Dataset Projection: Finding Target-aligned Subsets of Auxiliary Data

Anonymous authors
Paper under double-blind review

Abstract

To obtain more training data for a target task, one can draw upon related but distinct datasets, or auxiliary datasets. We put forth the problem of *dataset projection*—finding subsets of auxiliary datasets that are most aligned with a target dataset. These so-called projected datasets can be used as training data to improve performance on target tasks while being substantially smaller than the auxiliary dataset. We then develop a framework for solving such dataset projection problems and demonstrate in a variety of vision and language settings that the resulting projected datasets, when compared to the original auxiliary datasets, (1) are closer approximations of target datasets and (2) can be used to improve test performance or provide analysis for the target datasets.

1 Introduction

The use of larger datasets has been a key driver of recent progress in machine learning. Indeed, language and computer vision models trained on datasets containing billions of documents (Brown et al., 2020) and images (Sun et al., 2017; Mahajan et al., 2018) make the classic ImageNet (Russakovsky et al., 2015) benchmark look small in comparison. So, when training machine learning models, there is a strong inclination to use as much data as possible. This can amount to the costly process of collecting and labeling more data, or incorporating *auxiliary data* from related but separate datasets.

Indeed, auxiliary datasets can be used to improve generalization (Peng et al., 2019; Beery et al., 2020) and out of distribution robustness (Schneider et al., 2020; Mårtensson et al., 2020). They can also give rise to learning richer representations (Maurer et al., 2016; Qiu et al., 2021) and better initializations via pretraining (Radford et al., 2018; Caron et al., 2019). All of these reinforce the conventional wisdom that *more data leads to better performance* (Rosenfeld et al., 2020; Kaplan et al., 2020). As a result, when given the option of using an auxiliary dataset, a typical strategy is to use all of it.

However, blindly using auxiliary data turns out to hurt model performance in certain cases. For example, pooling medical imaging data from multiple hospitals causes models to detect hospital sources instead of just medical symptoms (Zech et al., 2018). Large language datasets can over-represent certain populations while excluding marginalized populations (Bender et al., 2021). The widely used ImageNet benchmark (Russakovsky et al., 2015) is often used as an auxiliary dataset for other tasks, yet was shown to contain spurious correlations such as men holding fish (Xiao et al., 2021). In light of the fact that auxiliary data may include irrelevant or harmful biases, how can we best use auxiliary data for a particular task while avoiding potential downsides?

To answer this question, we consider the task of identifying a subset of the auxiliary data that is most aligned with the target data, which we call the *dataset projection* problem. As we will demonstrate, using the "right" subset of auxiliary data can be important for certain target tasks. In particular, intelligently and automatically selected subsets of auxiliary data have the potential to decrease sources of unwanted biases, amplify useful features, and ultimately improve task performance.

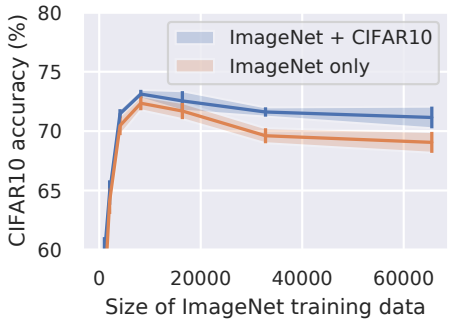


Figure 1: Adding auxiliary ImageNet data to a small CIFAR10 dataset (blue curve) can improve performance, but adding too much hurts performance. Training exclusively on auxiliary ImageNet data isolates this behavior (orange curve). This indicates that ImageNet contains patterns that hurt CIFAR10 generalization.

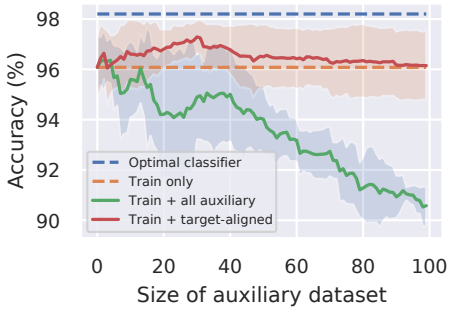


Figure 2: Accuracy of a linear classifier when augmenting a small dataset with biased auxiliary data for the Gaussian example in Figure 3. Increasing the amount of auxiliary data causes the classifier to lose accuracy. Adding target-aligned data can mitigate the bias and improve performance over the original training data.

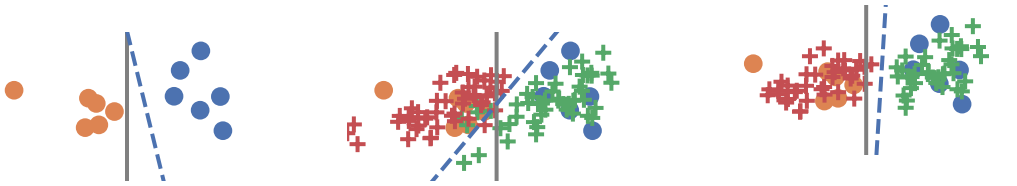
Our contributions. In this paper, we analyze the limitations of indiscriminately using auxiliary data, and develop methods to better leverage such data. Specifically:

1. Through three experiments, we exhibit scenarios where using the entire auxiliary dataset can hurt model performance, and demonstrate how manually selecting what auxiliary data to use improves performance.
2. We formulate the problem of projecting datasets, and develop methods to solve this problem that can *automatically* find target-aligned subsets of auxiliary datasets.
3. We demonstrate on a variety of vision and language tasks that our methods find projected datasets that are better approximations of target datasets than the original auxiliary dataset. In particular, we find that these datasets can improve downstream performance when augmenting the target dataset, even when compared to the original auxiliary dataset.
4. Our framework enables a new, model-free analysis that uses projections of auxiliary to characterize the composition of the target dataset in.

2 Can indiscriminate use of auxiliary datasets hurt performance?

The current trend in machine learning is to use more data when it is available. However, recent studies have raised concerns about the unintended consequences of blindly incorporating new dataset sources. For instance, pneumonia detectors learned to identify hospital-specific markers rather than pneumonia itself when data was pooled (Zech et al., 2018). Furthermore, combining data from multiple surveys can introduce new sources of sampling error and nonresponse bias (Lohr & Raghunathan, 2017).

Motivating example with CIFAR10 and ImageNet. The consequences of using auxiliary data manifest themselves in commonly-studied machine learning settings as well. Suppose we want to train a CIFAR10 classifier, but only have a limited amount of CIFAR10 data. To improve performance, we can augment our limited CIFAR10 dataset using relevant classes of ImageNet as auxiliary data. As we increase the amount of ImageNet data, performance initially improves (see blue curve in Figure 1). However, we find that adding *too much* ImageNet data degrades performance instead.



(a) Training data (b) Training + auxiliary data (c) Training + target-aligned data

Figure 3: A synthetic example showing how auxiliary data can harm performance. The optimal and estimated linear classifiers are plotted in the solid and dashed lines respectively. (a) Training data is sampled from Gaussian distributions, and (b) auxiliary data comes from a skewed Gaussian that biases the estimated classifier. (c) Subsampling the auxiliary data to be aligned with training data can reduce the auxiliary bias and improve performance.

Why does this happen? It turns out that the model is overfitting to the patterns in the ImageNet data. This trend becomes even clearer when we train a model only on the data sourced from ImageNet (see orange curve in Figure 1). While some amount of ImageNet data alone can provide useful features for CIFAR10, too much ImageNet data introduces irrelevant patterns that hurt performance.

Linear example. To better understand this phenomenon, we construct a simplified linear example which replicates the empirical trends observed with CIFAR10 and ImageNet. Consider a binary classification problem with labels $y \in \{-1, +1\}$ and data x drawn from a class conditional Gaussian distribution with mean $y \cdot \mu$:

$$x \sim \mathcal{N}(y \cdot \mu, I) \tag{1}$$

With only a few training datapoints, the estimated linear classifier can have high error (see Figure 3a). To improve performance, we can augment the training data with an auxiliary dataset. In this case, our auxiliary data z comes from a skewed and rotated Gaussian distribution instead:

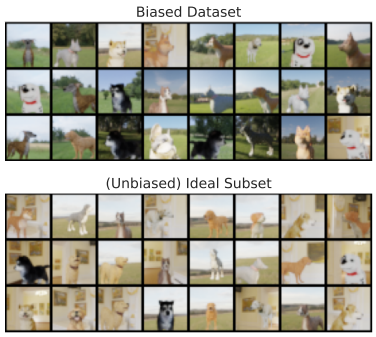
$$z \sim \mathcal{N}(y \cdot \mu, \Sigma) \tag{2}$$

where Σ is different from the identity. Adding too much of this auxiliary data biases the estimated linear classifier and actually hurts performance (see Figure 3b).

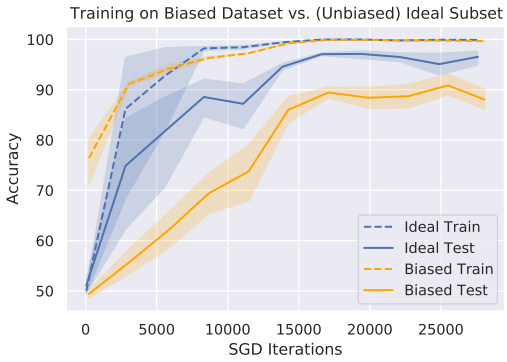
However, not all of the auxiliary data is harmful. Ideally, we would like to keep auxiliary data that is most aligned with our original training dataset, and discard irrelevant data. A natural way to achieve this is to fit multivariate Gaussian distributions to the training data, and restrict the auxiliary data to those that fall within a 95% confidence region (see Figure 3c). Augmenting the training data with this target-aligned subset reduces the bias and improves performance, as measured in Figure 2.

Controlled synthetic setting. We further recreate this phenomenon in a controlled, synthetic setting to show that indiscriminately using auxiliary data can be harmful due to spurious correlations. Our synthetic setting involves rendering photorealistic images with Blender and the 3DB framework (Leclerc et al., 2021). First, we generate a biased auxiliary dataset of cats and dogs, where 90% of cats are indoors and 90% of dogs are outdoors. For the target test dataset, we generate an unbiased dataset where the background is not correlated with the class at all. Training models on the full auxiliary dataset (containing the spurious correlation) leads to an average test accuracy of 90.4% while training models on a manually chosen subset of the auxiliary dataset (without the spurious correlation) leads to a significantly higher average test accuracy of 97.6%. Here, using strictly more data leads to worse performance due to the spurious background correlation in the full auxiliary dataset. Figure 4 shows examples of the biased auxiliary data and the unbiased subset.

Full details for reproducing all three of these examples are in Appendix A.



(a) Biased dataset and unbiased ideal subset of synthetic images generated via 3DB.



(b) Train and test accuracy curves.

Figure 4: (a) Training on the 15k-image biased dataset (top) results in 7% lower accuracy than training on the 3k-image unbiased, ideal subset of the dataset (bottom) when the model is tested on an unbiased test set. (b) Accuracy curves for each training set.

3 Dataset projection

Motivated by the limits of indiscriminately using auxiliary datasets, we aim to answer: *How can we extract the relevant portions of auxiliary datasets for a target dataset?* Unlike two of the three examples in Section 2 (linear example and the controlled synthetic example), real-world datasets do not always have clear patterns for manually filtering out irrelevant auxiliary data. Instead, we would like an automatic way to extract a target-aligned subset of auxiliary data.

More formally, let p be the target data distribution, and let q be the auxiliary data distribution. We assume that q is composed of k source distributions, $q_1, \ldots, q_k$. Our goal is to find a linear combination of source distributions $\{q_i\}$ that best matches the target distribution p. We can formulate this task as the following optimization problem:

$$\begin{aligned} &\min_{\alpha} \mathbb{E}_{X\sim p, Y\sim \hat{q}}\left[d(X,Y)\right] \\ &\text{subject to } \ \hat{q} = \sum_{i=1}^{k} \alpha_i q_i, \ \ \sum_{i=1}^{k} \alpha_i = 1, \ \ \alpha_i \geq 0 \end{aligned} \tag{3}$$

where d is a distance metric over datasets, (X, Y) are datasets sampled from $(p, \hat{q})$, and α_i denotes the proportion of each source distribution q_i used for approximating the target distribution p. Intuitively, this can be thought of as "projecting" the target distribution p onto the space spanned by source distributions q_i. Hence, we refer to this optimization problem as *dataset projection*.

Note that this formulation is well-defined for *any* two datasets drawn from distributions p and q. For example, both the source and the target could be unlabeled data, or subsets/concatenations of datasets. For the remainder of this paper, we will assume the target distribution q to be a class-wise subset of the target dataset, and leave the source distribution completely unrestricted and potentially unlabeled. Projecting target classes allows us to then subselect and pseudo-label source data for the target task. This differs from the closely-related field of domain adaptation, which typically has the opposite assumption of a labeled source distribution and an unrestricted target.

Source distributions. To apply our dataset projection framework, an auxiliary distribution q needs to be split into multiple source distributions q_i. How can we get these source distributions? In supervised settings, a natural way to split an auxiliary dataset is to use existing class or attribute labels. In completely unsupervised settings, we can automatically split an auxiliary distribution with unsupervised clustering methods. These two approaches can be combined to generate even finer-grained source distributions.

Algorithm 1 Active set algorithm for projecting datasets with soft active set estimate A_i that helps to dampen and stabilize the updates

1: $\alpha_i^0 = 1/k$, $A_i = 0.5$ for $i = 1 \dots k$ *// Initialize feasible point & soft active set estimate*
2: **for** $t = 0, 1, \dots$ **do**
3: **if** α^t is stationary point **then**
4: Return α^t
5: **end if**
6: $g = \nabla f(\alpha^t)$ *// Estimate numerical gradient of the dataset projection objective*
7: $(\tilde{\alpha}^t, \tilde{A}^t) = \texttt{DampedUpdate}(\alpha^t, A^t, g)$ *// Active set update*
8: $(\alpha^{t+1}, A^{t+1}) = \texttt{SearchUpdate}(\tilde{\alpha}^t, \tilde{A}^t, g)$ *// Line search update*
9: **end for**

Algorithm 2 Specialized dataset projection for MMD distances with quadratic programming of a target dataset Y onto the source datasets $X_1, \dots, X_k$ with parameters B, ϵ

1: **for** $i = 1 \dots k$ **do**
2: **for** $j = 1 \dots k$ **do**
3: $K_{ij} = \exp(-\gamma \cdot \texttt{MMD}(X_i, X_j))$ *// Calculate kernel distances of the source datasets*
4: **end for**
5: $k_i = \exp(-\gamma \cdot \texttt{MMD}(X_i, Y))$ *// Calculate kernel distances with the target dataset*
6: **end for**
7: $\alpha = \arg\min_\alpha \frac{1}{2}\alpha^T K \alpha + k^T \alpha$ subject to $\alpha \in [0, B]^k, \left|\sum_i \alpha_i - k\right| \leq \epsilon$

In this work, we adopt the following principle: use all available information to sub-divide datasets before resorting to unsupervised clustering. This provides a diverse array of settings with varying degrees of supervision: one can use the WordNet hierarchy for ImageNet, emoji multilabels for Twitter data, and clustering methods for unsupervised datasets like STL. Crucially, our framework can project *any* auxiliary dataset onto any target dataset, including those with different labels or no labels at all. A detailed discussion of these splitting strategies for dataset projection and the specific splits for each dataset is in Appendix B.

Distance metric. Our framework uses a metric to measure the distance between two datasets. In principle, one could use any distance metric. In this work, we employ the Maximum Mean Discrepancy (MMD) score (Gretton et al., 2012), a statistic that measures the similarity of two distributions. This metric has been successfully used in prior works to learn generative models (Kar et al., 2019; Dziugaite et al., 2015) and detect distribution shift (Rabanser et al., 2019). We provide an overview of the MMD score in Appendix B.

3.1 Solving the dataset projection problem.

Solving equation 3 has several challenges. There is a simplex constraint on the optimization variables α and the objective is stochastic and non-differentiable with respect to the variables α. To tackle this problem, we develop three approaches: (1) an active set solver motivated by simplex methods with theoretical convergence guarantees (Cristofari et al., 2020), (2) a projected gradient descent (PGD) solver based on the widely used proximal method, and (3) a quadratic programming reduction for the specific case of MMD metric. We provide a brief overview highlighting our technical work in developing these approaches.

Active set. Although active set solvers are well-suited for simplex contraints, they were not developed for stochastic problems. Indeed, noisy gradients can cause active set methods to oscillate variables between zero and non-zero. A technical contribution of our work is to stabilize the optimization with damped updates (Line 7 of Algorithm 1). This enables active set methods to converge, summarized in Algorithm 1.

Quadratic program. In the special case of the MMD metric, equation 3 can be reduced to a quadratic program and solved with an off-the-shelf solver. This can be viewed as an extension of Kernel Mean Matching (Gretton et al., 2009). The key difference, however, is to develop a kernel for datasets instead of samples. We summarize the solver in Algorithm 2.

Vision Datasets
CIFAR10 (Krizhevsky, 2009)
STL10 (Adam Coates, 2011)
Oxford-IIIT Pet (Parkhi et al., 2012)
ImageNet (Russakovsky et al., 2015)
3DB (Leclerc et al., 2021)
Language Datasets
SST (Socher et al., 2013)
Yelp (Zhang et al., 2015)
Emoji (Barbieri et al., 2018)
Emotion (Mohammad et al., 2018)
DailyDialog (Chapuis et al., 2020)

Table 1: All datasets used in our dataset projection benchmark.

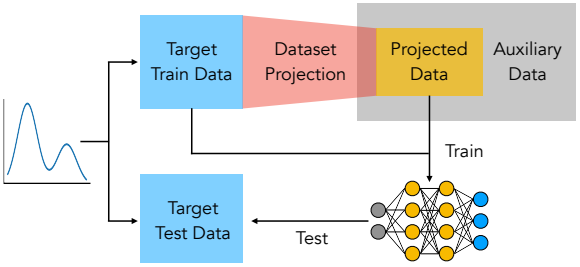


Figure 5: Pipeline for validating alignment and augmenting target data via training. After projecting onto the auxiliary dataset, we can (1) train a model on the projected data to validate target-alignment and (2) augment target training data to improve test performance.

In practice, we find the active set solver performs best in language settings while the quadratic programming solver works best for vision settings. The projected gradient solver is the simplest, but performs the worst and is not recommended for general use. Further details for all of our solvers, including convergence analysis and the quadratic programming reduction, are in Appendix B.

4 Projected datasets in practice

How well does our framework for projecting datasets work empirically? First, we perform an extensive evaluation of our framework, in order to validate that our projected datasets are indeed more target-aligned than the original auxiliary datasets. We then highlight potential use-cases for projected datasets. In particular, we find that projected datasets can improve downstream performance and provide insights on the composition of the target dataset.

Experimental setup. Our benchmark for projecting datasets spans five vision datasets for image classification and five language datasets for sentiment analysis, summarized in Table 1. Each scenario in the benchmark consists of one auxiliary dataset and one target dataset, resulting in a total of 36 scenarios. In all scenarios, we use our framework to project each class from the target dataset onto the auxiliary dataset, using either our active set or PGD solver, and compare to the random baseline of uniformly using all sources (i.e. $\alpha_i = 1/k$ for $i = 1 \dots k$). The complete description of the experimental setup and corresponding datasets is deferred to Appendix C.

4.1 Validating alignment with the target dataset

To solve the dataset projection problem, our framework aims to maximize alignment with the target dataset. But how can we validate whether our solvers actually achieve this goal?

To this end, we leverage a suite of metrics that quantify the alignment of a projected dataset with the target dataset. Note that it may not always be possible to reach perfect alignment with a projected dataset. This can occur when the auxiliary and target datasets are too distinct, or if the sources are too coarse. Nonetheless, we can still project datasets with our framework to find the *most* target-aligned subset. In this section, we validate whether projected datasets are more target-aligned than the original auxiliary dataset.

A natural metric is the MMD score from equation 3 between the target and projected dataset. However, a more holistic approach is to evaluate with additional metrics that are not directly optimized. Thus, we validate our results using distance metrics from (Zeng et al., 2017) for the vision settings and similarity metrics for the natural language settings (Manning et al., 2010). We confirm in Figure 6 that the projected datasets are indeed closer to the target than the original auxiliary dataset. Figure 7 demonstrates this improvement

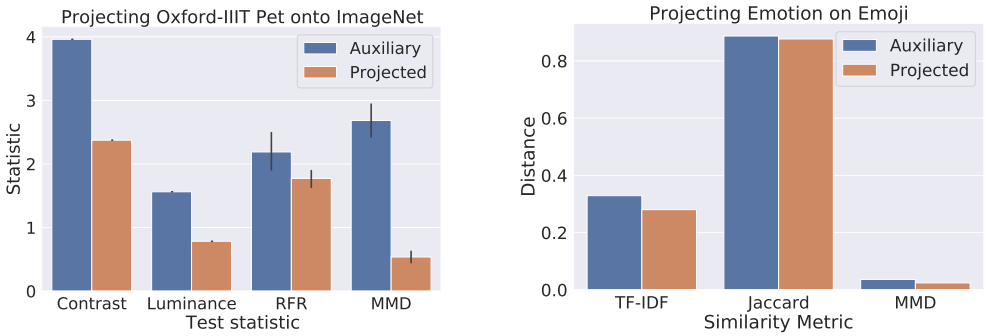


Figure 6: We validate that projected datasets have lower distance to the target dataset, using additional distance metrics. These metrics are (left) 1D-statistics based on contrast, luminance, and random filter response (RFR) for vision datasets and (right) similarity metrics based on TF-IDF and Jaccard coefficients for language datasets.

Figure 7: Visually comparing AS-PD (Middle) with random data from the auxiliary (Left) and target (Right) datasets. While the auxiliary dataset includes wild outdoor cats, the projected dataset contains mainly indoor household cats which aligns with the target dataset.

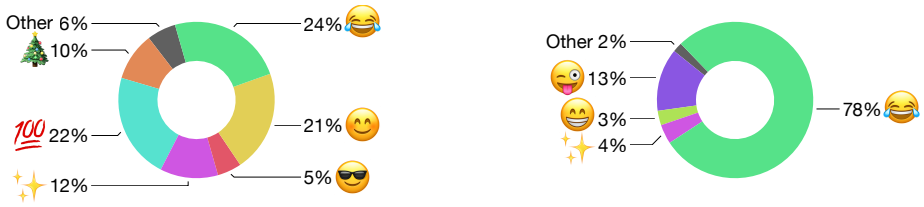
visually—projected cats from ImageNet have backgrounds that are more aligned with the target. This improvement in target approximation holds for nearly all choices of the auxiliary and target datasets, which we evaluate and plot for the rest of our benchmark in Appendix D.

4.2 Analyzing the target dataset

In this section, we highlight the use of our framework as an analysis tool for the target dataset *without needing to train and interpret any downstream models*. Specifically, our projection produces a subset of sources that characterizes the composition of the target dataset, thereby explaining what kind of data exists within the target. For example, Figure 8 plots the projection of the Emotion dataset onto the Emoji subclasses. The resulting proportions informs us about what kinds of Emoji data is closest to each Emotion class. "Optimistic" emotion data is closest to data with Christmas tree and sparkling emojis, while "sad" emotion data is closest to data with crying emojis. Figure 9 shows a similar analysis for projections onto ImageNet cats, revealing that CIFAR10 contains primarily household cats, while STL10 contains cats found in the wild. Additional examples can be found in Appendix E.

4.3 Augmenting the target dataset

A natural use case for auxiliary data is to augment the target dataset to reach higher test accuracy. In this section, we use projected datasets as extra training data (in addition to the target data itself), and evaluate performance on target test data, as illustrated in Figure 5. We compare to the baselines of using no auxiliary data (Target Only) and augmenting uniformly with all auxiliary sources (Target + Random). Tables 2 and 3 highlight a subset of our vision and language results, and the full suite of experiments is deferred to Appendix D.



(a) Optimistic (b) Sad

Figure 8: A visualization of the composition of Emotion classes (Optimistic and Sad) according to their projections onto the Emoji dataset. Additional examples showing the projections of the Angry and Joy classes onto the Emoji dataset are in Appendix E.

(a) CIFAR10 (b) STL10

Figure 9: A visualization of the composition of the CIFAR10 and STL10 cat classes, according to their projections onto ImageNet cats. Additional examples showing the projections of Oxford-IIIT Pet and 3DB cats are in Appendix E.

We find that augmenting the target dataset with projected data often improves performance in both sentiment analysis (Table 2) and image classification (Table 3). For example, augmenting DailyDialog with projected data from Emotion (AS-PD) increases test accuracy by 30% over both baselines. Similarly, augmenting STL10 with projected data from Oxford-IIIT (QP-PD) increases test accuracy by 18% over both baselines. This demonstrates that auxiliary data can more effectively augment existing data if we first apply dataset projection. Similar trends for the full benchmark are shown in Tables 8 and 12 for all 36 settings.

Data augmentation & domain adaptation. Two related areas are data augmentation and domain adaptation. In principle, these areas are not strictly in competition with dataset projection—any auxiliary data added via dataset projection can still be augmented or adapted. In Appendix D, we run experiments that study projected datasets with state of the art approaches in data augmentation and domain adaptation. To summarize, we find that projected datasets provide improvements that are indeed complementary to both of these areas, either outperforming or improving upon these methods. These experiments highlight how the benefits of dataset projection stem from increasing the size of target datasets, while data augmentation and domain adaptation aim to make better use of the target dataset.

5 Related work

Dataset projection is related to various methods including dataset pruning (Angelova et al., 2005), core-set construction (Tsang et al., 2005), and dataset distillation (Wang et al., 2018; Nguyen et al., 2021). These methods attempt to reduce the size of a single dataset while still retaining the most "valuable" training data signal. In contrast, our goal in dataset projection is to match a target dataset to the span of multiple auxiliary sources.

The use of an auxiliary dataset is related to the field of auxiliary learning (Vincent et al., 2008; Zhang et al., 2014; Mordan et al., 2018; Liu et al., 2019), in which additional tasks are used during training to help improve performance on a target task. Crucially, these

Auxiliary	Target	Target Only	Target + Random	Target + PGD-PD	Target + AS-PD	Target + QP-PD
Yelp	SST	74.2 ± 6.0	60.8 ± 8.9	65.6 ± 12.0	$\mathbf{78.6} \pm 2.9$	68.2 ± 4.0
DailyDialog	Emoji	12.4 ± 3.6	14.0 ± 3.3	12.8 ± 2.9	$\mathbf{24.4} \pm 1.0$	8.2 ± 1.3
SST	Emotion	37.6 ± 4.3	39.4 ± 4.1	42.8 ± 7.8	$\mathbf{45.2} \pm 4.7$	31.0 ± 6.2
Emoji	Yelp	28.6 ± 1.9	29.2 ± 3.0	27.0 ± 5.4	$\mathbf{34.4} \pm 5.7$	31.6 ± 4.1
Emotion	DailyDialog	37.4 ± 6.3	37.6 ± 5.0	35.0 ± 8.0	$\mathbf{67.4} \pm 6.0$	56.8 ± 8.5

Table 2: Augmenting a target dataset with auxiliary data for sentiment analysis. In language settings, augmenting with AS-PD typically performs the best.

Auxiliary	Target	Target Only	Target + Random	Target + PGD-PD	Target + AS-PD	Target + QP-PD
ImageNet	CIFAR10	43.6 ± 0.4	54.8 ± 2.1	55.3 ± 1.8	57.0 ± 3.0	$\mathbf{67.9} \pm 0.7$
CIFAR10	Oxford-IIIT	52.1 ± 1.9	54.2 ± 2.5	55.4 ± 2.0	59.5 ± 2.9	$\mathbf{73.4} \pm 0.5$
Oxford-IIIT	STL10	54.2 ± 2.8	54.3 ± 3.1	65.1 ± 1.4	61.6 ± 1.1	$\mathbf{72.5} \pm 0.9$
Oxford-IIIT	3DB	69.8 ± 1.6	73.4 ± 7.2	82.2 ± 3.3	82.3 ± 10.2	$\mathbf{96.3} \pm 1.6$

Table 3: Augmenting a target dataset with auxiliary data for image classification. In vision settings, augmenting with QP-PD typically performs the best.

approaches train on auxiliary tasks alongside the primary task, whereas dataset projection extracts target-aligned subsets of auxiliary data without requiring any auxiliary task.

Works have also explored modifying synthetic data generation processes to more closely resemble a target dataset. Kar et al. (2019) and Devaranjan et al. (2020) used a parameterized graphics engine and probabilistic scene grammars to match a target dataset of scenes. On the other hand, dataset projection is agnostic to the way in which auxiliary data is generated and can be used to select from any type of data source. Our work is also broadly related to the field of domain adaptation (Wang & Deng, 2018; Wilson & Cook, 2020; Farahani et al., 2021), which aims to adapt *models* trained on an auxiliary dataset to improve performance on a target dataset. In contrast, dataset projection is completely model-agnostic and answers the fundamental question of how to align the *data*. Furthermore, domain adaptation methods typically assume a labeled source dataset with a potentially unlabeled target dataset, whereas dataset projection assumes the opposite: a labeled target dataset with a potentially unlabeled source dataset. Using our framework with domain adaptation to align both the model and data jointly is an interesting future direction.

Our method builds on an extensive line of work on minimization over simplex constraints (Bertsekas, 1982; Birgin & Martínez, 2002), as well as distance metrics that measure the similarity of two image distributions (Gretton et al., 2012). Our simplex formulation of aligning multiple dataset sources uses constraints with similarities to the problems posed by Hashimoto (2021), but for a different purpose and objective. While Hashimoto (2021) uses their framework to study theoretical scaling laws for models generalization, we solve the optimization problem to search for target-aligned subsets of auxiliary data.

6 Conclusion

In this work, we pose the problem of dataset projection and develop methods to find the most target-aligned subset of auxiliary data. When using a biased auxiliary dataset, we demonstrate that it can be beneficial to use just a portion of the data rather than all of it. In these situations, dataset projection can select subsets of the data that better approximate the target dataset and can lead to better test performance when augmenting the target dataset. We highlight these trends empirically on a variety of language and vision datasets and further highlight the use of our framework as an analysis tool. Our work takes a step towards understanding when more data is useful, and finding more useful data.

References

Andrew Y. Ng Adam Coates, Honglak Lee. An analysis of single layer networks in unsupervised feature learning. In *AISTATS*, 2011.

A. Angelova, Y. Abu-Mostafam, and P. Perona. Pruning training sets for learning of object categories. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2005.

Francesco Barbieri, Jose Camacho-Collados, Francesco Ronzano, Luis Espinosa-Anke, Miguel Ballesteros, Valerio Basile, Viviana Patti, and Horacio Saggion. Semeval 2018 task 2: Multilingual emoji prediction. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pp. 24–33, 2018.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa-Anke, and Leonardo Neves. TweetEval:Unified Benchmark and Comparative Evaluation for Tweet Classification. In *Proceedings of Findings of EMNLP*, 2020.

Sara Beery, Yang Liu, Dan Morris, Jim Piavis, Ashish Kapoor, Neel Joshi, Markus Meister, and Pietro Perona. Synthetic examples improve generalization for rare classes. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 863–873, 2020.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 610–623, 2021.

Dimitri P Bertsekas. Projected newton methods for optimization problems with simple constraints. *SIAM Journal on control and Optimization*, 20(2):221–246, 1982.

Ernesto G Birgin and José Mario Martínez. Large-scale active-set box-constrained optimization method with spectral projected gradients. *Computational Optimization and Applications*, 23(1):101–125, 2002.

Carmo P Brás, Andreas Fischer, Joaquim J Júdice, Klaus Schönefeld, and Sarah Seifert. A block active set algorithm with spectral choice line search for the symmetric eigenvalue complementarity problem. *Applied Mathematics and Computation*, 294:36–48, 2017.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

Mathilde Caron, Piotr Bojanowski, Julien Mairal, and Armand Joulin. Unsupervised pre-training of image features on non-curated data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2959–2968, 2019.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020.

Emile Chapuis, Pierre Colombo, Matteo Manica, Matthieu Labeau, and Chloé Clavel. Hierarchical pre-training for sequence labelling in spoken dialog. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 2636–2648, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.239. URL `https://www.aclweb.org/anthology/2020.findings-emnlp.239`.

Andrea Cristofari, Marianna De Santis, Stefano Lucidi, and Francesco Rinaldi. A two-stage active-set algorithm for bound-constrained optimization. *Journal of Optimization Theory and Applications*, 172(2):369–401, 2017.

Andrea Cristofari, Marianna De Santis, Stefano Lucidi, and Francesco Rinaldi. An active-set algorithmic framework for non-convex optimization problems over the simplex. *Computational Optimization and Applications*, 77:57–89, 2020.

Jeevan Devaranjan, Amlan Kar, and Sanja Fidler. Meta-sim2: Learning to generate synthetic datasets. In *ECCV*, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

Gintare Karolina Dziugaite, Daniel M. Roy, and Zoubin Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. In *Uncertainty in Artificial Intelligence (UAI)*, 2015.

Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (python library), 2019a. URL `https://github.com/MadryLab/robustness`.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Brandon Tran, and Aleksander Madry. Adversarial robustness as a prior for learned representations. *arXiv preprint arXiv:1906.00945*, 2019b.

Francisco Facchinei, Andreas Fischer, and Christian Kanzow. On the accurate identification of active constraints. *SIAM Journal on Optimization*, 9(1):14–32, 1998.

Abolfazl Farahani, Sahar Voghoei, Khaled Rasheed, and Hamid R Arabnia. A brief review of domain adaptation. *Advances in Data Science and Information Engineering*, pp. 877–894, 2021.

Arthur Gretton, Alex Smola, Jiayuan Huang, Marcel Schmittfull, Karsten Borgwardt, and Bernhard Schölkopf. Covariate shift by kernel mean matching. *Dataset shift in machine learning*, 3(4):5, 2009.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research (JMLR)*, 13(25): 723–773, 2012. URL `http://jmlr.org/papers/v13/gretton12a.html`.

William W Hager and Hongchao Zhang. A new active set algorithm for box constrained optimization. *SIAM Journal on Optimization*, 17(2):526–557, 2006.

Tatsunori Hashimoto. Model performance scaling with multiple data sources. In *International Conference on Machine Learning (ICML)*, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. In *Arxiv preprint arXiv:2001.08361*, 2020.

Amlan Kar, Aayush Prakash, Ming-Yu Liu, Eric Cameracci, Justin Yuan, Matt Rusiniak, David Acuna, Antonio Torralba, and Sanja Fidler. Meta-sim: Learning to generate synthetic datasets. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Guillaume Leclerc, Hadi Salman, Andrew Ilyas, Sai Vemprala, Logan Engstrom, Vibhav Vineet, Kai Xiao, Pengchuan Zhang, Shibani Santurkar, Greg Yang, Ashish Kapoor, and Aleksander Madry. 3db: A framework for debugging computer vision models. In *Arxiv preprint arXiv:2106.03805*, 2021.

Shikun Liu, Andrew J Davison, and Edward Johns. Self-supervised generalisation with meta auxiliary learning. *arXiv preprint arXiv:1901.08933*, 2019.

Sharon L Lohr and Trivellore E Raghunathan. Combining survey data with other data sources. *Statistical Science*, 32(2):293–312, 2017.

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *European Conference on Computer Vision (ECCV)*, 2018.

Christopher Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to information retrieval. *Natural Language Engineering*, 16(1):100–103, 2010.

Gustav Mårtensson, Daniel Ferreira, Tobias Granberg, Lena Cavallin, Ketil Oppedal, Alessandro Padovani, Irena Rektorova, Laura Bonanni, Matteo Pardini, Milica G Kramberger, et al. The reliability of a deep learning model in clinical out-of-distribution mri data: a multicohort study. *Medical Image Analysis*, 66:101714, 2020.

Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81):1–32, 2016.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. Semeval-2018 task 1: Affect in tweets. In *Proceedings of the 12th international workshop on semantic evaluation*, pp. 1–17, 2018.

Taylor Mordan, Nicolas Thome, Gilles Henaff, and Matthieu Cord. Revisiting multi-task learning with rock: a deep residual auxiliary block for visual detection. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Samuel G Müller and Frank Hutter. Trivialaugment: Tuning-free yet state-of-the-art data augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 774–782, 2021.

Timothy Nguyen, Roman Novak, Lechao Xiao, and Jaehoon Lee. Dataset distillation with infinitely wide convolutional networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1406–1415, 2019.

Zhaofan Qiu, Ting Yao, Chong-Wah Ngo, Xiao-Ping Zhang, Dong Wu, and Tao Mei. Boosting video representation learning with multi-faceted integration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14030–14039, 2021.

Stephan Rabanser, Stephan Günnemann, and Zachary C. Lipton. Failing loudly: An empirical study of methods for detecting dataset shift. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *International Conference on Knowledge Discovery and Data Mining*, 2016.

Jonathan S. Rosenfeld, Amir Rosenfeld, Yonatan Belinkov, and Nir Shavit. A constructive prediction of the generalization error across scales. In *International Conference on Learning Representations (ICLR)*, 2020.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. In *International Journal of Computer Vision (IJCV)*, 2015.

Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. *arXiv preprint arXiv:2006.16971*, 2020.

Kendrick Shen, Robbie M Jones, Ananya Kumar, Sang Michael Xie, Jeff Z HaoChen, Tengyu Ma, and Percy Liang. Connect, not collapse: Explaining contrastive learning for unsupervised domain adaptation. In *International Conference on Machine Learning*, pp. 19847–19878. PMLR, 2022.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/D13-1170`.

Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *International Conference on Computer Vision (ICCV)*, 2017.

Jörg Tiedemann and Santhosh Thottingal. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal, 2020.

Ivor W. Tsang, James T. Kwok, and Pak-Ming Cheung. Core vector machines: Fast svm training on very large data sets. *Journal of Machine Learning Research (JMLR)*, 2005.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *International Conference on Machine Learning (ICML)*, 2008.

Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018.

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

Garrett Wilson and Diane J Cook. A survey of unsupervised deep domain adaptation. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(5):1–46, 2020.

Kai Xiao, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. Noise or signal: The role of image backgrounds in object recognition. In *International Conference on Learning Representations (ICLR)*, 2021.

John R Zech, Marcus A Badgeley, Manway Liu, Anthony B Costa, Joseph J Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study. *PLoS medicine*, 15(11): e1002683, 2018.

Yu Zeng, Huchuan Lu, and Ali Borji. Statistics of deep generated images. *arXiv preprint arXiv:1708.02688*, 2017.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Facial landmark detection by deep multi-task learning. In *European Conference on Computer Vision (ECCV)*, 2014.

A Examples from Section 2 on the limits of auxiliary data

In this section, we provide all the specifics for the three experiments discussed in Section 2. These experiments demonstrate how indiscriminate usage of *all* auxiliary data can actually harm performance.

A.1 Training on ImageNet data for CIFAR10

In the first example (Figure 1), we showed that training on more ImageNet data does not always improve CIFAR10 performance. Indeed, although adding a small amount of ImageNet does boost CIFAR10 performance, adding too much ultimately decreases accuracy. The blue line shows the performance of a CIFAR10 classifier when we add auxiliary ImageNet data to one thousand CIFAR10 training examples. The orange line shows the performance of a CIFAR10 classifier when we only train on the auxiliary ImageNet data. In both cases, we find that optimal CIFAR10 performance is reached after adding approximately ten thousand ImageNet datapoints. After this point, adding more ImageNet data degrades the classifier's accuracy.

Our experimental setup for this setting follows our main experimental setup but with one main difference: to examine the effect of auxiliary data, we vary the size of the auxiliary dataset in the amounts of 2^k for $k = 4 \dots 16$. Otherwise, the remaining specifics (how we select auxiliary data from ImageNet and training parameters) match our benchmark, and are described in Appendix C.

A.2 Gaussian example

In this section, we provide the specifics of the Gaussian example from Section 2, where we showed how restricting data from auxiliary Gaussians can improve the resulting classifier.

Data generation. The target data is generated from 2-dimensional Gaussians with a class-conditional distribution of

$$p(x|y) \sim \mathcal{N}(y \cdot \mu, I) \tag{4}$$

for $\mu = (2, 0)$. The auxiliary data is also generated from 2-dimensional Gaussians with the same mean but different covariance. Specifically, the auxiliary data has a conditional distribution of

$$p_{aux}(x|y) \sim \mathcal{N}(y \cdot \mu, \Sigma) \tag{5}$$

for $\Sigma = R\,\mathrm{diag}([s, 1/s])R^T$ where $s = 4$ and R is the standard rotation matrix

$$R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{6}$$

for $\theta = \pi/8$. In other words, p_{aux} is equivalent to a rotated and scaled version of p with the same mean. This rotation and scaling amounts to the bias introduced from the auxiliary dataset.

We generate $n = 12$ datapoints $\mathcal{D}_{target} = \{x_i, y_i\}$ from the target distribution split evenly between the two classes, and generate $m = 2 \dots 200$ datapoints $\mathcal{D}_{auxiliary} = \{x'_i, y'_i\}_{i=1 \dots m}$ from the auxiliary distribution, also split between the two classes.

Clipping auxiliary data. To clip the auxiliary data and get a target-aligned subset, we simply throw away all datapoints that lie outside a high confidence region of the Gaussians estimated from the target dataset. Specifically, use the following steps:

1. We estimate the class conditional mean and covariance matrix of the target dataset. Specifically, we calculate the sample mean μ_y and covariance Σ_y of each class in the target data, $\{x_i : y_i = y\} \subseteq \mathcal{D}_{target}$. This gives us an estimated distribution $\hat{p}(x|y) = \mathcal{N}(\mu_y, \Sigma_y)$ for the target distribution.

Figure 10: The full 15k-image auxiliary dataset (Left) contains a background bias where dogs appear more frequently outdoors and cats appear more frequently indoors. We only show dog images here to highlight the background bias. The unbiased 3k-image subset (Middle) does not have any background bias. Training on the unbiased subset results in better test accuracy on the unbiased test set (Right), showing that more data is not always more helpful.

2. We calculate the Mahalanobis distance of each auxiliary datapoint to the estimated Gaussian of its class $\hat{p}(x|y)$, i.e.

$$MD(x, y) = \sqrt{(x - \mu_y)^T \Sigma_y^{-1} (x - \mu_y)} \tag{7}$$

3. We then discard all auxiliary points whose Mahalanobis distances lies outside of a certain threshold. Specifically, the resulting subset is the following:

$$\mathcal{D}_{restricted} = \{(x', y') : MD(x', y') \leq r \text{ for } (x', y') \in \mathcal{D}_{auxiliary}\} \tag{8}$$

for $r = 3$.

Fitting the linear classifier. In the experiment shown in Figure 2, we measure how adding auxiliary data affects the performance of a linear classifier with a small amount of target data. Similar to the previous experiment on augmenting CIFAR10 data with ImageNet data, we find that adding too much auxiliary data hurts accuracy in this Gaussian example.

Specifically, we add data from the auxiliary dataset to the training data in various amounts from $2 \ldots 100$, and fit a linear classifier. We use the default `LogisticRegression` function from Scikit-learn. Test error is measured over an independent, random sample of 1000 additional samples from the original training distribution p. Error bars are averaged over five random seeds. We also compare to the analytically optimal classifier:

$$h_{opt}(x_1, x_2) = \begin{cases} 1, & \text{if } x_1 \geq 0 \\ -1, & \text{otherwise} \end{cases}$$

A.3 Training on biased 3DB data

Inspired by the common story of machine learning models picking up on biases between animals and their backgrounds (Ribeiro et al., 2016), we construct a binary classification task with explicitly planted biases. In contrast to the Gaussian example, we generate a more realistic setting of cats and dogs using the Blender renderer and the 3DB framework (Leclerc et al., 2021). In contrast to the CIFAR10 and ImageNet example, where the exact ImageNet patterns that hurt CIFAR10 generalization are unknown, we can instead construct a setting with an explicit bias. Since this is a controlled experiment, it enables us to do the following:

1. Validate in isolation that biased auxiliary data can hurt performance in situations more complex than the Gaussian example.
2. Demonstrate how an "ideal" target-aligned subset can remove this bias and improve performance over using all of the original auxiliary data

Data generation. We begin by constructing an auxiliary dataset with a background bias. Specifically, we generate 50 images for each (3D-model, background) pairing. We use 15 cat

3D-models and 15 dog 3D-models for the training data. We choose 1 indoor background and 1 outdoor background to be shared by both classes. Then, we choose 8 new indoor backgrounds to appear with cats, and 8 new outdoor backgrounds to appear with dogs. We hold out 1 indoor background, 1 outdoor background, and three 3D-models per class for the test set.

Our resulting training set has a strong background bias; 90% of cats are indoors, and 90% of dogs are outdoors. Meanwhile, the test set consists of two entirely new backgrounds and is unbiased: half of the images use an outdoor background and the other half use an indoor background irrespective of the class. We show examples of generated train and test dog images on the left and right panels of Figure 10 respectively.

The ideal target-aligned subset. The ideal subset of auxiliary data is one that does not introduce a background bias. Specifically, we can accomplish this goal by choosing images that use the shared indoor and outdoor backgrounds as the "ideal" subset. Then, cats and dogs are equally likely to appear on either indoor or outdoor backgrounds. This ensures that the backgrounds do not confer any useful correlations for predicting the class, removing the background bias. Examples of the ideal dog subset are shown shown in the middle of Figure 10.

Fitting a classifier. We train ResNet-18 models on both the full auxiliary dataset and the target-aligned subset and evaluate their test performance on the unbiased test set. Most training details are the same as the training details for the vision experiments in the rest of the paper, which are described in Appendix C.3. The main differences are the following.

- We train for 120 epochs for the full 15k-image auxiliary dataset, and we train for 600 epochs for the unbiased 3k-image subset (so that the total number of SGD iterations across both settings is the same).
- We use a learning rate of 0.1, and we only do a learning rate drop one time, halfway through training.

Indiscriminately training on the full, biased dataset results in a test accuracy of $90.4\% \pm 2.1$. On the other hand, training on the smaller, target-aligned subset improves the accuracy by over 7% to $97.6\% \pm 0.6$, respectively. This suggests that when a bias exists in the auxiliary data, it can be harmful to use the full dataset as opposed to just a target-aligned subset.

B How to project datasets

In this section, we describe in detail the framework that we developed to solve the dataset projection problem. Specifically:

1. We provide a brief background on the active set optimization method that we build upon (Appendix B.1).
2. We describe the exact algorithms of our active set (Appendix B.2) and PGD (Appendix B.3) solvers.
3. We provide a more detailed description of how we obtain source distributions (Appendix B.6).
4. We outline specifically how we compute the distance metric between datasets (Appendix B.7).

B.1 Overview of active set framework

We build upon an extensive line of work on active set optimization frameworks for minimization over the simplex (Bertsekas, 1982; Birgin & Martínez, 2002; Brás et al., 2017; Cristofari et al., 2017; Facchinei et al., 1998; Hager & Zhang, 2006). This family of algorithms can

Algorithm 3 The damped active set update `DampedUpdate`(α, A, g) for simplex optimization, which applies a momentum update to the active set A and performs a coordinate step to the iterate α.

// Select coordinate to update and calculate hard active set for current iterate
$j = \arg\min_i \{g_i\}$
$\tilde{A} = \{i : \alpha_i \leq \epsilon g^T (e_i - \alpha)\}$

// Apply damped update to the soft active set estimate to stabilize the current active set
$A = \beta A$
for $i \in \tilde{A}$ **do**
  $A = A + (1 - \beta)$
end for
$\tilde{A} = \{i : A_i > 0.5\}$
$N = \{i :_i \leq 0.5, i \neq j\}$

// Apply coordinate update
Set $\tilde{\alpha}_{\tilde{A}} = 0,\ \tilde{\alpha}_N = \alpha_N,\ \tilde{\alpha}_j = \alpha_j + \sum_{h \in \tilde{A}} \alpha_h$
return $(\tilde{\alpha}, A)$

solve problems of the following form:

$$\min_{\alpha} f(\alpha) \ \text{ subject to } \ \sum_{i=1}^{k} \alpha_i = 1, \ \ \alpha_i \geq 0 \tag{9}$$

where f is the objective function to be minimized over the simplex. Existing algorithms have favorable properties, such as global convergence to stationary points in non-convex settings (Cristofari et al., 2020).

Our algorithm is an adaptation of the framework from Cristofari et al. (2020) to the stochastic and non-differentiable setting, in order to solve the dataset projection problem from Equation equation 3.

As a brief overview, the simplex algorithm from Cristofari et al. (2020) consists of two main steps. First, the method calculates an estimate of the active set, and performs a single coordinate update based on this update. Second, the method then calculates a search direction, and performs a line search in this direction. With these updates, Cristofari et al. (2020) show that this method has linear convergence to a stationary point for non-convex problems with simplex constraints.

However, the method as originally proposed was not intended for stochastic and non-differentiable optimization. Indeed, if we attempt to directly apply this algorithm to solve the dataset projection from Equation equation 3, we run into two main problems. First, the estimate of the active set can vary due to the stochasticity in the objective when sampling a dataset. This causes coordinates to rapidly fluctuate between being in and out of the active set between iterations, which prevents the method from converging. Second, the method requires a gradient calculation for both the active set estimate and the search direction. However, the objective from Equation equation 3 is non-differentiable with respect to the simplex variables.

B.2 Extending the active set framework from Cristofari et al. (2020)

In order to address these two problems and solve the dataset projection problem, we modify the active set simplex algorithm in two ways.

Damped updates for estimating the active set. In order to stabilize the algorithm, we need to stop the active set estimates from fluctuating too much. To do this, we instead introduce a soft active set estimate which varies between $[0, 1]$ for each coordinate. Then, to determine if a coordinate is in the active set or not, we simply threshold the soft estimate at

Algorithm 4 The search direction update $\texttt{SearchUpdate}(\alpha, A, g)$, which uses a stale gradient and line search to update the iterate α with line search parameters (γ, δ). Here, $\mathcal{P}_\Delta$ is the projection operator onto the simplex.

// Compute a search direction that obeys the active set estimate
$d = \mathcal{P}_\Delta(\alpha - sg) - \alpha$
for $i : A_i > 0.5$ **do**
  $d_i = 0$
end for

// Armijo line search
if $g^T d < 0$ **then**
  $\lambda = 1$
  while $f(\alpha + \lambda d) > f(\alpha) + \gamma\lambda g^T d$ **do**
    $\lambda = \delta\lambda$
  end while
else
  $\lambda = 0$
end if
return $\alpha + \lambda d$ *// Second update*

Algorithm 5 Projected gradient descent (PGD) solver for projecting datasets with step size γ

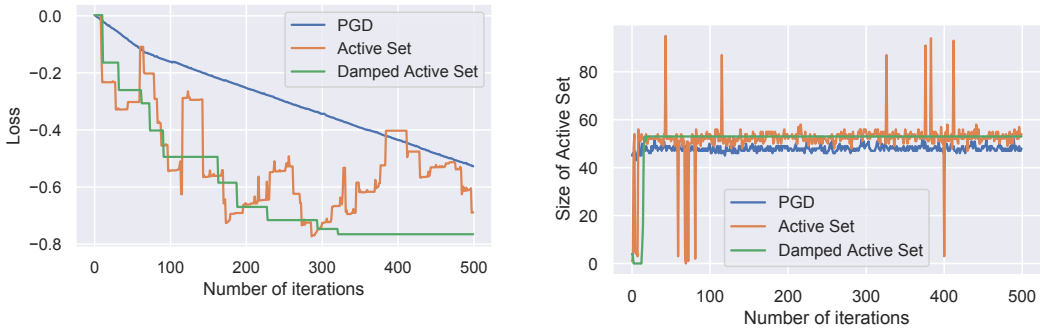
$\alpha_i^0 = 1/k$ for $i = 1 \dots k$ *// Initialize feasible point*
for $t = 0, 1, \dots$ **do**
  $g = \nabla f(\alpha^t)$ *// Estimate numerical gradient of the dataset projection objective*
  $\alpha^t = \texttt{Proj}_\Delta(\alpha^t + \gamma \cdot g)$ *// Gradient update with projection onto simplex*
end for

0.5: coordinates with a soft estimate greater than 0.5 are in the active set, and coordinates below are not. Finally, each iteration we update the soft estimate with a momentum-style update to damp the variability at each iteration. This stabilizes the soft estimate, and creates a more consistent active set across iterations that enables the method to converge. The specific steps for calculating and updating the soft active set are in lines 5-11 of Algorithm 3. The rest of the active set update remains the same as in Cristofari et al. (2020).

Numerical estimation for gradients. The active set algorithm has two updates that require gradient directions. However, the dataset projection problem is non-differentiable due to the sampling procedure. Instead, we use numerical estimation to calculate the gradient. Specifically, we use the central difference formula for estimating the gradient. Furthermore, in the second step of the framework, we use a stale gradient from the previous update to keep computational overhead low. This contrasts with the original framework, which uses a fresh gradient for the iterate after the initial active set update step. Otherwise, the remainder of the second update is the same as in Cristofari et al. (2020), and is shown in Algorithm 4.

B.3 Projected gradient descent

As an another approach, we can apply projected gradient descent (PGD) to solve Equation 3. Similarly to the active set approach, we can use numerical gradient estimates to directly optimize α, and project α back to the simplex after every step. Although PGD does not have the global convergence guarantee that the active set method does, PGD is an simple technique for solving constrained optimization problems in deep learning settings. The PGD solver is shown in Algorithm 5.



(a) Loss convergence of each solver over iterations.

(b) Size of the active set, or number of zeroed out variables, of each solver over iterations.

Figure 11: Convergence of the loss objective and number of zero entries for each solver (PGD, Active Set, and Active Set with damped updates) over iterations. The vanilla active set method struggles with the stochasticity, and has large jumps in loss and size of the active set. The PGD solver makes consistent but slow progress, and oscillates around the optimal active set. Our solver using damped updates can leverage the superior convergence of active set methods but stabilizes the updates, and is able to find the optimal active set within 16 iterations.

B.4 Convergence analysis

In this section we demonstrate how active set methods struggle to converge with stochasticity, and how our damped update addresses this problem. To isolate the effect, we construct a minimal problem of minimizing $f(x) = w^T x + \|x\|_2^2$ subject to $\|x\|_1 \leq 1$ for a randomly generated w. This is a non-stochastic convex problem with a unique solution, and indeed both active set and projected gradient descent methods can both solve this problem.

The issue arises when there is noise in the gradients. We can simulate this by adding standard Gaussian noise to 10% of the coordinates in the gradient at each step. If we attempt to apply the standard active set method with noisy gradients, we see that the loss and active set becomes highly unstable in Figure 11. We also see that the PGD solver converges stably, but takes much longer to do so and oscillates around the optimal active set. In contrast, our damped version of the active set solver allows us to utilize the faster convergence of the active set method without the oscillating convergence. Indeed, in this setting our method finds the optimal active set within 16 iterations, which neither vanilla active set nor PGD is able to do in 500 iterations.

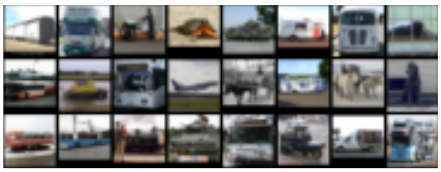
B.5 Quadratic Programming reduction

In the special case of MMD metric, equation 3 can be reduced to a quadratic program. Specifically, if we assume the distance to be the squared MMD distance, we can rewrite equation 3 as

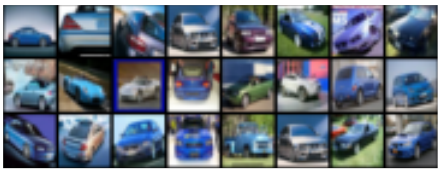
$$\begin{aligned} &\min_{\alpha} \mathbb{E}_{Y\sim\hat{q},Y'\sim\hat{q}}\left[k(Y,Y')\right] - 2\mathbb{E}_{X\sim p,y\sim\hat{q}}\left[k(X,Y)\right] \\ &\text{subject to } \hat{q} = \sum_{i=1}^{k} \alpha_i q_i, \quad \sum_{i=1}^{k} \alpha_i = 1, \quad \alpha_i \geq 0 \end{aligned} \tag{10}$$

Moving one of the constraints into the objective, we get

$$\begin{aligned} &\min_{\alpha} \sum_i \sum_j \alpha_i \alpha_j \mathbb{E}_{Y\sim q_i,Y'\sim q_j}\left[k(Y,Y')\right] - 2\sum_i \alpha_i \mathbb{E}_{X\sim p,y\sim q_i}\left[k(X,Y)\right] \\ &\text{subject to } \sum_{i=1}^{k} \alpha_i = 1, \quad \alpha_i \geq 0 \end{aligned} \tag{11}$$



(a) STL10 cluster.



(b) CIFAR10 "blue" cluster of the car class.

Figure 12: Examples of clusters created when using a robust representation and unsupervised learning to separate auxiliary datasets into source distributions. (a) A cluster from the STL10 dataset, which has no label information and (b) a cluster from the CIFAR10 car class, which contains mostly blue images.

Rewriting this in matrix form and relaxing the simplex constraint to be within a tolerance , we have

$$\begin{aligned} &\min_{\alpha} \alpha^T K \alpha - 2\alpha^T k \\ &\text{subject to} \quad \left| \sum_i \alpha_i - k \right| \leq \epsilon, \;\; B \geq \alpha_i \geq 0 \end{aligned} \tag{12}$$

where $K_{ij} = \mathbb{E}_{Y \sim q_i, Y' \sim q_j}[k(Y, Y')]$ and $k_i = \mathbb{E}_{X \sim p, y \sim q_i}[k(X, Y)]$ for some kernel k. This formulation carries similarities to the Moment Matching Algorithm from Gretton et al. (2009), with one major difference: the MMD metric uses a kernel k that is defined over *datasets*, whereas the original Moment Matching Algorithm uses a kernel defined over individual *data points*.

Thus, to create the kernel matrix for dataset projection, we need to define a kernel over datasets. To this end, we can recursively use the MMD metric again, but this time as a kernel for an MMD over datasets. Specifically, we define the following dataset kernel:

$$k(X, Y) = \exp(-\gamma \cdot \text{MMD}(X, Y)^2) \tag{13}$$

where γ is a hyperparameter and $\text{MMD}(X, Y)$ calculates the typical MMD distance between two datasets (X, Y).

B.6 Determining source distributions

To apply dataset projection to real-world datasets, we need to divide an auxiliary distribution into multiple source distributions to search over. We do so primarily with two main strategies: either by using existing labels when available, or generating source distributions with unsupervised clustering methods.

Using label information. In some cases, datasets may already have existing class or attribute labels. For example, suppose that the target dataset is CIFAR10 and the auxiliary dataset is ImageNet. When projecting the CIFAR10 *dog* class onto ImageNet, each ImageNet class that is a dog breed can be considered a separate source distribution. Note that the labels for the auxiliary dataset do not need to strictly line up with labels in the target dataset. For example, the Emoji dataset has labels corresponding to emoticons such as a Christmas tree, a camera, and the sun. These labels may not have an obvious correspondence to the labels of other datasets such as the five star ratings from the Yelp dataset. Our framework can simply use the auxiliary labels as a way to divide the data into source distributions, and then project each target class onto the auxiliary data to re-assign labels for the target task.

Unsupervised data. Sometimes no label information is available. For example, suppose we want to project CIFAR10 classes onto the unsupervised STL10 dataset. The STL10 dataset is completely unlabeled and can contain images of objects that are completely irrelevant to the base CIFAR10 class. In such cases, we can use unsupervised clustering

techniques to find source distributions. For our vision settings, we use k-means clustering based on robust representations because it has been shown to lead to good visual alignment within each cluster (Engstrom et al., 2019b). We show several examples from an STL10 cluster in Figure 12.

Combined label and clustering. The previous two approaches for dividing an auxiliary distribution into source distributions can be combined to obtain even finer-grained source distributions. This can enable source distributions to capture a narrower population, which can then be used to find more accurate projections of the target dataset. For example, when using CIFAR10 as an auxiliary dataset, we can not only break the dataset into subsets corresponding to its classes, but also use clustering techniques to break each class subset into finer-grained categories. We show an example of this in Figure 12, which shows a blue-car cluster within the CIFAR10 car class.

B.7 Distance metric for dataset projection

In this section, we expand on the specific distance metric used in the objective of the dataset projection problem. This objective is the fundamental metric that guides the solvers towards "target-aligned" subsets.

We use what is known as the unbiased estimate for the Maximum Mean Discrepency (Gretton et al., 2012), as defined in Equation 14:

$$\mathrm{MMD}(x, y) = \left[\frac{1}{m^2} \sum_{i,j=1}^{m} k(x_i, x_j) + \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} k(x_i, y_j) + \frac{1}{n^2} \sum_{i,j=1}^{n} k(y_i, y_j) \right]^{\frac{1}{2}} \tag{14}$$

where $k(x, y) = \exp(-\alpha \cdot \frac{1}{2}\|x - y\|_2^2)$ is a kernel with hyperparameter α.

This score was originally proposed as a way to distinguish whether two distributions were different or indistinguishable via a two-sample hypothesis test. In our setting, we consider the datasets as sampled from the source and target distributions, and use the MMD score as the metric for distance without the hypothesis testing component.

However, these MMD scores run into numerical issues when the feature dimension of the dataset is extremely large. These issues make it not possible for the MMD score to distinguish between, for example, two different image distributions. However, Rabanser et al. (2019) found that calculating the MMD score in the encoded representation space of a neural network was significantly more effective than the original feature space, and in fact was the most powerful way to perform the traditional two-sample hypothesis test for detecting distribution shift.

Thus, in this work we use the MMD score in the feature space of a neural network, as suggested by Rabanser et al. (2019). For the vision setting we use a randomly initialized encoder network, which found to be as effective as pretrained variants (Rabanser et al., 2019). For the language setting we use a pretrained BERT model from HuggingFace (Devlin et al., 2018). To select the hyperparameter α, we measured the MMD statistic over a range of possible α and selected one that resulted in a statistically significant hypothesis test in distinguishing the original auxiliary data from the target data. For the vision settings, this turned out to be $\alpha = 500$, and for the language settings this was $\alpha = 0.01$.

C Experimental details

In this section, we provide a complete description of the datasets used and the experimental setup for projecting datasets and training models.

C.1 Datasets

In this section, we describe each of the datasets used in the experiments of Section 4. A table summary of how we assigned the auxiliary dataset, the target dataset, and the held-out test set of the target dataset is shown in Table 4.

Our computer vision experiments use four standard classification datasets — ImageNet (Russakovsky et al., 2015), Oxford-IIIT Pet (Parkhi et al., 2012), CIFAR10 Krizhevsky (2009), and STL10 (Adam Coates, 2011) — and one synthetic dataset generated using 3DB (Leclerc et al., 2021). Our natural language experiments use five sentiment analysis datasets — Stanford Sentiment Treebank (Socher et al., 2013), Emotion Recognition (Mohammad et al., 2018; Barbieri et al., 2020), Emoji Prediction (Barbieri et al., 2018; 2020), Yelp Reviews (Zhang et al., 2015), and DailyDialog Act Corpus (Chapuis et al., 2020).

	Target	Test	Auxiliary	Source Distribution
ImageNet	-	-	Train Set	Class labels
CIFAR10	Train Set (1000)	Test Set (1000)	Train Set	Combined
Oxford-IIIT	All but test set	Random 250 per class	All but test set	Class labels
STL10	Train Set (500)	Test Set (800)	Unlabeled data	Clustering
3DB	Validation Set (1000)	Test Set (1000)	Train Set	Attribute labels
SST	Validation set	Test set	Train Set	Sentiment scores
Emoji	Validation set	Test set	Train Set	Emoji labels
Emotion	Validation set	Test set	Train Set	Emotion labels
Yelp	Train set (2nd half)	Train set (1st half)	Test set	Review scores
DailyDialog	Validation set (100)	Test set	Train Set	Emotion labels

Table 4: We summarize the portions of each dataset used when the given dataset is chosen as the target, test, or auxiliary dataset. For example, when CIFAR10 is the target distribution, we project 1000 images from each class of CIFAR10's training set onto the auxiliary dataset. When testing a model (e.g., one trained on the projected dataset) on CIFAR10, we test on the held-out test set of CIFAR10. When CIFAR10 is used as an auxiliary dataset, we use both the CIFAR10 class labels *and* unsupervised clustering to get source distributions.

Personally identifiable information or offensive content. The datasets we use are all open source and widely-used in the community. Nonetheless, there is a non-zero chance that the data contains personal information or offensive content. For example, unsupervised datasets such as the images in STL10 may contain such images since the dataset has, by definition, not been supervised. The sentiment analysis datasets may contain negative sentences that are possibly offensive towards people, such as racist messages posted on Twitter or ad-hominen attacks in negative Yelp reviews. To our knowledge, we are not aware of any such data within these datasets, and have not explicitly encountered them in our research.

C.1.1 Image classification

We standardize all of the image classification tasks into a single unified setting. Specifically, all images are resized to have the same resolution (32×32), and for each scenario, we use classes that are shared between datasets. For example, when projecting Oxford-IIIT onto CIFAR10, we choose the auxiliary dataset to be the the CIFAR10 classes *cat* and *dog*, and project the corresponding classes in Oxford-IIIT onto the cluster-based sources of the cat and dog classes.

ImageNet. ImageNet is the largest vision dataset we consider, and it is substantially larger (both in size and number of classes) than all the other vision datasets. Hence, we use ImageNet primarily as an auxiliary dataset. Specifically, we use the training set of ImageNet.

To get source distributions for a particular class, we find that class in the WordNet hierarchy and use every descendant ImageNet class as a source for the class. For example, for the "dog" class, we find the dog node in the WordNet hierarchy and use all the various breeds of dogs under this node as individual source distributions for the dog class.

CIFAR10. The target and test datasets for CIFAR10 are random subsets of the training and test data respectively. Specifically, we randomly subsample 1000 examples from each class to match the ImageNet dataset.

When CIFAR10 is used as an auxiliary dataset, we split the training data into sources with a combination of class labels and unsupervised clustering. Specifically, we first encode each example into a robust feature representation. We do this with an adversarially robust ImageNet classifier (Engstrom et al., 2019a), as this is known to be more aligned with human visual features (Engstrom et al., 2019b). We used the open-source pre-trained ℓ_2 robust model for $\epsilon = 3$. For each class, we then cluster the training data into 16 clusters using the robust representations and the `MiniBatchKMeans` function from Scikit-learn. These 16 clusters form the source distributions for the corresponding CIFAR10 class.

Oxford-IIIT Pet. The Oxford-IIIT Pet dataset is originally not split into a training set and test set. Thus, we randomly select 250 cats and 250 dogs for use as a test set, and use the remaining data as the train set.

When Oxford-IIIT Pet is used as auxiliary data, we use the entire Oxford-IIIT Pet dataset, other than the 250 cats and 250 dogs that were set aside as the test set. Specifically, we use the fine-grained labels of dog breeds and cat breeds to divide the dataset into source distributions. In other words, the dog breeds form the source distributions for the dog class, and the cat breeds form the source distributions for the cat class. In total, there are 12 cat breeds and 23 dog breeds, each with (on average) 200 images per cluster.

STL10. The target and test datasets are random subsets of the training and test data, with 500 and 800 examples each, respectively.

We use the unlabeled data in STL10 as the auxiliary data. Similar to CIFAR10, since there are no sources, we use unsupervised clustering to generate the sources. However, in this case since there are no class labels, we rely purely on unsupervised clustering. Specifically, we use the same methodology as for CIFAR10, but instead create 160 clusters total that are not necessarily separated by class (instead of 16 clusters for each of 10 classes).

3DB. 3DB is a synthetic rendering platform that allows the user to generate synthetic image data by specifying a 3D-model, an HDRI background, and a variety of other parameters such as the camera location, camera and object orientation, and the scene brightness (Leclerc et al., 2021). In our case, we generate a "CIFAR10-like" synthetic dataset with this renderer. Specifically, we use this framework with free, Blender-compatible 3D-models that we find online from SKETCHFAB.COM, as well as free HDRI backgrounds from POLYHAVEN.COM.

We first describe how we make the training set for 3DB. For each class that we want to generate (e.g., *airplane* or *dog*), we find 15 high-quality 3D-models and 25 HDRI backgrounds. We also choose 4 distinct camera settings in terms of height and zoom — in particular, we allow the camera to have a zoom factor of either 1.5 or 3.0, and we choose the camera height to be 0 or 1[1]. This gives a total of 1500 unique triplets of (3D-model, HDRI background, camera setting) per class. For each such choice of 3D-model, background, and camera setting, we generate 1000 different images by rotating the camera around the object by a random angle and randomly varying the brightness of the generated image.

Our validation and test sets use similar 3DB parameters, but they are generated independently with different 3D models and HDRI backgrounds. Specifically, we choose 3 new 3D-models per class and find 10 new HDRI backgrounds in total. For each class, we randomly choose 4 out of the 10 backgrounds to be associated with that class, in order to add a realistic background bias. This helps us mimic real image datasets like CIFAR10, where there also exists a background bias. For each class, we use exactly one of the 4 possible height and zoom settings for the camera instead of using all 4. These settings are shown in Table 5.

Using this combination of 3D-models and backgrounds for each class, we generate 1000 images for each validation set and test set.

Thus, we now associate the train, validation, and test sets of 3DB with the dataset projection setting. When using 3DB as the auxiliary dataset, we use the training set, which has 1500 unique labeled source distributions per class. Each source corresponds to a specific triplet of

[1]We choose camera heights to be 0 or -1 instead in the case of the two classes *airplane* and *bird* because those classes are usually photographed from below rather than from above.

Class	Height	Zoom
Airplane	-1	1.5
Automobile	0	3.0
Bird	0	3.0
Cat	0	1.5
Deer	1	1.5
Dog	0	1.5
Frog	1	3.0
Horse	0	1.5
Ship	0	3.0
Truck	0	1.5

Table 5: Height and zoom values (randomly) chosen for each class in the 3DB validation and test sets.

(3D-model, HDRI background, camera setting). When using 3DB as the target dataset, we use the validation set of 3DB, and when using 3DB as the test set, we use the test set of 3DB.

C.1.2 Sentiment analysis.

Similar to the computer vision setting, we standardize all of the language datasets into a single unified setting. Specifically, all datasets are loaded via the `Datasets` package built by HuggingFace. All sentences are tokenized with the `BertTokenizerFast` tokenizer from the pre-trained `bert-base-case` model on `https://huggingface.co/` with maximum length padding and truncation. All datasets are subsampled to 100 examples total.

Note that, since each sentiment analysis task has a different goal, we cannot canonicalize all scenarios to predict the same set of labels like we could in the computer vision setting. For example, it is unclear at what point a 1 through 5 star rating on a Yelp review translates to positive or negative sentiment in the Stanford Sentiment Treebank.

Stanford Sentiment Treebank (SST). The target and test datasets for SST are random subsets of the corresponding validation and test splits. When SST is used as an auxiliary dataset, we take the SST train set and discretize the sentiment scores into 10 bins of size 0.1. Each bin forms a source distribution for the auxiliary SST dataset. The dataset is available at `https://huggingface.co/datasets/sst`.

Emoji. The Emoji dataset is a subset of the `tweet_eval` dataset on HuggingFace. Specifically, the target and test datasets for Emoji are random subsets of the corresponding validation and test splits. When used as an auxiliary dataset, we use the emoji labels to divide the dataset into 20 source distributions. The dataset is available at `https://huggingface.co/datasets/tweet_eval`.

Emotion. The Emotion dataset is a subset of the `tweet_eval` on HuggingFace. Specifically, the target and test datasets for Emotion are random subsets of the corresponding validation and test splits. When used as an auxiliary dataset, we use the emotion labels to divide the dataset into 4 source distributions. The dataset is available at `https://huggingface.co/datasets/tweet_eval`.

Yelp. The Yelp dataset does not have a validation set, so we split the training data to get a target dataset. Specifically, the target dataset is the second half of the Yelp training split, and the test dataset is the test split.

The Yelp auxiliary data comes from the first half of the Yelp training split. This subset is further divided into source distributions by using the review scores of each example. Specifically, we divide the subset into 5 source distributions corresponding to 1 star reviews through 5 star reviews. The dataset is available at `https://huggingface.co/datasets/yelp_review_full`.

DailyDialog. The DailyDialog dataset is the `dyda_e` subset of the `sillicone` dataset, a collection of resources designed for spoken language. We make one modification to the dataset: since some of the classes have fewer than 100 examples while others have thousands, we subset the dataset to classes $(3, 4, 6)$ (corresponding to happiness, no emotion, and surprise) which have sufficient representation. We then balance the classes to have no more than 100 examples per class.

The target and test datasets thus come from the corresponding validation and test splits. When used as auxiliary data, we divide the training set into 3 source distributions corresponding to the emotion labels. The dataset is available at `https://huggingface.co/datasets/silicone`.

C.2 Dataset projection benchmark

In this section, we provide the full experimental setup for projecting datasets, and training neural networks to either evaluate the projection or to use the projection as dataset augmentation.

C.2.1 Experimental setup

For each experiment we choose two datasets, one auxiliary dataset and one target dataset. For final evaluation, we evaluate on the test set of the target dataset.

At a high level, we first use active set or PGD to project each class of the target dataset onto the auxiliary dataset as described in Appendix B. After finding the projected dataset, we validate its effectiveness by training deep learning models on it and reporting the mean and standard deviation of the test accuracy over 5 training runs. These training runs can be in combination with the target dataset (to measure augmentation performance) or without the target dataset (to validate the projection method).

C.2.2 Baselines

We compare three different methods of choosing a subset of the auxiliary data to determine the most effective one. In the rest of this section, we refer to them as follows.

- **AS-PD:** We find the projected dataset via active set.
- **PGD-PD:** We find the projected dataset via PGD.
- **QP-PD:** We find the projected dataset via quadratic programming.
- **Random** (Baseline): We use a equally-sized subset of the auxiliary dataset chosen uniformly at random from the sources. This baseline isolates the importance of finding the "right" subset of the auxiliary dataset.

C.3 Training specifics for image classification

Our computer vision models are trained with settings that are standardized across all experiments in this paper.

- We use a standard ResNet-18 architecture (He et al., 2016).
- We always randomly subsample 100 examples for each dataset we use during model training. In other words, all training datasets (e.g. the target dataset, the projected dataset, the random portion of the auxiliary dataset) have size 100.
- We use SGD to train for 100 epochs. We set the batch size to 32, the learning rate to 0.1 (with learning rate drops every 33 epochs), the momentum parameter to 0.9, and the weight decay to 5e−4.
- For data augmentation of the regular training runs (the ones not labeled as "DA"), we use random crop and random horizontal flip. When using stronger data augmentation (the ones labeled as "DA"), we use the default settings of TrivialAugment (Müller & Hutter, 2021).

- For each experiment, we repeat each training run with 5 different seeds in order to get the mean and standard deviation of the test accuracies.

C.4 Training specifics for sentiment analysis

Our language models are trained with settings that are standardized across all experiments in this paper.

- We use a standard BERT architecture (He et al., 2016), specifically the pretrained `bert-base-cased` from HuggingFace.
- We use the HuggingFace `Trainer` to fine-tune with the default arguments.
- Data augmentation via backtranslation is done via the French language using the open source translation Opus-MT models `Helsinki-NLP/opus-mt-en-ROMANCE` and `Helsinki-NLP/opus-mt-ROMANCE-en` from HuggingFace (Tiedemann & Thottingal, 2020).
- For each experiment, we repeat each training run with 5 different seeds in order to get the mean and standard deviation of the test accuracies.

C.5 Projecting dataset specifics

When projecting datasets, we use the same following settings for both AS-PD and PGD-PD, most of which are typical settings from the optimization literature:

- We use $\epsilon \in \{1, 0.1, 0.01, 0.001, 0.0001\}$
- We use a learning rate of $\gamma = 1$
- We use a tolerance level of 10^{-6}. If an iteration does not change by more than this amount, we terminate the algorithm
- We run for a maximum of 1000 iterations
- We use a momentum parameter of $\beta = 0.9$ for the soft active set update
- We use $(\delta, \gamma) = (0.9, 0.9)$ as parameters for the Armijo line search
- We use $\alpha = 500$ as the MMD kernel hyperparameter

For QP-PD we use the following settings:

- We use $\epsilon = 0.01$ as the tolerance threshold for the quadratic program
- We use $B = \#$ of sources to allow any individual source to have maximum influence
- We use $\gamma = 100$ as the MMD kernel hyperparameter between datasets
- We use $\alpha = 500$ as the MMD kernel hyperparameter between examples
- We used CVXPY to solve the QP with the default solver, OSQP (`https://osqp.org/`)

C.6 Compute requirements

All experiments across all scenarios can in theory be run with a single 1080Ti with 1 CPU core (double-threaded). Most projection and training runs can be completed within a few hours, depending on the number of source distributions. For the projections onto the 3DB auxiliary dataset, we use multi-processing with 4 CPU cores and 2 GPUs to accelerate the projection process due to the large number of source distributions, which makes the numerical gradient estimate expensive. Due to the large number of scenarios and experiments, we run our scenarios in parallel across a cluster consisting of 56 GPUs (1080Ti). In total, there are 36 scenarios, 3 training experiments per scenario, 4-5 methods per experiment, and 5 random seeds, for a total of 2340 models trained.

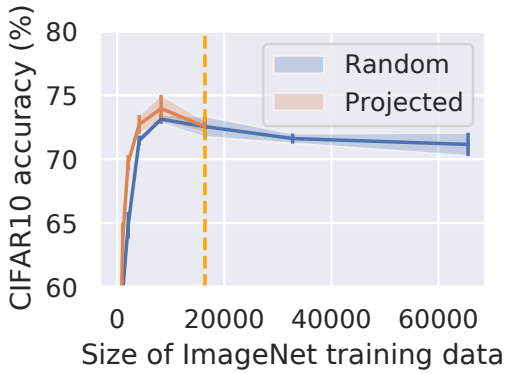


Figure 13: Comparing test accuracies after projecting CIFAR10 on ImageNet and training on either (1) the projected dataset (2) a random subset of the full auxiliary dataset. Even though the entire projected dataset (marked by the dotted line) is a strict subset of the full auxiliary dataset, training on just the projected dataset gives higher test accuracy than training on the full auxiliary dataset (marked by the datapoint at the far right of the blue line of the graph).

D Additional experimental results

We include additional experimental results not presented in the main paper here. We begin with an overview of all our additional experiments, followed by the full set of the results.

Measuring target-alignment via training. One approach to validate our projected datasets is to train a model on the projected data, and evaluate that model on the target dataset, as depicted in Figure 5. Intuitively, a better alignment with the target dataset should lead to better accuracy. Indeed, we find that our framework can find projections that are more accurate at predicting the target dataset. The full table of results evaluating our projections for PGD-PD and AS-PD on our benchmark are in Table 7 and Table 11.

Ablation: alignment of the complementary subset. Since our framework searches for the most target-aligned subset, we would expect the complementary subset to have poor alignment with the target dataset. Indeed, this sanity check turns out to be the case: in Table 7 we evaluate the complementary subsets of those chosen by our framework (Not PGD-PD and Not AS-PD), and find that these subsets have even worse performance than the random baseline.

Comparing AS-PD with training on the full auxiliary dataset. Figure 13 compares training on the projected dataset (AS-PD) with training on the full auxiliary dataset. The projected dataset is smaller, and training on an equally-sized random subset of the full auxiliary dataset results in worse performance at all dataset sizes up to the total size of AS-PD. Using even more auxiliary data, including the full auxiliary dataset, only degrades performance further, due to the biases present in the auxiliary dataset. In this case, training only on the smaller projected dataset is beneficial compared to training on all available auxiliary data, even though the full auxiliary dataset has strictly more datapoints than the projected dataset.

Comparing AS-PD and PGD-PD. In Table 6, we examine the differences between the projected datasets found using PGD and active set. Overall, AS-PD performs slightly better, and both methods of solving dataset projection outperform sampling uniformly at random from the auxiliary dataset.

Quantitative comparison of AS-PD and the random baseline via distance metrics. In Figure 6, we showed that when projecting Oxford-IIIT onto ImageNet, the projected

Aux. \Target	CIFAR10	Oxford-IIIT Pet	STL10	3DB
ImageNet	2.54	1.64	1.40	−0.10
CIFAR10	1.83	2.84	−1.68	5.49
Oxford-IIIT Pet	2.83	−2.20	0.82	−2.74
STL10	1.73	2.56	2.03	0.90
3DB	1.33	0.88	2.72	2.94

Table 6: Improvement of active set over PGD in approximating target datasets, measured by the difference in test accuracy between a model trained on AS-PD and a model trained on PGD-PD. Both methods improve approximation over uniformly random sampling from the full auxiliary dataset, but active set is slightly better than PGD (that is, the measured difference is positive) in 16 out of 20 settings.

dataset is much closer in distance to the target dataset than the auxiliary dataset is. We showed this for the MMD distance, which we explicitly use active set and PGD to opimize for, as well as for 3 alternate distance metrics discussed in (Zeng et al., 2017) — the contrast, luminance, and random filter response (RFR). For completeness, we produce the same plot for every possible choice of auxiliary dataset and target dataset. In the majority of cases, such as in Figure 14, the quantitative distance metrics add further confirmation that the projected dataset is more effective at approximating the target dataset than the original auxiliary dataset. In a few cases, although dataset projection is able to decrease the MMD distance to the target dataset, it does not always decrease the other distance metrics.

D.1 When projected datasets can and can't help

Surprisingly, projected datasets that approximate the target better do not always lead to better results when augmenting the target. When using 3DB as the auxiliary data, projected datasets are better at approximating the target, but the Random baseline is often better at augmenting the target. This may be because the 3DB dataset is generated by pairing random backgrounds with random 3D-models of each class—thus, training on the baseline dataset encourages invariance to background and different 3D-models, which may actually be more beneficial for generalization. Understanding exactly what properties of a projected dataset make it most useful for augmentation is an interesting future direction.

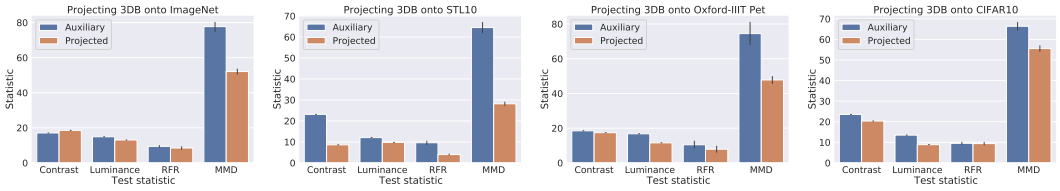


Figure 14: 1D distance metrics showing how close the projected and auxiliary datasets are to the target dataset (lower is better). 3DB is the target dataset.

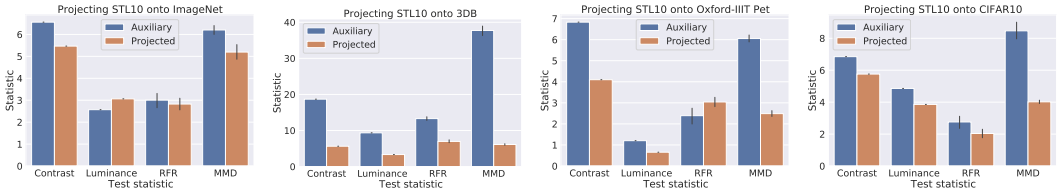


Figure 15: 1D distance metrics showing how close the projected and auxiliary datasets are to the target dataset (lower is better). STL is the target dataset.

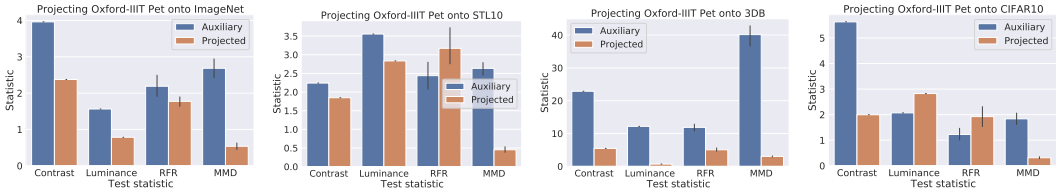


Figure 16: 1D distance metrics showing how close the projected and auxiliary datasets are to the target dataset (lower is better). Oxford-IIIT Pet is the target dataset.

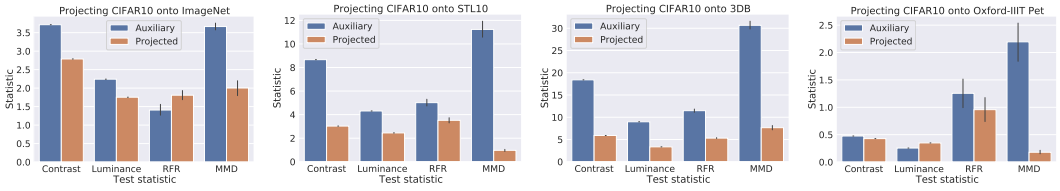


Figure 17: 1D distance metrics showing how close the projected and auxiliary datasets are to the target dataset (lower is better). CIFAR10 is the target dataset.

D.2 Experiments for vision benchmark

We present the complete set of results for our image classification scenarios. In Table 7, we record the target-alignment of our projected datasets as measured by training on the projected dataset. In Table 8, we record the performance of training on the target dataset augmented with projected data. In Table 9, we record similar results but combined with TrivialAugment data augmentation.

D.3 Domain adaptation

In this section, we compare to SwAV (Caron et al., 2020), a unsupervised domain adaptation approach based on contrastive pre-training. This method was found to be state of the art in domain adaptation (Shen et al., 2022). We compare our method with SwAV in the setting

Auxiliary	Target	Random	PGD-PD	Not PGD-PD	AS-PD	Not AS-PD
ImageNet	CIFAR10	33.9 ± 0.4	35.5 ± 0.7	27.3 ± 0.7	**37.0** ± 1.0	30.7 ± 0.8
Oxford-IIIT Pet		53.6 ± 0.8	50.5 ± 2.9	**54.6** ± 1.9	53.0 ± 4.1	54.5 ± 1.1
STL10		12.2 ± 1.9	28.3 ± 0.7	10.7 ± 2.5	**33.8** ± 1.2	13.1 ± 1.5
3DB		22.6 ± 1.0	24.3 ± 0.3	22.2 ± 0.8	**25.4** ± 0.8	22.6 ± 0.7
ImageNet	Oxford-IIIT Pet	50.1 ± 2.5	51.8 ± 1.5	48.0 ± 2.1	**55.4** ± 1.6	51.8 ± 2.4
CIFAR10		53.0 ± 1.0	49.4 ± 1.5	49.4 ± 1.7	**55.6** ± 2.1	51.3 ± 1.9
STL10		51.6 ± 2.4	**54.0** ± 1.9	49.2 ± 0.8	53.3 ± 2.1	50.9 ± 2.3
3DB		52.0 ± 2.9	49.0 ± 1.9	48.8 ± 2.3	**53.7** ± 1.6	49.1 ± 1.1
ImageNet	STL10	37.4 ± 1.0	37.7 ± 0.5	37.0 ± 0.5	**38.3** ± 1.1	37.3 ± 0.8
CIFAR10		41.2 ± 1.2	34.8 ± 4.3	35.5 ± 1.1	**38.8** ± 0.5	38.1 ± 1.5
Oxford-IIIT Pet		**55.5** ± 1.5	50.1 ± 3.5	49.7 ± 2.2	55.1 ± 0.9	52.2 ± 1.2
3DB		24.6 ± 1.2	26.3 ± 1.3	21.0 ± 1.1	**30.4** ± 1.5	24.5 ± 1.9
ImageNet	3DB	27.0 ± 5.0	22.2 ± 1.7	16.3 ± 2.4	**29.5** ± 1.7	22.1 ± 1.6
CIFAR10		30.8 ± 3.7	32.7 ± 4.5	19.6 ± 3.1	**35.4** ± 1.0	23.8 ± 0.9
Oxford-IIIT Pet		66.8 ± 8.8	68.4 ± 14.4	55.9 ± 12.8	**71.2** ± 8.0	56.3 ± 12.1
STL10		12.6 ± 4.0	34.8 ± 3.5	5.7 ± 4.0	**37.7** ± 4.0	12.2 ± 3.4

Table 7: Approximating target datasets with auxiliary data. In this experiment, we train on auxiliary data and test on target data. Higher test accuracy corresponds to better approximation quality. AS-PD typically performs the best, and PGD-PD also typically outperforms Random.

Auxiliary	Target	Target Only	Target + Random	Target + PGD-PD	Target + AS-PD	Target + QP-PD
ImageNet	CIFAR10	43.6 ± 0.4	54.8 ± 2.1	55.3 ± 1.8	57.0 ± 3.0	67.9 ± 0.7
Oxford-IIIT Pet		51.9 ± 2.1	54.2 ± 2.6	54.8 ± 2.0	59.3 ± 0.2	**70.0** ± 1.2
STL10		41.0 ± 1.6	37.2 ± 1.8	44.6 ± 0.4	44.3 ± 1.1	56.0 ± 0.6
3DB		43.6 ± 1.1	50.0 ± 2.2	43.2 ± 0.9	49.4 ± 0.6	-
ImageNet	Oxford-IIIT Pet	49.3 ± 1.3	58.2 ± 2.7	55.1 ± 2.2	57.4 ± 3.7	71.4 ± 1.0
CIFAR10		52.1 ± 1.9	54.2 ± 2.5	55.4 ± 2.0	59.5 ± 2.9	**73.4** ± 0.5
STL10		47.6 ± 0.9	54.2 ± 2.5	55.7 ± 1.9	55.0 ± 2.3	64.2 ± 1.4
3DB		51.1 ± 1.9	53.8 ± 3.8	51.8 ± 2.7	55.7 ± 1.0	-
ImageNet	STL10	41.7 ± 2.3	55.9 ± 1.0	53.2 ± 2.1	55.1 ± 1.3	59.3 ± 9.7
CIFAR10		39.5 ± 0.9	58.6 ± 1.8	53.2 ± 0.8	58.4 ± 1.6	68.4 ± 1.2
Oxford-IIIT Pet		54.2 ± 2.8	54.3 ± 3.1	65.f1 ± 1.4	61.6 ± 1.1	**72.5** ± 0.9
3DB		42.8 ± 2.4	52.2 ± 1.5	46.9 ± 0.8	51.0 ± 1.4	-
ImageNet	3DB	84.0 ± 2.1	89.4 ± 1.6	83.2 ± 0.6	87.7 ± 1.4	91.1 ± 1.3
CIFAR10		80.5 ± 2.0	89.0 ± 3.3	86.3 ± 0.4	89.5 ± 0.5	94.8 ± 0.6
Oxford-IIIT Pet		69.8 ± 1.6	73.4 ± 7.2	82.2 ± 3.3	82.3 ± 10.2	**96.3** ± 1.6
STL10		68.5 ± 5.8	76.5 ± 6.9	77.8 ± 1.6	79.6 ± 1.7	87.4 ± 2.5

Table 8: Augmenting a target dataset with auxiliary data for vision scenarios. In this experiment, we add auxiliary data to a target dataset for training, and test on target data. For all target datasets, we can find an auxiliary dataset in which augmenting with AS-PD or PGD-PD performs the best. For each target dataset, we highlight the choice of auxiliary dataset where either AS-PD or PGD-PD has the largest benefit in augmenting the target dataset.

where ImageNet is the auxiliary dataset with the goal of improving performance on CIFAR10, Oxford-IIIT Pet, STL10, and 3DB.

We use the released ImageNet models from `https://github.com/facebookresearch/swav`. We note several differences in the experimental setup. (1) The released model uses a ResNet-50 architecture, which is larger than the ResNet-18 architecture we used in our experiments, and (2) the model was pretrained on the entire ImageNet dataset, whereas our projected

Auxiliary	Target	DA	Random + DA	PGD-PD + DA	AS-PD + DA
ImageNet	CIFAR10	45.5 ± 1.0	52.6 ± 0.9	55.2 ± 1.5	52.3 ± 3.3
Oxford-IIIT Pet		53.1 ± 2.2	52.6 ± 1.7	53.5 ± 2.7	**57.8** ± 3.3
STL10		47.5 ± 1.8	37.1 ± 0.9	44.8 ± 1.3	44.9 ± 1.2
3DB		46.2 ± 2.1	48.2 ± 0.6	47.0 ± 1.9	48.0 ± 1.4
ImageNet	Oxford-IIIT Pet	47.0 ± 2.2	50.6 ± 4.4	57.0 ± 2.8	50.9 ± 3.5
CIFAR10		54.0 ± 3.1	53.5 ± 2.8	56.4 ± 2.0	**58.1** ± 2.0
STL10		48.9 ± 2.0	48.9 ± 1.4	55.8 ± 3.1	53.2 ± 1.3
3DB		52.4 ± 3.5	50.7 ± 2.9	50.7 ± 2.3	50.4 ± 2.5
ImageNet	STL10	48.9 ± 1.6	54.0 ± 1.1	53.8 ± 1.5	52.9 ± 1.0
CIFAR10		49.7 ± 0.9	57.8 ± 1.5	57.3 ± 1.5	56.1 ± 3.1
Oxford-IIIT Pet		54.5 ± 4.3	51.9 ± 1.7	**63.0** ± 1.1	58.7 ± 2.5
3DB		48.6 ± 1.1	48.5 ± 0.5	49.6 ± 0.3	49.0 ± 1.2
ImageNet	3DB	92.1 ± 1.9	93.6 ± 1.4	92.4 ± 0.8	94.3 ± 0.5
CIFAR10		91.4 ± 1.8	94.4 ± 0.6	92.4 ± 1.3	94.5 ± 1.0
Oxford-IIIT Pet		73.0 ± 7.6	59.5 ± 2.9	**91.5** ± 0.9	86.3 ± 13.3
STL10		86.9 ± 2.3	91.3 ± 0.8	89.1 ± 1.0	88.2 ± 0.6

Table 9: Augmenting a target dataset with auxiliary data, combined with TrivialAugment data augmentation. In this experiment, we add auxiliary data and TrivialAugment to a target dataset for training, and test on target data. For all target datasets, we can find an auxiliary dataset in which augmenting with AS-PD or PGD-PD performs the best. For each target dataset, we highlight the choice of auxiliary dataset where either AS-PD or PGD-PD has the largest benefit in augmenting the target dataset.

Method	CIFAR10	Oxford-IIIT	STL10	3DB
QP-PD	67.9 ± 0.7	71.4 ± 51.0	59.3 ± 9.7	91.1 ± 1.3
SwAV (Caron et al., 2020)	63.9 ± 3.9	56.0 ± 1.7	19.0 ± 2.7	90.8 ± 0.7

Table 10: A comparison between projected datasets and a contrastive pre-training approach called SwAV (Caron et al., 2020) recently found to be competitive with state of the art (Shen et al., 2022). Here, we adapt ImageNet as the auxiliary dataset to CIFAR10, Oxford-IIIT Pets, STL10, and 3DB. Note that SwAV uses a much larger auxiliary dataset and model architecture but struggles to adapt to the low amount of data in the target dataset.

datasets use a much smaller subset corresponding to the WordNet hierarchy. Both of these differences favor SwAV.

The results are summarized in Table 10. We find that in our projected dataset setting, state-of-the-art domain adaptation does not perform as well as directly adding ImageNet data via projected datasets. This trend is because in these settings, the amount of data in the target dataset is low enough such that adding more data helps significantly more than pretraining. This is highlighted by the STL10 target, where the pretrained SwAV grossly overfits to the STL10 target dataset and has poor performance.

D.4 Experiments for language benchmark

We present the complete set of results for our sentiment analysis scenarios. In Table 11, we record the target-alignment of our projected datasets as measured by training on the projected dataset. In Table 12, we record the performance of training on the target dataset augmented with projected data. In Table 13, we record similar results but combined with back-translation data augmentation.

Auxiliary	Target	Random	PGD-PD	Not PGD-PD	AS-PD	Not AS-PD
Emoji	SST	**57.6** ± 10.1	54.8 ± 7.5	56.2 ± 6.6	57.0 ± 0.0	54.0 ± 4.1
Emotion		52.2 ± 4.7	50.6 ± 3.9	57.0 ± 7.8	55.8 ± 3.5	**57.4** ± 0.5
Yelp		50.4 ± 7.7	49.4 ± 5.2	**57.8** ± 6.0	55.8 ± 6.7	50.6 ± 9.1
DailyDialog		51.6 ± 4.8	**61.2** ± 9.8	51.4 ± 4.5	56.6 ± 11.4	55.6 ± 1.9
SST	Emoji	3.2 ± 1.0	4.0 ± 1.5	**5.4** ± 2.6	**5.4** ± 2.1	4.0 ± 1.5
Emotion		3.8 ± 1.5	4.2 ± 1.5	**5.2** ± 1.3	4.2 ± 1.2	3.6 ± 0.8
Yelp		4.2 ± 1.8	4.8 ± 1.7	4.8 ± 1.7	**6.4** ± 3.1	3.2 ± 0.4
DailyDialog		2.6 ± 0.5	**6.2** ± 3.1	3.6 ± 1.2	5.0 ± 2.8	3.0 ± 0.0
SST	Emotion	16.6 ± 7.0	27.0 ± 6.3	13.4 ± 4.1	29.2 ± 2.8	**31.0** ± 0.0
Emoji		16.2 ± 5.3	20.4 ± 3.3	15.0 ± 1.8	**28.6** ± 2.3	**28.6** ± 3.5
Yelp		16.4 ± 5.7	20.4 ± 8.8	22.8 ± 10.0	**29.0** ± 2.7	24.6 ± 6.7
DailyDialog		15.0 ± 6.7	23.8 ± 7.6	20.2 ± 6.8	29.8 ± 2.9	**30.4** ± 1.4
SST	Yelp	21.8 ± 1.6	22.6 ± 3.2	19.4 ± 2.7	22.2 ± 2.4	**24.4** ± 0.8
Emoji		21.0 ± 0.0	19.0 ± 2.6	21.0 ± 0.0	**23.2** ± 4.1	16.8 ± 2.9
Emotion		21.0 ± 0.0	**22.8** ± 2.2	20.8 ± 0.4	21.2 ± 4.3	19.6 ± 4.1
DailyDialog		21.0 ± 0.0	21.8 ± 1.3	21.6 ± 1.4	**23.2** ± 2.6	20.4 ± 6.6
SST	DailyDialog	14.6 ± 3.1	16.8 ± 1.0	15.8 ± 2.0	**17.4** ± 3.4	14.2 ± 2.4
Emoji		13.6 ± 2.2	13.6 ± 1.5	14.4 ± 2.9	**23.6** ± 2.5	19.6 ± 4.2
Emotion		14.6 ± 2.4	19.4 ± 5.9	15.0 ± 4.3	**23.4** ± 5.8	12.0 ± 2.7
Yelp		**16.6** ± 3.3	13.6 ± 0.5	14.2 ± 1.5	**16.6** ± 5.0	13.2 ± 3.1

Table 11: Approximating target datasets with auxiliary data. In this experiment, we train on auxiliary data and test on target data. Higher test accuracy corresponds to better approximation quality. AS-PD typically performs the best, and PGD-PD also typically outperforms Random.

Auxiliary	Target	Target Only	Target + Random	Target + PGD-PD	Target + AS-PD	Target + QP-PD
Emoji	SST	74.2 ± 6.0	66.2 ± 8.8	71.4 ± 6.1	77.0 ± 6.7	64.8 ± 8.8
Emotion			61.2 ± 7.4	64.8 ± 9.9	75.8 ± 9.0	62.8 ± 6.9
Yelp			60.8 ± 8.9	65.6 ± 12.0	**78.6** ± 2.9	68.2 ± 4.0
DailyDialog			63.6 ± 12.1	76.4 ± 5.4	76.4 ± 6.1	60.8 ± 6.9
SST	Emoji	12.4 ± 3.6	12.4 ± 3.8	14.6 ± 3.4	24.0 ± 2.7	7.2 ± 2.1
Emotion			15.0 ± 4.1	13.8 ± 4.1	23.6 ± 2.2	7.6 ± 2.3
Yelp			13.0 ± 3.8	13.6 ± 3.1	22.6 ± 2.2	8.4 ± 2.2
DailyDialog			14.0 ± 3.3	12.8 ± 2.9	**24.4** ± 1.0	8.2 ± 1.3
SST	Emotion	37.6 ± 4.3	39.4 ± 4.1	42.8 ± 7.8	**45.2** ± 4.7	31.0 ± 6.2
Emoji			39.2 ± 1.3	39.4 ± 4.7	44.0 ± 4.3	30.2 ± 6.9
Yelp			41.6 ± 3.2	40.4 ± 4.9	44.8 ± 4.2	31.8 ± 8.1
DailyDialog			43.2 ± 4.1	42.6 ± 8.0	42.6 ± 4.5	30.0 ± 6.6
SST	Yelp	28.6 ± 1.9	27.4 ± 2.4	29.4 ± 4.8	30.0 ± 4.1	32.8 ± 6.0
Emoji			29.2 ± 3.0	27.0 ± 5.4	**34.4** ± 5.7	31.6 ± 4.1
Emotion			27.8 ± 6.9	28.6 ± 4.9	33.6 ± 4.3	34.4 ± 3.9
DailyDialog			29.0 ± 6.4	33.2 ± 4.8	32.0 ± 5.2	30.0 ± 6.6
SST	DailyDialog	37.4 ± 6.3	37.0 ± 5.7	42.6 ± 5.4	61.2 ± 3.4	55.0 ± 8.7
Emoji			36.0 ± 4.9	42.6 ± 5.9	66.2 ± 6.1	58.8 ± 9.4
Emotion			37.6 ± 5.0	35.0 ± 8.0	**67.4** ± 6.0	56.8 ± 8.5
Yelp			38.2 ± 5.6	36.0 ± 7.3	64.6 ± 7.5	56.0 ± 10.0

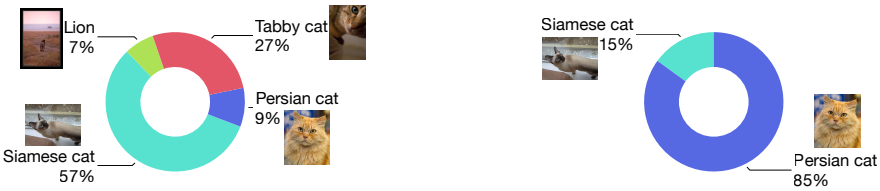
Table 12: Augmenting a target dataset with auxiliary data for language scenarios. In this experiment, we add auxiliary data to a target dataset for training, and test on target data. For each target dataset, we highlight the choice of auxiliary dataset where either AS-PD or PGD-PD has the largest benefit in augmenting the target dataset.

Auxiliary	Target	DA	Random + DA	PGD-PD + DA	AS-PD + DA
Emoji	SST	78.4 ± 3.5	78.4 ± 3.5	78.2 ± 2.1	78.4 ± 2.1
Emotion			76.8 ± 6.5	76.0 ± 4.0	76.2 ± 5.7
Yelp			77.8 ± 1.9	78.0 ± 3.4	80.2 ± 2.7
DailyDialog			76.2 ± 4.7	76.0 ± 2.6	**80.8** ± 2.7
SST	Emoji	22.8 ± 3.2	22.8 ± 3.2	22.2 ± 1.7	24.2 ± 1.9
Emotion			23.2 ± 1.3	22.8 ± 1.7	24.0 ± 0.6
Yelp			22.8 ± 1.3	23.6 ± 2.7	24.2 ± 4.0
DailyDialog			23.8 ± 2.0	23.6 ± 1.9	**25.0** ± 2.4
SST	Emotion	50.6 ± 9.0	50.6 ± 9.0	56.0 ± 4.7	59.2 ± 7.4
Emoji			52.6 ± 6.7	52.6 ± 5.1	**59.6** ± 4.4
Yelp			53.4 ± 6.7	53.6 ± 3.3	58.2 ± 3.8
DailyDialog			51.8 ± 8.0	51.2 ± 4.2	57.6 ± 9.0
SST	Yelp	32.4 ± 4.3	32.4 ± 4.3	32.2 ± 6.5	35.8 ± 3.7
Emoji			33.8 ± 6.6	33.6 ± 6.6	**36.8** ± 4.2
Emotion			30.2 ± 3.2	33.2 ± 5.3	36.6 ± 5.4
DailyDialog			29.4 ± 1.9	32.8 ± 5.6	35.8 ± 4.4
SST	DailyDialog	76.2 ± 2.0	76.2 ± 2.0	75.6 ± 1.2	75.4 ± 2.3
Emoji			75.4 ± 1.7	75.2 ± 1.9	76.2 ± 1.5
Emotion			76.0 ± 2.7	76.0 ± 1.1	75.0 ± 2.5
Yelp			74.8 ± 2.9	**76.8** ± 1.5	75.0 ± 1.8

Table 13: Augmenting a target dataset with auxiliary data, combined with back-translation augmentation via the French language. In this experiment, we add auxiliary data and back-translated data to a target dataset for training, and test on target data. For each target dataset, we highlight the choice of auxiliary dataset where either AS-PD or PGD-PD has the largest benefit in augmenting the target dataset.

(a) Angry (b) Joy

Figure 18: A visualization depicting the result of projecting various classes of the Emotion dataset (angry and joy) onto the Emoji dataset. Each doughnut corresponds to the subset of the Emoji dataset that is closest to the target Emotion class as calculated by our framework.



(a) Oxford-IIIT Pet (b) 3DB

Figure 19: A visualization depicting the result of projecting cat classes from various target datasets (Oxford-IIIT Pet and 3DB) onto the ImageNet cats. Each doughnut represents the subset of the ImageNet cats that is closest to the cat class from the target dataset as calculated by our framework.

E Visualizations of projected datasets

In this section, we present additional visualizations of projected datasets. These visualizations provide a way to analyze the composition of a target dataset via inspection of the resulting source distributions proportions.

Figure 18 shows the projection of the Angry and Joy classes of the Emotion dataset onto the Emoji dataset. Here, we see that the Angry projection consists of an Emoji with tears and an Emoji with a tongue sticking out. We suspect this is due to the usage of these emojis in sarcastic or cynical expressions that are most closely aligned with the Angry class, since the Emoji dataset does not contain any clearly angry emojis. The Joy projection contains generally happy emojis, including a heart emoji that none of the other Emotion classes had.

Figure 19 shows the projection Oxford-IIIT Pet and 3DB onto ImageNet cats. The Oxford-IIIT Pet projection has largely household cats, except for a small proportion of lions. The 3DB projection consists primarily of two types of cats, suggesting that the 3DB cats do not contain very much variation.