smaller distances between data points correlate with more similar asset wealth measures. The pairs are binned by distance on a log (base 10) scale (100 bins), and the mean value of each bin is plotted at the midpoint distance of each bin.

roof type (e.g. thatched or metal roofing) has been shown to be a reliable proxy for wealth (Abelson et al., 2014), and contextual factors such as the health of nearby croplands, the presence of paved roads, and connections to urban areas are plausibly reliable signals for measuring poverty. Poverty measures are also known to be highly correlated across space, meaning nearby villages will likely have similar poverty measures, and methods can utilize this spatial structure (using the provided location coordinate metadata) to improve predictions (Jean et al., 2018; Rolf et al., 2020). We show the correlation with distance in Figure 24, which plots the distance between pairs of data points against the absolute differences in asset wealth between pairs.

H.8.2. BASELINE RESULTS

Model. For all experiments, we follow Yeh et al. (2020) and train a ResNet-18 model (He et al., 2016) to minimize

squared error. We use the Adam optimizer (Kingma & Ba, 2015) with an initial learning rate of 10^{-3} that decays by 0.96 per epoch, and train for 200 epochs for with early stopping (on OOD r) and with a batch size of 64.

ERM results and performance drops. When shifting across country borders, Table 21 shows that ERM suffers a 0.04 drop in r on OOD test examples compared to ID test examples (MSE results are in Appendix H.8.4). Moreover, the drop in performance is exacerbated when looking at urban and rural subpopulations, even though all splits contain urban and rural examples. Table 21 shows that the difference between ID and OOD test r in the ERM model nearly triples from 0.04 to 0.11 when considering the overall vs. rural r . Correlation is consistently lower on the rural subpopulation than the urban subpopulation.

We ran an additional experiment where we considered an alternative training set with data that was uniformly sampled from all countries, while keeping the overall training set size constant (i.e., compared to the standard training set, it has fewer examples from each country, but data from more countries). A model trained on this mixed split had a much smaller drop in performance between the ID and OOD test sets (Table 21), as the mixed training set contains examples from the countries in both the ID and OOD test sets. This comparison implies that the performance drop between the ID and OOD test sets is largely due to the distribution shift from seen to unseen countries.

Finally, we also ran an ablation where we removed the night-time light intensity channel. This resulted in a drop in OOD r of 0.04 on average and 0.06 on the rural subpopulation, demonstrating the usefulness of the nightlight data in asset wealth estimation.

Additional baseline methods. We trained models with CORAL, IRM, and Group DRO, taking examples from dif-

Table 20: Pearson correlation r (higher is better) on in-distribution and out-of-distribution (unseen countries) held-out sets in POVERTYMAP-WILDS, including results on the worst urban/rural subpopulations. All results are averaged over 5 different OOD country folds taken from [Yeh et al. \(2020\)](#), with standard deviations across different folds in parentheses.

	Validation (ID)	Validation (OOD)	Test (ID)	Test (OOD)
Overall				
ERM	0.82 (0.02)	0.80 (0.04)	0.82 (0.03)	0.78 (0.04)
CORAL	0.82 (0.00)	0.80 (0.04)	0.83 (0.01)	0.78 (0.05)
IRM	0.82 (0.02)	0.81 (0.03)	0.82 (0.02)	0.77 (0.05)
Group DRO	0.78 (0.03)	0.78 (0.05)	0.80 (0.03)	0.75 (0.07)
Worst urban/rural subpopulation				
ERM	0.58 (0.07)	0.51 (0.06)	0.57 (0.07)	0.45 (0.06)
CORAL	0.59 (0.04)	0.52 (0.06)	0.59 (0.03)	0.44 (0.06)
IRM	0.57 (0.06)	0.53 (0.05)	0.57 (0.08)	0.43 (0.07)
Group DRO	0.49 (0.08)	0.46 (0.04)	0.54 (0.11)	0.39 (0.06)

Table 21: Performance drops for ERM models on POVERTYMAP-WILDS. In the standard split, we train on data from one set of countries, and then test on a different set of countries. In the mixed split, we train on the same amount of data but sampled uniformly from all countries. Models trained on the standard split degrade in performance, especially on rural subpopulations, while models trained on the mixed split do not.

	Test (ID)			Test (OOD)		
	Overall r	Rural r	Urban r	Overall r	Rural r	Urban r
Standard split (ID examples)	0.82 (0.03)	0.57 (0.07)	0.66 (0.04)	0.78 (0.04)	0.46 (0.05)	0.59 (0.11)
Mixed split (ID + OOD examples)	0.83 (0.01)	0.62 (0.01)	0.65 (0.03)	0.83 (0.03)	0.60 (0.06)	0.65 (0.06)

ferent countries as coming from distinct domains. Table 20 shows that these baselines are generally comparable to ERM, and that they continue to be susceptible to shifts across countries and urban/rural areas. As with most other datasets, our grid search selected the lowest values of the penalty weights for CORAL ($\lambda = 0.1$) and IRM ($\lambda = 1$).

Discussion. These results corroborate performance drops seen in previous out-of-country generalization tests for poverty prediction from satellite imagery ([Jean et al., 2016](#)). In general, differences in infrastructure, economic development, agricultural practices, and even cultural differences can cause large shifts across country borders. Differences between urban and rural subpopulations have also been well-documented ([Jean et al., 2016; Yeh et al., 2020](#)). Models based on nighttime light information could suffer more in rural areas where nighttime light intensity is uniformly low or even zero.

Since survey years are also available, we could also investigate the robustness of the model over time. This would enable the models to be used for a longer time before needing more updated survey data, and we leave this to future work. [Yeh et al. \(2020\)](#) investigated predicting the change in asset wealth for individual villages in the World Bank Living Standards Measurement Surveys (LSMS), which is a longitudinal study containing multiple samples from the

same village. POVERTYMAP-WILDS only contains cross-sectional samples which do not provide direct supervision for changes over time at any one location, but it is still possible to consider aggregate shifts across years.

As with FMoW-WILDS, there are important ethical considerations associated with remote sensing applications, e.g., around surveillance and privacy issues, as well as the potential for systematic biases that negatively affect particular populations. As we describe in Section H.8.4, noise has been added to the location metadata in POVERTYMAP-WILDS to protect privacy. The distribution shifts across country and urban/rural boundaries that we study in POVERTYMAP-WILDS are an example of a bias that affects model performance and therefore could have adverse policy consequences. We refer interested readers to the UNICEF discussion paper by [Berman et al. \(2018\)](#) for a more in-depth discussion of the ethics of remote sensing especially as it pertains to development and humanitarian endeavors.

H.8.3. BROADER CONTEXT

Computational sustainability applications in the developing world also include tracking child mortality ([Burke et al., 2016; Osgood-Zimmerman et al., 2018; Reiner et al., 2018](#)), educational attainment ([Graetz et al., 2018](#)), and food security and crop yield prediction ([You et al., 2017; Wang et al., 2020b; Xie et al., 2020](#)). Remote sensing data and

satellite imagery has the potential to enable high-resolution maps of many of these sustainability challenges, but as with poverty measures, ground truth labels in these applications come from expensive surveys or observations from human workers in the field. Some prior works consider using spatial structure (Jean et al., 2018; Rolf et al., 2020), unlabeled data (Xie et al., 2016; Jean et al., 2018; Xie et al., 2020), or weak sources of supervision (Wang et al., 2020b) to improve global models despite the lack of ground-truth data. We hope that POVERTYMAP-WILDS can be used to improve the robustness of machine learning techniques on satellite data, providing an avenue for cheaper and faster measurements that can be used to make progress on a general set of computational sustainability challenges.

H.8.4. ADDITIONAL DETAILS

Data processing. The POVERTYMAP-WILDS dataset is derived from Yeh et al. (2020), which gathers LandSat imagery and Demographic and Health Surveys (DHS) data from 19669 villages across 23 countries in Africa . The images are 224×224 pixels large over 7 multispectral channels and an eighth nighttime light intensity channel. The LandSat satellite has a 30m resolution, meaning that each pixel of the image covers a $30m^2$ spatial area. The location metadata is perturbed by the DHS as a privacy protection scheme; urban locations are randomly displaced by up to 2km and rural locations are perturbed by up to 10km. While this adds noise to the data, having a large enough image can guarantee that the location is in the image most of the time. The target is a real-valued composite asset wealth index computed as the first principal component of survey responses about household assets, which is thought to be a less noisy measure of households’ longer-run economic well-being than other welfare measurements like consumption expenditure (Sahn & Stifel, 2003; Filmer & Scott, 2011). Asset wealth also has the advantage of not requiring adjustments for inflation or for purchasing power parity (PPP), as it is not based on a currency.

We normalize each channel by the pixel-wise mean and standard deviation for each channel, following (Yeh et al., 2020). We also do a similar data augmentation scheme, adding random horizontal and vertical flips as well as color jitter (brightness factor 0.8, contrast factor 0.8, saturation factor 0.8, hue factor 0.1).

The data download process provided by Yeh et al. (2020) involves downloading and processing imagery from Google Earth Engine. We process each image into a compressed NumPy array with 8 channels. We also provide all the metadata in a CSV format.

Modifications to the original dataset. We report a much larger drop in correlation due to spatial shift than in Yeh

et al. (2020). To explain this, we note that our data splitting method is slightly different from theirs. They have two separate experiments (with different data splits) to test in-distribution vs. out-of-distribution generalization. In contrast, our data splits on both held-out in-distribution and out-of-distribution points at the same time with respect to the same training set, thus allowing us to compare both metrics simultaneously on one model as a more direct comparison. We use the same OOD country folds as the original dataset. However, Yeh et al. (2020) split the ID train/val/test while making sure that the spatial extent of the images between each split never overlap, while we simply take uniformly random splits of the ID data. This means that between our ID train/val/test splits, we may have images that have share some overlapping spatial extent, for example for two very nearby locations. Thus, a model can utilize some memorization here to improve ID performance. We believe this is reasonable since, with more ID data, more of the spatial area will be labeled and memorization should become an increasingly viable strategy for generalization in-domain.

H.9. AMAZON-WILDS

In many consumer-facing ML applications, models are trained on data collected on one set of users and then deployed across a wide range of potentially new users. These models can perform well on average but poorly on some individuals (Tatman, 2017; Caldas et al., 2018; Li et al., 2019b; Koenecke et al., 2020). These large performance disparities across users are practical concerns in consumer-facing applications, and they can also indicate that models are exploiting biases or spurious correlations in the data (Badgeley et al., 2019; Geva et al., 2019). We study this issue of inter-individual performance disparities on a variant of the AMAZON-WILDS Reviews dataset (Ni et al., 2019).

	Reviewer ID (d)	Review Text (x)	Stars (y)
Train	Reviewer 1	They are decent shoes. Material quality is good but the color fades very quickly. Not as black in person as shown.	5
	Reviewer 2	Super easy to put together. Very well built.	5
	Reviewer 10,000	This works well and was easy to install. The only thing I don’t like is that it tilts forward a little bit and I can’t figure out how to stop it. Perfect for the trail camera ...	4
Test	Reviewer 10,000	I am disappointed in the quality of these. They have significantly deteriorated in just a few uses. I am going to stick with using foil.	1
	Reviewer 10,001	Very sturdy especially at this price point. I have a memory foam mattress on it with nothing underneath and the slats perform well. Solidly built plug in. I have had 4 devices plugged in and all charge just fine. Works perfectly on the wall to hang our wreath without having to do any permanent damage. ...	5

Figure 25: The AMAZON-WILDS dataset involves predicting star ratings from reviews of Amazon products. The goal is to do consistently well on new reviewers who are not in the training set.

H.9.1. SETUP

Problem setting. We consider a hybrid domain generalization and subpopulation problem where the domains correspond to different reviewers. The task is multi-class sentiment classification, where the input x is the text of a review, the label y is a corresponding star rating from 1 to 5, and the domain d is the identifier of the reviewer who wrote the review. Our goal is to perform consistently well across a wide range of reviewers, i.e., to achieve high tail performance on different subpopulations of reviewers in addition to high average performance. In addition, we consider disjoint set of reviewers between training and test time.

Data. The dataset comprises 539,502 customer reviews on Amazon taken from the Amazon Reviews dataset (Ni et al., 2019). Each input example has a maximum token length of 512. For each example, the following additional metadata is also available at both training and evaluation time: reviewer ID, product ID, product category, review time, and summary.

To reliably measure model performance on each reviewer, we include at least 75 reviews per reviewer in each split. Concretely, we consider the following splits, where reviewers are randomly assigned to either in-distribution or out-of-distribution sets:

1. **Training:** 245,502 reviews from 1,252 reviewers.
2. **Validation (OOD):** 100,050 reviews from another set of 1,334 reviewers, distinct from training and test (OOD).
3. **Test (OOD):** 100,050 reviews from another set of 1,334 reviewers, distinct from training and validation (OOD).
4. **Validation (ID):** 46,950 reviews from 626 of the 1,252 reviewers in the training set.
5. **Test (ID):** 46,950 reviews from 626 of the 1,252 reviewers in the training set.

The reviewers in the train and in-distribution splits; the validation (OOD) split; and the test (OOD) split are all disjoint, which allows us to test generalization to unseen reviewers. See Appendix H.9.4 for more details.

Evaluation. To assess whether models perform consistently well across reviewers, we evaluate models by their accuracy on the reviewer at the 10th percentile. This follows the federated learning literature, where it is standard to measure model performance on devices and users at various percentiles in an effort to encourage good performance across many devices (Caldas et al., 2018; Li et al., 2019b).

Potential leverage. We include more than a thousand reviewers in the training set, capturing variation across a wide range of reviewers. In addition, we provide reviewer ID annotations for all reviews in the dataset. These annotations could be used to directly mitigate performance disparities across reviewers seen during training time.

H.9.2. BASELINE RESULTS

Model. For all experiments, we finetuned DistilBERT-base-uncased models (Sanh et al., 2019), using the implementation from Wolf et al. (2019), and with the following hyperparameter settings: batch size 8; learning rate 1×10^{-5} with the AdamW optimizer (Loshchilov & Hutter, 2019); L_2 -regularization strength 0.01; 3 epochs with early stopping; and a maximum number of tokens of 512. We selected the above hyperparameters based on a grid search over learning rates $\{1 \times 10^{-6}, 2 \times 10^{-6}, 1 \times 10^{-5}, 2 \times 10^{-5}\}$, and all other hyperparameters were simply set to standard/default values.

ERM results and performance drops. A DistilBERT-base-uncased model trained with the standard ERM objective performs well on average, but performance varies widely across reviewers (Figure 26, Table 22). Despite the high average accuracy of 71.9%, per-reviewer accuracies vary widely between 100.0% and 12.0%, with accuracy at the 10th percentile of 53.8%. The above variation is larger than expected from randomness: a random binomial baseline with equal average accuracy would have a 10th percentile accuracy of 65.4%. We observe low tail performance on both previously seen and unseen reviewers, with low 10th percentile accuracy on in-distribution and out-of-distribution sets (Table 22). In addition, we observe drops on both average and 10th percentile accuracies upon evaluating on unseen reviewers, as evident in the performance gaps between the in-distribution and the out-of-distribution sets.

Additional baseline methods. We now consider models trained by existing robust training algorithms, and show that these models also perform poorly on tail reviewers, failing to mitigate the performance drop (Table 22). We observe that reweighting to achieve uniform class balance fails to improve the 10th percentile accuracy, showing that variation across users cannot be solved simply by accounting for label imbalance. In addition, CORAL, IRM, and Group DRO fail to improve both average and 10th percentile accuracies on both ID and OOD sets. Our grid search selected $\lambda = 1.0$ for the CORAL penalty and $\lambda = 1.0$ for the IRM penalty.

Discussion. The distribution shift and the evaluation criteria for AMAZON-WILDS focus on the tail performance, unlike the other datasets in WILDS. Because of this, AMAZON-

Table 22: Baseline results on AMAZON-WILDS. We report the accuracy of models trained using ERM, CORAL, IRM, and group DRO, as well as a reweighting baseline that reweights for class balance. To measure tail performance across reviewers, we report the accuracy for the reviewer in the 10th percentile. While the performance drop on AMAZON-WILDS is primarily from subpopulation shift, there is also a performance drop from evaluating on unseen reviewers, as evident in the gaps in accuracies between the in-distribution and the out-of-distribution sets.

Algorithm	Validation (ID)		Validation (OOD)		Test (ID)		Test (OOD)	
	10th percentile	Average	10th percentile	Average	10th percentile	Average	10th percentile	Average
ERM	58.7 (0.0)	75.7 (0.2)	55.2 (0.7)	72.7 (0.1)	57.3 (0.0)	74.7 (0.1)	53.8 (0.8)	71.9 (0.1)
CORAL	56.2 (1.7)	74.4 (0.3)	54.7 (0.0)	72.0 (0.3)	55.1 (0.4)	73.4 (0.2)	52.9 (0.8)	71.1 (0.3)
IRM	56.4 (0.8)	74.3 (0.1)	54.2 (0.8)	71.5 (0.3)	54.7 (0.0)	72.9 (0.2)	52.4 (0.8)	70.5 (0.3)
Group DRO	57.8 (0.8)	73.7 (0.6)	54.7 (0.0)	70.7 (0.6)	55.8 (1.0)	72.5 (0.3)	53.3 (0.0)	70.0 (0.5)
Reweight (label)	55.1 (0.8)	71.9 (0.4)	52.1 (0.2)	69.1 (0.5)	54.4 (0.4)	70.7 (0.4)	52.0 (0.0)	68.6 (0.6)

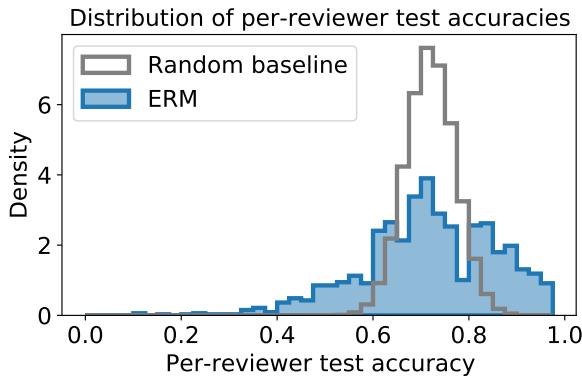


Figure 26: Distribution of per-reviewer accuracy on the test set for the ERM model (blue). The corresponding random baseline would have per-reviewer accuracy distribution in grey.

WILDS might have distinct empirical trends or be conducive to different algorithms compared to other datasets. Potential approaches include extensions to algorithms for worst-group performance, for example to handle a large number of groups, as well as adaptive approaches that yield user-specific predictions.

H.9.3. BROADER CONTEXT

Performance disparities across individuals have been observed in a wide range of tasks and applications, including in natural language processing (Geva et al., 2019), automatic speech recognition (Koenecke et al., 2020; Tatman, 2017), federated learning (Li et al., 2019b; Caldas et al., 2018), and medical imaging (Badgeley et al., 2019). These performance gaps are practical limitations in applications that call for good performance across a wide range of users, including many user-facing applications such as speech recognition (Koenecke et al., 2020; Tatman, 2017) and personalized recommender systems (Patro et al., 2020), tools used for analysis of individuals such as sentiment classification in computational social science (West et al., 2014) and user analytics (Lau et al., 2014), and applications in federated

learning. These performance disparities have also been studied in the context of algorithmic fairness, including in the federated learning literature, in which uniform performance across individuals is cast as a goal toward fairness (Li et al., 2019b; Dwork et al., 2012). Lastly, these performance disparities can also highlight models’ failures to learn the actual task in a generalizable manner; instead, some models have been shown to learn the biases specific to individuals. Prior work has shown that individuals—technicians for medical imaging in this case—can not only be identified from data, but also are predictive of the diagnosis, highlighting the risk of learning to classify technicians rather than the medical condition (Badgeley et al., 2019). More directly, across a few natural language processing tasks where examples are annotated by crowdworkers, models have been observed to perform well on annotators that are commonly seen at training time, but fail to generalize to unseen annotators, suggesting that models are merely learning annotator-specific patterns and not the task (Geva et al., 2019).

H.9.4. ADDITIONAL DETAILS

Data processing. We consider a modified version of the Amazon reviews dataset (Ni et al., 2019). We consider disjoint reviewers between the training, OOD validation, and OOD test sets, and we also provide separate ID validation and test sets that include reviewers seen during training for additional reporting. These reviewers are selected uniformly at random from the reviewer pool, with the constraint that they have at least 150 reviews in the pre-processed dataset. Statistics for each split are described in Table 23. Notably, each reviewer has at least 75 reviews in the training set and exactly 75 reviews in the validation and test sets.

To process the data, we first eliminate reviews that are longer than 512 tokens, reviews without any text, and any duplicate reviews with identical star rating, reviewer ID, product ID, and time. We then obtain the 30-core subset of the reviews, which contains the maximal set of reviewers and products such that each reviewer and product has at least 30 reviews; this is a standard preprocessing procedure used

Table 23: Dataset details for AMAZON-WILDS.

Split	# Reviews	# Reviewers	# Reviews per reviewer (mean / minimum)
Training	245,502	1,252	196 / 75
Validation (OOD)	100,050	1,334	75 / 75
Test (OOD)	46,950	662	75 / 75
Validation (ID)	100,050	1,334	75 / 75
Test (ID)	46,950	662	75 / 75

in the original dataset (Ni et al., 2019). To construct the dataset for reviewer shifts in particular, we further eliminate the following reviews: (i) reviews that contain HTML, (ii) reviews with identical text within a user in order to ensure sufficiently high effective sample size per reviewer, and (iii) reviews with identical text across users to eliminate generic reviews. Once we have the filtered set of reviews, we consider reviewers with at least 150 reviews and sample uniformly at random until the training set contains approximately 250,000 reviews and each evaluation set contains at least 100,000 reviews. As we construct the training set, we reserve a random sample of 75 reviews for each user for evaluation and put all other reviews in the training set. For the evaluation set, we put a random sample of 75 reviews for each user.

Modifications to the original dataset. The original dataset does not prescribe a specific task or split. We consider a standard task of sentiment classification, but instead of using a standard i.i.d. split, we instead consider disjoint users between training and evaluation time as described above. In addition, we preprocess the data as detailed above.

H.10. PY150-WILDS

Code completion models—autocomplete tools used by programmers to suggest subsequent source code tokens, such as the names of API calls—are commonly used to reduce the effort of software development (Robbes & Lanza, 2008; Bruch et al., 2009; Nguyen & Nguyen, 2015; Proksch et al., 2015; Franks et al., 2015). These models are typically trained on data collected from existing codebases but then deployed more generally across other codebases, which may have different distributions of API usages (Nita & Notkin, 2010; Proksch et al., 2016; Allamanis & Brockschmidt, 2017). This shift across codebases can cause substantial performance drops in code completion models. Moreover, prior studies of real-world usage of code completion models have noted that these models can generalize poorly on some important subpopulations of tokens such as method names (Hellendoorn et al., 2019).

We study this problem using a variant of the Py150 Dataset, originally developed by Raychev et al. (2016) and adapted to a code completion task by Lu et al. (2021).

	Repository ID (d)	Source code context (x)	Next tokens (y)
Train	Repository 1	... from easyrecc.gateway import EasyRec <EOL> gateway = EasyRec('tenant','key') <EOL> item_type = gateway. _____	get_item_type
	Repository 2	... response = gateway.get_other_users() <EOL> get_params = HtlPretty. _____	last_request
	:	import numpy as np <EOL> if np.linalg.norm(target - prev_target) > far_threshold: <EOL> norm = np. _____	linalg
Test	Repository 6,001	... new_trans = np.zeros((n_beats + max_beats, n_beats) <EOL> new_trans[:n_beats,:n_beats] = np. _____	max
	:	... if e.errno == errno.ENOENT: <EOL> continue <EOL> p = subprocess.Popen () <EOL> stdut = p. _____	communicate
		... command = shlex.split(command) <EOL> command = map(str, command) <EOL> env = os. _____	environ

Figure 27: The PY150-WILDS dataset comprises Python source code files taken from a variety of public repositories on GitHub. The task is code completion: predict token names given the context of previous tokens. We evaluate models on their accuracy on the subpopulation of API calls (i.e., method and class tokens), which are the most common code completion queries in real-world settings. Our goal is to learn code completion models that generalize to source code in new repositories that are not seen in the training set.

H.10.1. SETUP

Problem setting. We consider a hybrid domain generalization and subpopulation shift problem, where the domains are codebases (GitHub repositories), and our goal is to learn code completion models that generalize to source code written in new codebases. Concretely, the input x is a sequence of source code tokens taken from a single file, the label y is the next token (e.g., "environ", "communicate" in Figure 27), and the domain d is an integer that identifies the repository that the source code belongs to. We aim to solve both a domain generalization problem across codebases and improve subpopulation performance on class and methods tokens.

Data. The dataset comprises 150,000 Python source code files from 8,421 different repositories on GitHub (github.com). Each source code file is associated with the repository ID so that code from the same repository can be linked.

We split the dataset by randomly partitioning the data by repositories:

1. **Training:** 79,866 code files from 5,477 repositories.
2. **Validation (OOD):** 5,160 code files from different 261

repositories.

3. **Test (OOD):** 39,974 code files from different 2,471 repositories.
4. **Validation (ID):** 5,000 code files from the same repositories as the training set (but different files).
5. **Test (ID):** 20,000 code files from the same repositories as the training set (but different files).

The repositories are randomly distributed across the training, validation (OOD), and test (OOD) sets. As we use models pre-trained on the CodeSearchNet dataset (Husain et al., 2019), which partially overlaps with the Py150 dataset, we ensured that all GitHub repositories used in CodeSearchNet only appear in the training set in PY150-WILDS and not in the validation/test sets.

Table 24 shows the token statistics of the source code files, as well as the token type breakdown (e.g., class, method, punctuator, keyword, literal). The tokens are defined by the built-in Python tokenizer and the CodeGPT tokenizer, following Lu et al. (2021). Training and evaluation are conducted at the token-level (more details are provided below).

Evaluation. We evaluate models by their accuracy on predicting class and method tokens in the test set code files. This subpopulation metric is inspired by Hellendoorn et al. (2019), which finds that in real-world settings, developers primarily use code completion tools for completing class names and method names; in contrast, measuring average token accuracy would prioritize common tokens such as punctuators, which are often not a problem in real-world settings.

Potential leverage. We provide the GitHub repository that each source code files was derived from, which training algorithms can leverage. As programming tools like code completion are expected to be used across codebases in real applications (Nita & Notkin, 2010; Allamanis & Brockschmidt, 2017), it is important for models to learn generalizable representations of code and extrapolate well on unseen codebases. We hope that approaches using the provided repository annotations can learn to factor out common features and codebase-specific features, resulting in more robust models.

Additionally, besides the (integer) IDs of repositories, we also provide the repository names and file names in natural language as extra metadata. While we only use the repository IDs in our baseline experiments described below, the extra natural language annotations can potentially be leveraged as well to adapt models to target repositories/files.

H.10.2. BASELINE RESULTS

Model. For all experiments, we use the CodeGPT model (Lu et al., 2021) pre-trained on CodeSearchNet (Husain et al., 2019) as our model and finetune it on PY150-WILDS, using all the tokens in the training set. We tokenize input source code by the CodeGPT tokenizer and take blocks of length 256 tokens. We then train the CodeGPT model with a batch size of 6 (with $6 \times 256 = 1,536$ tokens), a learning rate of 8×10^{-5} , no L_2 regularization, and the AdamW optimizer (Loshchilov & Hutter, 2019) for 3 epochs with early stopping. Using the hyperparameters from Lu et al. (2021) as a starting point, we selected the above hyperparameters by a grid search over learning rates $\{8 \times 10^{-4}, 8 \times 10^{-5}, 8 \times 10^{-6}\}$ and L_2 regularization strength $\{0, 0.01, 0.1\}$. All other hyperparameters were simply set to standard/default values.

ERM results and performance drops. Table 25 shows that model performance on class and method tokens dropped substantially from 75.4% on the in-distribution repositories in the Test (ID) set to 67.9% on the out-of-distribution repositories in the Test (OOD) set. This gap shrinks if we evaluate the model on all tokens (instead of class and method tokens): accuracy drops from 74.5% on Test (ID) to 69.6% on Test (OOD). This is because the evaluation across all tokens includes many tokens that are used universally across repositories, such as punctuators and keywords.

Additional baseline methods. We trained CORAL, IRM, and Group DRO baselines, treating each repository as a domain. For CORAL and IRM, we find that the smaller penalties give slightly better generalization performance ($\lambda = 1$ for CORAL and $\lambda = 1$ for IRM). Compared to the ERM baseline, while CORAL and IRM reduced the performance gap between ID and OOD, neither of them improved upon ERM on the final OOD performance.

H.10.3. BROADER CONTEXT

Machine learning can aid programming and software engineering in various ways: automatic code completion (Raychev et al., 2014; Svyatkovskiy et al., 2019), program synthesis (Bunel et al., 2018; Kulal et al., 2019), program repair (Vasic et al., 2019; Yasunaga & Liang, 2020), code search (Husain et al., 2019), and code summarization (Allamanis et al., 2015). However, these systems face several forms of distribution shifts when deployed in practice. One major challenge is the shifts across codebases (which our PY150-WILDS dataset focuses on), where systems need to adapt to factors such as project content, coding conventions, or library or API usage in each codebase (Nita & Notkin, 2010; Allamanis & Brockschmidt, 2017). A second source of shifts is programming languages, which includes adaptation across different domain-specific languages (DSLs), e.g., in

Table 24: Token statistics for PY150-WILDS.

Split	#Files	#Total tokens	#Class	#Method	#Punctuator	#Keyword	#Literal
Training	79,866	14,129,619	894,753	789,456	4,512,143	1,246,624	1,649,653
Validation (ID)	5,000	882,745	55,645	48,866	282,568	77,230	105,456
Test (ID)	20,000	3,539,524	222,822	194,293	1,130,607	313,008	420,232
Validation (OOD)	5,160	986,638	65,237	56,756	310,914	84,677	111,282
Test (OOD)	39,974	7,340,433	444,713	412,700	2,388,151	640,939	869,083

Table 25: Baseline results on PY150-WILDS. We report both the model’s accuracy on predicting class and method tokens and accuracy on all tokens trained using ERM, CORAL, IRM and group DRO. Standard deviations over 3 trials are in parentheses.

Algorithm	Validation (ID)		Validation (OOD)		Test (ID)		Test (OOD)	
	Method/class	All	Method/class	All	Method/class	All	Method/class	All
ERM	75.5 (0.5)	74.6 (0.4)	68.0 (0.1)	69.4 (0.1)	75.4 (0.4)	74.5 (0.4)	67.9 (0.1)	69.6 (0.1)
CORAL	70.7 (0.0)	70.9 (0.1)	65.7 (0.2)	67.2 (0.1)	70.6 (0.0)	70.8 (0.1)	65.9 (0.1)	67.9 (0.0)
IRM	67.3 (1.1)	68.4 (0.7)	63.9 (0.3)	65.6 (0.1)	67.3 (1.1)	68.3 (0.7)	64.3 (0.2)	66.4 (0.1)
Group DRO	70.8 (0.0)	71.2 (0.1)	65.4 (0.0)	67.3 (0.0)	70.8 (0.0)	71.0 (0.0)	65.9 (0.1)	67.9 (0.0)

robotic environments (Shin et al., 2019); and across different versions of languages, e.g., Python 2 and 3 (Malloy & Power, 2017). Another challenge is the shift from synthetic training sets to real usage: for instance, (Hellendoorn et al., 2019) show that existing code completion systems, which are typically trained as language models on source code, perform poorly on the real completion instances that are most commonly used by developers in IDEs, such as API calls (class and method calls).

H.10.4. ADDITIONAL DETAILS

Data split. We generate the splits in the following steps. First, to avoid test set contamination, we took all of the repositories in CodeSearchNet (which, as a reminder, is used to pretrain our baseline model) and assigned them to the training set. Second, we randomly split all of the remaining repositories into three groups: Validation (OOD), Test (OOD), and Others. Finally, to generate the ID splits, we randomly split the files in the Others repositories into three sets: Training, Validation (ID), and Test (ID).

Modifications to the original dataset. The original Py150 dataset (Raychev et al., 2016) splits the total 150k files into 100k training files and 50k test files, regardless of the repository that each file was from. In PY150-WILDS, we re-split the dataset based on repositories to construct the aforementioned train, validation (ID), validation (OOD), test (ID), and test (OOD) sets.

Additionally, in the Py150 code completion task introduced in Lu et al. (2021), models are evaluated by the accuracy of predicting every token in source code. However, according to developer studies, this evaluation may include various tokens that are rarely used in real code completion, such as punctuators, strings, numerals, etc. (Robbes & Lanza, 2008;

Proksch et al., 2016; Hellendoorn et al., 2019). To define a task closer to real applications, in PY150-WILDS we focus on class name and method name prediction (which are used most commonly by developers).

I. Datasets with distribution shifts that do not cause performance drops

I.1. SQF: Criminal possession of weapons across race and locations

In this section, we provide more details on the stop-and-frisk dataset discussed in Section C.1. The original data was provided by the New York City Police Department, and has been widely used in previous ML and data analysis work (Goel et al., 2016; Zafar et al., 2017; Pierson et al., 2018; Kallus & Zhou, 2018; Srivastava et al., 2020). For our analysis, we use the version of the dataset that was processed by Goel et al. (2016). Our problem setting and dataset structure closely follow theirs.

I.1.1. SETUP

Problem setting. We study a subpopulation shift in a weapons prediction task, where each data point corresponds to a pedestrian who was stopped by the police on suspicion of criminal possession of a weapon. The input x is a vector that represents 29 observable features from the UF-250 stop-and-frisk form filled out by the officer after each stop: e.g., whether the stop was initiated based on a radio run or at an officer’s discretion, whether the officer was uniformed, and any reasons the officer gave for the stop (encoded as a categorical variable). Importantly, these features can all

be observed by the officer prior to making the stop.⁶ The binary label y is whether the pedestrian in fact possessed a weapon (i.e., whether the stop fulfilled its stated purpose). We consider, separately, two types of domains d : 1) race groups and 2) locations (boroughs in New York City). We consider location and race as our domains because previous work has shown that they can produce substantial disparities in policing practices and in algorithmic performance (Goel et al., 2016).

Data. Each row of the dataset represents one stop of one pedestrian. Following Goel et al. (2016), we filter for the 621,696 stops where the reason for the stop is suspicion of criminal possession of a weapon. We then filter for rows with complete data for observable features; with stopped pedestrians who are Black, white, or Hispanic; and who are stopped during the years 2009–2012 (the time range used in Goel et al. (2016)). These filters yield a total of 506,283 stops, 3.5% of which are positive examples (in which the officer finds that the pedestrian is illegally possessing a weapon).

The training versus validation split is a random 80%-20% partition of all stops in 2009 and 2010. We test on stops from 2011–2012; this follows the experimental setup in Goel et al. (2016). Overall, our data splits are as follows:

1. **Training:** 241,964 stops from 2009 and 2010.
2. **Validation:** 60,492 stops from 2009 and 2010, disjoint from the training set.
3. **Test:** 289,863 stops from 2011 and 2012.

In the experiments below, we do not use the entire training set, as we observed in our initial experiments that the model performed less well on certain subgroups (Black pedestrians and pedestrians from the Bronx). To determine whether this inferior performance might be ameliorated by training specifically on those groups, we controlled for training set size by downsampling the training set to the size of the disadvantaged population of interest for a given split. Specifically, we consider the following (overlapping) training subsets, each of which is subsampled from the overall training set described above:

1. **Black pedestrians only:** 155,929 stops of Black pedestrians from 2009 and 2010.
2. **All pedestrians, subsampled to # Black pedestrians:** 155,929 stops of all pedestrians from 2009 and 2010.

⁶When we consider subpopulation shifts over race groups, the input x additionally includes 75 one-hot indicators corresponding to the precinct that the stop was made in. We do not include those features when we consider shifts over locations, as they prevent the model from generalizing to new locations.

3. **Bronx pedestrians only:** 69,129 stops of pedestrians in the Bronx from 2009 and 2010.

4. **All pedestrians, subsampled to # Bronx pedestrians:** 69,129 stops of all pedestrians from 2009 and 2010.

Evaluation. Our metric for classifier performance is the precision for each race group and each borough at a global recall of 60%—i.e., when using a threshold which recovers 60% of all weapons in the test data, similar to the recall evaluated in Goel et al. (2016). The results are similar when using different recall thresholds. Examining the precision for each race/borough captures the fact, discussed in Goel et al. (2016), that very low-precision stops may violate the Fourth Amendment, which requires *reasonable suspicion* for conducting a police stop; thus, the metric encapsulates the intuition that the police are attempting to avoid Fourth Amendment violations for any race group or borough while still recovering a substantial fraction of the illegal weapons.

I.1.2. BASELINE RESULTS

Model. For all experiments, we use a logistic regression model trained with the Adam optimizer (Kingma & Ba, 2015) and early stopping. We trained one model on each of the 4 training sets, separately picking hyperparameters through a grid search across 7 learning rates logarithmically-spaced in $[5 \times 10^{-8}, 5 \times 10^{-2}]$ and batch sizes in $\{4, 8, 16, 32, 64\}$. Table 28 provides the hyperparameters used for each training set. All models were trained with a reweighted cross-entropy objective that upsampled the positive examples to achieve class balance.

ERM results and performance drops. Performance differed substantially across race and location groups: precision was lowest on Black pedestrians (Table 26, top row) and pedestrians in the Bronx (Table 27, top row). To assess whether in-distribution training would improve performance on these groups, we trained the model only on Black pedestrians (Table 26, bottom row) and pedestrians in the Bronx (Table 27, bottom row). However, this did not substantially improve performance on Black pedestrians or pedestrians from the Bronx; the difference in precision was less than 0.005 for both groups relative to the original model trained on all races and locations. This is consistent with the fact that groups with the lowest performance are not necessarily small minorities of the dataset: for example, more than 90% of the stops are of Black or Hispanic pedestrians, but performance on these groups is worse than that for white pedestrians. The lack of improvement from in-distribution training suggests that approaches like group DRO would be unlikely to further improve performance, and we thus did not assess these approaches.

Table 26: Comparison of precision scores for each race group at 60% global weapon recall. Train set size is 69,129 for both rows.

Training dataset	Precision at 60% recall		
	Black	Hispanic	White
Black pedestrians only	0.131	0.174	0.360
All pedestrians, subsampled to # Black pedestrians	0.135	0.183	0.362

Table 27: Comparison of precision scores for each borough at a threshold which achieves 60% global weapon recall. Train set size is 155,929 for both rows.

Training dataset	Precision at 60% recall				
	Bronx	Brooklyn	Manhattan	Queens	Staten Island
Bronx pedestrians only	0.074	0.158	0.207	0.157	0.105
All pedestrians, subsampled to # Bronx pedestrians	0.075	0.162	0.224	0.168	0.107

Table 28: Model parameters used in this analysis. Learning rates are rounded to the first significant digit.

Training data	Batch size	Learning rate	Number of epochs
Black pedestrians only	4	5e-04	1
All pedestrians, subsampled to # Black pedestrians	4	5e-04	4
Bronx pedestrians only	4	5e-04	2
All pedestrians, subsampled to # Bronx pedestrians	4	5e-03	4

Discussion. We observed large disparities in performance across race and location groups. However, the fact that in-distribution training did not ameliorate these disparities suggests that they do not occur because some groups comprise small minorities of the original dataset, and thus suffer worse performance. Instead, our results suggest that classification performance on some race and location groups are intrinsically noisier; it is possible, for example, that collection of additional features would be necessary to improve performance on these groups (Chen et al., 2018).

I.1.3. ADDITIONAL DETAILS

Modifications to the original dataset. The features we use are very similar to those used in Goel et al. (2016). The two primary differences are that 1) we remove features which convey information about a stopped pedestrian’s race, since those might be illegal to use in real-world policing contexts and 2) we do not include a “local hit rate” feature which captures the fraction of historical stops in the vicinity of a stop which resulted in discovery of a weapon; we omit this latter feature because it was unnecessary to match performance in Goel et al. (2016).

I.2. BDD100K: Object recognition in autonomous driving across locations

As discussed in Section C.6, autonomous driving, and robotics in general, is an important application that requires

effective and robust tools for handling distribution shift. Here, we discuss our findings on a modified version of the BDD100K dataset that evaluates on shifts based on time of day and location. Our results below suggest that more challenging tasks, such as object detection and segmentation, may be more suited to evaluations of distribution shifts in an autonomous driving context.

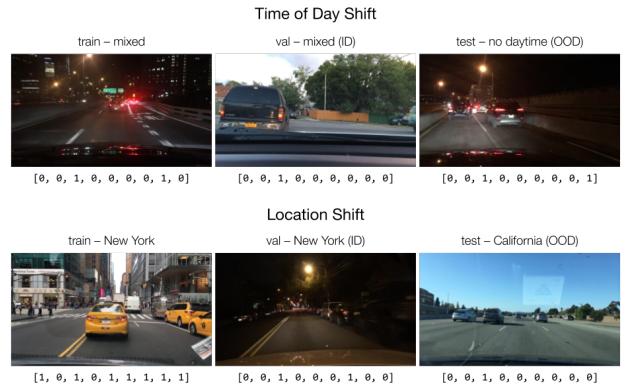


Figure 28: For BDD100K, we study two different types of shift, based on time of day and location. We visualize randomly chosen images and their corresponding labels from the training, validation, and test splits for both shifts. The labels are 9-dimensional binary vectors indicating the presence (1) or absence (0) of, in order: bicycles, buses, cars, motorcycles, pedestrians, riders, traffic lights, traffic signs, and trucks.

Table 29: Average multi-task classification accuracy of ERM trained models on BDD100K. All results are reported across 3 random seeds, with standard deviation in parentheses. We observe no substantial drops in the presence of test time distribution shifts.

Algorithm	Time of day shift		Location shift	
	Validation (ID)	Test (OOD)	Validation (ID)	Test (OOD)
ERM	87.1 (0.3)	89.7 (0.2)	87.9 (0.0)	86.9 (0.0)

I.2.1. SETUP

Task. In line with the other datasets in WILDS, we evaluate using a classification task. Specifically, the task is to predict whether or not 9 different categories appear in the image x : bicycles, buses, cars, motorcycles, pedestrians, riders, traffic lights, traffic signs, and trucks. This is a multi-task binary classification problem, and the label y is thus a 9-dimensional binary vector.

Data. The BDD100K dataset is a large and diverse driving dataset crowd-sourced from tens of thousands of drivers, covering four different geographic regions and many different times of day, weather conditions, and scenes (Yu et al., 2020). The original dataset contains 80,000 images in the combined training and validation sets and is richly annotated for a number of different tasks such as detection, segmentation, and imitation learning. We use bounding box labels to construct our task labels, and as discussed later, we use location and image tags to construct the shifts we evaluate.

Evaluation. In evaluating the trained models, we consider average accuracy across the binary classification tasks, averaged over each of the validation and test sets separately. We next discuss how we create and evaluate two different types of shift based on time of day and location differences.

I.2.2. TIME OF DAY SHIFT

Distribution shift and evaluation. We evaluate two different types of shift, depicted in Figure 28. For time of day shift (Figure 28 top row), we use the original BDD100K training set, which has roughly equal proportions of daytime and non daytime images (Yu et al., 2020). However, we construct a test set using the original BDD100K validation set that only includes non-daytime images. We then split roughly the same number of images randomly from the training set to form an in-distribution validation set. There are 64,993, 4,860, and 4,742 images in the training, validation, and test splits, respectively.

ERM results. Table 29 summarizes our findings. For time of day shift, we actually observe slightly *higher* test performance, on only non daytime images, than validation performance on mixed daytime and non daytime images. We contrast this with findings from Dai & Van Gool (2018);

Yu et al. (2020), who showed worse test performance for segmentation and detection tasks, respectively, on non daytime images. We believe this disparity can be attributed to the difference in tasks – for example, it is likely more difficult to draw an accurate bounding box for a car at night than to simply recognize tail lights and detect the presence of a car.

I.2.3. LOCATION SHIFT

Distribution shift. For location shift (Figure 28 bottom row), we combine all of the data from the original BDD100K training and validation sets. We construct training and validation sets from all of the images captured in New York, and we use all images from California for the test set. The validation set again is in distribution with respect to the training set and has roughly the same number of images as the test set. There are 53,277, 9,834, and 9,477 images in the training, validation, and test splits, respectively.

ERM results. In the case of location shift, we see from Table 29 that there is a small drop in performance, possibly because this shift is more drastic as the locations are disjoint between training and test time. However, the performance drop is relatively small and the test time accuracy is still comparable to validation accuracy. In general, we believe that these results lend support to the conclusion that, for autonomous driving and robotics applications, other more challenging tasks are better suited for evaluating performance. Generally speaking, incorporating a wide array of different applications will likely require a simultaneous effort to incorporate different tasks as well. table*

I.3. Amazon: Sentiment classification across different categories and time

Our benchmark dataset AMAZON-WILDS studies user shifts. In Section 7, we discussed empirical trends on other types of distribution shifts on the same underlying 2018 Amazon Reviews dataset (Ni et al., 2019). We now present the detailed setup and empirical results for the time and category shifts.

I.3.1. SETUP

Model. For all experiments in this section, we finetune BERT-base-uncased models, using the implementation from Wolf et al. (2019), and with the following hyperparam-

eter settings: batch size 8; learning rate 2×10^{-6} ; L_2 -regularization strength 0.01; 3 epochs; and a maximum number of tokens of 512. These hyperparameters are taken from the AMAZON-WILDS experiments.

I.3.2. TIME SHIFTS

Problem setting. We consider the domain generalization setting, where the domain d is the year in which the reviews are written. As in AMAZON-WILDS, the task is multi-class sentiment classification, where the input x is the text of a review, the label y is a corresponding star rating from 1 to 5.

Data. The dataset is a modified version of the Amazon Reviews dataset (Ni et al., 2019) and comprises customer reviews on Amazon. Specifically, we consider the following split:

1. **Training:** 1,000,000 reviews written in years 2000 to 2013.
2. **Validation (OOD):** 20,000 reviews written in years 2014 to 2018.
3. **Test (OOD):** 20,000 reviews written in years 2014 to 2018.

To construct the above split, we first randomly sample 4,000 reviews per year for the evaluation splits. For years in which there are not sufficient reviews, we split the reviews equally between validation and test. After constructing the evaluation set, we then randomly sample from the remaining reviews to form the training set.

Evaluation. To assess whether models generalize to future years, we evaluate models by their average accuracy on the OOD test set.

ERM results and performance drops. We only observed modest performance drops due to time shift. Our baseline model performs well on the OOD test set, achieving 76.0% accuracy on average and 75.4% on the worst year (Table 30). To measure performance drops due to distribution shifts, we compared the above results with an oracle in-distribution baseline model, which is trained on reviews written in years 2014 to 2018 (Table 31). The performance gaps with the in-distribution baseline model are consistent but modest across the years, with the biggest drop of 1.1% for 2018.

I.3.3. CATEGORY SHIFTS

Shifts across categories—where a model is trained on reviews in one category and then tested on another—have been studied extensively (Blitzer et al., 2007; Mansour et al., 2009; Hendrycks et al., 2020c). In line with prior work, we

observe that model performance drops upon evaluating on a few unseen categories. However, the observed difference between out-of-distribution and in-distribution baselines varies from category to category and is not consistently large (Hendrycks et al., 2020c). In addition, we find that training on more diverse data with more product categories tends to improve generalization to unseen categories and reduce the effect of the distribution shift; similar phenomena have also been reported in prior work (Mansour et al., 2009; Guo et al., 2018).

Problem setting. We consider the domain generalization setting, where the domain d is the product category. As in AMAZON-WILDS, the task is multi-class sentiment classification, where the input x is the text of a review, the label y is a corresponding star rating from 1 to 5.

Data. The dataset is a modified version of the Amazon Reviews dataset (Ni et al., 2019) and comprises customer reviews on Amazon. Specifically, we consider the following split for a given set of training categories:

1. **Training:** up to 1,000,000 reviews in training categories.
2. **Validation (OOD):** reviews in categories unseen during training.
3. **Test (OOD):** reviews in categories unseen during training.
4. **Validation (ID):** reviews in training categories.
5. **Test (ID):** reviews in training categories.

To construct the above split, we first randomly sample 1,000 reviews per category for the evaluation splits (for categories with insufficient number of reviews, we split the reviews equally between validation and test) and then randomly sample from the remaining reviews to form the training set.

Evaluation. To assess whether models generalize to unseen categories, we evaluate models by their average accuracy on each of the categories in the OOD test set.

ERM results. We first considered training on four categories (Books, Movies and TV, Home and Kitchen, and Electronics) and evaluating on unseen categories. We observed that a BERT-base-uncased model trained via ERM yields a test accuracy of 75.4% on the four in-distribution categories and a wide range of accuracies on unseen categories (Table 32, columns Multiple). While the accuracies on some unseen categories are lower than the in-distribution accuracy, it is unclear whether the performance gaps stem from the distribution shift or differences in intrinsic difficulty across categories; in fact, the accuracy is higher

Table 30: Baseline results on time shifts on the Amazon Reviews Dataset. We report the accuracy of models trained using ERM. In addition to the average accuracy across all years in each split, we report the accuracy for the worst-case year.

Algorithm	Train		Validation (OOD)		Test (OOD)	
	Average	Worst year	Average	Worst year	Average	Worst year
ERM	75.0 (0.0)	72.4 (0.1)	75.7 (0.1)	74.6 (0.1)	76.0 (0.1)	75.4 (0.1)

Table 31: Comparison with in-distribution baselines for time shifts on Amazon Reviews Dataset. We observe only modest performance drops due to time shifts.

Year	2014	2015	2016	2017	2018
OOD baseline (ERM)	75.4 (0.1)	75.8 (0.1)	76.3 (0.1)	76.4 (0.4)	76.1 (0.1)
ID baseline (oracle)	76.1 (0.2)	76.8 (0.1)	77.1 (0.2)	77.5 (0.2)	77.0 (0.0)

Table 32: Baseline results on category shifts on the Amazon Reviews Dataset. We report the accuracy of models trained using ERM on a single category (Books) versus four categories (Books, Movies and TV, Home and Kitchen, and Electronics). Across many categories unseen at training time, corresponding to each row, the latter model modestly but consistently outperforms the former.

Category	Validation (OOD)		Test (OOD)	
	Single	Multiple	Single	Multiple
All Beauty	87.8 (0.8)	85.6 (1.4)	82.9 (0.8)	83.1 (0.8)
Arts Crafts and Sewing	81.6 (0.7)	83.4 (0.4)	79.5 (0.2)	81.7 (0.2)
Automotive	78.2 (0.4)	80.4 (0.4)	76.5 (0.2)	78.9 (0.2)
CDs and Vinyl	78.1 (0.7)	78.6 (0.2)	78.5 (0.7)	79.7 (0.3)
Cell Phones and Accessories	76.8 (0.3)	79.0 (0.7)	78.0 (0.5)	80.2 (1.0)
Clothing Shoes and Jewelry	69.8 (0.6)	72.6 (0.2)	73.3 (0.2)	75.2 (0.2)
Digital Music	77.5 (0.5)	77.8 (0.5)	80.7 (1.0)	81.7 (0.6)
Gift Cards	88.2 (1.5)	90.7 (3.1)	90.7 (0.8)	91.2 (0.0)
Grocery and Gourmet Food	79.0 (0.3)	79.0 (0.1)	79.3 (0.7)	79.2 (0.2)
Industrial and Scientific	77.0 (0.4)	78.1 (0.6)	77.4 (0.2)	78.9 (0.1)
Kindle Store	75.0 (0.3)	74.5 (0.3)	73.2 (0.3)	73.1 (0.5)
Luxury Beauty	67.2 (0.2)	70.2 (0.6)	67.4 (0.7)	69.4 (0.9)
Magazine Subscriptions	74.2 (3.2)	71.0 (0.0)	90.3 (0.0)	89.2 (1.9)
Musical Instruments	76.1 (0.3)	78.3 (0.3)	78.8 (0.8)	80.9 (0.2)
Office Products	78.5 (0.3)	80.0 (0.5)	76.7 (0.5)	78.9 (0.4)
Patio Lawn and Garden	70.8 (0.6)	72.9 (0.3)	75.5 (0.6)	79.7 (0.6)
Pet Supplies	74.5 (0.4)	77.1 (0.9)	74.4 (0.4)	76.8 (0.5)
Prime Pantry	80.5 (0.3)	80.2 (0.2)	78.5 (0.6)	79.4 (0.3)
Software	65.8 (1.7)	67.1 (1.1)	71.3 (1.5)	72.6 (0.5)
Sports and Outdoors	74.2 (0.5)	76.0 (0.2)	75.8 (0.2)	78.3 (0.6)
Tools and Home Improvement	74.0 (1.1)	76.4 (0.3)	73.1 (0.6)	74.4 (0.2)
Toys and Games	78.9 (0.4)	79.9 (0.2)	77.6 (0.2)	80.9 (0.2)
Video Games	76.0 (0.2)	76.6 (0.8)	76.9 (0.6)	78.0 (0.6)

on many unseen categories (e.g., All Beauty) than on the in-distribution categories, illustrating the importance of accounting for intrinsic difficulty.

In an attempt to control for this and measure performance drops due to distribution shifts, we compared the above model against an oracle in-distribution model, which is trained on each target category. We controlled for the number of training reviews to the extent possible; the main, non-oracle model is trained on 1 million reviews, and each oracle model is trained on 1 million reviews or less, as limited by the number of reviews per category. We observed performance drops on some categories, for example

on Clothing, Shoes, and Jewelry (83.0% with the oracle model versus 75.2% with the main model trained on the four different categories) and on Pet Supplies (78.8% to 76.8%). However, on the remaining categories, we observed more modest performance gaps, if at all. While we thus found no evidence for significant performance drops for many categories, these results do not rule out such drops either: one confounding factor is that some of the oracle models are trained on significantly smaller training sets and therefore underestimate the in-distribution performance.

In addition, we compared training on four categories (Books, Movies and TV, Home and Kitchen, and Electronics), as

above, to training on just one category (Books), while keeping the training set size constant. We found that decreasing the number of training categories in this way lowered out-of-distribution performance: across many OOD categories, accuracies were modestly but consistently higher for the model trained on four categories than for the model trained on a single category (Table 32).

I.4. Yelp: Sentiment classification across different users and time

We present empirical results on time and user shifts in the Yelp Open Dataset⁷.

I.4.1. SETUP

Model. For all experiments in this section, we finetune BERT-base-uncased models, using the implementation from Wolf et al. (2019), and with the following hyperparameter settings: batch size 8; learning rate 2×10^{-6} ; L_2 -regularization strength 0.01; 3 epochs with early stopping; and a maximum number of tokens of 512. We select the above hyperparameters based on a grid search over learning rates $1 \times 10^{-6}, 2 \times 10^{-6}, 1 \times 10^{-5}, 2 \times 10^{-5}$, using the time shift setup; for the user shifts, we adopted the same hyperparameters.

I.4.2. TIME SHIFTS

Problem setting. We consider the domain generalization setting, where the domain d is the year in which the reviews are written. As in AMAZON-WILDS, the task is multi-class sentiment classification, where the input x is the text of a review, the label y is a corresponding star rating from 1 to 5.

Data. The dataset is a modified version of the Yelp Open Dataset and comprises 1 million customer reviews on Yelp. Specifically, we consider the following split:

1. **Training:** 1,000,000 reviews written in years 2006 to 2013.
2. **Validation (OOD):** 20,000 reviews written in years 2014 to 2019.
3. **Test (OOD):** 20,000 reviews written in years 2014 to 2019.

To construct the above split, we first randomly sample 1,000 reviews per year for the evaluation splits. For years in which there are not sufficient reviews, we split the reviews equally between validation and test. After constructing the evaluation set, we then randomly sample from the remaining reviews to form the training set.

⁷<https://www.yelp.com/dataset>

Evaluation. To assess whether models generalize to future years, we evaluate models by their average accuracy on the OOD test set.

ERM results and performance drops. We observe modest performance drops due to time shift. A BERT-base-uncased model trained with the standard ERM objective performs well on the OOD test set, achieving 76.0% accuracy on average and 73.9% on the worst year (Table 33). To measure performance drops due to distribution shifts, we compare the above results with an oracle in-distribution baseline model, which is trained on reviews written in years 2014 to 2019 (Table 34). While there are consistent performance gaps between the out-of-distribution and the in-distribution baselines in later years, they are modest in magnitude with the largest drop of 3.1% for 2018.

I.4.3. USER SHIFT

Problem setting. As in AMAZON-WILDS, we consider the domain generalization setting, where the domains are reviewers and the task is multi-class sentiment classification. Concretely, the input x is the text of a review, the label y is a corresponding star rating from 1 to 5, and the domain d is the identifier of the user that wrote the review.

Data. The dataset is a modified version of the Yelp Open Dataset and comprises 1.2 million customer reviews on Yelp. To measure generalization to unseen reviewers, we train on reviews written by a set of reviewers and consider reviews written by *unseen* reviewers at test time. Specifically, we consider the following random split across reviewers:

1. **Training:** 1,000,104 reviews from 11,856 reviewers.
2. **Validation (OOD):** 40,000 reviews from another set of 1,600 reviewers, distinct from training and test (OOD).
3. **Test (OOD):** 40,000 reviews from another set 1,600 reviewers, distinct from training and validation (OOD).
4. **Validation (ID):** 40,000 reviews from 1,600 of the 11,856 reviewers in the training set.
5. **Test (ID):** 40,000 reviews from 1,600 of the 11,856 reviewers in the training set.

The training set includes at least 25 reviews per reviewer, whereas the evaluation sets include exactly 25 reviews per reviewer. While we primarily evaluate model performance on the above OOD test set, we also provide in-distribution validation and test sets for potential use in hyperparameter tuning and additional reporting. These in-distribution splits comprise reviews written by reviewers in the training set.

Evaluation. To assess whether models perform consistently well across reviewers, we evaluate models by their accuracy on the reviewer at the 10th percentile.

ERM results and performance drops. We observe modest variations in performance across reviewers. A BERT-base-uncased model trained with the standard ERM objective achieves 71.5% accuracy on average and 56.0% accuracy at the 10th percentile reviewer (Table 35). The above variation is modestly larger than expected from randomness; a random binomial baseline with equal average accuracy would have a tenth percentile accuracy of 60.1%.

Table 33: Baseline results on time shifts on the Yelp Open Dataset. We report the accuracy of models trained using ERM.

Algorithm	Train		Validation (OOD)		Test (OOD)	
	Average	Worst year	Average	Worst year	Average	Worst year
ERM	71.4 (0.7)	65.7 (1.1)	76.1 (0.1)	73.1 (0.2)	76.0 (0.4)	73.9 (0.4)

Table 34: Comparison with in-distribution baselines for time shifts on Yelp Open Dataset. We observe only modest performance drops due to time shifts.

Year	2014	2015	2016	2017	2018	2019
OOD baseline (ERM)	75.8 (0.6)	75.2 (0.9)	73.9 (0.4)	77.0 (0.4)	76.7 (0.3)	77.2 (0.6)
ID baseline (oracle)	75.2 (0.5)	75.0 (0.5)	76.4 (0.7)	78.8 (0.6)	79.6 (0.4)	79.5 (0.5)

Table 35: Baseline results on user shifts on the Yelp Open Dataset. We report the accuracy of models trained using ERM. In addition to the average accuracy across all reviews, we compute the accuracy for each reviewer and report the performance for the reviewer in the 10th percentile.

Algorithm	Validation (OOD)		Test (OOD)		Validation (ID)		Test (ID)	
	10th percentile	Average	10th percentile	Average	10th percentile	Average	10th percentile	Average
ERM	56.0 (0.0)	70.5 (0.0)	56.0 (0.0)	71.5 (0.0)	56.0 (0.0)	70.6 (0.0)	56.0 (0.0)	70.9 (0.1)