A Data-Based Perspective on Transfer Learning

Saachi Jain* MIT saachij@mit.edu
Hadi Salman* MIT hady@mit.edu
Alaa Khaddaj * MIT alaakh@mit.edu
Eric Wong MIT wongeric@mit.edu
Sung Min Park MIT sp765@mit.edu
Aleksander Mądry MIT madry@mit.edu

Abstract

It is commonly believed that in transfer learning including more pre-training data translates into better performance. However, recent evidence suggests that *removing* data from the source dataset can actually help too. In this work, we take a closer look at the role of the source dataset's composition in transfer learning and present a framework for probing its impact on downstream performance. Our framework gives rise to new capabilities such as pinpointing transfer learning brittleness as well as detecting pathologies such as data-leakage and the presence of misleading examples in the source dataset. In particular, we demonstrate that removing detrimental datapoints identified by our framework improves transfer learning performance from ImageNet on a variety of target tasks.[1]

1 Introduction

Transfer learning enables us to adapt a model trained on a *source* dataset to perform better on a downstream *target* task. This technique is employed in a range of machine learning applications including radiology [WPL+17; KEB+21], autonomous driving [KP17; DGS19], and satellite imagery analysis [XJB+16; WAL19]. Despite its successes, however, it is still not clear what the drivers of performance gains brought by transfer learning actually are.

So far, a dominant approach to studying these drivers focused on the role of the source *model*—i.e., the model trained on the source dataset. The corresponding works involve investigating the source model's architecture [KEB+21], accuracy [KSL19], adversarial vulnerability [SIE+20; UKE+20], and training procedure [JLH+19; KRJ+22]. This line of work makes it clear that the properties of the source model has a significant impact on transfer learning. There is some evidence, however, that the source *dataset* might play an important role as well [HAE16; NPV+18; KBZ+19]. For example, several works have shown that while increasing the size of the source dataset generally boosts transfer learning performance, *removing* specific classes can help too [HAE16; NPV+18; KBZ+19]. All of this motivates a natural question:

How can we pinpoint the exact impact of the source dataset in transfer learning?

Our Contributions. In this paper, we present a framework for measuring and analyzing the impact of the source dataset's composition on transfer learning performance. In particular, our framework allows us to study how the quality of the transfer learning model's predictions changes when subsets of the source dataset are removed. This enables us, in turn, to automatically find subsets of the source dataset that—positively or negatively—impact downstream behavior. Using our framework, we can:

*Equal contribution.
[1]Code is available at https://github.com/MadryLab/data-transfer

- Pinpoint what parts of the source dataset are most utilized by the downstream task.
- Automatically extract granular subpopulations in the target dataset through projection of the fine-grained labels of the source dataset.
- Remove detrimental data from the source dataset to improve transfer learning performance.

Finally, we demonstrate how our framework can be used to find subsets of ImageNet [DDS+09] that, when removed, give rise to better downstream performance on a variety of image classification tasks.

2 A Data-Based Framework for Studying Transfer Learning

In order to pinpoint the role of the source dataset in transfer learning, we need to understand how the composition of that source dataset impacts the downstream model's performance. To do so, we draw inspiration from supervised machine learning approaches that study the impact of the training data on the model's subsequent predictions. In particular, these approaches capture this impact via studying (and approximating) the counterfactual effect of excluding certain training datapoints. This paradigm underlies a number of techniques, from influence functions [CW82; KL17; FZ20], to datamodels [IPE+22], to data Shapley values [KZ21; KDI+22; GZ19].

Now, to adapt this paradigm to our setting, we study the counterfactual effect of excluding datapoints from the *source* dataset on the downstream, *target* task predictions. In our framework, we will focus on the inclusion or exclusion of entire *classes* in the source dataset, as opposed to individual examples[2]. This is motivated by the fact that, intuitively, we expect these classes to be the ones that embody whole concepts and thus drive the formation of (transferred) features. We therefore anticipate the removal of entire classes to have a more measurable impact on the representation learned by the source model (and consequently on the downstream model's predictions).

Once we have chosen to focus on removal of entire source classes, we can design counterfactual experiments to estimate their influences. A natural approach here, the *leave-one-out* method [CW82; KL17], would involve removing each individual class from the source dataset separately and then measuring the change in the downstream model's predictions. However, in the transfer learning setting, we suspect that removing a single class from the source dataset won't significantly change the downstream model's performance. Thus, leave-one-out methodology may be able to capture meaningful influences only in rare cases. This is especially so as many common source datasets contain highly redundant classes. For example, ImageNet contains over 100 dog-breed classes. The removal of a single dog-breed class might thus have a negligible impact on transfer learning performance, but the removal of all of the dog classes might significantly change the features learned by the downstream model. For these reasons, we adapt the *subsampling* [FZ20; IPE+22] approach, which revolves around removing a random collection of source classes at once.

Computing transfer influences. In the light of the above, our methodology for computing the influence of source classes on transfer learning performance involves training a large number[3] of models with random subsets of the source classes removed, and fine-tuning these models on the target task. We then estimate the influence value of a source class $\mathcal{C}$ on a target example t as the expected difference in the transfer model's performance on example t when class $\mathcal{C}$ was either included in or excluded from the source dataset:

$$\text{Infl}[\mathcal{C} \to t] = \mathbb{E}_S \left[f(t; S) \mid \mathcal{C} \subset S \right] - \mathbb{E}_S \left[f(t; S) \mid \mathcal{C} \not\subset S \right], \tag{1}$$

where $f(t; S)$ is the softmax output[4] of a model trained on a subset S of the source dataset. A positive influence value indicates that including the source class $\mathcal{C}$ helps the model predict the target example t correctly. On the other hand, a negative influence value suggests that the source class $\mathcal{C}$ actually hurts the model's performance on the target example t. We outline the overall procedure in Algorithm 1, and defer a detailed description of our approach to Appendix A.

[2] In Section 4.3, we adapt our framework to calculate more granular influences of individual source examples too.
[3] In this paper, we train 7540 models. Details are in Appendix A.
[4] We experiment with other outputs such as logits, margins, or correctness too. We discuss the corresponding results in Appendix B.

Algorithm 1 Estimation of source dataset class influences on transfer learning performance.

Require: Source dataset $\mathcal{S} = \cup_{k=1}^{K} \mathcal{C}_k$ (with K classes), a target dataset $\mathcal{T} = (t_1, t_2, \cdots, t_n)$, training algorithm $\mathcal{A}$, subset ratio α, and number of models m

Sample m random subsets $S_1, S_2, \cdots, S_m \subset \mathcal{S}$ of size $\alpha \cdot |\mathcal{S}|$:
for $i \in 1$ to m **do**
    Train model f_i by running algorithm $\mathcal{A}$ on S_i
end for
for $k \in 1$ to K **do**
    for $j \in 1$ to n **do**
        $$\text{Infl}[\mathcal{C}_k \to t_j] = \frac{\sum_{i=1}^{m} f_i(t_j; S_i) \mathbb{1}_{\mathcal{C}_k \subset S_i}}{\sum_{i=1}^{m} \mathbb{1}_{\mathcal{C}_k \subset S_i}} - \frac{\sum_{i=1}^{m} f_i(t_j; S_i) \mathbb{1}_{\mathcal{C}_k \not\subset S_i}}{\sum_{i=1}^{m} \mathbb{1}_{\mathcal{C}_k \not\subset S_i}}$$
    end for
end for
return $\text{Infl}[\mathcal{C}_k \to t_j]$, for all $j \in [n], k \in [K]$

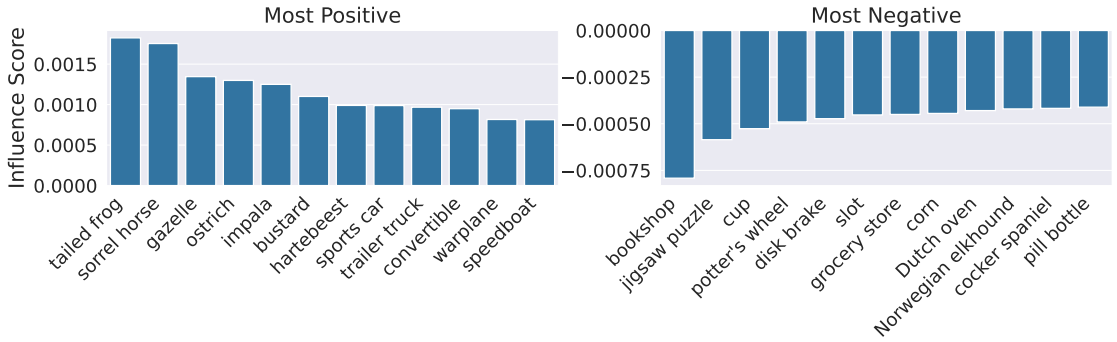


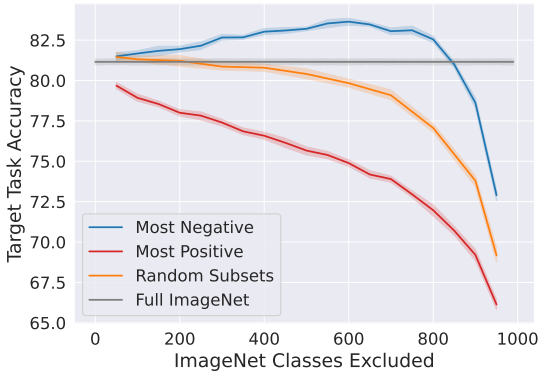
Figure 1: Most positive and negative ImageNet classes ordered based on their overall influence on the CIFAR-10 dataset. The top source classes (e.g., tailed frog and sorrel horse) turn out to be semantically relevant to the target classes (e.g., frog and horse).

3 Identifying the Most Influential Classes of the Source Dataset

In Section 2, we presented a framework for pinpointing the role of the source dataset in transfer learning by estimating the influence of each source class on the target model's predictions. Using these influences, we can now take a look at the classes from the source dataset that have the largest positive or negative impact on the overall transfer learning performance. We focus our analysis on the fixed-weights transfer learning setting (and defer results for full model fine-tuning to Appendix E).

As one might expect, not all source classes have large influences. Figure 1 displays the most influential classes of ImageNet with CIFAR-10 as the target task. Notably, the most positively influential source classes turn out to be directly related to classes in the target task (e.g., the ImageNet label "tailed frog" is an instance of the CIFAR class "frog"). This trend holds across all of the target datasets and transfer learning settings we considered (see Appendix C). Interestingly, the source dataset also contains classes that are overall negatively influential for the target task (e.g., "bookshop" and "jigsaw puzzle" classes). (In Section 4, we will take a closer look at the factors that can cause a source class to be negatively influential for a target prediction.)

How important are the most influential source classes? We now remove each of the most influential classes from the source dataset to observe their actual impact on transfer learning performance (Figure 2a). As expected, removing the most positively influential classes severely degrades transfer learning performance as compared to removing random classes. This counterfactual experiment confirms that these classes are indeed important to the performance of transfer learning. On the other hand, removing the most negatively influential classes actually improves the overall transfer learning performance *beyond what using the entire ImageNet dataset provides* (see Figure 2b).



(a) CIFAR-10 results

Target Dataset	*Source Dataset* Full ImageNet	Removing Bottom Infl.	Hand-picked
AIRCRAFT	36.08 ± 1.07	$\mathbf{36.88 \pm 0.74}$	N/A
BIRDSNAP	38.42 ± 0.40	$\mathbf{39.19 \pm 0.38}$	26.74 ± 0.31
CALTECH101	86.69 ± 0.79	$\mathbf{87.03 \pm 0.30}$	82.28 ± 0.40
CALTECH256	74.97 ± 0.27	$\mathbf{75.24 \pm 0.21}$	67.42 ± 0.39
CARS	39.55 ± 0.32	$\mathbf{40.59 \pm 0.57}$	21.71 ± 0.40
CIFAR10	81.16 ± 0.30	$\mathbf{83.64 \pm 0.40}$	75.53 ± 0.42
CIFAR100	59.37 ± 0.58	$\mathbf{61.46 \pm 0.59}$	55.21 ± 0.52
FLOWERS	$\mathbf{82.92 \pm 0.52}$	82.89 ± 0.48	N/A
FOOD	56.19 ± 0.14	$\mathbf{56.85 \pm 0.27}$	39.36 ± 0.39
PETS	83.41 ± 0.55	$\mathbf{87.59 \pm 0.24}$	87.16 ± 0.24
SUN397	50.15 ± 0.23	$\mathbf{51.34 \pm 0.29}$	N/A

(b) Summary of 11 target tasks

Figure 2: Target task accuracies after removing the K most positively or negatively influential ImageNet classes from the source dataset. Mean/std are reported over 10 runs. **(a)** Results with CIFAR-10 as the target task after removing different numbers of classes from the source dataset. We also include baselines of using the full ImageNet dataset and removing random classes. One can note that, by removing negatively influential source classes, we can obtain a test accuracy that is 2.5% larger than what using the entire ImageNet dataset would yield. Results for other target tasks can be found in Appendix C. **(b)** Peak performances when removing the most negatively influential source classes across a range of other target tasks. We compare against using the full ImageNet dataset or a relevant subset of classes (hand-picked, see Appendix A for details).

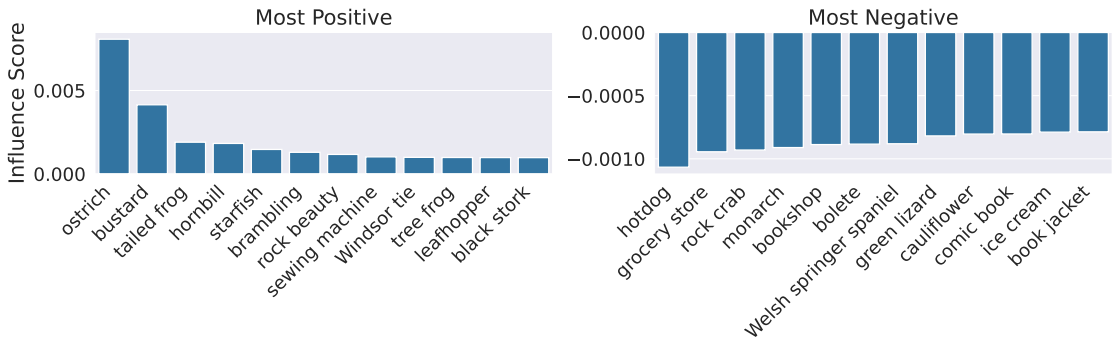


Figure 3: Most positive and negative influencing ImageNet classes for the CIFAR-10 class "bird". These are calculated by averaging the influence of each source class over all bird examples. We find that the most positively influencing ImageNet classes (e.g., "ostrich" and "bustard") are related to the CIFAR-10 class "bird". See Appendix E for results on other CIFAR-10 classes.

4 Probing the Impact of the Source Dataset on Transfer Learning

In Section 3, we developed a methodology for identifying source dataset classes that have the most impact on transfer learning performance. Now, we demonstrate how this methodology can be extended into a framework for probing and understanding transfer learning, including: (1) identifying granular target subpopulations that correspond to source classes, (2) debugging transfer learning failures, and (3) detecting data leakage between the source and target datasets. We focus our demonstration of these capabilities on a commonly-used transfer learning setting: ImageNet to CIFAR-10 (experimental details are in Appendix A).

Figure 4: **Projecting source labels onto the target dataset.** The CIFAR-10 images that were most positively influenced by the ImageNet classes "ostrich" and "fire engine." We find that these images look similar to the corresponding images in the source dataset.

4.1 Capability 1: Extracting target subpopulations by projecting source class labels

Imagine that we would like to find all the ostriches in the CIFAR-10 dataset. This is not an easy task as CIFAR-10 only has "bird" as a label, and thus lacks sufficiently fine-grained annotations. Luckily, however, ImageNet *does* contain an ostrich class! Our computed influences enable us to "project" this ostrich class annotation (and, more broadly, the fine-grained label hierarchy of our source dataset) to find this subpopulation of interest in the target dataset.

Indeed, our examination from Section 3 suggests that the most positively influencing source classes are typically those that directly overlap with the target classes (see Figure 1). In particular, for our example, "ostrich" is highly positively influential for the "bird" class (see Figure 3). To find ostriches in the CIFAR-10 dataset, we thus need to simply surface the CIFAR-10 images which were most positively influenced by the "ostrich" source class (see Figure 4).

It turns out that this type of projection approach can be applied more broadly. Even when the source class is not a direct sub-type of a target class, the downstream model can still leverage salient features from this class — such as shape or color — to predict on the target dataset. For such classes, projecting source labels can extract target subpopulations which share such features. To illustrate this, in Figure 5, we display the CIFAR-10 images that are highly influenced by the classes "starfish" and "rapeseed" (both of which do not directly appear in the CIFAR-10 dataset). For these classes, the most influenced CIFAR-10 images share the same shape ("starfish") or color ("rapeseed") as their ImageNet counterparts. More examples of such projections can be found in Appendix E.

4.2 Capability 2: Debugging the failures of a transferred model

Our framework enables us to also reason about the possible mistakes of the transferred model caused by source dataset classes. For example, consider the CIFAR-10 image of a dog in Figure 6, which our transfer learning model often mispredicts as a horse. Using our framework, we can demonstrate that this image is strongly negatively influenced by the source class "sorrel horse." Thus, our downstream model may be misusing a feature introduced by this class. Indeed, once we remove "sorrel horse" from the source dataset, our model predicts the correct label more frequently. (See Appendix E for more examples, as well as a quantitative analysis of this experiment.)

Figure 5: The CIFAR-10 images that were most positively (or negatively) influenced by the ImageNet classes "starfish" and "rapeseed." CIFAR-10 images that are highly influenced by the "starfish" class have similar shapes, while those influenced by "rapeseed" class have yellow-green colors.

4.3 Capability 3: Detecting data leakage and misleading source examples

Thus far, we have focused on how the *classes* in the source dataset influence the predictions of the transferred model on target examples. In this section, we extend our analysis to the *individual* datapoints of the source dataset. We do so by adapting our approach to measure the influence of each individual source datapoint on each target datapoint. Further details on how these influences are computed can be found in Appendix D.

Figure 7 displays the ImageNet training examples that have highly positive or negative influences on CIFAR-10 test examples. We find that the source images that are highly positively influential are often instances of *data leakage* between the source training set and the target test set. On the other hand, the ImageNet images that are highly negatively influential are typically mislabeled, misleading, or otherwise surprising. For example, the presence of the ImageNet image of a flying lawnmower hurts the performance on a CIFAR-10 image of a regular (but similarly shaped) airplane (see Figure 7).

5 Related Work

Transfer learning. Transfer learning is a technique commonly used in domains ranging from medical imaging [MGM18; KEB+21], language modeling [CK18], to object detection [RHG+15; DLH+16; GDD+14; CPK+17]. Therefore, there has been considerable interest in understanding the drivers of transfer learning's success. For example, by performing transfer learning on block-shuffled images, Neyshabur, Sedghi, and Zhang [NSZ20] demonstrate that at least some of the benefits of transfer learning come from low-level image statistics of source data. There is also an important line of work studying transfer learning by investigating the relationship between different properties of the source model and performance on the target task [KEB+21; KSL19; SIE+20; UKE+20].

The works that are the most relevant to ours are those which studied how modifying the source dataset can affect the downstream performance. For example, Kolesnikov et al. [KBZ+19] showed that pre-training

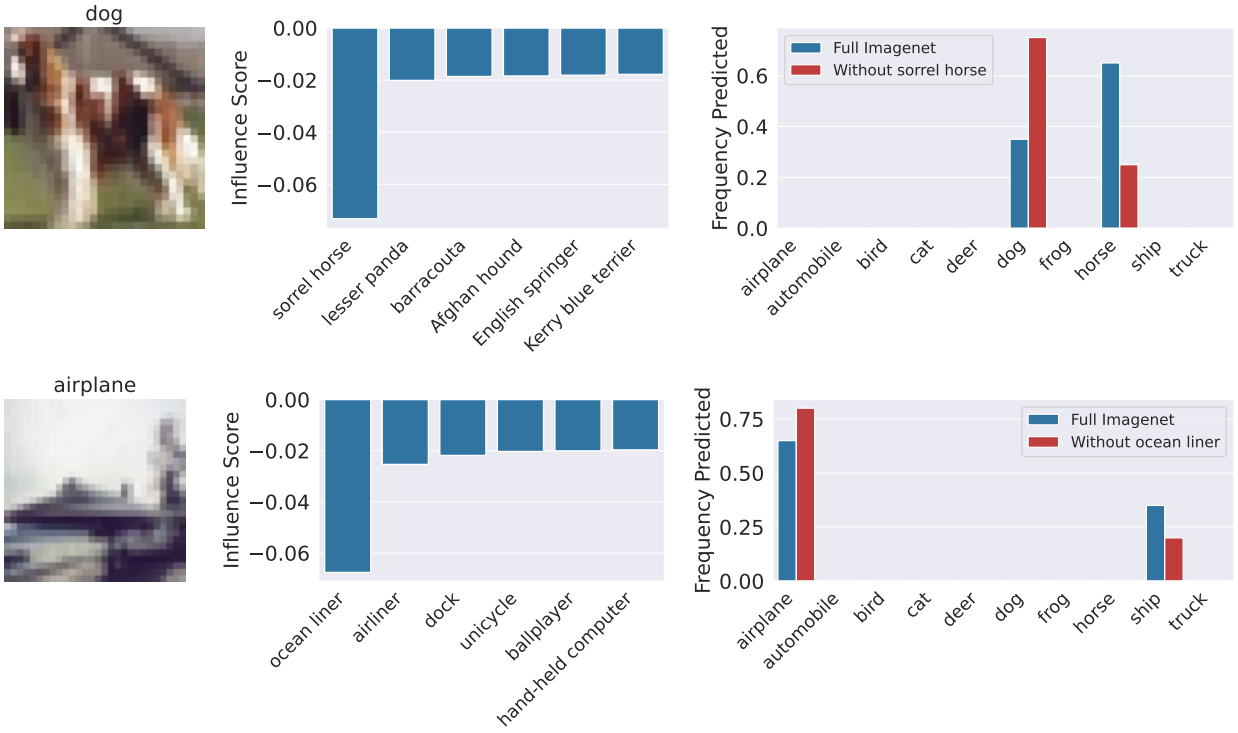


Figure 6: Pinpointing highly negatively influential source classes can help explain model mistakes. **Left:** For two CIFAR-10 images, we plot the most negatively influential source classes. **Right:** Over 20 runs, the fraction of times that our downstream model predicts each label for the given CIFAR-10 image. When the most negatively influential class is removed, the model predicts the correct label more frequently. More examples can be found in Appendix E.

with an enormous source dataset (approximately 300 million) of noisily labeled images can outperform pre-training with ImageNet. Azizpour et al. [ARS+15] and Huh, Agrawal, and Efros [HAE16] investigated the importance of the number of classes and the number of images per class in transfer learning. Finally, Ngiam et al. [NPV+18] demonstrated that more pre-training data does not always help, and transfer learning can be sensitive to the choice of pre-training data. They also presented a framework for reweighting the source datapoints in order to boost transfer learning performance.

Influence functions and datamodels. Influence functions are well-studied statistical tools that have been recently applied in machine learning settings [HRR+11; CW82; KL17]. For a given model, influence functions analyze the effect of a training input on the model's predictions by estimating the expected change in performance when this training input is added or removed. A recent line of work estimates this quantity more efficiently by training on different subsets of the training set [FZ20]. In a similar vein, Ghorbani and Zou [GZ19] proposed running a Monte Carlo search to estimate the effect of every training input via Shapley values. More recently, Ilyas et al. [IPE+22] proposed datamodeling framework as an alternative way to estimate the effect of a training input on the models' prediction. Datamodels are represented using parametric functions (typically, linear functions) that aim to map a subset of the training set to the model's output.

6 Conclusions

In this work, we presented a new framework for examining the impact of the source dataset in transfer learning. Specifically, our approach estimates the influence of a source class (or datapoint) that captures how including that class (or datapoint) in the source dataset impacts the downstream model's predictions.

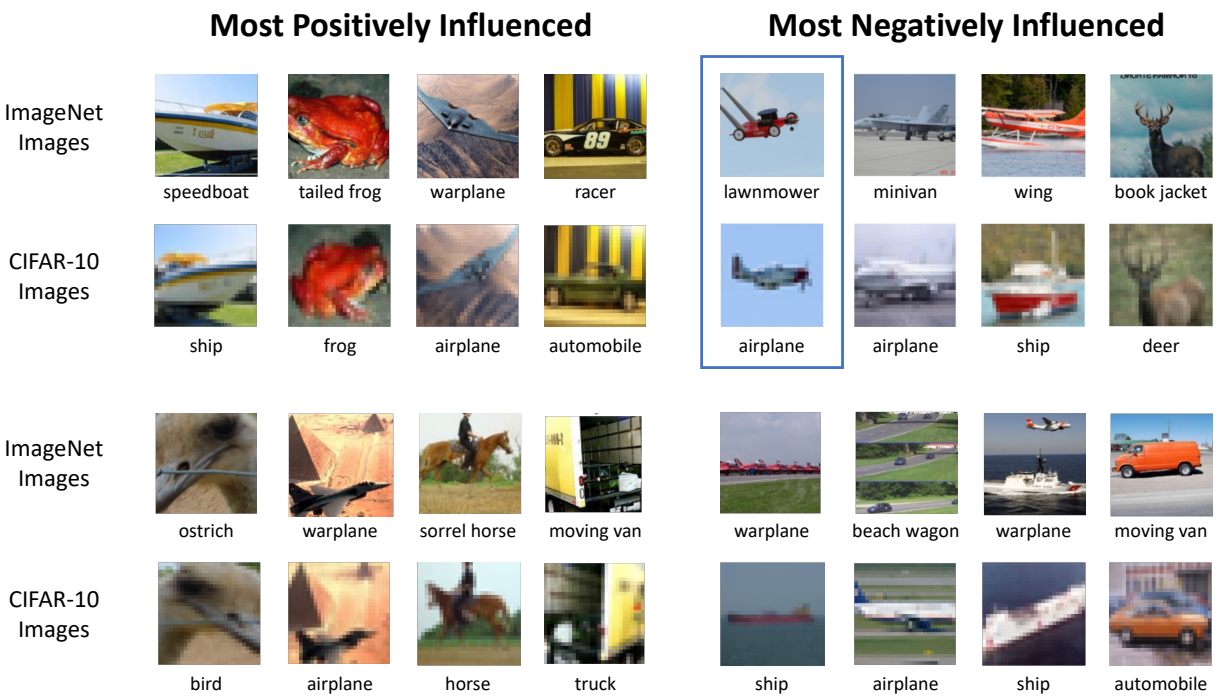


Figure 7: ImageNet training images with highest positive (**left**) or negative (**right**) example-wise (average) influences on CIFAR-10 test images. We find that ImageNet images that are highly positively influential often correspond to data leakage, while ImageNet images that are highly negatively influential are often either mislabeled, ambiguous, or otherwise misleading. For example, the presence of a flying lawnmower in the ImageNet dataset hurts the downstream performance on a similarly shaped airplane (boxed).

Leveraging these estimates, we demonstrate that we can improve the transfer learning performance on a range of downstream tasks by identifying and removing detrimental datapoints from the source dataset. Furthermore, our framework enables us to identify granular subpopulations in the target dataset by projecting fine-grained labels from the source dataset, better understand model failures on the downstream task and detect potential data-leakages from the source to the downstream dataset. We believe our framework provides a new perspective on transfer learning: one that enables us to perform a fine-grained analysis of the impact of the source dataset.

7 Acknowledgements

Work supported in part by the NSF grants CCF-1553428 and CNS-1815221, and Open Philanthropy. This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001120C0015.

Research was sponsored by the United States Air Force Research Laboratory and the United States Air Force Artificial Intelligence Accelerator and was accomplished under Cooperative Agreement Number FA8750-19-2-1000. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the United States Air Force or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

We thank the MIT Supercloud cluster [RKB+18] for providing computational resources that supported part of this work.

References

[ARS+15] Hossein Azizpour, Ali Sharif Razavian, Josephine Sullivan, Atsuto Maki, and Stefan Carlsson. "Factors of transferability for a generic convnet representation". In: *IEEE transactions on pattern analysis and machine intelligence* (2015).

[BGV14] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. "Food-101–mining discriminative components with random forests". In: *European conference on computer vision*. 2014.

[BLW+14] Thomas Berg, Jiongxin Liu, Seung Woo Lee, Michelle L Alexander, David W Jacobs, and Peter N Belhumeur. "Birdsnap: Large-scale fine-grained visual categorization of birds". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014.

[CFJ+18] Emmanuel Candes, Yingying Fan, Lucas Janson, and Jinchi Lv. "Panning for gold:'model-X'knockoffs for high dimensional controlled variable selection". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 80.3 (2018), pp. 551–577.

[CK18] Alexis Conneau and Douwe Kiela. "Senteval: An evaluation toolkit for universal sentence representations". In: *Language Resources and Evaluation Conference (LREC)* (2018).

[CPK+17] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs". In: *IEEE transactions on pattern analysis and machine intelligence* (2017).

[CW82] R Dennis Cook and Sanford Weisberg. *Residuals and influence in regression*. New York: Chapman and Hall, 1982.

[DDS+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A large-scale hierarchical image database". In: *Computer Vision and Pattern Recognition (CVPR)*. 2009.

[DGS19] Shuyang Du, Haoli Guo, and Andrew Simpson. "Self-driving car steering angle prediction based on image recognition". In: *arXiv preprint arXiv:1912.05440* (2019).

[DLH+16] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. "R-fcn: Object detection via region-based fully convolutional networks". In: *Advances in neural information processing systems (NeurIPS)*. 2016.

[FFP04] Li Fei-Fei, Rob Fergus, and Pietro Perona. "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories". In: *2004 conference on computer vision and pattern recognition workshop*. IEEE. 2004, pp. 178–178.

[FZ20] Vitaly Feldman and Chiyuan Zhang. "What Neural Networks Memorize and Why: Discovering the Long Tail via Influence Estimation". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 33. 2020, pp. 2881–2891.

[GDD+14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *computer vision and pattern recognition (CVPR)*. 2014, pp. 580–587.

[GHP07] Gregory Griffin, Alex Holub, and Pietro Perona. "Caltech-256 object category dataset". In: (2007).

[GZ19] Amirata Ghorbani and James Zou. "Data shapley: Equitable valuation of data for machine learning". In: *International Conference on Machine Learning (ICML)*. 2019.

[HAE16] Minyoung Huh, Pulkit Agrawal, and Alexei A Efros. "What makes ImageNet good for transfer learning?" In: *arXiv preprint arXiv:1608.08614* (2016).

[HRR+11] Frank R Hampel, Elvezio M Ronchetti, Peter J Rousseeuw, and Werner A Stahel. *Robust statistics: the approach based on influence functions*. Vol. 196. John Wiley & Sons, 2011.

[IPE+22] Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. "Datamodels: Predicting Predictions from Training Data". In: *International Conference on Machine Learning (ICML)*. 2022.

[JLH+19] Yunhun Jang, Hankook Lee, Sung Ju Hwang, and Jinwoo Shin. "Learning what and where to transfer". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 3030–3039.

[KBZ+19] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. "Big Transfer (BiT): General Visual Representation Learning". In: *arXiv preprint arXiv:1912.11370* (2019).

[KDI+22] Bojan Karlaš, David Dao, Matteo Interlandi, Bo Li, Sebastian Schelter, Wentao Wu, and Ce Zhang. "Data Debugging with Shapley Importance over End-to-End Machine Learning Pipelines". In: *arXiv preprint arXiv:2204.11131* (2022).

[KDS+13] Jonathan Krause, Jia Deng, Michael Stark, and Li Fei-Fei. "Collecting a large-scale dataset of fine-grained cars". In: (2013).

[KEB+21] Alexander Ke, William Ellsworth, Oishi Banerjee, Andrew Y Ng, and Pranav Rajpurkar. "CheX-transfer: performance and parameter efficiency of ImageNet models for chest X-Ray interpretation". In: *Proceedings of the Conference on Health, Inference, and Learning*. 2021, pp. 116–124.

[KL17] Pang Wei Koh and Percy Liang. "Understanding Black-box Predictions via Influence Functions". In: *International Conference on Machine Learning*. 2017.

[KP17] Jiman Kim and Chanjong Park. "End-to-end ego lane estimation based on sequential transfer learning for self-driving cars". In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 2017, pp. 30–38.

[Kri09] Alex Krizhevsky. "Learning Multiple Layers of Features from Tiny Images". In: *Technical report*. 2009.

[KRJ+22] Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. "Fine-Tuning can Distort Pretrained Features and Underperform Out-of-Distribution". In: *arXiv preprint arXiv:2202.10054* (2022).

[KSL19] Simon Kornblith, Jonathon Shlens, and Quoc V Le. "Do better imagenet models transfer better?" In: *computer vision and pattern recognition (CVPR)*. 2019.

[KZ21] Yongchan Kwon and James Zou. "Beta Shapley: a Unified and Noise-reduced Data Valuation Framework for Machine Learning". In: *arXiv preprint arXiv:2110.14049* (2021).

[LIE+22] Guillaume Leclerc, Andrew Ilyas, Logan Engstrom, Sung Min Park, Hadi Salman, and Aleksander Madry. *ffcv*. https://github.com/libffcv/ffcv/. 2022.

[MGM18] Romain Mormont, Pierre Geurts, and Raphaël Marée. "Comparison of deep transfer learning strategies for digital pathology". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018.

[Mil95] George A Miller. "WordNet: a lexical database for English". In: *Communications of the ACM* (1995).

[MRK+13] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. "Fine-grained visual classification of aircraft". In: *arXiv preprint arXiv:1306.5151* (2013).

[NPV+18] Jiquan Ngiam, Daiyi Peng, Vijay Vasudevan, Simon Kornblith, Quoc V Le, and Ruoming Pang. "Domain adaptive transfer learning with specialist models". In: *arXiv preprint arXiv:1811.07056* (2018).

[NSZ20] Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. "What is being transferred in transfer learning?" In: *Advances in neural information processing systems* 33 (2020), pp. 512–523.

[NZ08] Maria-Elena Nilsback and Andrew Zisserman. "Automated flower classification over a large number of classes". In: *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*. 2008.

[PVZ+12] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. "Cats and dogs". In: *2012 IEEE conference on computer vision and pattern recognition*. IEEE. 2012, pp. 3498–3505.

[RHG+15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems (NeurIPS)*. 2015.

[RKB+18] Albert Reuther, Jeremy Kepner, Chansup Byun, Siddharth Samsi, William Arcand, David Bestor, Bill Bergeron, Vijay Gadepally, Michael Houle, Matthew Hubbell, Michael Jones, Anna Klein, Lauren Milechin, Julia Mullen, Andrew Prout, Antonio Rosa, Charles Yee, and Peter Michaleas. "Interactive supercomputing on 40,000 cores for machine learning and data analysis". In: *2018 IEEE High Performance extreme Computing Conference (HPEC)*. IEEE. 2018, pp. 1–6.

[SIE+20] Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. "Do Adversarially Robust ImageNet Models Transfer Better?" In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2020.

[UKE+20] Francisco Utrera, Evan Kravitz, N. Benjamin Erichson, Rajiv Khanna, and Michael W. Mahoney. "Adversarially-Trained Deep Nets Transfer Better". In: *ArXiv preprint arXiv:2007.05869*. 2020.

[WAL19] Sherrie Wang, George Azzari, and David B Lobell. "Crop type mapping without field-level labels: Random forest transfer and unsupervised clustering techniques". In: *Remote sensing of environment* 222 (2019), pp. 303–317.

[WPL+17] Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M Summers. "Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases". In: *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*. 2017.

[XHE+10] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. "Sun database: Large-scale scene recognition from abbey to zoo". In: *Computer Vision and Pattern Recognition (CVPR)*. 2010.

[XJB+16] Michael Xie, Neal Jean, Marshall Burke, David Lobell, and Stefano Ermon. "Transfer learning from deep features for remote sensing and poverty mapping". In: *Thirtieth AAAI Conference on Artificial Intelligence*. 2016.

A Experimental Setup

A.1 ImageNet Models

In this paper, we train a large number of models on various subsets of ImageNet in order to estimate the influence of each class of ImageNet on the model's transfer performance for multiple downstream tasks. We focus on the ResNet-18 architecture from PyTorch's official implementation found here https://pytorch.org/vision/stable/models.html[5].

Training details. We fix the training procedure for all of our models. Specifically, we train our models from scratch using SGD to minimize the standard cross-entropy multi-class classification loss. We use a batch size of 1024, momentum of 0.9, and weight decay of 5×10^{-4}. The models are trained for 16 epochs using a Cyclic learning rate schedule with an initial learning rate of 0.5 and learning rate peak epoch of 2. We use standard data-augmentation: *RandomResizedCrop* and *RandomHorizontalFlip* during training, and *RandomResizedCrop* during testing. Our implementation and configuration files are available in our code https://github.com/MadryLab/data-transfer.

A.2 ImageNet transfer to classification datasets

Dataset	Classes	Train Size	Test Size
Birdsnap [BLW+14]	500	32,677	8,171
Caltech-101 [FFP04]	101	3,030	5,647
Caltech-256 [GHP07]	257	15,420	15,187
CIFAR-10 [Kri09]	10	50,000	10,000
CIFAR-100 [Kri09]	100	50,000	10,000
FGVC Aircraft [MRK+13]	100	6,667	3,333
Food-101 [BGV14]	101	75,750	25,250
Oxford 102 Flowers [NZ08]	102	2,040	6,149
Oxford-IIIT Pets [PVZ+12]	37	3,680	3,669
SUN397 [XHE+10]	397	19,850	19,850
Stanford Cars [KDS+13]	196	8,144	8,041

Table 1: Image classification datasets used in this paper.

Datasets. We consider the transfer image classification tasks that are used in [SIE+20; KSL19], which vary in size and number of classes. See Table 1 for the details of these datasets. We consider two transfer learning settings for each dataset: *fixed-feature* and *full-network* transfer learning.

Fixed-feature transfer. For this setting, we *freeze* the layers of the ImageNet source model[6], except for the last layer, which we replace with a random initialized linear layer whose output matches the number of classes in the transfer dataset. We now train only this new layer for using SGD, with a batch size of 1024 using cyclic learning rate. For more details and hyperparameter for each dataset, please see config files in our code https://github.com/MadryLab/data-transfer.

Full-network transfer. For this setting, we *do not freeze* any of the layers of the ImageNet source model, and all the model weights are updated. We follow the exact same hyperparameters as the fixed-feature setting.

A.3 Compute and training time.

We leveraged the FFCV data-loading library for fast training of the ImageNet models [LIE+22][7]. Our experiments were run on two GPU clusters: an A100 and a V100 cluster.

[5] Our framework is agnostic to the choice of the model's architecture.

[6] For all of our experiments, we do not freeze the batch norm statistics. We only freeze the weights of the model, similar to Salman et al. [SIE+20].

[7] Using FFCV, we can train a model on the ImageNet dataset in around 1 hour, and reach ∼63% accuracy

Training ImageNet models and influence calculation. We trained 7,540 ImageNet models on random subsets of ImageNet, each containing half of ImageNet classes. On a single V100, training a single model takes around 30 minutes. After training these ImageNet models, we compute the influence of each class as outlined in Algorithm 1. Computing the influences is fast, and takes few seconds on a single V100 GPU.

A.4 Handpicked baseline details

In our counterfactual experiments in Section 3, we automatically selected, via our framework, the most influential subsets of ImageNet classes for various downstream tasks. We then removed the classes that are detrimental to the transfer performance, and measured the transfer accuracy improvement after removing these classes. The results are summarized in Table 2b.

What happens if we hand-pick the source dataset classes that are relevant to the target dataset? Indeed, Ngiam et al. [NPV+18] found that hand-picking the source dataset classes can sometimes boost transfer performance. We validate this approach for our setting using the WordNet hierarchy [Mil95]. Specifically, for each class from the target task, we look up all the ImageNet classes that are either children or parents of this target class. The set of all such ImageNet classes are used as the handpicked most influential classes. Following this manual selection, we train an ImageNet model on these classes, then apply transfer learning to get the baseline performance that we report in Table 2b.

A.5 Error and Convergence Analysis

We compute our class influence values using 7,540 source models, each of which were trained using 500 randomly chosen ImageNet classes. How many models do we actually need to compute our transfer influences? In order to analyze the convergence of the transfer influences, we track the standard deviation of the influence values after bootstrap resampling.

We consider the ImageNet $\rightarrow$ CIFAR-10 transfer setting with fixed-feature fine-tuning. Given N models, we randomly sample, with replacement, N models to recompute our transfer influences. Specifically, we evaluate the overall transfer influences (i.e., the influence value of each ImageNet class averaged over all target examples). We perform this resampling 500 times, and measure the standard deviation of the computed overall transfer influence value for each class over these 500 resamples.

Below, we plot this standard deviation (averaged over the 1000 classes) for various number of models N. We find that the standard deviation goes down as more models are used, indicating that our estimate of the influence values has less variance. This metric roughly plateaus by the time we are using 7000 models.

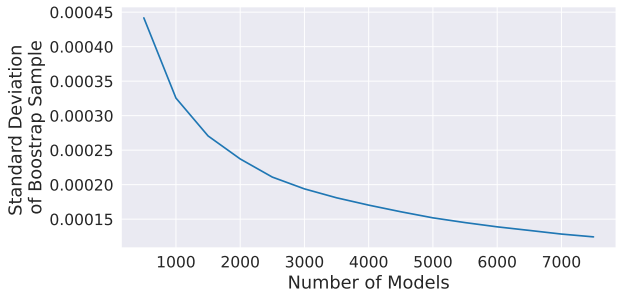


Figure 8: Standard deviation of the overall influence values (averaged over classes) after bootstrap resampling for various numbers of models.

B Variants of Computing Influences

B.1 Variations of targets for computing transfer influences

In our paper, we used the softmax output of the groundtruth class as the target for our influence calculation. What happens if we use a different target? We compare using the following types of targets.

- Softmax Logits: the softmax output of the groundtruth class
- Is Correct: the binary value of whether the image was predicted correctly
- Raw Margins: the difference in raw output between the correct class and the most confidently predicted incorrect class
- Softmax Margins: the same as raw margins, but use the output after softmax

In Figure 9, we replicate the counterfactual experiment from the main paper in Figure 2b using these different targets. Specifically (over 2 runs), we rank the overall influence of the ImageNet classes on CIFAR-10 for fixed-feature transfer. We then remove the classes in order most most or least influence.

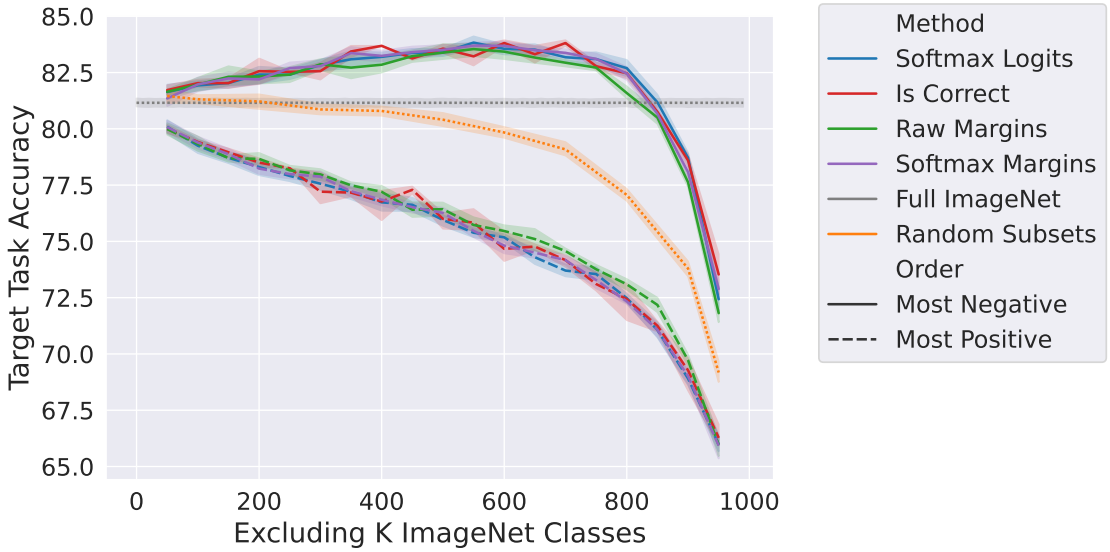


Figure 9: Target task accuracies after removing the most positively or negatively influential ImageNet classes from the source dataset with various influence targets.

We find that our method is not particularly sensitive to the individual target used. We found that using the softmax logits provided the highest benefit when removing negative influencers, and thus used that target for the reset of our experiments.

Datamodels vs. Influences. Datamodels [IPE+22] is another method that, similar to influences, seeks to compute the importance of a training point on a test set prediction. Specifically, instead of computing the difference in the expected accuracy of the model when a training point is removed, the method fits a linear model that, given a binary vector that denotes the composition of the training dataset, predicts the raw margin (i.e., the difference in raw output between the correct class and the most confidently predicted incorrect class). The importance of each training point is then the coefficient of the linear model for that particular example.

We adapt this method to our framework by training a linear model with ridge regression to predict the softmax output of the transfer model on the target images given a binary vector that denotes which source classes were included in the source dataset. However, we find that datamodels were more effective for computing example-based values (see Appendix D).

In Figure 10, we compare using influence values (as described in the main paper) to using these adapted datamodels. Specifically (over 5 runs), we rank the overall importance of the ImageNet classes on CIFAR-10 for fixed-feature transfer using influences or datamodels. We then remove the classes in order most most or least influence. We find that our framework is not sensitive to the choice of datamodels or influences. However, influences performed marginally better in this counterfactual experiment, so we used influences for all other experiments in this paper.

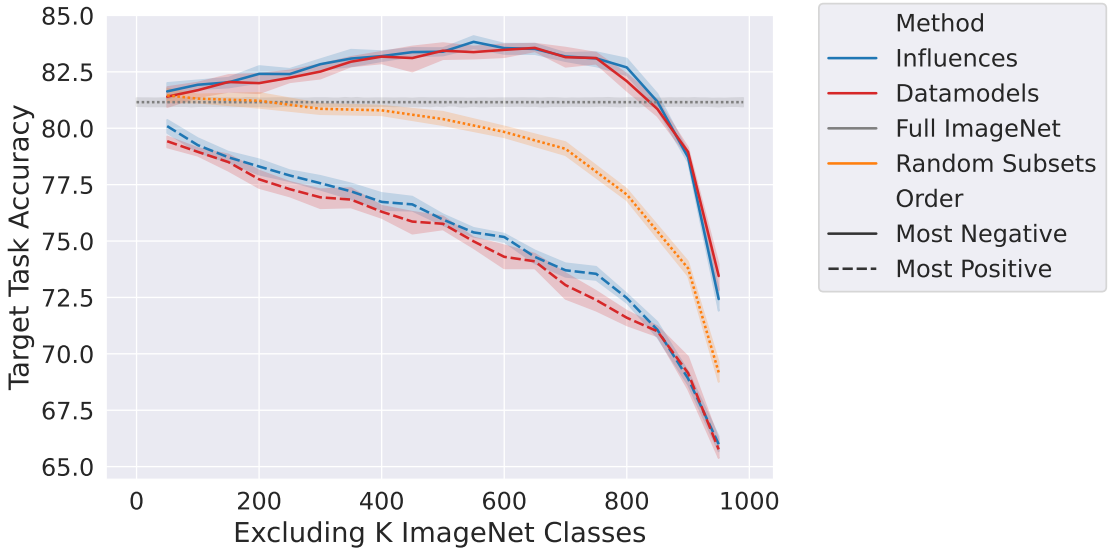


Figure 10: Target task accuracies after removing the most positively or negatively influential ImageNet classes from the source dataset using datamodels or influences.

C Full Counterfactual Experiment

In this section, we display the full results for the counterfactual experiment in the main paper (Figure 2b). Specifically, for each target task, we display the target task accuracies after removing the most positive (top) and negative (bottom) influencers from the dataset over 10 runs. We find that our results hold across datasets.

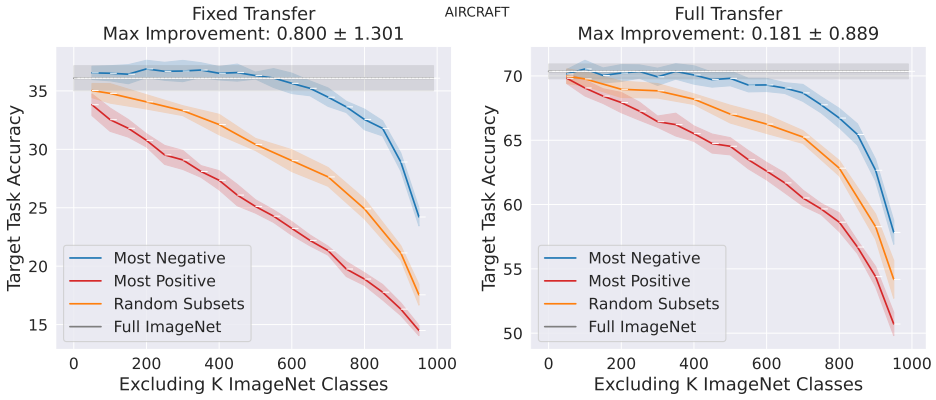


Figure 11: AIRCRAFT

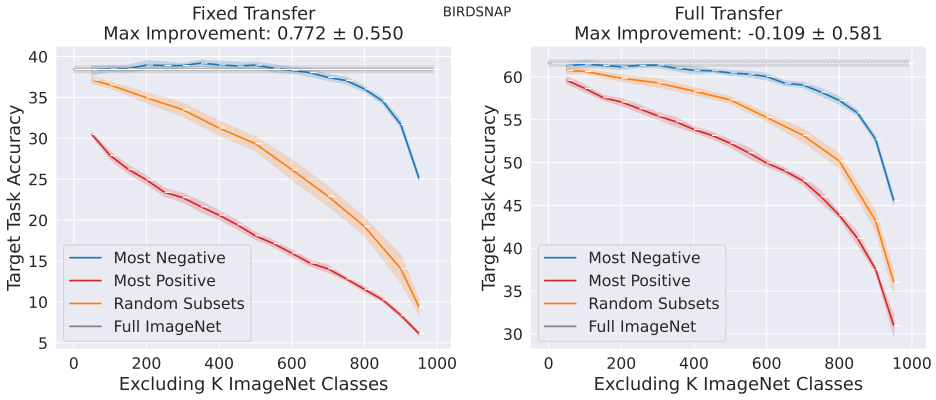


Figure 12: BIRDSNAP

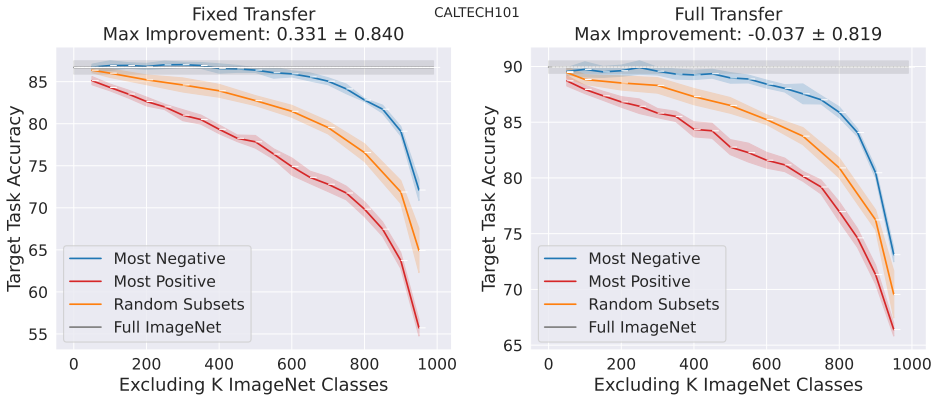


Figure 13: CALTECH101

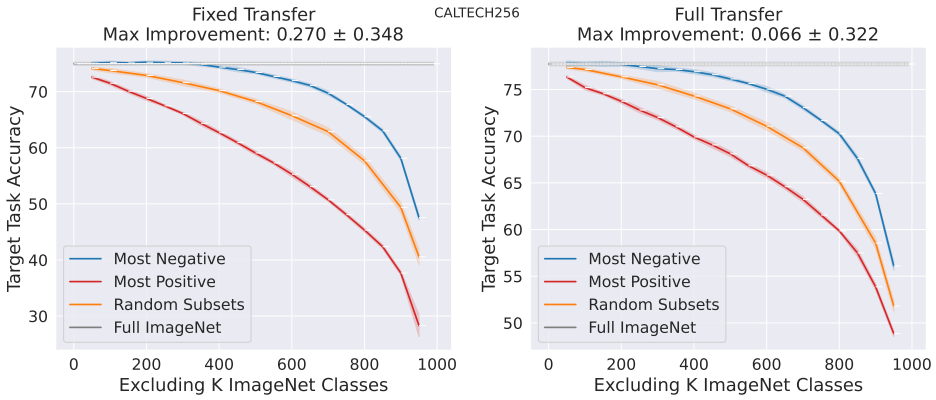


Figure 14: CALTECH256

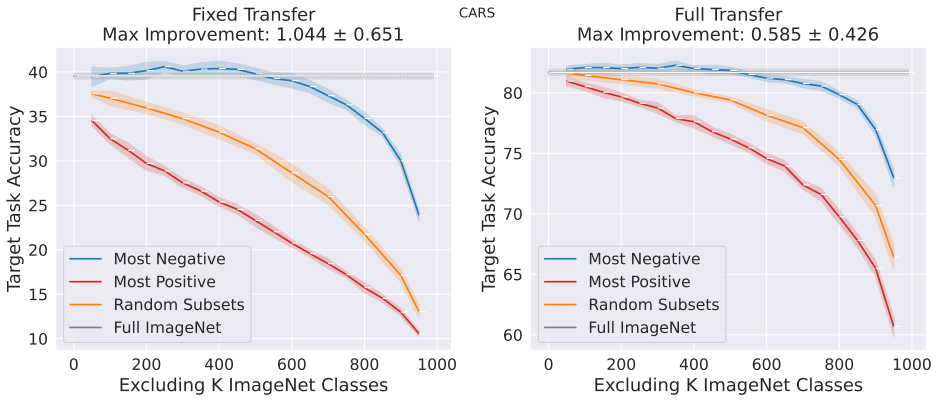


Figure 15: CARS

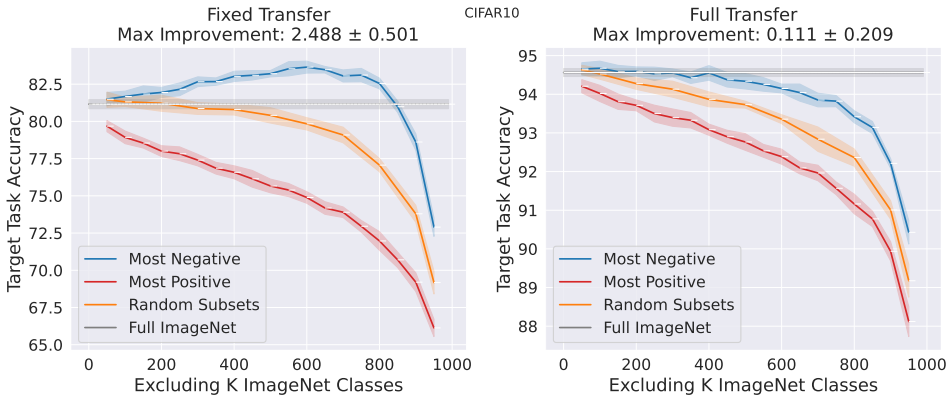


Figure 16: CIFAR10

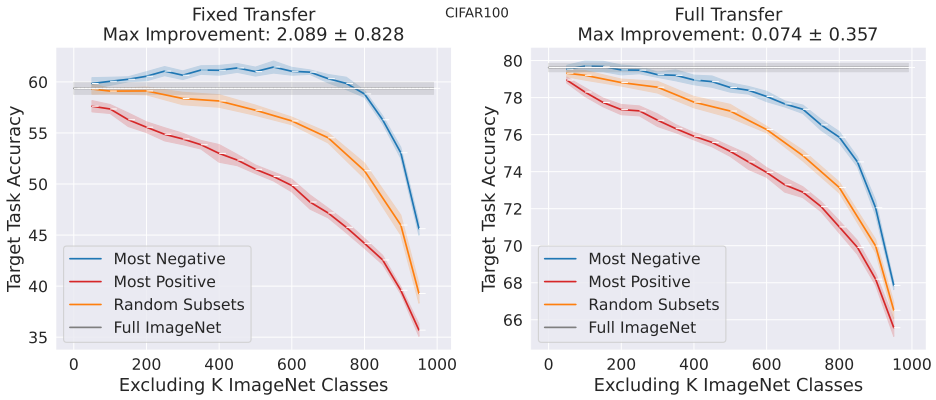


Figure 17: CIFAR100

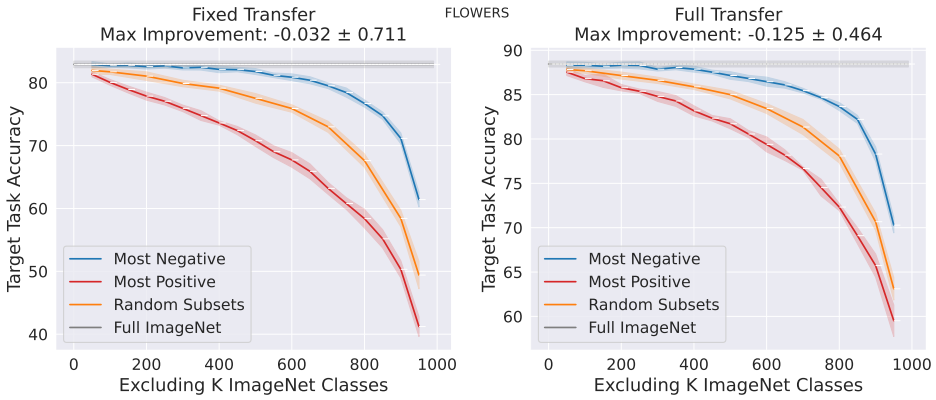


Figure 18: FLOWERS

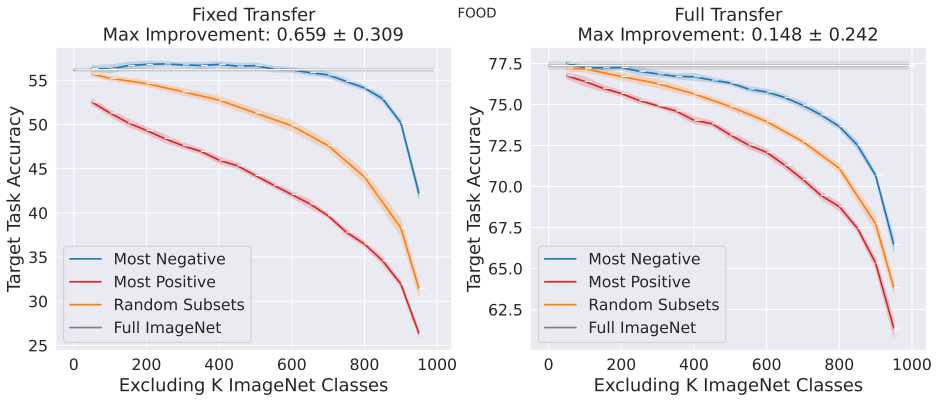


Figure 19: FOOD

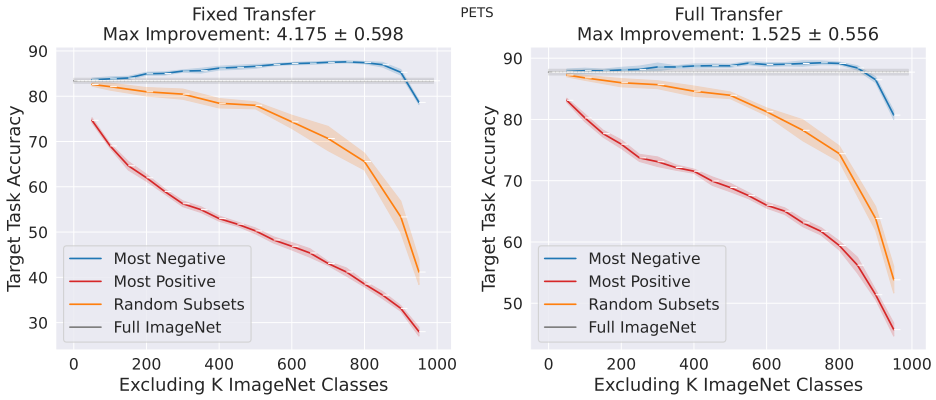


Figure 20: PETS

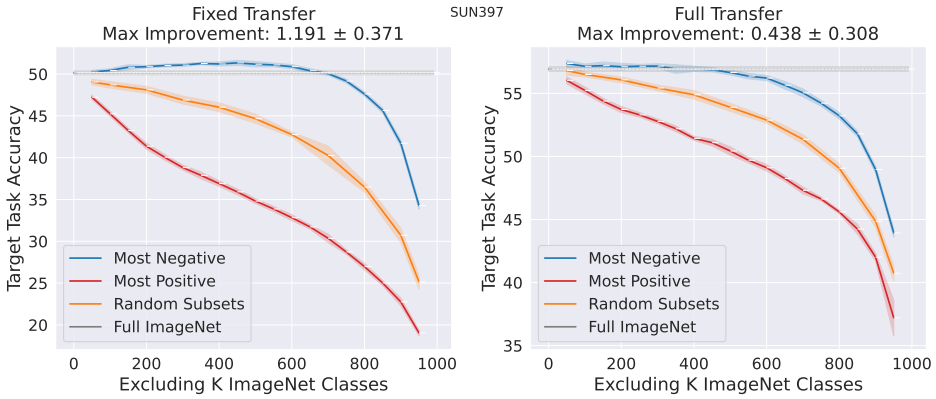


Figure 21: SUN397

D Adapting our Framework to Compute the Effect of Every Source Datapoint on Transfer Learning

We have presented in the main paper how to compute the influences of every class in the source dataset on the predictions of the model on the target dataset. In that setup, we demonstrated multiple capabilities of our framework, such as improving overall transfer performance, detecting particular subpopulations in the target dataset, etc. Given the wide range of capabilities class-based influences provide, one natural question that arises: Can we compute the influence of every source datapoint on the predictions of the model on the target dataset? Furthermore, what do these influences tell us about the transfer learning process?

Mathematically, the computation of example-based influences (i.e., the influence of every source datapoint) is very similar to the computation of class-based influences. Specifically, to compute example-based influences, we start by training a large number of models on different subsets of the source datapoints (as opposed to source classes for class-based influences). Next, we estimate the influence value of a source datapoint s on a target example t as the expected difference in the transfer model's performance on example t when datapoint s was either included or excluded from the source dataset:

$$\text{Infl}[s \to t] = \mathbb{E}_S \left[f(t; S) \mid s \in S \right] - \mathbb{E}_S \left[f(t; S) \mid s \notin S \right] \tag{2}$$

where $f(t; S)$ is the softmax output of a model trained on a subset S of the source dataset. Similar to class-based influences, a positive (resp. negative) influence value indicates that including the source datapoint s improves (resp. hurts) the model's performance on the target example t.

While example-based influences provide some insights about the transfer process, we found that—in this regime—datamodels [IPE+22] provide cleaner results and better insights. Generally, influences and datamodels measure similar properties: the effect of the source datapoints on the target datapoints. For a particular target datapoint t, we measure the effect of every source datapoint s with datamodels by solving a regression problem. Specifically, we train a large number of models on different subsets of the source dataset. For every model f_i, we record 1) a binary mask $\mathbb{1}_{\mathcal{S}_i}$ that indicates which source datapoints were included in the subset $\mathcal{S}_i$ of the source dataset, and 2) the transfer performance $f_i(t; \mathcal{S}_i)$ of the model f_i on the target datapoint t after fine-tuning on the target dataset. Following the training and the fine-tuning stages, we fit a linear model g_w that predicts the transfer performance $f(t; \mathcal{S})$ from a random subset $\mathcal{S}$ of the source dataset as follows: $f(t; \mathcal{S}) \approx g_w(\mathbb{1}_{\mathcal{S}}) = w^\top \mathbb{1}_{\mathcal{S}}$. Given this framework, $w = (w_1, w_2, \ldots, w_L)$ measures the effect of every source datapoint s on the target datapoint t[8]. We present the overall procedure in Algorithm 2.

Algorithm 2 Example-based datamodels estimation for transfer learning.

Require: source dataset $\mathcal{S} = \cup_{l=1}^{L} s_l$ (with L datapoints), a target dataset $\mathcal{T} = (t_1, t_2, \cdots, t_n)$, training algorithm $\mathcal{A}$, subset ratio α, number of models m

Sample m random subsets $S_1, S_2, \cdots, S_m \subset \mathcal{S}$ of size $\alpha \cdot |\mathcal{S}|$:
for $i \in 1$ to m **do**
    Train model f_i by running algorithm $\mathcal{A}$ on S_i
end for
Fine-tune f_i on the training target dataset
for $j \in 1$ to n **do**
    Collect datamodels training set $\mathcal{D}_j = \{(\mathbb{1}_{\mathcal{S}_i}, f_i(t_j, \mathcal{S}_i))\}_{i=1}^{m}$
    Compute w_j by fitting LASSO on $\mathcal{D}_j$
end for
return $w_j \ \forall \ j \in [n]$

[8] To estimate the datamodels, we train 71,828 models on different subsets of the source dataset.

E Omitted Results

E.1 Per-class influencers

We display for the ImageNet → CIFAR-10 the top (most positive) and bottom (most negative) influencing classes for each CIFAR-10 class. This is the equivalent to the plot in Figure 3 in the main paper.

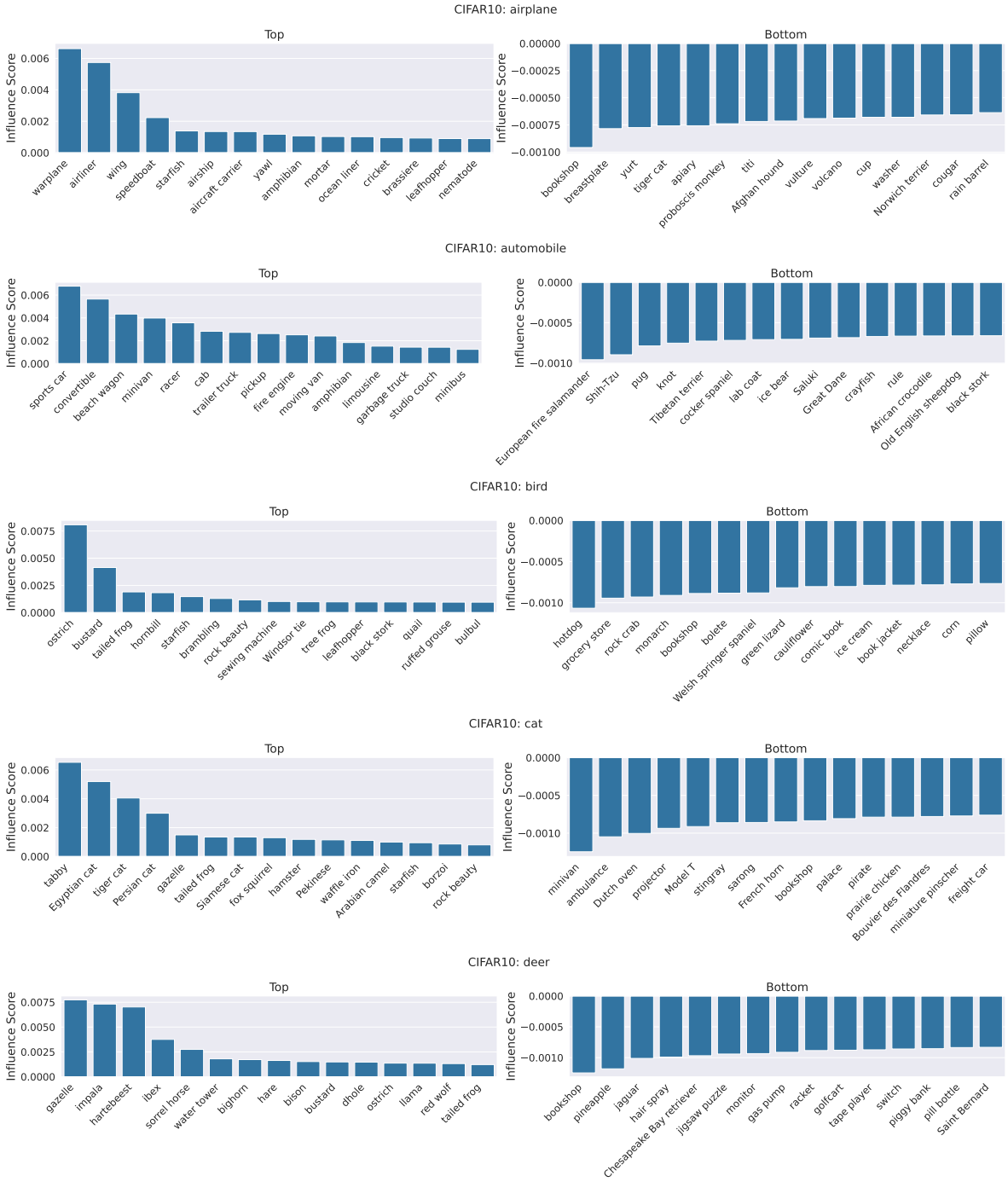


Figure 22: Top and bottom influencing ImageNet classes for all CIFAR-10 classes.

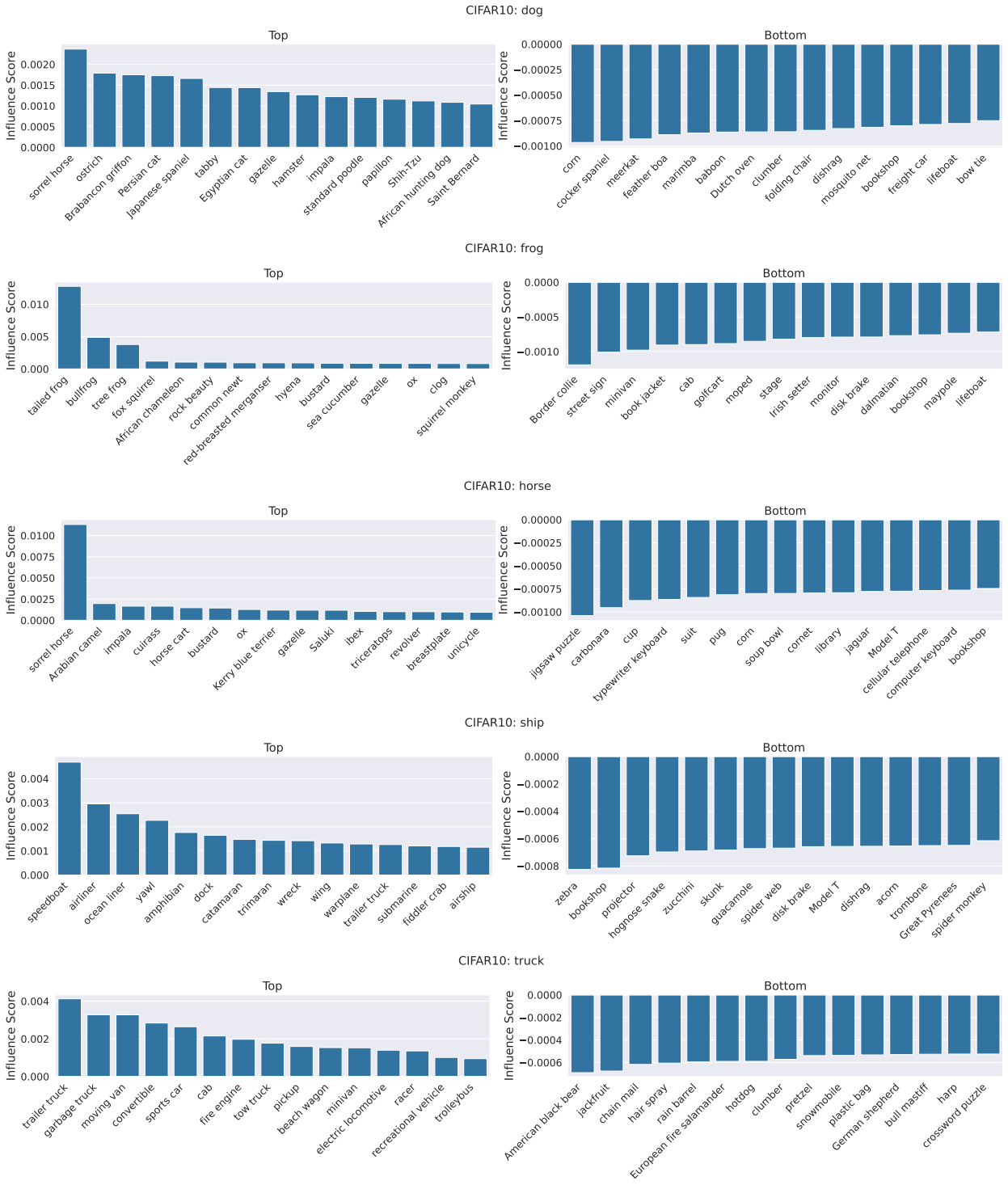
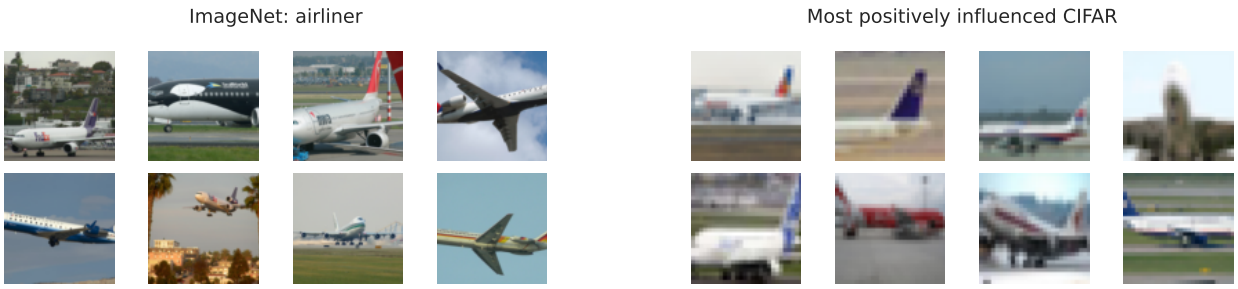


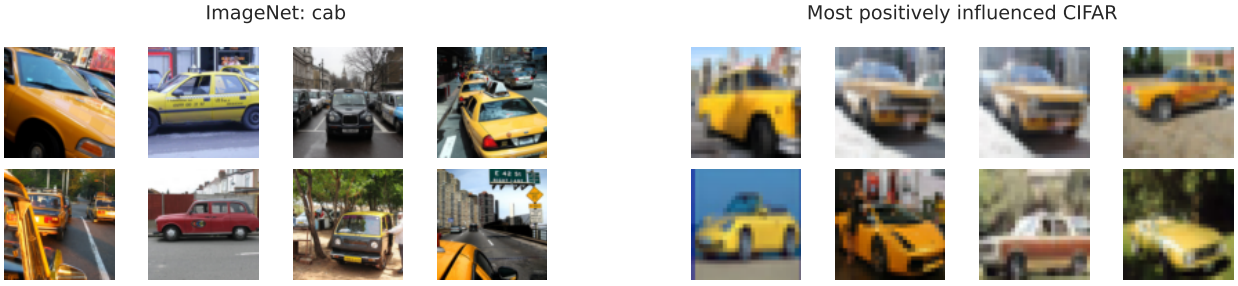
Figure 23: Top and bottom influencing ImageNet classes for all CIFAR-10 classes.

E.2 More examples of extracted subpopulations from the target dataset

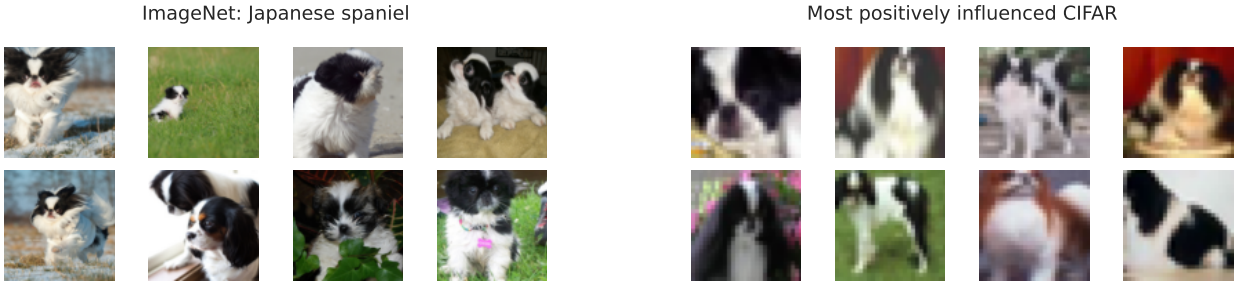
Here, we depict more examples of extracting subpopulations from the target dataset (as in Figure 4 of the main paper).



(a) Airliner



(b) Cab



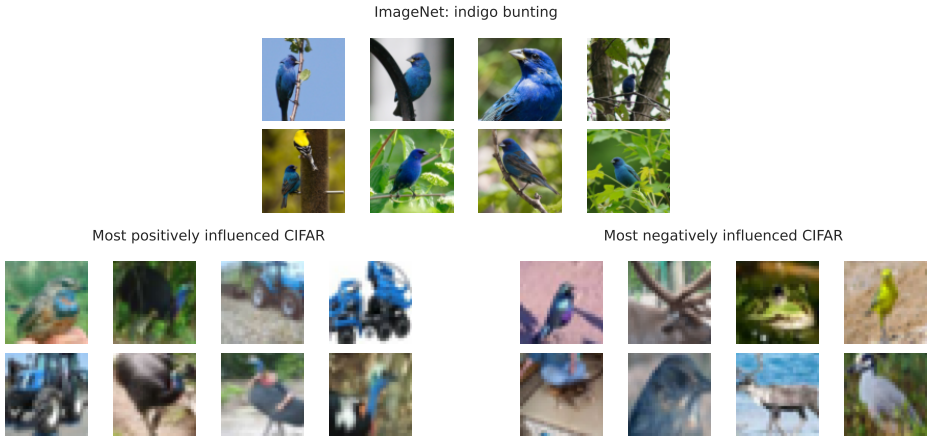
(c) Japanese Spaniel

(d) Ocean Liner

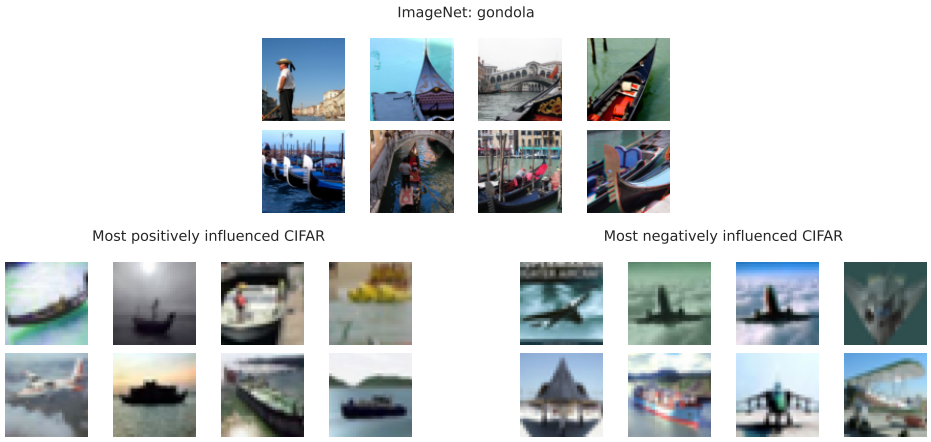
Figure 24: For each ImageNet class, we show the CIFAR-10 examples which were most positively influenced by that ImageNet class.

E.3 More examples of transfer of shape and texture feature

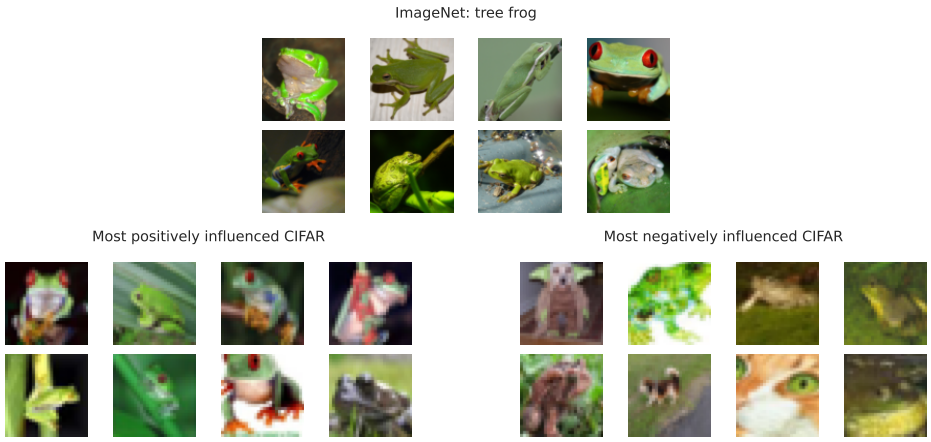
We depict more examples of ImageNet influencers which transfer shape or texture features (as in Figure 5).



(a) Indigo Bunting



(b) Gondola



(c) Tree Frog

Figure 25: For each ImageNet class, we show the CIFAR-10 examples which were most highly influenced by that ImageNet class.

E.4 More examples of debugging mistakes of transfer model using influencers

We display more examples of how our influences can be used to debug the mistakes of the transfer model, as presented in Figure 6 in the main paper. We find that, in most cases (Figure 26a, 26b, 26c), removing the top negative influencer improves the model's performance on the particular image. There are a few examples where removing the top negative influencer hurts the model's performance on the image (Figure 26d).

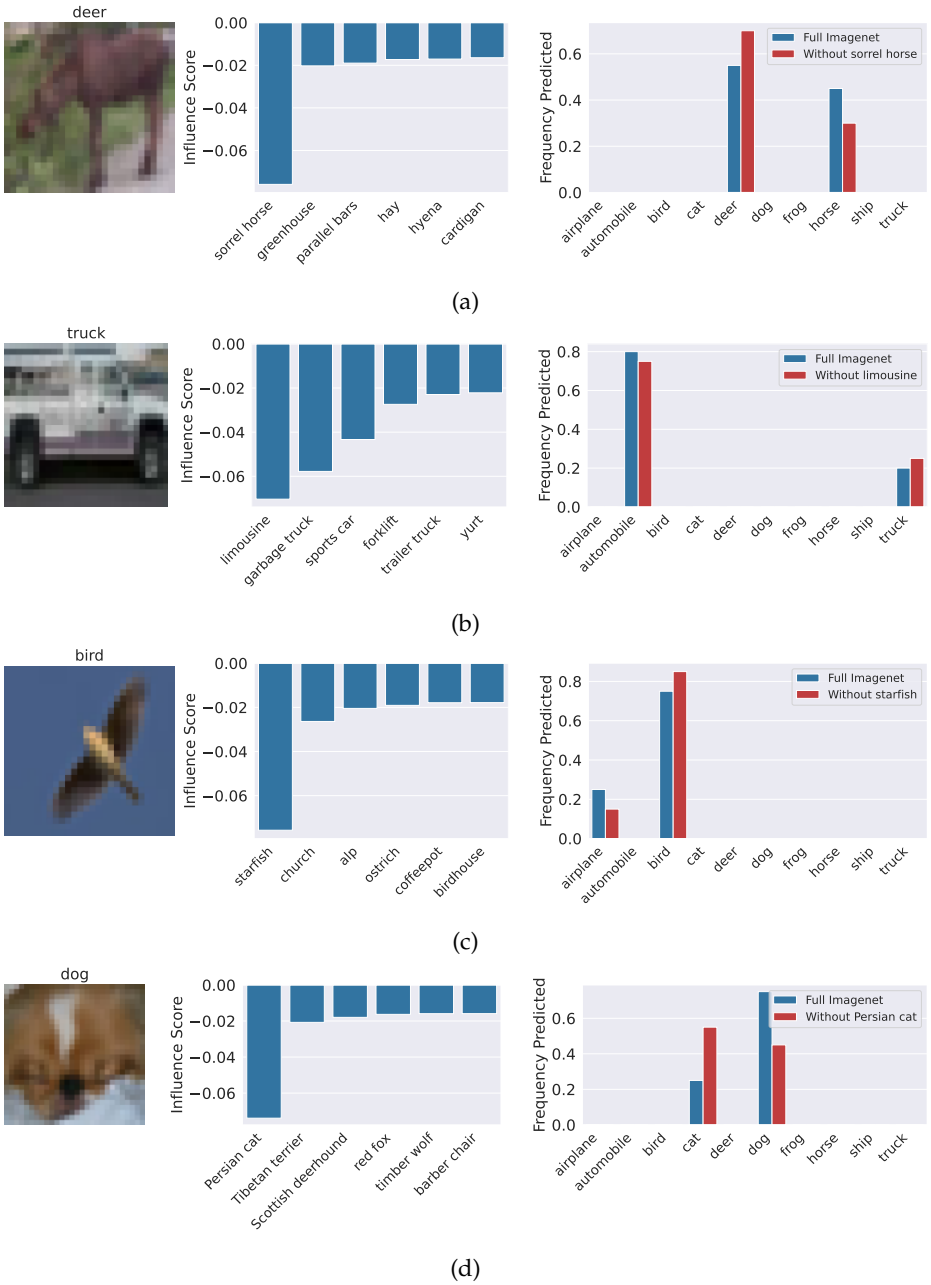


Figure 26: More examples of debugging transfer mistakes through our framework (c.f. Figure 6 in the main paper). For each CIFAR-10 image (**left**), we plot their most negative influencers (**middle**). On the **right**, we plot for each image the fraction (over 20 runs) of times that our transfer model predicts each class with and without the most negative influencer. While in most cases (a, b, c) we find that removing the most negative influencer helps the model predict the image correctly, in some cases removing the negative influencer does not outweigh the impact of removing images from the source dataset (d).

Quantitative analysis. How often does removing the most negative influencer actually improve the prediction on an image? For each of the following 14 classes, we run 20 runs of the ImageNet $\rightarrow$ CIFAR-10 fixed transfer pipeline while excluding that single class from the source dataset: ["sorrel horse", "limousine", "minivan", "fireboat", "ocean liner", "Arabian camel", "Persian cat", "ostrich", "gondola", "pool table", "starfish", "rapeseed", "tailed frog", "trailer truck"]. We compare against running the pipeline with 20 runs of the entire ImageNet dataset. Then, we look at individual CIFAR-10 images which were highly negatively influenced by one of these ImageNet classes, and check whether the images were predicted correctly more or less often when the top negative influencers were removed from the source dataset.

Of the 30 most negatively influenced ImageNet class/CIFAR-10 image pairs, 26 of them had the most negative ImageNet influencer in the above classes. Of those, 61.5% were predicted correctly more often when the negatively influential ImageNet class was removed, 34.6% were predicted incorrectly more often, and 3.9% were predicted correctly the same number of times.

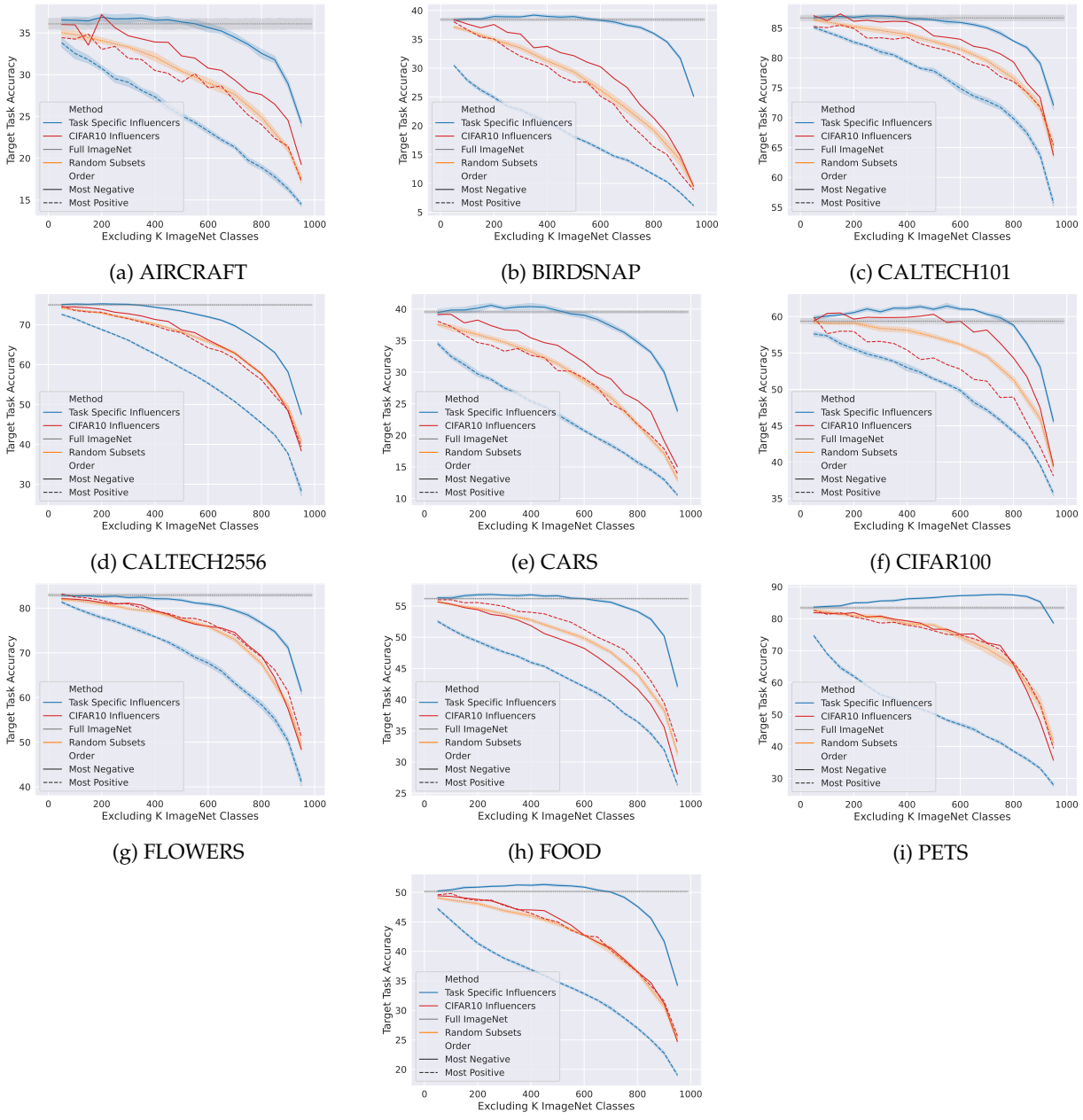
We then examine the top 8 most influenced CIFAR-10 images for each of the above 14 ImageNet classes. Of those 112 images, 53% were predicted correctly more often when the image was removed, 34% were predicted incorrectly more often, and 14% were predicted correctly the same number of times.

We thus find that, for the most influenced CIFAR-10 images, removing the top negative influencer usually improves that specific image's prediction (even though we are removing training data from the source dataset).

E.5 Do Influences Transfer?

E.5.1 Transfer across datasets

In this section, we seek to understand how much task-specific information is in the transfer influences that we compute. To do so, we use the transfer influences computed for CIFAR-10 in order to perform the counterfactual experiments for other datasets. We find that while using the CIFAR-10 influence values for other target datasets is more meaningful than random, they do not provide the same boost in performance when removing bottom influencers as using the task-specific influences. We thus conclude that the influence values computed by our framework are relatively task-specific.



(a) AIRCRAFT (b) BIRDSNAP (c) CALTECH101

(d) CALTECH2556 (e) CARS (f) CIFAR100

(g) FLOWERS (h) FOOD (i) PETS

(j) SUN397

Figure 27: We repeat the counterfactual experiments (c.f. Figure 2b from the main paper) but using the influence values computed for CIFAR-10 on other target datasets.

E.5.2 Transfer across architectures

How well do our transfer influences work across architectures? Recall that we computed our transfer influences using a ResNet-18. We now repeat the counterfactual experiment from Figure 2b, using these influences to remove classes from the source dataset when training a ResNet-50. We find that these influences transfer relatively well.

Figure 28: We repeat the counterfactual experiments (c.f. Figure 2b from the main paper) but use our influence values computed using a ResNet-18 on a ResNet-50.

F Further Convergence Analysis

In this section, we analyze the sample complexity of our influence estimation in more detail. In particular, our goal is to understand how the quality of influence estimates improves with the number of models trained. We also look at the impact of different choices of model outputs.

Instead of directly measuring our downstream objective (transfer accuracy on target dataset after removing the most influential classes), which is expensive, we design two proxy metrics to gauge the convergence of our estimates:

Rank correlation. As Ilyas et al. [IPE+22] shows, we can associate influences with a particular linear regression problem: given features $\mathbb{1}_{\mathcal{S}_i}$, an indicator vector of the subset of classes in the source dataset, predict the labels $f(t; \mathcal{S}_i)$, the model's output after it is finetuned on target dataset $\mathcal{T}$. In fact, we can interpret influences as weights corresponding to these binary features for presence of each class. That is, the influence vector $w_t = \{\text{Infl}[C_i \to t]\}_i$ defines a linear function that, given a subset of classes that the source model is trained on, predicts the corresponding model's output when trained on that subset. Here, we focus on analyzing average model accuracy, so in fact we consider the aggregate output $\sum_i f(t; \mathcal{S}_i)$ and the corresponding aggregated influences $w = \{\text{Infl}[C_i]\}_i$.

Given this view, we can measure the quality of the influence estimates by measuring their performance on the above regression problem on a held-out[9] set of examples $\{\mathcal{S}_i, \sum_i f(t; \mathcal{S}_i)\}$. In order to make different choices of model outputs (logit, confidence, etc.) comparable, we measure performance with spearman rank correlation between the ground truth model outputs and the predictions of the linear model (whose weights are given by the influences).

We measure this correlation while varying both the number of trained models used in the influence estimation and the choice of the model output, and the results are shown in Figure 29.

False discovery rate. Above we considered a measure based on predictive performance. Here, we focus on a more parameter-centric notion of False Discovery Rate (FDR). Intuitively, FDR here quantifies the following: how often are the top influencers actually just due to noise?

The Knockoffs [CFJ+18] framework allows one to perform feature selection and also estimate the FDR. At a high level, it consists of two steps: First, one constructs "knockoff" versions of the original features which are distributed "indistinguishably" (more formally, exchangable) from the original features, and at the same time are independent of the response by design. Second, one applies an estimation algorithm of choice (e.g., OLS or LASSO) to the augmented data consisting of both the original and the knockoff features. Then, the relative frequency at which a variable X_i has higher statistical signal than its knockoff counterpart $\tilde{X}_i$ indicates how likely the algorithm chooses true features, and this can be used to estimate the FDR (intuitively, if X_i is independent of the response y, X_i is indistinguishable from its knockoff $\tilde{X}_i$ and both are equally likely to have higher score).

We adapt this framework here (particularly, the verion known as model-X knockoffs) as follows:

1. Sample an independent knockoff matrix $\tilde{X}$ consisting of 1,000 binary features from the same distribution as the original mask matrix X (namely, each instance has 500 active features).[10]
2. Estimate influences for both original and knockoff features using the difference-in-means estimator.
3. Consider the top $k = 100$ features by positive influence, and count the proportion of features that are knockoff. This yields an estimate of FDR among the top 100 features. An FDR of 0.5 indicates chance-level detection.[11]

As with the previous metric, we measure the above FDR for each target dataset while varying the number of trained models and the target output type (Figure 30).

[9] We split the 7,540 models into a training set of 6,000 and a validation set of the remainder.

[10] Technically, this procedure is not exactly valid in the original FDR framework as X_i and $\tilde{X}_i$ are exchangable due to depenencies in the features. Nonetheless, it is accurate up to some approximation.

[11] This is different from the usual manner of controlling the FDR, but we look at this alternative metric for simplicity.

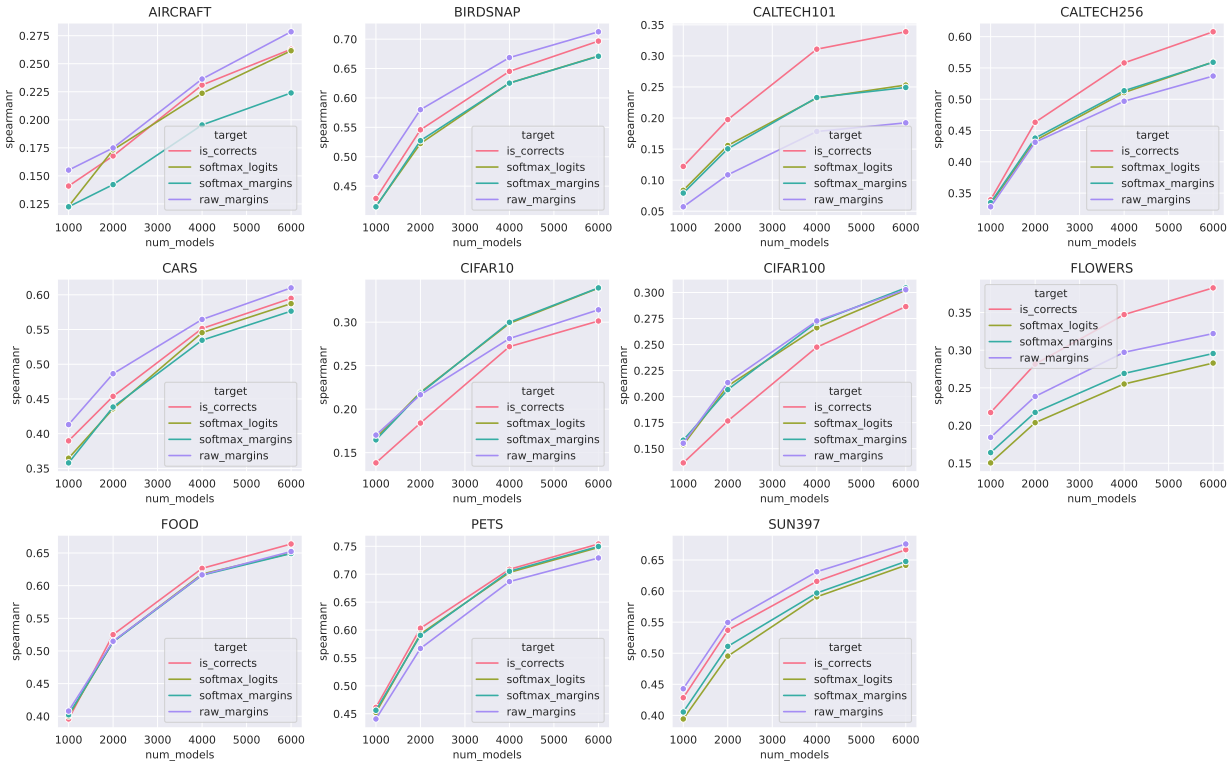


Figure 29: Measuring improvement in influence estimates using the **rank correlation** metric. The rank correlation here measures how well influence estimates perform in the underlying regression problem of predicting target accuracy from the subset of classes included in the training set. We evaluate on a held-out set of subsets independent from those used to estimate influences. Across all datasets, correlation improves significantly with more trained models.

Discussion. We observe the following from the above analyses using our two statistics:

- There are significant gains (higher correlation and lower FDR) with increasing number of trained models.
- But neither metric appears to have plateaued with 6,000 models, so this indicates that we can improve the accuracy of our influence estimates with more trained models, which may in turn improve the max improvement in transfer accuracies (Section 3), among other results.
- The choice of the target type does not appear to have a significant or consistent impact across different datasets, which is also consistent with the results of our counterfactual experiments (Appendix B.1).

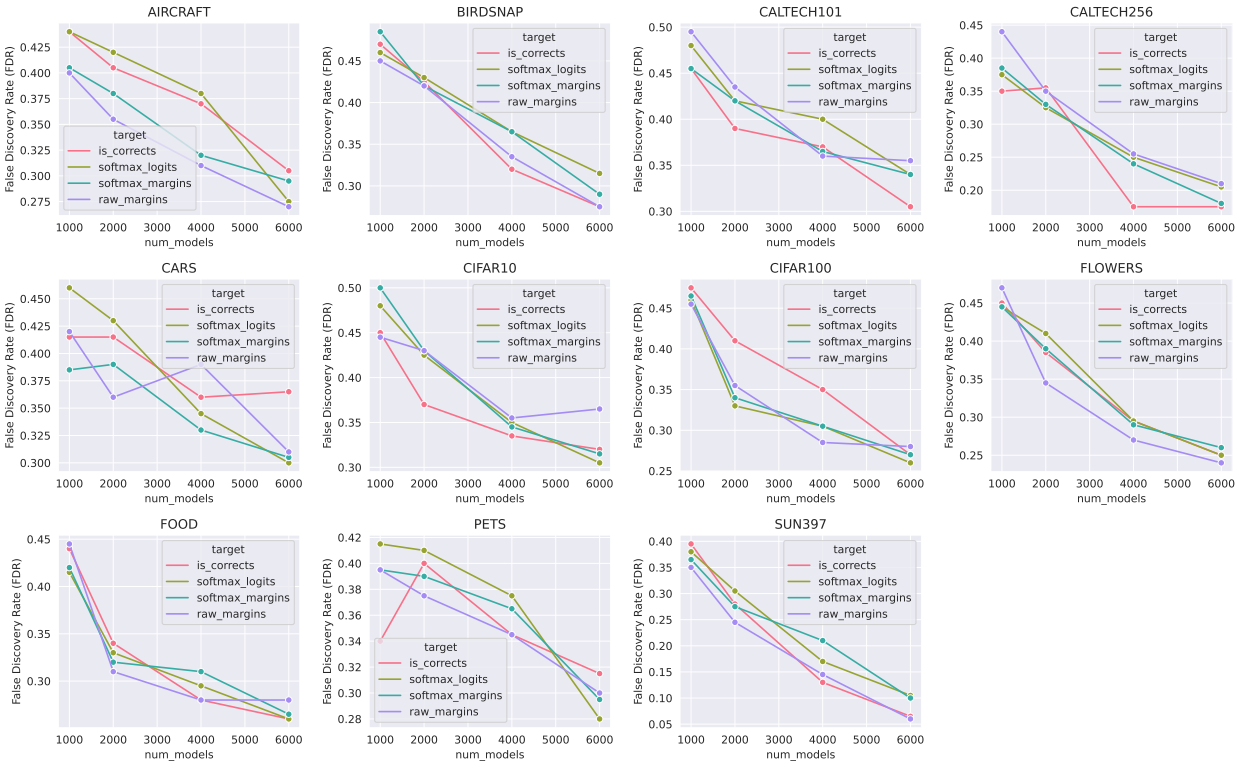


Figure 30: Measuring improvement in influence estimates using the **False Discovery Rate** heuristic. Using a procedure (loosely) based on the Knockoffs framework, we estimate the proportion of false discoveries within the top 100 features ranked by esteimated influences. Across all datasets, FDR decreases generally with more trained models.