Dataset Deduplication with Datamodels

by

Yunxing Liao

S.B., Computer Science and Engineering, Massachusetts Institute of Technology (2021)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2022

Author . . . .
Department of Electrical Engineering and Computer Science
May 12, 2022

Certified by . . . .
Aleksander Mądry
Cadence Design Systems Professor of Computing
Thesis Supervisor

Accepted by . . . .
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Dataset Deduplication with Datamodels

by

Yunxing Liao

Submitted to the Department of Electrical Engineering and Computer Science
on May 12, 2022, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Large curated datasets have been essential to the development of deep learning models across many disciplines. Consequently, the properties of these datasets have a large impact on the behavior of these models. As machine learning pipelines increasingly leverage more unlabelled datasets—which tend to undergo less curation than labelled datasets—controlling data quality becomes even more important. We focus on a particular aspect of data quality: *train-test leakage* or *duplicate examples*. These can cause overestimation of models' performance on benchmarks among other issues. In this work, we apply datamodels, a framework for analyzing the behavior of a model class as a function of its training data, to deduplicate unlabelled datasets. Inspired by the recent CLIP model, we focus on detecting duplicates between YFCC15M and the ImageNet validation dataset. Our results demonstrate how to adapt datamodels effectively for these filtering tasks in unsupervised, large-scale settings. We finish by discussing the challenges of our method and duplicate detection more broadly.

Thesis Supervisor: Aleksander Mądry
Title: Cadence Design Systems Professor of Computing

Acknowledgments

I would like to thank my advisor, Prof. Aleksander Mądry, for his supervision and support. I would like to thank Sam Park for his mentorship in the past month and his feedback on my thesis draft revisions. Thank you for your guidance on this project and for teaching me how to become a better writer and researcher. I would also like to thank Andrew Ilyas for his mentorship over the past 2 years.

I would like to thank Guillaume Leclerc for making an awesome, fast library for dataloading, ffcv. All of my models would not have finished training without him. I would like to thank Hedi Driss for performing miracles on the machines in the lab's compute cluster. I would like to thank Logan Engstrom for always making the lab an interesting place to be. I would also like to thank Kristian Georgiev for introducing me to the lab fish and the lab snacks (two very different things).

I would like to thank the 6.036 instructors for having me as a course staff for three semesters. I was first introduced to machine learning through 6.036, and I learned a lot both as a student and as a TA in this class.

I acknowledge the MIT SuperCloud and Lincoln Laboratory Supercomputing Center for providing resources that have contributed to the research results reported within this thesis.

Finally, I would like to thank my parents for their unconditional support throughout my journey at MIT.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A key driver in the progress of deep neural architectures in machine learning is the development of large datasets. Labelled datasets such as the ILSVRC[6] image classification benchmark ("ImageNet") have been central to the development and evaluation of image classification models. Recent state of the art machine learning models use even larger uncurated and unlabelled datasets. The best ImageNet classification models use massive neural network architectures and both labelled and unlabeled training examples. OpenAI trained CLIP[15], a self-supervised model that achieves state of the art accuracy across many image classification benchmarks, with 400 million image-text pairs. Large datasets have also been central to recent progress in natural language processing. GPT-3[3], a state of the art language model, was trained using 500 billion tokens.

Although large, unlabelled datasets are helpful for achieving state of the art results, these datasets tend to be more poorly curated compared to labelled datasets. In particular, train-test duplication can be a problem within such datasets. Classification accuracy on ImageNet and other datasets serve as an important benchmark for evaluating new computer vision algorithms. If duplicates exist between the training and the validation dataset, the benchmark could encourage the emergence of training algorithms that are better at training examples than generalizing to unseen data. Prior work shows that removing train-val duplicates can cause significant reductions in the evaluation accuracy of state of the art computer vision models on CIFAR[1].

In addition, training on datasets with train-test duplication can lead to lower-quality models. Language models trained on a deduplicated version of C4 have lower perplexity scores and output fewer memorized sequences[11].

More specifically, we are interested in finding duplicates between the CLIP dataset and the ImageNet validation set. The CLIP training dataset is a model of interest because CLIP's evaluation datasets are known to have duplicates within the its respective training dataset. For instance, FairFace[10], a CLIP evaluation dataset, is drawn entirely from YFCC100M[17], a dataset that was also used to draw examples for the CLIP training dataset. The same also applies for Country211, another CLIP evaluation dataset. This thesis can help future work better understand how duplicates affect downstream task performance by improving duplicate detection techniques. We are particularly interested in finding duplicates between the CLIP dataset and the ImageNet evaluation dataset because ImageNet validation accuracy is an important metric for evaluating computer vision models. Prior work has shown that ImageNet accuracy is correlated with performance on downstream tasks.

Most traditional methods for dataset deduplication in computer vision are fairly limited. It is common to use pixel-wise distance metrics (e.g l2 distance) to identify duplicates. However, this method is not robust to small differences in saturation, scaling, or translation. Distance in representation space[1] (e.g using features of a pretrained model) is another common heuristic for dataset deduplication. Datamodels [8] outperformed distance in representation space for dataset deduplication on CIFAR. Datamodels were able to find 10% train-test leakage in CIFAR compared to 3% using distance in representation space. Similarly, datamodels outperformed distance in representation space when finding train-val duplicates in FMoW. In this work, we apply datamodels to dataset deduplication between YFCC15M and the ImageNet validation dataset.

In this thesis, we use datamodels for dataset deduplication between YFCC15M and the ImageNet validation dataset. YFCC15M is a dataset filtered from YFCC100M by OpenAI in 2020. This dataset is a subset of YFCC100M that contains image and English text pairs. YFCC15M is also a subset of the dataset used to train the OpenAI

CLIP model. Our contributions in this thesis are as follows:

- **Extending datamodel-based deduplication to unlabelled datasets:** This work presents the first attempt at using datamodels to deduplicate an *unlabelled* dataset. Previously, datamodels have only been used for deduplicating labelled datasets. We present modifications to the datamodel training process that allow it to deduplicate unlabelled datasets.

- **(Partially) Quantifying leakage between YFCC15M and ImageNet:** We detect only a small degree of duplication between YFCC15M and the ImageNet validation set. Of the 600 samples we manually inspect (and are predicted to be most likely to contain duplicates), we only find 21 duplicates.

- **Comparing datamodel-based and representation-based methods:** We compare the performance of the datamodel with a baseline method based on embedding or representation distance. Although datamodels detect more duplicates than the ResNet50 baseline, we find that datamodels do not perform significantly better than the baseline at detecting ImageNet-YFCC15M duplicates. Nonetheless, both methods detect non-overlapping set of examples, suggesting that combining different methods can be a more effective duplicate detection method.

- **Analyzing challenges of automated duplicate detection:** We analyze the failure cases of the datamodel in duplicate detection. Specifically, we observe that datamodels struggle to detect duplicates among evaluation images that models can more easily classify. We propose various strategies to improve the datamodel's ability to detect duplicates.

In Chapter 2, we present background relevant to this work. In Chapter 3, we describe the methods we used to find duplicates between YFCC15M and the ImageNet validation set. In Chapter 4, we describe the results of applying our deduplication strategy to a baseline method and offer an analysis.

Chapter 2

Related Work

2.1 Large-scale computer vision datasets

Large-scale datasets have been essential to the growth of deep learning in computer vision. In this section, we will describe some important computer vision datasets relevant to this thesis.

2.1.1 ImageNet

ImageNet Large Scale Visual Recognition Challenge (ILSVRC)[6] dataset is a supervised image classification dataset with around 1.2 million labeled examples from 1000 classes. The ILSVRC dataset has served as an important benchmark for computer vision algorithms. Prior work has shown that good classification accuracy on ImageNet is correlated with transfer accuracy on downstream tasks[9]. It has been shown that there are exceptions to this trend. Some adversarially robust ImageNet models have higher transfer accuracy despite having lower performance on ILSVRC[16]. Self-supervised models tend to perform worse on ILSVRC dataset, but they tend to have higher transfer accuracies and do better on downstream tasks[12].

Some recent works have also addressed the quality of labelling within the ImageNet dataset. Each image in the ImageNet dataset has a single label assigned through Amazon Mechanical Turk workers. However, there exist multiple classes present in some ImageNet images, leading to ambiguity concerning the true label in the image. In [2], reassessed labels (ReaL) are assigned to each image such that each image in the ILSVRC dataset can have multiple labels. They found that state of the art ImageNet models tend to perform well even in the presence of label ambiguity, which suggests that these models tend to be good at memorizing label noise.

2.1.2 YFCC100M

YFCC100M[17] is a dataset of 100 million images collected from Flickr from 2004 until 2014. YFCC100M is an unlabelled dataset. Although there are no label annotation for each image, some images in YFCC100M contain natural text titles, descriptions, and tags. The presence of natural language titles and descriptions has made YFCC100M a valuable resource for training multimodal models such as CLIP and DALLE.

Due to the massive size of YFCC100M, most people do not work with the entirety of YFCC100M. Some notable subsets of YFCC100M include:

- **YFCC15M**[15] - A subset of 15 million images in YFCC100M that contains natural language titles and descriptions. This subset was used as part of an internal OpenAI dataset to train CLIP and DALLE.
- **FairFace**[10] - A public dataset of around 100,000 images balanced across 7 race groups. All images are labelled with age, race, and gender information. This subset is used to calibrate models across race, age, and gender groups.
- **Country211**[15] - A subset designed to evaluate a model's performance on geolocation. This subset was constructed from image-geotag pairs within YFCC100M.

2.2 CLIP

Supervised labels are expensive to collect. To avoid paying the price of labelling images, some learning algorithms have been designed to take advantage of large unlabelled datasets. CLIP[15] uses image-text pairs to jointly train an image encoder and a text encoder. During CLIP training, we assume that each image is most similar to its corresponding caption and dissimilar to all other captions. CLIP enforces this assumption through the InfoNCE[13] objective, which maximizes the cosine similarity of image-caption pairs and minimizes the cosine similarity of the image with all other captions in the training batch.

CLIP achieves state of the art results across a wide range of evaluation benchmarks and image domains. CLIP's training dataset is central to its success. OpenAI trained CLIP using WIT, an internal dataset of 400 million image-caption pairs. Training on smaller datasets causes a significant decrease in CLIP's performance on downstream tasks. CLIP can only achieve 62% accuracy on ImageNet after training exclusively on YFCC15M, but it can achieve 73.3% accuracy after training on WIT using a comparable architecture.

2.3 Dataset Deduplication

2.3.1 Dataset Deduplication in NLP

Previous works find that dataset duplication is highly prevalent in natural language datasets, and removing exact and near-duplicates can improve the performance of language models. [11] finds that train-test overlap is highly prevalent in language datasets. Specifically, the authors deduplicate C4, a large dataset used to train T5 Text-to-Text Transformers. They discovered that 7.18% of tokens from the C4 training set are duplicated in the validation set through exact substring matching.

No negative consequences arose from deduplication, despite the reduction in the size of the training corpus. In fact, the authors found that duplication leads to several positive outcomes. Duplication causes the model to generate less diverse text. By removing near-duplicates, the model emits 10 times fewer memorized sequences. In addition, removing duplication causes the language model to generate better higher-quality text. Model perplexity can decrease by up to 10% after removing duplicates.

This work uses methods that are specific to text processing, such as MinHash and substring matching. These techniques cannot be applied to images.

2.3.2 Deduplicating CLIP Dataset

Various previous works have attempted to deduplicate the CLIP training dataset due to the large degree of known overlap between the CLIP training dataset and evaluation datasets. For instance, CLIP reports metrics on the Country211 geolocation dataset, which has 20 percent overlap with the CLIP training dataset. Prior work use the following strategies for deduplicating the CLIP dataset:

- In the CLIP paper[15], the authors use a self-supervised model to detect duplicates between the CLIP training set and various evaluation sets. The self-supervised model is trained by aligning image representations under different augmentations.
- In the LiT paper[19], the authors use a B/32 Vision Transformer pretrained on the JFT300M dataset to perform deduplication.

Previous work show that YFCC-ImageNet validation duplicates do not significantly affect CLIP training. In the CLIP paper, the authors compare the performance of CLIP on various downstream tasks before and after deduplication. Even under the presence of a 20 percent train-validation dataset overlap, the CLIP model trained on a deduplicated dataset does not perform much worse than a CLIP model trained on the original dataset. Similar results have also been observed by LiT.

2.3.3 Deduplicating Image Datasets

In addition to the techniques described in Section 2.3.2, several other works are related to image deduplication.

- Embedding distance is commonly used to deduplicate datasets. As described in [1], we can compute the distance between image embeddings to determine if an image pair is a duplicate. If the embedding distance between an image pair is less than a threshold value, this technique would flag the image pair to be a potential duplicate.

- Near-duplicate detection is related to instance-level content-based image retrieval (CIR). Instance-level CIR is the task of searching in a large database to find identical instances of objects present in the query image under various conditions (ie illumination, cropping, different perspectives) [5]. Prior works propose Siamese networks and manifold learning for this task.

- Identifying near-duplicates is related to copy detection. Various neural network architectures have been developed to detect instances of copyright violation. These models are trained to identify duplicates even when the duplicate has been intentionally modified to evade detection. Examples of models that achieve state of the art on copy detection evaluation datasets include DINO[4] and SEER[7].

2.4 Datamodels

Datamodels[8] can be used to understand the behavior of a model class in terms of its training data. A datamodel is a function that predicts the outcome of training a model on an arbitrary subset of the training data and evaluating the resulting model on a fixed target. We can use datamodels for a wide range of tasks, including feature representation, duplicate detection, and counterfactual identification. Datamodels are estimated through the following procedure:

1. Randomly sample N number of subsets S_i from the training dataset. The size of each subset S_i depends on the parameter $\alpha \in [0, 1]$: $|S_i| = \alpha \cdot |D_T|$, where D_T is the training dataset.

2. Train one model m_i per subset S_i. After this step, there should be N models.

3. Record the model output $O_{i,j} = f(m_i(x))$ for each example x_j in the evaluation dataset D_E. This results in an $N \times |D_E|$ array O of predicted model outputs.

4. For a particular training example j, train a datamodel D that predicts model m_i output $O_{i,j}$ from S_i, the subset of the training dataset that was used to train m_i.

Datamodels can find more train-test duplicates than some existing methods. This approach found a 10% train-test overlap between the CIFAR training dataset and the CIFAR test dataset, significantly more than the 3% found through embedding distances. Similarly, datamodels detected more duplicates in FMoW than through the embedding distance method.

Chapter 3

Design

We use datamodels to detect duplicates between the ImageNet validation set and a filtered subset of YFCC15M. [8] uses datamodels to detect duplicates between the the CIFAR validation dataset and the CIFAR training dataset. More specifically, they use a linear datamodel g_θ to analyze the influences between training samples and validation samples. By examining the learned θ, we can determine the impact of a training sample on a learned model's prediction. If a training sample has a large influence on a validation sample, then it is likely that there is a duplicate of the validation sample on the training sample. We cannot use the same process to detect duplicates between YFCC15M and the ImageNet validation set without modifications. Since YFCC15M does not have labels, we cannot train a datamodel to predict $f_A(x; S_i)$, where f_A is the correct class margin.

The next sections describe how we extend datamodels to analyze an unlabelled dataset. Roughly, this process can be split into the following steps:

1. **Training Models on the ImageNet validation dataset**: 100,000 models were trained on different random 50 percent subsets of the ImageNet validation dataset.

2. **Filtering Candidate YFCC15M Examples**: Evaluating 100,000 models on the entirety of YFCC15M is too slow and time-intensive. This evaluation would have taken a few months, even assuming the availability of 16 V100 GPUs. In

order make the evaluation process more efficient, we used 20 resnet18 models to remove out-of-distribution samples from the YFCC15M dataset. In addition, we assigned pseudolabels to the filtered samples.

3. **Estimating Influences**: We computed influences from the ImageNet validation dataset to YFCC15M using predictions from the 100,000 models on the YFCC15M dataset. Using the influence values, we can estimate the importance of each training example on the model prediction of YFCC15M. We expect that high magnitude influence values correspond to potential duplicates.

3.1 Training Models

We wish to predict the influence of one example i on another example j. Let i be any datapoint in D_1, j be any datapoint in D_2, where D_1 and D_2 are two different datasets. To measure the influence of training sample i on validation sample j, we can train models on many subsets $\{S_n\}$ of D_1 and evaluate these models on j. Let $S_i = \{S_n : i \in S_n\}$. Let $\overline{S_i} = \{S_n : i \notin S_n\}$. Let $f_A(x, S_n)$ be the evaluation of some validation point x on a model trained on set S_n. If $|\mu_{S_i}(f_A(j, S_n)) - \mu_{\overline{S_i}}(f_A(j, S_n))|$ is large, then training example i has a large influence on the prediction of example j.

In most cases, this relationship should be symmetric. Assume a training example i is highly influential on the model's evaluation on validation j. Then, sample j should also have high influences in training sample i, assuming that sample j is part of the training set and sample i is part of the validation set. Thus, we have the option of either training many models over subsets of the ImageNet validation set or training models over subsets of YFCC15M. We decided on training models over subsets of the ImageNet validation set because the YFCC15M is an unlabeled dataset; training classification models over YFCC15M is impossible. In addition, YFCC15M is the bigger dataset, and it would be faster to train models over YFCC15M.

During training, we randomly sampled 100,000 random 50 percent subsets of the ImageNet validation dataset and trained 100,000 ResNet18 models on the random 50 percent subsets. Before training, we performed grid search the select the best

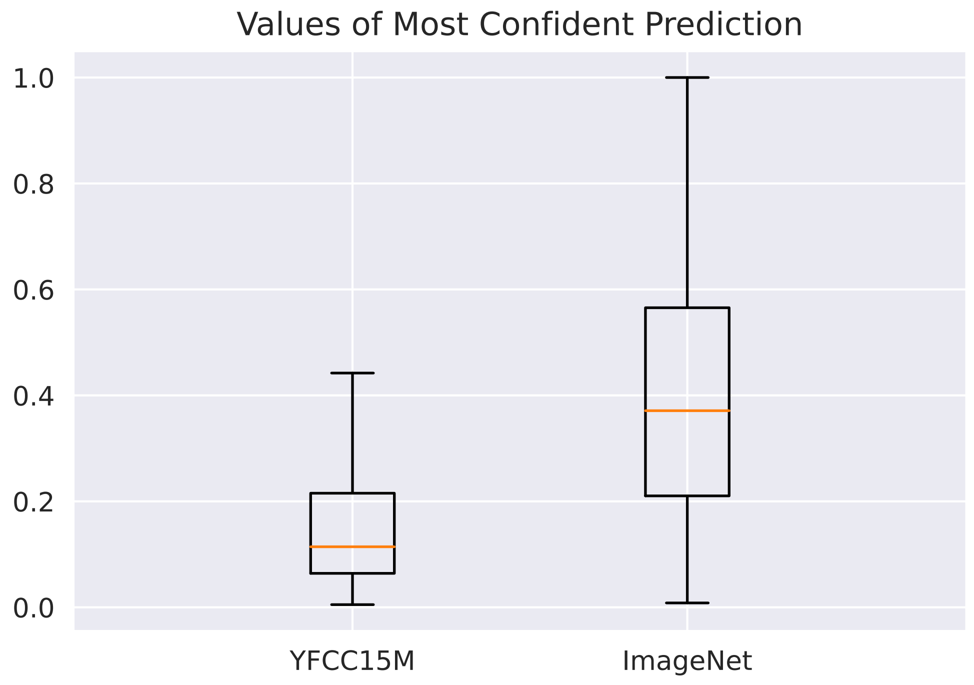


Figure 3-1: Average ImageNet Model Confidences on YFCC Compared to ImageNet Training Set

hyperparameters for training on the ImageNet validation set. The hyperparameters we used can be found in Table A.1.

3.2 Filtering YFCC15M

Due to the massive size of YFCC15M, we decided to only examine samples in YFCC15M that are likely to be duplicated in ImageNet. In order to perform this filtering, we trained 20 resnet18 models on the entire ImageNet training dataset. Then, recorded the output of each 20 resnet18 model on the YFCC15M dataset. We calculated the average prediction of each resnet18 model on each YFCC15M datapoint. Let $\mu(y) = \sum_{i=1}^{20} f_i(y)/20$, where f_i is one of our resnet18 classifiers, $y \in Y$ is a YFCC15M image. We assign top 5 largest values of $\mu(y)$ as the model confidence and record the top 5 largest classes in $\mu(y)$ for each y.

Ultimately, we decided to select the 10,000 samples with the largest average confidences for computing influences. This technique is commonly used in related work in semi-supervised learning to remove out-of-distribution data[18, 14]. After manual inspection of randomly selected subsets of the filtered data, we believe that the filtered data can belong to ImageNet classes. Examining the confidence values $\mu(y)$ also suggests that the filtered data belongs to ImageNet classes. Figure 3-1 shows the distribution of model confidences $\mu(y)$ on YFCC15M compared to the ImageNet training set. We observe that model confidences for the majority of YFCC15M images is less than 0.45, which suggests that the outliers with confidence values greater than 0.45 belong to ImageNet classes. However, there exist many ImageNet training samples on which the trained models have low confidence. There could exist many duplicates in the remaining, unexamined portions of YFCC15M on which the trained models predict lower average confidence.

3.3 Computing Influences

Using the 100,000 models trained on 50 percent of the ImageNet validation set and evaluated on the filtered subset of YFCC15M, we can compute the influence of examples from the ImageNet validation set on examples from YFCC15M. [8] shows that calculating influences is a special form of datamodelling. Let i be an example from the ImageNet validation set. We can compute S_i and $\overline{S_i}$ by examining the training set for each 50 percent ImageNet model. Then, we compute influences for each data i from the ImageNet validation set and j from the YFCC15M filtered subset through the following equation:

$$|\frac{1}{|S_i|}\sum_{S_n \in S_i} f_A(j, S_n) - \frac{1}{|\overline{S_i}|}\sum_{S_n \in \overline{S_i}} f_A(j, S_n)| \tag{3.1}$$

We recorded influences for five different choices of f_A. Let p_i be the ith largest class index recorded after averaging the outputs of 20 ImageNet models from section 3.2. We let $f_{A_i} = o_{p_i}$, where o are the logits of a trained model. We record the top 5 largest logit values and classes. We will refer to the top-k largest logit classes as the top-k pseudolabel. After computing influences, we ranked each ImageNet validation image i by the maximum influence over YFCC images j. We manually inspected the top 300 ImageNet validation images by maximum influence on YFCC image to find proposals for duplicates.

Chapter 4

Results

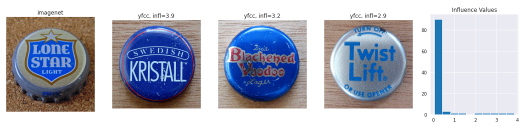
In this chapter, we compare the results of applying datamodels to duplicate detection to a baseline method that uses embedding distance. After applying the method described in Section 3, we realized that both the baseline and datamodels propose false positives and that manual review of candidates was necessary. We describe the procedure we use to select duplicates for manual review in Section 4.1. In Section 4.2, we compare the performance of datamodels to a baseline method. We find that datamodels do not significantly outperform the baseline method. We describe proposals for improving the datamodels' performance in detecting duplicates in Section 4.3.

4.1 Selecting Duplicate Candidates

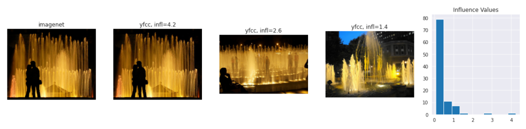
We discovered many false positives when analyzing the top predictions from the datamodels and the baseline method. Section 4.1.1 describes how false positives are virtually indistinguishable from true duplicates using embedding distance or influence scores alone. The next section describes the methods we use to select the top duplicates for review.

4.1.1 Challenges of Fully Automated Duplicate Detection

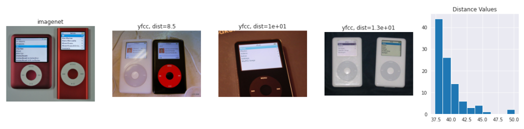
We encountered many false positives when examining the top image proposals from the baseline method and the datamodel. It was quite difficult to distinguish between false positive proposals by examining influence values or embedding distances alone. In Figure 4-1b and Figure 4-1a, we present histograms of the 100 highest influence values across YFCC images for an ImageNet image with an exact duplicate and a histogram of the same values for an ImageNet image with no duplicate. The histogram of influence values for these two images are virtually indistinguishable. Similarly, the distribution of the distances to the 100 closest ImageNet embeddings is virtually indistinguishable between duplicated ImageNet examples and non-duplicated ImageNet examples. Due to the similarity of influence values and embedding distance between duplicates and false positives, we need to manually review duplicate proposals from the datamodels and the baseline.



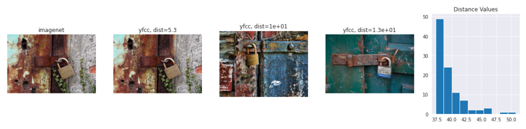
(a) Top Influences - No Duplicate



(b) Top Influences - Duplicate



(c) Top Distances - No Duplicate



(d) Top Distances - Duplicate

Figure 4-1: Top 100 Distance and influences for duplicate and non-duplicate image pairs

4.1.2 Narrowing Duplicate Candidates

To minimize the number of candidates for manual review, we only inspected the most promising duplicate proposals from the datamodels and the baseline method. We use the following heuristic to select the most promising duplicate proposals from the datamodels. We only manually reviewed duplicate proposals from the 300 ImageNet images with the highest maximum influences on top-1 pseudolabel value across

YFCC images. For each of these ImageNet images, we propose the 10 YFCC images with the highest influence on top-1 pseudolabel value as potential duplicate candidates. We originally examined images by their influence across all top-5 pseudolabel values, but we found that duplicates usually occur between ImageNet images and the YFCC images with largest influence on the top-1 pseudolabel value.

The baseline model proposes ImageNet-YFCC image pairs with the smallest Euclidean distance between ImageNet-YFCC image embedding pairs as potential duplicates. Similar to the datamodels approach, we manually review duplicate proposals from the 300 ImageNet images with the smallest minimum embedding distance across YFCC images. For each of these 300 ImageNet images, we inspect the 10 YFCC images with the smallest embedding distance to the ImageNet image for duplicates.

4.2 Detected Duplicates

In this section, we provide data on duplicates found by the embedding method and the datamodels. We are interested in detecting both *exact duplicates* and *near duplicates*. Exact duplicate images are image pairs that are almost pixel-wise identical. They could be detected by comparing the pixel-wise difference between an image pair to a preset threshold. Near duplicates are duplicate images from the same scene, but not necessarily from the same camera perspective or the under the same illumination, lighting conditions, etc.

4.2.1 Duplicates found by both Datamodels and Baseline

In Figure 4-2a, we show the duplicates detected by both the datamodel and the baseline model. Within the top 300 image proposals, we detect 14 exact duplicates. Both the baseline method and datamodels are able to detect exact duplicates. However, the datamodels require more samples to detect the same duplicates. Table 4.1 shows that the top 14 image proposals from the baseline model are exact duplicates. One must examine 144 duplicate proposals to detect exact the same duplicates using the datamodel. This result suggests that the baseline model could be a better tool for detecting exact duplicates.

ImageNet Index	Datamodel Rank	Datamodel Influence	Baseline Rank	Embedding Distance
34209	82	2.24	6	3.72
34787	50	2.58	9	5.29
35236	35	2.86	5	3.50
28100	4	4.24	1	2.60
38566	46	2.64	11	5.47
47687	58	2.45	3	3.12
49509	92	2.18	0	2.03
30571	1	4.97	13	7.35
25998	53	2.56	7	3.99
47667	61	2.44	8	4.18
26423	3	4.55	2	2.88
48568	73	2.28	10	5.43
24505	144	1.96	4	3.32
43899	87	2.20	12	6.55

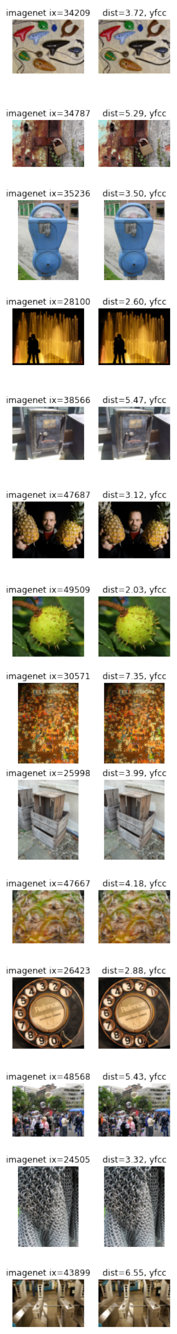
Table 4.1: Relative rank for duplicates found by both datamodel and baseline

4.2.2 Duplicates only found through Baseline

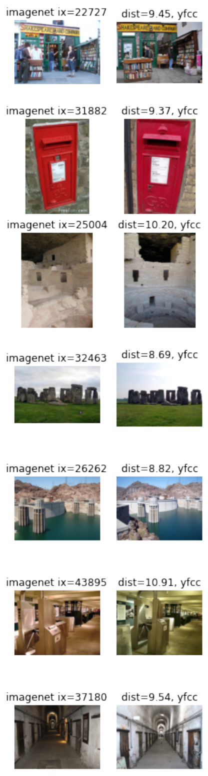
The baseline was able to detect some near-duplicates that the datamodel was unable to detect. In Figure 4-2b, we show the images that were only found by the baseline. The baseline model is able to detect ImageNet duplicates with remarkably few proposals. In Table 4.2, we show that 7/8 duplicates detected by only the baseline were found after examining fewer than 100 samples.

ImageNet Index	Datamodel Rank	Datamodel Influence	Baseline Rank	Embedding Distance
22727	8711	0.32	31	9.45
31882	6194	0.45	27	9.37
25004	694	1.26	82	10.20
32463	1088	1.08	18	8.69
26262	350	1.52	20	8.82
43895	4430	0.57	178	10.91
37180	4158	0.62	33	9.54

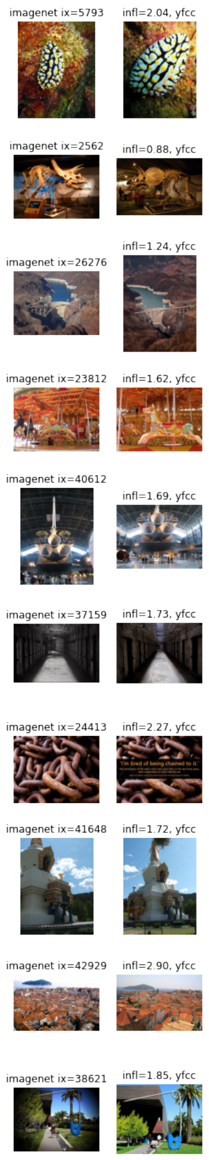
Table 4.2: Relative rank for duplicates found by only the baseline model



(a) Both



(b) Baseline Only



(c) Datamodel Only

Figure 4-2: Duplicates found by each model type

ImageNet Index	Datamodel Rank	Datamodel Influence	Baseline Rank	Embedding Distance
5793	123	2.04	1986	13.73
2562	220	1.71	8016	16.33
26276	182	1.79	910	12.59
23812	270	1.62	1971	13.71
40612	230	1.69	545	11.99
37159	207	1.73	976	12.70
24413	75	2.27	3681	14.76
41648	212	1.72	4958	15.34
42929	15	3.72	1494	13.29
38621	173	1.85	717	12.29

Table 4.3: Relative rank for duplicates found by only the datamodel

4.2.3 Duplicates only found through Datamodels

10 duplicate pairs were only detected by the datamodel. In Figure 4-2c, we show the image pairs that were only detected by the datamodel. In Table 4.3, we show the ranks of the image pairs that were only detected by the datamodel. Interestingly, one must examine over 200 duplicate proposals to find most of the near-duplicates only detected by the datamodel. It is possible that the top proposals from the datamodel do not typically contain near-duplicates, and that some near-duplicates are proposed beyond the 300th proposal.

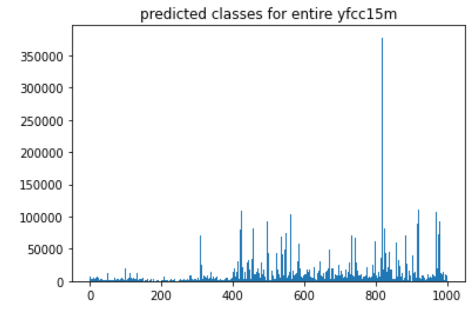
4.3 Additional Discussion

We were hoping to find more duplicates between the ImageNet training dataset and YFCC. Datamodels were able to find many more duplicates between the CIFAR and FMoW training and validation datasets than the baseline method. Encouraged by these results, we also expected that the datamodel would perform better than the baseline on the duplicate detection task. However, we found that the baseline performs comparably to the datamodel duplicate detector. In addition, we believe that there are not as many duplicates between YFCC100M and the ImageNet validation dataset as we had previously suspected. We randomly select 300 ImageNet images and examine the corresponding 10 YFCC images with the highest influence on top-1 pseudolabel value for each ImageNet image. We found no duplicates of the 300 randomly sampled ImageNet images, so we suspect that the overlap rate between YFCC15M and ImageNet is relatively low. In the subsequent sections, we will discuss potential explanations for why so few duplicates were detected between ImageNet and YFCC and why our datamodel did not perform substantially better than the baseline.

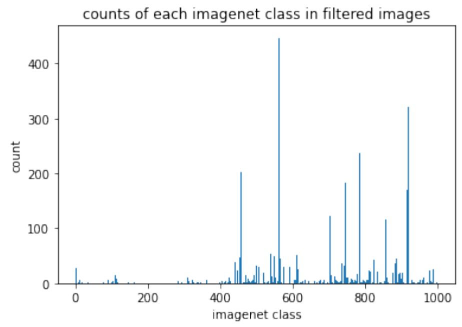
4.3.1 Little Overlap between examined subset and ImageNet Validation

There could inherently exist few duplicates between YFCC15M and the ImageNet validation dataset. In 4-3a, we show the distribution of pseudolabels across the entire YFCC15M dataset. Each ImageNet class is present at least once in the YFCC15M pseudolabels. Some classes are better-represented than others. The most common YFCC15M pseudolabel is “stage”, with 377948 images in the YFCC sharing this pseudolabel. The least common YFCC15M pseudolabel is "Sealyham terrier", with only 42 images. This is not a surprising outcome; we can expect that some ImageNet classes are unlikely to be posted on Flickr, the source of all YFCC images. We could easily expect that there are more images of stages, lakeshores, and movie theaters (the most common YFCC pseudolabels) compared to Sealyhams, Dandie Dinmont terriers, and Japanese spaniels (the least common pseudolabels). Because many ImageNet classes are underrepresented in YFCC, we would expect fewer duplicates from these classes.

In addition, the relatively poor performance of YFCC15M CLIP on ImageNet could imply that there are very few duplicates between YFCC15M and the ImageNet validation dataset. CLIP[15] has fairly low linear probe accuracy on the ImageNet validation dataset when trained on only YFCC15M (not the entire OpenAI dataset). A CLIP trained with YFCC15M only achieves around 60 percent linear probe accuracy and 30 percent zero-shot accuracy on the ImageNet validation dataset. It is fairly straightforward to achieve higher accuracies on ImageNet validation dataset simply by training a supervised model on the ImageNet training dataset. CLIP does not outperform the best semi-supervised models on ImageNet even when trained on the entire OpenAI dataset. CLIP performs 3% worse on ImageNet compared Noisy Student, a semi-supervised approach. There could exist more train-val duplicates in the 21 datasets where CLIP outperforms Noisy Student, such as Country211 (+22.7%), StanfordCars (+15.9%), or GTSRB (+14.7%). However, it is also possible that Noisy Student is overfitted to the ImageNet dataset.



(a) YFCC15M



(b) Filtered 10k YFCC15M

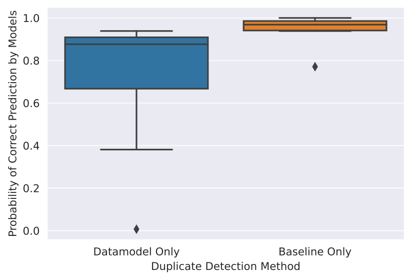
Figure 4-3: Counts for each pseudolabel class by dataset

It is possible that our filtering process from Section 3.2 removed good candidates for deduplication from the YFCC dataset. The class distribution in our filtered YFCC subset is very skewed. Of the 1000 ImageNet classes, 521 classes are not represented in the filtered subset's pseudolabels. This could be due to the method that we used to filter YFCC images. We selected images from YFCC that had the highest average predicted confidence. Our filtered subset could be limited to images that are easier for the trained ImageNet model to correctly classify. The ImageNet models that we used for filtering could output lower confidences for more fine-grained ImageNet classes, leading to their underrepresentation in the filtered YFCC subset. Figure 3-1 shows that trained ImageNet models could output low confidence values even on images present in their own training set. We could have even removed exact duplicates of ImageNet validation images from YFCC15M while performing filtering.

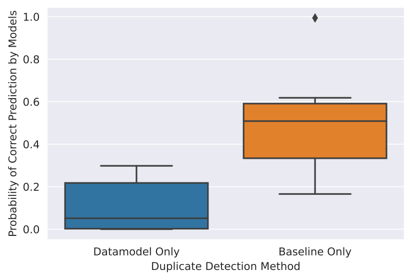
4.3.2 Suboptimal Tuning of Datamodel Hyperparameters

The hyperparameters we used for deduplicating YFCC15M and the ImageNet validation dataset could have been non-optimal. Specifically, we suspect that we used an incorrect α value, a hyperparameter controlling the size of each subset S_i on which we train each ImageNet model. We believe that we should decrease α and train on smaller subsets S_i. Quantitatively, we observe that datamodels are most effective at identifying near-duplicates of examples where the models struggle to make a correct prediction, which is in line with our intuitions. More easy examples tend to have larger data support in the subsampled sets S_i. We would then calculate lower influences between individual images within the data support and a query evaluation image, even if a duplicate exists. In Figure 4-4, we compare the probability that models correctly classify exclusively datamodel-detected duplicate images and exclusively baseline-detected duplicate images. We observe that models tend to have higher average accuracies on exclusively baseline-detected examples. By decreasing α, we can decrease the size of the data support for each evaluation image and perhaps allow the datamodel to detect duplicates that only the baseline can currently detect. Qualitatively, we observe that exclusively datamodel-detected duplicates indeed seem more difficult to classify. There appear to be more copies of the exclusively baseline-detected duplicates within the ImageNet validation dataset, implying that exclusively baseline-detected duplicate images are easier to classify and enjoy larger data support. The datamodel was able to identify duplicate pairs in Figure 4-6. There tend to be no copies of successfully detected duplicates within the model training dataset. Figure 4-5 shows that the datamodel does not successfully detect duplicates when there are multiple copies of the duplicate in the model training dataset. Decreasing α in turn decreases the probability that we sample a copy of a duplicate candidate while selecting the model training subset S_i, hence increasing the influence of the duplicate.

We expect that the resulting datamodels would be more sensitive to individual images in the training dataset if α were reduced, but reducing α comes with its own risks. Insufficient training data could have negative implications for the ability of our



(a) Seen During Training



(b) Unseen During Training

Figure 4-4: These figures show the probability that the trained models would correctly classify near-duplicate images. The datamodel tends to detect duplicates among samples that are more difficult for models to classify. The baseline tend to detect duplicates among samples that are easier for models to classify.

datamodel to detect near-duplicates. In a low data regime, it is possible that the trained ImageNet models would be unable to learn good features and exhibit high variance behaviors that do not depend on the presence of particular images in their training set. The calculated influence values from such ImageNet models would not be useful for detecting near-duplicate pairs.

However, experimenting with different values of α is computationally expensive. The most time-intensive step of applying datamodels is the model training step; experimenting with additional values of α would force us to train thousands of more models. Due to the difficulty of experimenting with α, datamodels could be a useful tool to use in conjunction with a distance-based embedding approach because they tend to find duplicates among different kinds of images. We can also attempt to reduce the time necessary for training models by using pretrained feature extractors during model training. This approach would decrease the amount of time necessary to train models for datamodel estimation. In addition, we could then safely set α to a low value without sacrificing the quality of the learned feature representation. However, this approach can lead to incorrect estimations of the true influence values since the pretrained feature extractor would have been trained on a larger set of examples than the model training set. We could also attempt to decrease the model's sensitivity to the α parameter by deduplicating the model training set with itself before datamodel training.

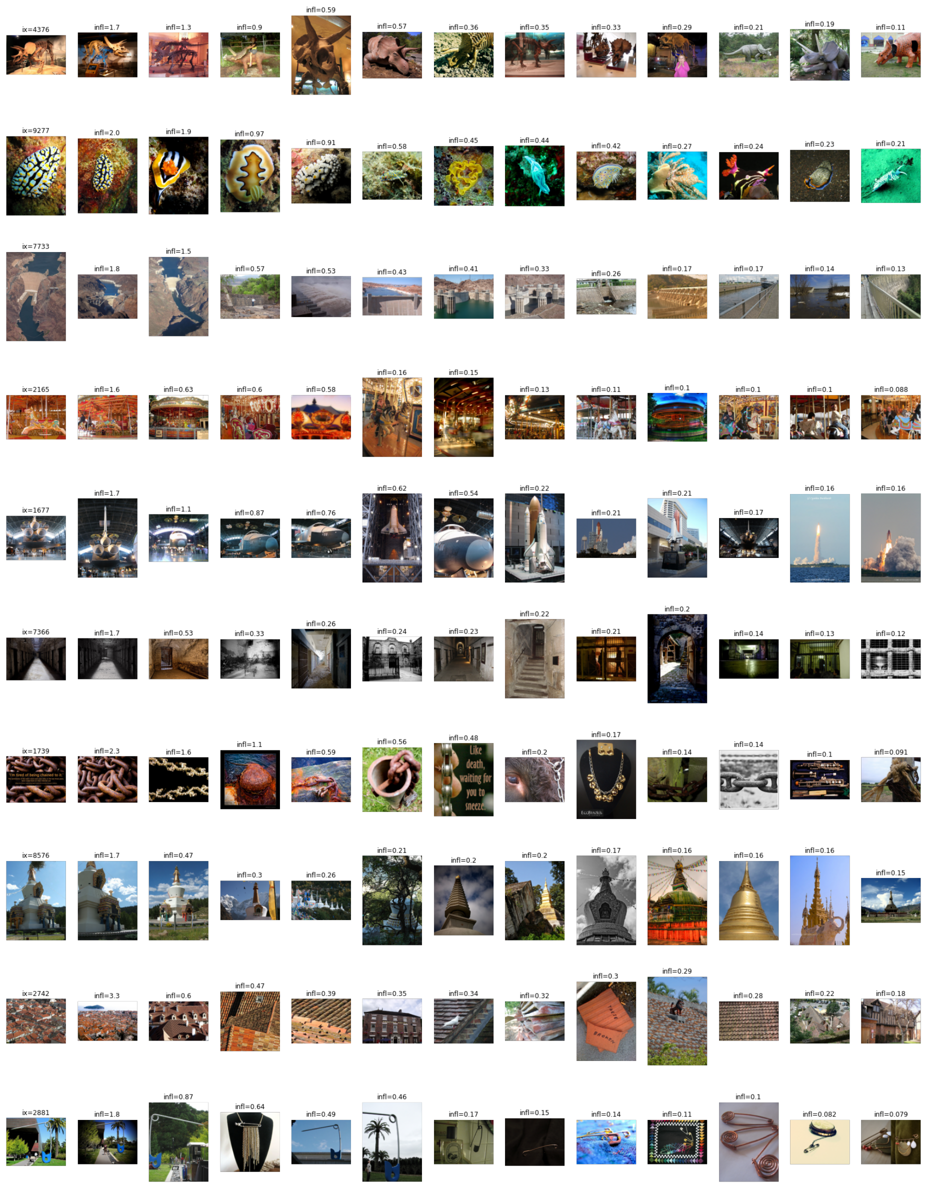


Figure 4-5: Highest ImageNet influencers for duplicates only detected by datamodel

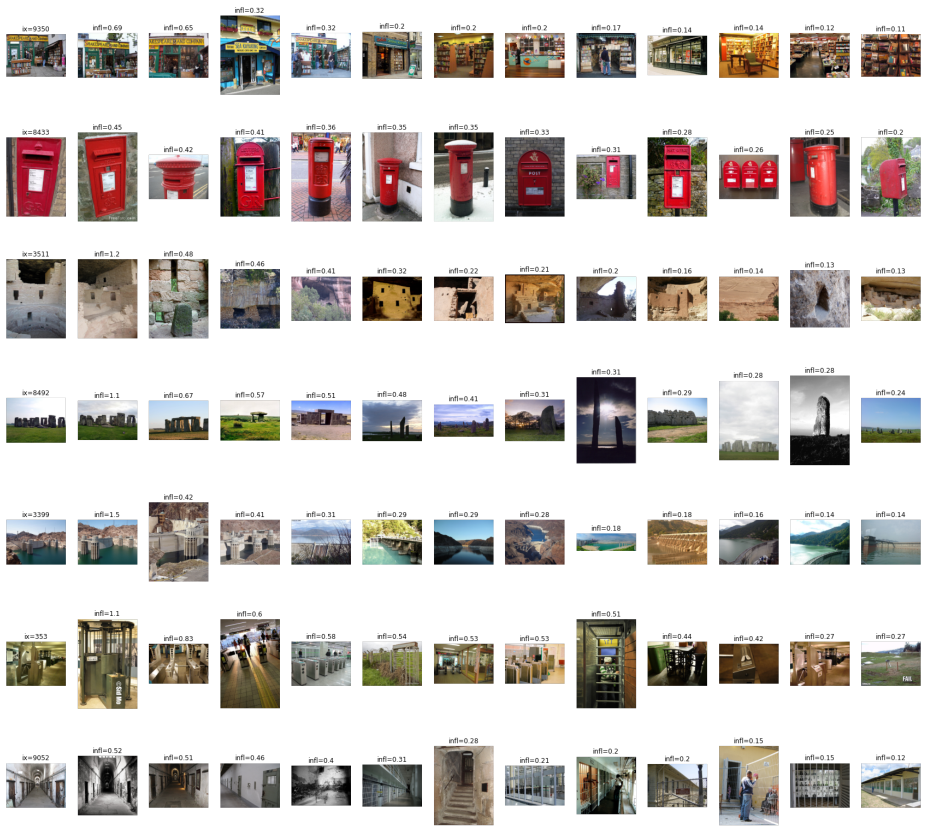
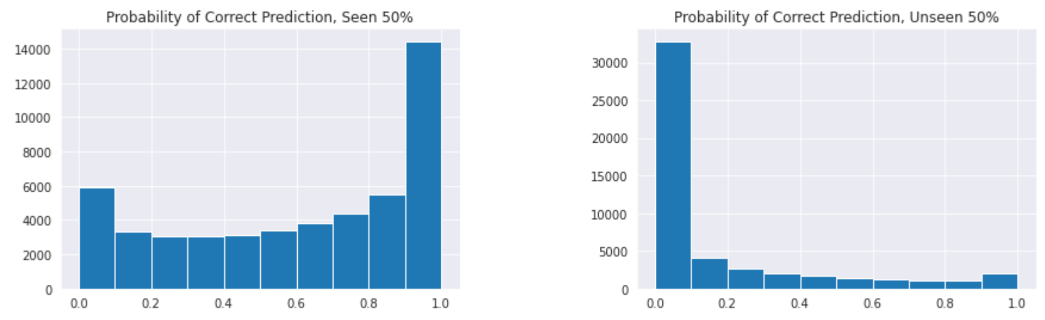


Figure 4-6: Highest ImageNet influencers for duplicates only detected by baseline



(a) Seen During Training (b) Unseen During Training

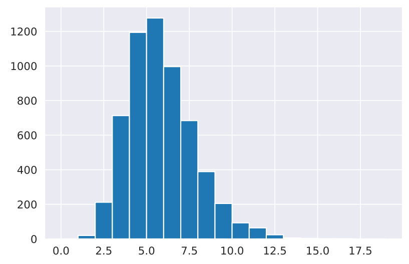
Figure 4-7: Probability of Correctly Classifying Image Per ImageNet Validation Example

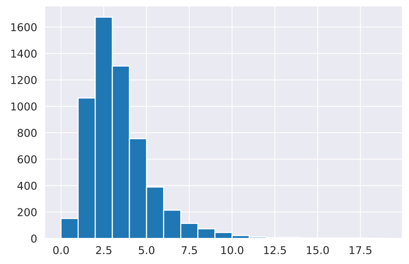
Although most examples in the model training set are "difficult", we believe that duplicates are more likely to be "easy" examples. We plot the probability of correctly predicting labels for each image in the model training set in Figure 4-7a. The models perform poorly on most of the examples not present in the training set as shown in Figure 4-7b, which suggests that the datamodel could correctly identify more duplicates than the baseline over the entire dataset. However, it is more likely that more train-val near-duplicates exist where the model performs well. Some images are more likely to be duplicated than others due to the nature of their contents. For instance, it's more likely for an image of a famous landmark to be duplicated than an image of a common item. If such an example is duplicated between the training and evaluation dataset, it is also more likely to be copied within the training dataset itself. Multiple copies of the same image within the training dataset could lead to a higher probability of correctly classifying the example, which in turn could lead the datamodel to be less effective at identifying the duplicate.

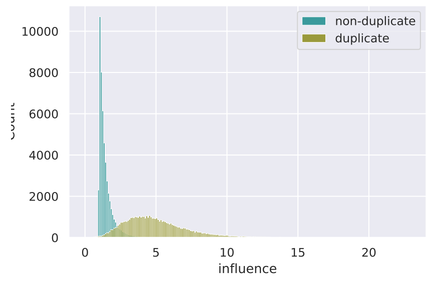
4.3.3 Naïve Selection of Duplicate Detection Thresholds

We observed many more false positives in the datamodels' top duplicate proposals than in the baseline's top proposals. This result may be due to the shortcomings of max influence as a heuristic. We can quantitively explain why there are many false positives in the datamodels' top proposals by computing influence values from the training dataset to itself. From Figure 4-8c, we notice an overlap in the range of influence values for duplicate and non-duplicate pairs in the training dataset.

It is possible that there are many duplicated images within the model training dataset; this would cause an overestimate in the influence values between non-duplicate pairs. Despite this possibility, there are still many images from the model training dataset that have very low self-influence values, which makes these images difficult to distinguish from non-duplicated images by influence values alone. This overlap suggests that the maximum influence value over the evaluation set could be a bad heuristic. Duplicates that have low self-influence scores would be unlikely to be proposed, even if they have a high degree of similarity.


(a) Images with Lowest Probability of Correct Prediction


(b) Images with Best Probability of Correct Prediction



(c) Influence Overlap Between Duplicates and Non-Duplicates

Figure 4-8: These figures show the value of the influence function from each ImageNet image to itself (an exact duplicate).

We could potentially decrease the datamodels' false positives by taking self-influence scores into account when choosing the top training images to manually inspect. We note that self-influence scores behave differently for the most challenging ImageNet examples compared to the easiest ImageNet examples. Figure 4-8a shows the self-influence values for the most difficult training examples, while 4-8b shows self-influence values for the easiest training examples. We could try to dynamically set duplicate thresholds based on the difficulty of a particular model training example. We should calibrate the influence values of each training-validation pair with the training image's self-influence value before ranking the top training images by max influence across the evaluation set.

4.3.4 Subjectivity of Near-Duplicate Detection

We end this section with a discussion on the nature of duplicativeness. First, duplicativeness can be subjective. Even different people can have different opinions of whether or not a pair of images are near-duplicates. Experts in a particular object domain can perhaps notice differences between pairs of images that an untrained eye would not detect (e.g., two dogs of the same breed). Deciding if a pair of images are near-duplicates or are just highly similar can cause disagreement even among people.

More importantly, human understanding of what makes two objects the same can be extremely different from a machine learning model's understanding. This difference in understanding could be due to the way that neural networks come to understand "sameness." The neural networks used in this work only have an understanding of "sameness" from training on various classification tasks. Humans have a much richer understanding of what makes an image one of a kind. For this reason, near duplicates as judged by humans could be irrelevant to how current models behave. Indeed, the high number of false positives proposed by both the embedding-based method and the datamodels indicate that near-duplicates do not contribute significantly more to a model's predictions compared other examples. Hence, duplicate examples may not be the most useful datapoints on which to analyze these models—at the end of the day, we mainly care about duplicates insofar as how they affect models.

Chapter 5

Future Work and Conclusion

In this work, we use datamodels to find duplicates between YFCC15M and the ImageNet validation set. We found that datamodels have the potential to be a useful tool for dataset deduplication, but some additional modifications are likely to improve their usefulness. Some notable results from this work include:

- **Deduplicating Unlabelled Datasets with Datamodels** - This is the first work that uses datamodels to deduplicate an unlabelled dataset. We use predictions from trained models to generate pseudolabels for the unlabelled dataset and to filter examples that are unlikely to be duplicated. We train a datamodel using these pseudolabels to detect duplicates.

- **Concrete Data on YFCC-ImageNet Overlap** - We present 31 instances of duplication between YFCC and ImageNet along with their ImageNet identifiers out of the 300 examples we examined. We do not believe that this result implies a high degree of overlap between YFCC15M and the ImageNet validation dataset; these 300 examples are the most likely candidates for duplication out of the filtered subset we examined. Although this is a relatively insignificant number in the context of the ImageNet and YFCC dataset size, these examples can be used to evaluate future deduplication attempts and to train future models.

- **Comparing Datamodels to Representation Baseline** - We find that datamodels do not significantly outperform the baseline. However, we believe that datamodels can be improved by applying some modifications.

- **Example Difficulty and Datamodel Duplicate Detection** - We find that the baseline can more easily find duplicates among easily-classified examples, while the datamodels can find more duplicates among examples that are more difficult to classify. This could imply that datamodels should be used in conjunction with feature representation models because they have different strengths.

- **Analysis of Self-Influence Scores** - We observe that self-influence scores can take on a wide range of values, and they are not always greater than influence scores between non-duplicates. We propose that duplicate proposals should be calibrated against self-influence scores.

Some recommendation we have for future work include:

- **Tuning of Datamodel Subsampling Parameter** - We observed some instances where the datamodel cannot detect duplicates due to the presence of duplicates within the ImageNet validation dataset itself. We propose that evaluation datasets should be deduplicated before applying datamodels or that the α parameter of the datamodel should be reduced for larger evaluation datasets.

- **More Efficient Datamodel Estimation** - We are unsure if this modification will lead to good influence estimates. However, using pretrained backbones can lead to faster model training, less compute requires, and less storage consumption. Also, this modification will enable us to better examine the effects of individual training examples.

- **Quantifying the Impact of Data Deduplication for CLIP Models** - It is not well-understood how train-val duplication affects CLIP models. Prior works indicate that train-val duplication does not affect CLIP performance on downstream tasks. However, it is possible that duplication does play a role in CLIP performance in ways that are currently not well-understood. Understanding if and how label leakage occurs in multimodal self-supervised learning could be an interesting line of work. It is also possible that the largest effects of data deduplication could be observed in semi-supervised models, which are trained on both labelled and unlabelled datasets. Prior work [2] show that semi-supervised models perform well even in the presence of label ambiguity. In addition, CLIP cannot outperform state of the art semi-supervised models trained to maximize ImageNet validation accuracy.

- **Deduplicating Additional Datasets** - Machine learning practitioners usually evaluate self-supervised models across a wide range of datasets. Self-supervised models usually cannot outperform semi-supervised models on ImageNet, but they can achieve better results across a wider range of tasks. There could be a high degree of overlap between YFCC15M and these other evaluation datasets.

- **Deduplication Within Training and Validation Sets** - We discovered several duplicates within the ImageNet validation set (see the first row of images in Figure 4-5) and within YFCC15M. Although train-val duplication is well-studied, we do not currently have a good understanding of how duplication within training and validation sets affect model training.

Bibliography

[1] Björn Barz and Joachim Denzler. Do we train on test data? purging cifar of near-duplicates. 2019.

[2] Lucas Beyer, Olivier J. Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. Are we done with imagenet?, 2020.

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher 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, 2020.

[4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers, 2021.

[5] Wei Chen, Yu Liu, Weiping Wang, Erwin Bakker, Theodoros Georgiou, Paul Fieguth, Li Liu, and Michael S. Lew. Deep learning for instance retrieval: A survey, 2021.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[7] Priya Goyal, Quentin Duval, Isaac Seessel, Mathilde Caron, Ishan Misra, Levent Sagun, Armand Joulin, and Piotr Bojanowski. Vision models are more robust and fair when pretrained on uncurated images without supervision, 2022.

[8] Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. Datamodels: Predicting predictions from training data, 2022.

[9] Simon Kornblith, Jonathon Shlens, and Quoc V. Le. Do better imagenet models transfer better?, 2018.

[10] Kimmo Kärkkäinen and Jungseock Joo. Fairface: Face attribute dataset for balanced race, gender, and age, 2019.

[11] Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better, 2021.

[12] Niv Nayman, Avram Golbert, Asaf Noy, Tan Ping, and Lihi Zelnik-Manor. Diverse imagenet models transfer better, 2022.

[13] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2018.

[14] Hieu Pham, Zihang Dai, Qizhe Xie, Minh-Thang Luong, and Quoc V. Le. Meta pseudo labels, 2020.

[15] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[16] Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. Do adversarially robust imagenet models transfer better? *CoRR*, abs/2007.08489, 2020.

[17] Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. 2015.

[18] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. Self-training with noisy student improves imagenet classification, 2019.

[19] Xiaohua Zhai, Xiao Wang, Basil Mustafa, Andreas Steiner, Daniel Keysers, Alexander Kolesnikov, and Lucas Beyer. Lit: Zero-shot transfer with locked-image text tuning, 2021.

Appendix A

Supplement

A.1 Tables

In Table A.1, we show the hyperparameters used to train the models in this work.

Model Type	LR Schedule Type	Learning Rate	LR Peak Epoch	Epochs	Momentum	Weight Decay
resnet18	cyclic	0.5	16	40	0.9	1e-3

Table A.1: Model Training Hyperparameters