Coreset Selection via LLM-based Concept Bottlenecks

Akshay Mehra[1*], Trisha Mittal[1*], Subhadra Gopalakrishnan[2], and Joshua Kimball[1]
[*]Equal Contribution [1]Dolby Laboratory [2]Apple
{akshay.mehra, trisha.mittal, joshua.kimball}@dolby.com

Abstract

Coreset Selection (CS) identifies a subset of training data that achieves model performance comparable to using the entire dataset. Many state-of-the-art CS methods, select coresets using scores whose computation requires training the downstream model on the entire dataset and recording changes in its behavior on samples as it trains (training dynamics). These scores are inefficient to compute and hard to interpret as they do not indicate whether a sample is difficult to learn in general or only for a specific model. Our work addresses these challenges by proposing an interpretable score that gauges a sample's difficulty using human-understandable textual attributes (concepts) independent of any downstream model. Specifically, we measure the alignment between a sample's visual features and concept bottlenecks, derived via large language models, by training a linear concept bottleneck layer and compute the sample's difficulty score using it. We then use this score and a stratified sampling strategy to identify the coreset.Through experiments on CIFAR-10, CIFAR-100, and ImageNet-1K, we show our coresets outperform random subsets, even at high pruning rates, and achieve model performance comparable to or better than coresets found by training dynamics-based methods.

1. Introduction

Machine learning (ML) pipelines are becoming increasingly intensive regarding their data and compute requirements [2, 59]. Coreset Selection (CS) [10, 17, 36, 37, 43, 65, 73] approaches have emerged to improve the efficiency of these pipelines, since they focus on pruning large datasets and retaining only a small subset of representative samples, making training more efficient. Many state-of-the-art (SOTA) CS methods use the downstream model's training dynamics — the model's behavior on a sample during training, to generate a difficulty score for each sample. This enables accurate estimation of the data difficulty but requires training the downstream model on the entire dataset at least once, which can be costly when training a large model on a large dataset. Moreover, these scores are difficult to interpret as they are downstream model-dependent and cannot inform if the sample will be hard or easy to learn for a different downstream model (without training it first). Thus, we aim to answer the following question *"How to efficiently estimate a sample's difficulty in an interpretable and model-agnostic way i.e. without the knowledge of the architecture or training dynamics of the downstream model?"*

To address this, we use Concept Bottleneck Models (CBMs) [28, 71] which are effective at making ML models more interpretable. CBMs map a model's input on to a set of human-understandable concepts, referred to as the "bottleneck" and then use them to make a prediction. However, CBMs require concept annotation for every sample in the training data, which can be costly to obtain. Recently [67, 70] showed that this limitation of CBMs can be overcome by leveraging the advances in Large Language Models (LLMs) and Vision Language Models (VLMs) that can be prompted to generate concept annotation for the training samples without requiring any task-specific fine-tuning. Our Fig. 1 (block 1) shows the prompt we used to obtain the concept bottleneck using an LLM.

Once the bottleneck is formed, we use a VLM (such as CLIP [45]) to measure the alignment between the visual features and the concept bottleneck (denoted as concept similarity in Fig. 1 (block 2)). A simple linear concept bottleneck layer is then trained to align the visual and concept features to make predictions on the training samples. We use the average margin of a training sample during the training of this bottleneck layer as our concept-based difficulty score. We use a stratified sampling [73] (block 3) using our score to form the coreset which can then be used to train various downstream models Fig. 1 (block 4). Our method speeds up the computation of the coreset by 15 times compared to approaches based on the downstream model's training dynamics across three benchmark datasets. Moreover, compared to coresets found by other training dynamics-free approaches our coresets lead to downstream models with significantly better accuracy, especially at high data pruning rates, and achieve performance close to SOTA CS methods. We also show that our approach is effective for the label-free CS problem where the dataset is unlabeled. Since

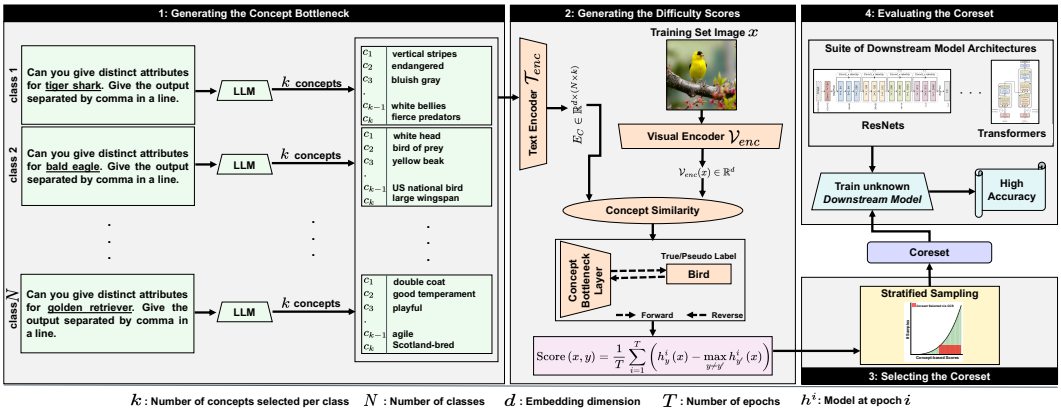


Figure 1. **: Overview of our approach:** We start by prompting an LLM to generate concept annotation for N class labels in the dataset and select k most discriminative attributes (per class) to form the concept bottleneck, which are passed through a text encoder ($\mathcal{T}_{enc}$) to obtain the bottleneck embedding matrix (E_C). The visual information of a training sample x extracted via the visual encoder ($\mathcal{V}_{enc}$) is then aligned to the embedding matrix using a linear concept bottleneck layer trained for T epochs. Our difficulty score for x is then computed as the average margin (i.e., the difference between the likelihood of the correct and other classes) over T epochs. Finally, a coreset is selected using stratified sampling which is then used to train an unknown downstream model.

our CS method is model agnostic, we show that our coreset leads to high performance regardless of the architecture of the downstream model, unlike other methods. Lastly, we show that our concept-based difficulty score is aligned with human intuition, allowing us to interpret why examples are easy/hard, independent of the downstream model.

2. Preliminaries

2.1. Coreset selection (CS) problem formulation

Consider a classification task and data distribution P. Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ denote the dataset of n training examples sampled i.i.d. from the distribution P where x_i denotes the data and $y_i \in \mathcal{Y}$ denotes the label from a set of N classes. CS [11, 73] aims to find a subset $\mathcal{S}$ of $\mathcal{D}$ consisting of $m \leq n$ samples such that the models trained on $\mathcal{S}$ achieve performance comparable to models trained on $\mathcal{D}$. Formally, the CS problem is defined as follows,

$$\min_{\mathcal{S}:|\mathcal{S}|=m} \mathbb{E}_{(x,y)\sim P}[\ell(x, y|\theta_{\mathcal{S}})] - \mathbb{E}_{(x,y)\sim P}[\ell(x, y|\theta_{\mathcal{D}})], \quad (1)$$

where $\theta_{\mathcal{D}}$ and $\theta_{\mathcal{S}}$ denote the "*downstream model*" trained on $\mathcal{D}$ and $\mathcal{S}$ (coreset), respectively and ℓ is the loss function.

To find this subset $\mathcal{S}$, previous works have proposed scores to gauge a sample's difficulty for a model, and are later used to form the coreset. These approaches use information from the training dynamics of the downstream model to estimate the a difficulty score. Scores such as the forgetting score [58] computed as the number of times a sample gets misclassified after being correctly classified earlier during model training and the area under the margin (AUM) [44] which identifies mislabeled/difficult samples are popular examples. While approaches based on the training dynamics of the downstream model have achieved SOTA results, the requirement of knowledge/training the downstream model or a relatively big proxy model on the entire dataset is inefficient for large datasets/models.

2.2. Concept bottleneck models (CBMs)

Recent advances in language model-guided CBMs utilize an LLM to obtain concept bottlenecks which are then used to predict the labels. These works rely on pre-trained multimodal models (such as CLIP [45]) which consists of a visual encoder $\mathcal{V}_{enc}$ and a text encoder $\mathcal{T}_{enc}$ that can map images and text to a d-dimensional representation space. Let $C = \{c_1, c_2, \cdots, c_{N_C}\}$ be the set of N_C concepts (bottleneck) generated via a LLM, we can then construct a bottleneck embedding matrix $E_C \in \mathcal{R}^{N_C \times d}$ such that each row of the matrix is mapping of the concept $c \in C$ after passing it through textual encoder $\mathcal{T}_{enc}$. Based on this, a CBM [70] produces a prediction $h(x) = f(g(\mathcal{V}_{enc}(x), E_C))$ for a sample x where $g : \mathbb{R}^d \to \mathbb{R}^{N_C}$ computes the similarity of the visual features to each concept in the bottleneck and $f : \mathbb{R}^d \to \Delta$ outputs the probability of each class in the label set $\mathcal{Y}$, where Δ is a N simplex.

3. Methodology

3.1. Generating the concept bottleneck via LLMs

Since obtaining data with concept annotation is costly we use LLMs to generate concept annotation for the samples. However, generating attributes (word-level concepts) for all the samples in the dataset via LLMs is still costly, hence we generate attribute-level concepts only for class label names. This approach was recently shown to be effective at generating the concept bottleneck for interpretable image

Table 1. Comparison of the model's (RN-18 for CIFAR10/100 and RN-34 for Imagenet) test accuracy after training on coresets found by various approaches shows that our coresets lead to significantly better performance than Random achieving competitive results compared to the methods using the downstream model's training dynamics, even for high pruning rates. (Best in each category is highlighted).

	Method	Datasets and Pruning Rates											
		CIFAR-10				*CIFAR-100*				*Imagenet*			
		30%	50%	70%	90%	30%	50%	70%	90%	30%	50%	70%	90%
Needs Training Dynamics	Entropy	94.44	92.11	85.67	66.52	72.26	63.26	50.49	28.96	72.34	70.76	64.04	39.04
	Forgetting	**95.40**	**95.04**	92.97	85.70	**77.14**	**74.45**	**68.92**	**55.59**	**72.60**	**70.89**	66.51	52.28
	AUM	95.27	94.93	**93.00**	**86.08**	76.84	73.77	68.85	55.03	72.29	70.52	**67.78**	**57.36**
Doesn't Need Training Dynamics	Random	94.33	93.40	90.94	79.08	74.59	71.07	65.30	44.76	72.18	70.34	66.67	52.34
	$\text{Random}_{\text{FFCV}}$	-	-	-	-	-	-	-	-	73.37 ±0.08	71.71 ±0.10	67.85 ±0.04	51.29 ±0.20
	Ours	**94.77** ±0.09	**93.44** ±0.61	**91.80** ±0.21	**84.63** ±0.24	**75.98** ±0.26	**72.22** ±0.22	**66.53** ±0.42	**51.85** ±0.29	**73.39** ±0.12	**72.34** ±0.13	**69.44** ±0.17	**55.92** ±0.02

classification [67, 70]. In Figure 1 (block 1), we present the prompts provided to the LLMs to extract the concepts for various class label names. The responses of the LLM are then processed to remove formatting errors to obtain the concept sets. Details of our prompt design, robustness check of different prompts, and examples of the generated attributes are mentioned in App. F. Once the per-class concepts are extracted, we select k discriminative concepts per class (concepts that are unique to a class) to form the concept bottleneck. Our final list of k concepts for a class contains its name and $k-1$ attributes generated by the LLM. In App. E we show an ablation of using different methods to obtain the concept sets, and k.

3.2. Concept-based score for CS

Next, we describe how to use the concept bottleneck to produce a difficulty score for the samples in the dataset. We start by discussing how we learn the functions f and g described in Sec. 2.2 (see Fig. 1(block 2) for an overview of the method). We use the dot product between the visual embeddings of an image x denoted as $\mathcal{V}_{enc}(x)$ and the bottleneck embedding matrix E_C to measure the alignment between the visual and textual features of the concepts [67, 70]. The concept similarity score is computed as

$$g(x, E_C) := \mathcal{V}_{enc}(x) \cdot E_C^{\intercal}. \tag{2}$$

To map the concept similarity scores to predictions in the label space $\mathcal{Y}$, we propose to use a linear (concept bottleneck layer) predictor as f. Concretely, the function f with parameters $W \in \mathbb{R}^{N \times N_C}$ is given by $f(x; W) := g(x, E_C) \cdot W^{\intercal}$. We learn the parameters W using

$$W^* = \arg\min_W \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i; W), y_i), \tag{3}$$

where $\ell(f(x; W), y) = -\log(f(x; W)_y)$ is the cross-entropy loss. We define the concept bottleneck layer's output, $h(x) := f(g(x, E_C); W^*)$. In practice, we learn the optimal W using mini-batch gradient descent by running the optimization for T epochs. To gauge the difficulty of each training sample we use the area under the margin (AUM) [44] while solving Eq. 3 which quantifies the data difficulty as the margin of a training sample averaged over T training epochs. Concretely, the margin of a sample (x, y) at a training epoch t is $M^t(x, y) = h_y^t(x) - \max_{y' \neq y} h_{y'}^t(x)$, where $h_{y'}^t(x)$ is the prediction likelihood of the bottleneck layer at epoch h^t for class y'. Thus, AUM (concept-based score) is computed as

$$\text{AUM}(x, y) = \frac{1}{T} \sum_{t=1}^{T} M^t(x, y). \tag{4}$$

Recent works [44, 73, 74] have demonstrated the effectiveness of AUM for gauging the sample's difficulty for coreset selection. However, [73, 74] requires computing AUM for the downstream model by training it on the entire dataset once, which is computationally costly. On the other hand, our method integrates AUM with the training of the linear concept bottleneck layer h, which is computationally cheaper (training a linear layer takes only 7 minutes for Imagenet compared to 8 hours for training a ResNet-34) than training the downstream model (θ).

Sampling training examples to form a coreset. After obtaining data difficulty scores, a crucial step is choosing the samples to form the coreset. While many previous works [11, 58] have reported encouraging results keeping the most challenging samples (for our concept-based score this means samples with the smallest margin), recent works [55, 73] have shown that this could lead to a catastrophic drop in accuracies after training the downstream model on the coreset, especially when the size of the coreset is small. This is mainly due to poor sample coverage and potentially mislabeled data in the datasets. To remedy this, we use Coverage-centric Coreset Selection (CCS) proposed by [73] (see Alg. 1 in App. G.3) which uses a stratified sampling approach and filters out (potentially) mislabeled samples to form the coreset.

4. Experiments

Datasets, models, and training: We focus on CS for classification tasks on three benchmark datasets; CIFAR-10,

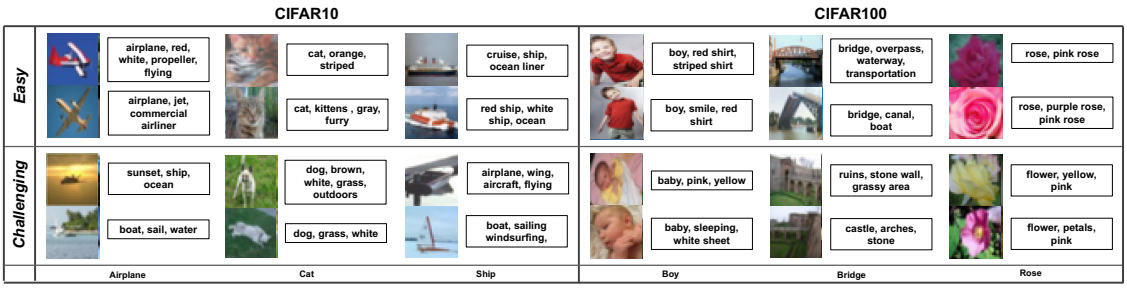


Figure 2. Visualizing samples according to our concept-based score for a subset of classes in CIFAR-10/100 showing that easy (challenging) samples are aligned (unaligned) with their assigned label. Image-level concepts (in boxes) extracted via LLaVA confirm that easy (challenging) examples are aligned (unaligned) with concepts of their labels, explaining the reason for a high (low) concept-based score.

CIFAR-100 [29], and Imagenet-1K [13] datasets consisting of 50000, 50000, and 1.28 million samples spread across 10, 100, and 1000 classes, respectively. We also report results on Emotion recognition and biomedical image recognition tasks using the Affectnet [38] and BloodMNIST [1] datasets. We report the results for different pruning rates where a pruning rate of 90% refers to removing 90% of the samples from the original training dataset (additional details in Appendix G.1). We compare our method against various baselines and SOTA methods proposed by previous works (explained in App. G.2).

4.1. Evaluating performance on CS

Table 1 shows the accuracy of models trained on coresets found by various approaches on the test sets of the three datasets for the standard CS problem (where the dataset is labeled). We find that coresets found by our approach lead to significantly better performance, even at higher pruning rates, compared to the random subsets. Moreover, our method which does not use any information about the training dynamics of the downstream models provides competitive performance to coresets found by the SOTA approaches based on forgetting score and AUM, and even outperforms them on Imagenet for smaller pruning rates.

To test coreset performance of our method on emotion recognition, we used the EfficientNet model [57] and report F1 scores for our coresets in Table 2. When compared against randomly selected coresets, coresets selected via our concept-based approach achieve better performance. To evaluate the coreset performance on BloodMNIST, we use a ResNet-18 model and report accuracy of our coresets in Table 3. Our method achieves better performance than random for higher pruning rates and is competitive at lower pruning rates. In App. B, we show that our approach can be extended to label-free datasets as well.

4.2. Visualizing Concept-based Scores

In Fig. 2, the top row shows the images with the highest concept-based scores (easiest) and the bottom shows the images with the lowest scores (challenging) for a subset of classes in CIFAR-10/100. As observed the easiest images are typical images associated with the label where as the challenging images are confusing (and even potentially mislabeled) as they look like images from a different class. For example, some challenging images in the class "boy" from CIFAR-100 are actually images of a baby which is also a class in CIFAR-100. More examples of such images are presented in Fig. 3 in App. D. Since the challenging examples are confusing for humans too our score is well aligned with the human-perceived difficulty of the samples.

Next, we demonstrate why certain samples get low/high concept-based scores in our approach by extracting attributes specific to these images using LLaVA (note that these attributes are different from the per-class concepts used in the concept bottleneck). To generate these, we prompt LLavA to produce concepts using both the sample image and its class label (see image-level concept extraction in App. F). These image-level attributes are shown in the boxes in Fig. 2. As observed in Fig. 2, image-level attributes provided by LLaVA are related to the class label for easy images whereas they are unrelated for challenging images. For example, attributes provided by LLaVA for the challenging images of "airplane" align more with those of a "ship" , and concepts provided for challenging images of "bridges" align more with those of "castles". Since our score also assigns a small value for these images, our concept-based score in Eq. 4 can correctly capture when the visual information in the sample is not aligned with the associated label of the sample and vice-versa.

5. Conclusion

In this work, we proposed a coreset selection method based on scores obtained via concept bottlenecks that allow us to gauge the difficulty of a sample in terms of human-understandable concepts and is independent of the downstream model. Our experiments show that training downstream models on these coresets lead to better performance than random subsets and achieves accuracy similar to or better than the SOTA approaches.

References

[1] Andrea Acevedo, Anna Merino, Santiago Alférez, Ángel Molina, Laura Boldú, and José Rodellar. A dataset of microscopic peripheral blood cell images for development of automatic recognition systems. *Data in brief*, 30:105474, 2020. 4, 11

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 1

[3] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6541–6549, 2017. 8

[4] Usha Bhalla, Alex Oesterling, Suraj Srinivas, Flavio P Calmon, and Himabindu Lakkaraju. Interpreting clip with sparse linear concept embeddings (splice). *arXiv preprint arXiv:2402.10376*, 2024. 11

[5] Sebastian Bujwid and Josephine Sullivan. Large-scale zero-shot image classification from rich and diverse textual descriptions. *arXiv preprint arXiv:2103.09669*, 2021. 8

[6] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924, 2020. 12

[7] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 12

[8] Samprit Chatterjee and Ali S Hadi. Influential observations, high leverage points, and outliers in linear regression. *Statistical science*, pages 379–393, 1986. 8

[9] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. *Advances in neural information processing systems*, 32, 2019. 8

[10] Hoyong Choi, Nohyun Ki, and Hye Won Chung. Bws: Best window selection based on sample scores for data pruning across broad ranges. *arXiv preprint arXiv:2406.03057*, 2024. 1

[11] Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Selection via proxy: Efficient data selection for deep learning. *arXiv preprint arXiv:1906.11829*, 2019. 2, 3, 12

[12] Aron Culotta and Andrew McCallum. Reducing labeling effort for structured prediction tasks. In *AAAI*, pages 746–751, 2005. 8

[13] 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. 4

[14] Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 569–578, 2011. 8

[15] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca, and projective clustering. *SIAM Journal on Computing*, 49(3):601–657, 2020. 8

[16] Brent A Griffin, Jacob Marks, and Jason J Corso. Zero-shot coreset selection: Efficient pruning for unlabeled data. *arXiv preprint arXiv:2411.15349*, 2024. 8

[17] Chengcheng Guo, Bo Zhao, and Yanbing Bai. Deepcore: A comprehensive library for coreset selection in deep learning. In *International Conference on Database and Expert Systems Applications*, pages 181–195. Springer, 2022. 1, 8

[18] Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. Generating visual explanations. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 3–19. Springer, 2016. 8

[19] Lisa Anne Hendricks, Ronghang Hu, Trevor Darrell, and Zeynep Akata. Grounding visual explanations. In *Proceedings of the European conference on computer vision (ECCV)*, pages 264–279, 2018. 8

[20] Evan Hernandez, Sarah Schwettmann, David Bau, Teona Bagashvili, Antonio Torralba, and Jacob Andreas. Natural language descriptions of deep visual features. In *International Conference on Learning Representations*, 2021. 8

[21] Lingxiao Huang, Shaofeng Jiang, and Nisheeth Vishnoi. Coresets for clustering with fairness constraints. *Advances in neural information processing systems*, 32, 2019. 8

[22] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, 2021. If you use this software, please cite it as below. 12

[23] Kaggle. Affectnet-trainingdata. https://www.kaggle.com/datasets/noamsegal/affectnet-training-data. [Accessed 24-03-2025]. 11

[24] Jihyung Kil and Wei-Lun Chao. Revisiting document representations for large-scale zero-shot learning. *arXiv preprint arXiv:2104.10355*, 2021. 8

[25] Krishnateja Killamsetty, Sivasubramanian Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *International Conference on Machine Learning*, pages 5464–5474. PMLR, 2021. 8

[26] Jinkyu Kim, Anna Rohrbach, Trevor Darrell, John Canny, and Zeynep Akata. Textual explanations for self-driving vehicles. In *Proceedings of the European conference on computer vision (ECCV)*, pages 563–578, 2018. 8

[27] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017. 8

[28] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pages 5338–5348. PMLR, 2020. 1

[29] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. ., 2009. 4

[30] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE transactions on pattern analysis and machine intelligence*, 36(3):453–465, 2013. 8

[31] Guillaume Leclerc, Andrew Ilyas, Logan Engstrom, Sung Min Park, Hadi Salman, and Aleksander Madry. Ffcv: Accelerating training by removing data bottlenecks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12011–12020, 2023. 11, 12

[32] David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Machine learning proceedings 1994*, pages 148–156. Elsevier, 1994. 8

[33] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning, 2023. 11

[34] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023. 10, 11

[35] Zhuoming Liu, Hao Ding, Huaping Zhong, Weijia Li, Jifeng Dai, and Conghui He. Influence selection for active learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9274–9283, 2021. 8

[36] Adyasha Maharana, Prateek Yadav, and Mohit Bansal. D2 pruning: Message passing for balancing diversity and difficulty in data pruning. *arXiv preprint arXiv:2310.07931*, 2023. 1, 8, 9

[37] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, pages 6950–6960. PMLR, 2020. 1, 8

[38] Ali Mollahosseini, Behzad Hasani, and Mohammad H Mahoor. Affectnet: A database for facial expression, valence, and arousal computing in the wild. *IEEE Transactions on Affective Computing*, 10(1):18–31, 2017. 4, 11

[39] Jesse Mu and Jacob Andreas. Compositional explanations of neurons. *Advances in Neural Information Processing Systems*, 33:17153–17163, 2020. 8

[40] Meike Nauta, Ron Van Bree, and Christin Seifert. Neural prototype trees for interpretable fine-grained image recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14933–14943, 2021. 8

[41] Kosuke Nishida, Kyosuke Nishida, and Shuichi Nishioka. Improving few-shot image classification using machine-and user-generated natural language descriptions. *arXiv preprint arXiv:2207.03133*, 2022. 8

[42] Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Anna Rohrbach, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. Multimodal explanations: Justifying decisions and pointing to the evidence. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8779–8788, 2018. 8

[43] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. *Advances in neural information processing systems*, 34:20596–20607, 2021. 1, 8

[44] Geoff Pleiss, Tianyi Zhang, Ethan Elenberg, and Kilian Q Weinberger. Identifying mislabeled data using the area under the margin ranking. *Advances in Neural Information Processing Systems*, 33:17044–17056, 2020. 2, 3, 8, 12

[45] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1, 2, 9, 12

[46] Karsten Roth, Oriol Vinyals, and Zeynep Akata. Integrating language guidance into vision-based deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16177–16189, 2022. 8

[47] Olga Russakovsky and Li Fei-Fei. Attribute learning in large-scale datasets. In *Trends and Topics in Computer Vision: ECCV 2010 Workshops, Heraklion, Crete, Greece, September 10-11, 2010, Revised Selected Papers, Part I 11*, pages 1–14. Springer, 2012. 8

[48] Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In *International conference on learning representations*, 2018. 8

[49] Andrea Schioppa, Polina Zablotskaia, David Vilar, and Artem Sokolov. Scaling up influence functions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8179–8186, 2022. 8

[50] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 8

[51] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017. 8

[52] Sheng Shen, Chunyuan Li, Xiaowei Hu, Yujia Xie, Jianwei Yang, Pengchuan Zhang, Zhe Gan, Lijuan Wang, Lu Yuan, Ce Liu, et al. K-lite: Learning transferable visual models with external knowledge. *Advances in Neural Information Processing Systems*, 35:15558–15573, 2022. 8

[53] Chandan Singh, John X Morris, Jyoti Aneja, Alexander M Rush, and Jianfeng Gao. Explaining data patterns in natural language with language models. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 31–55, 2023. 8

[54] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017. 8

[55] Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536, 2022. 3, 8, 9, 12

[56] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI conference on artificial intelligence*, 2017. 8

[57] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 4

[58] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018. 2, 3, 8, 12

[59] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 1

[60] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016. 8

[61] Yan Wang, Wei Song, Wei Tao, Antonio Liotta, Dawei Yang, Xinlei Li, Shuyong Gao, Yixuan Sun, Weifeng Ge, Wei Zhang, et al. A systematic review on affective computing: Emotion models, databases, and recent advances. *Information Fusion*, 83:19–52, 2022. 11

[62] Sandareka Wickramanayake, Wynne Hsu, and Mong Li Lee. Explanation-based data augmentation for image classification. *Advances in neural information processing systems*, 34:20929–20940, 2021. 8

[63] Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, et al. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7959–7971, 2022. 9

[64] Qi Wu, Chunhua Shen, Lingqiao Liu, Anthony Dick, and Anton van den Hengel. What value do explicit high level concepts have in vision to language problems?, 2016. 8

[65] Xiaobo Xia, Jiale Liu, Jun Yu, Xu Shen, Bo Han, and Tongliang Liu. Moderate coreset: A universal method of data selection for real-world data-efficient deep learning. In *The Eleventh International Conference on Learning Representations*, 2022. 1

[66] Wenjia Xu, Yongqin Xian, Jiuniu Wang, Bernt Schiele, and Zeynep Akata. Attribute prototype network for zero-shot learning. *Advances in Neural Information Processing Systems*, 33:21969–21980, 2020. 8

[67] An Yan, Yu Wang, Yiwu Zhong, Chengyu Dong, Zexue He, Yujie Lu, William Yang Wang, Jingbo Shang, and Julian McAuley. Learning concise and descriptive attributes for visual recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3090–3100, 2023. 1, 3, 8, 11, 12

[68] Jiancheng Yang, Rui Shi, and Bingbing Ni. Medmnist classification decathlon: A lightweight automl benchmark for medical image analysis. In *IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pages 191–195, 2021. 11

[69] Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification. *Scientific Data*, 10(1):41, 2023. 11

[70] Yue Yang, Artemis Panagopoulou, Shenghao Zhou, Daniel Jin, Chris Callison-Burch, and Mark Yatskar. Language in a bottle: Language model guided concept bottlenecks for interpretable image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19187–19197, 2023. 1, 2, 3, 8, 10, 12

[71] Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc concept bottleneck models. *arXiv preprint arXiv:2205.15480*, 2022. 1, 8

[72] Tian Yun, Usha Bhalla, Ellie Pavlick, and Chen Sun. Do vision-language pretrained models learn composable primitive concepts? *arXiv preprint arXiv:2203.17271*, 2022. 8, 12

[73] Haizhong Zheng, Rui Liu, Fan Lai, and Atul Prakash. Coverage-centric coreset selection for high pruning rates. *arXiv preprint arXiv:2210.15809*, 2022. 1, 2, 3, 12, 14

[74] Haizhong Zheng, Elisa Tsai, Yifu Lu, Jiachen Sun, Brian R Bartoldson, Bhavya Kailkhura, and Atul Prakash. Elfs: Enhancing label-free coreset selection via clustering-based pseudo-labeling. *arXiv preprint arXiv:2406.04273*, 2024. 3, 8, 9, 12

Appendix

We present additional related work in Appendix A followed by experimental results on label-free CS in Appendix B and experiments on transferability and efficiency of our approach in Appendix C. Then we visualize easy/hard images in Appendix D followed by ablation studies in Appendix E and describe the details of our methodology for extracting the concepts from LLaVA in Appendix F. Finally, we present additional experimental and implementation details of our in Appendix G including the algorithm for stratified sampling used in our work in Appendix G.3. We conclude in Appendix H with limitations and future work.

A. Related Work

Coreset selection (CS): CS improves the efficiency of model training by selecting a subset of influential samples. Various approaches have been proposed to generate such a subset [17]. A popular approach uses influence functions [8, 27, 35, 49] which measures the influence of a sample by considering the effect of removing it from the model's training. While effective, these approaches are computationally costly due to their dependence on higher-order derivatives. Another set of approaches selects a subset by either matching the gradients to those computed on the entire dataset [25, 37] or uses training dynamics of a model [12, 32, 43, 44, 58] to compute the importance of a sample. However, such approaches require repeated training of the downstream model to produce accurate importance scores. In comparison, our approach avoids using any knowledge of the downstream model for computing the difficulty scores. Finally, using the dataset's geometric properties via clustering is another popular choice for CS [14, 15, 21, 51, 55]. However, the high computational complexity due to their dependence on pairwise distances between the samples prohibits their use on large datasets.

Concept-based interpretability approaches: Concepts are defined as high-level semantics that refers to the abstract and human-interpretable meanings of the visual data, such as objects, actions, and scenes, as opposed to low-level features like edges or textures [64]. Concepts have been used in interpretable computer vision to bridge the gap between human understanding and machine perception in various tasks such as image classification. Such interpretability methods can be broadly classified as *post-hoc methods* (do not impose any model constraints) or *by-design* methods.

Post-hoc methods include Gradient-weighted Class Activation Mapping approaches [3, 20, 39, 50] that trace network gradients to identify the input areas that guide predictions and Explanation Generation methods [18, 26, 41, 53] that require models to produce explanations for visual tasks by conditioning their predictions on captioning models or incorporating visual evidence to ground explanations [19, 42]. Interpretable-by-design methods, such as Prototype methods, optimize a metric space where classifications are based on distances to class prototypes, identifying important input regions but often obscuring their semantic content [9, 40, 48, 54, 60].

Concept Bottleneck Models (CBMs) extend interpretable-by-design approaches by using human-understandable attributes as an intermediate layer for predictions, as used in few-shot learning [30] and attribute learning [47, 66]. A recent advancement, Computational Derivation Learning (CompDL), utilizes a CBM architecture by applying a linear layer over CLIP scores between expert-designed concepts and images, improving evaluation of how well CLIP grounds concepts [72]. While interpretable, CBMs reliance on costly annotations and lower accuracy compared to end-to-end models limit their usage.

Post-hoc Concept Bottleneck Models (PCBMs) address these issues by incorporating static knowledge bases (e.g., ConceptNet [56]) and residual connections to boost accuracy [71]. High-level semantic-driven descriptions are also used to guide data augmentation to build an informative set [62] to make model training efficient with a good enough training set. Prior works use external knowledge bases to obtain these textual semantic concepts to guide vision models [5, 24, 46, 52]. Recently [67, 70] incorporated LLMs to identify the concept bottleneck to make classification more explainable. We build on this literature and use CBMs for CS.

Table 2. Performance of concept-based coreset selection for emotion recognition task, Affectnet dataset. Our approach's coresets result in higher F1 scores compared to Random for standard CS.

Model Arch.	Method	Pruning Rates			
		30%	50%	70%	90%
EfficientNet	Random	$0.607_{\pm 0.006}$	$0.573_{\pm 0.025}$	$0.517_{\pm 0.006}$	$0.347_{\pm 0.055}$
	Ours	$0.603_{\pm 0.006}$	$0.577_{\pm 0.021}$	$0.537_{\pm 0.015}$	$0.450_{\pm 0.035}$

Table 3. Performance of concept-based coreset selection on biomedical image recognition (BloodMNIST dataset). CS selected by our approach results in superior accuracy compared to Random for standard CS.

Model Arch.	Method	Pruning Rates			
		30%	50%	70%	90%
ResNet-18	Random	$94.99_{\pm 0.27}$	$94.79_{\pm 0.18}$	$91.71_{\pm 0.29}$	$86.50_{\pm 0.66}$
	Ours	$94.64_{\pm 0.60}$	$94.06_{\pm 0.10}$	$92.26_{\pm 0.19}$	$87.44_{\pm 1.46}$

B. Concept-based score for label-free CS

Recently, there has also been an interest [16, 36, 74] in identifying the representative samples from an unlabeled dataset

Table 4. Comparison of the model's (RN-18 for CIFAR10/100 and RN-34 for Imagenet) test accuracy after training on coresets, found in a **label-free** manner, shows that our coresets lead to better performance than Random and those found by Prototypicality and is competitive/better than the coresets found by ELFS (which is dependent on the training dynamics of the downstream model).

	Method	Datasets and Pruning Rates											
		CIFAR-10				*CIFAR-100*				*Imagenet*			
		30%	50%	70%	90%	30%	50%	70%	90%	30%	50%	70%	90%
Needs Training Dynamics	ELFS (SwAV)	95.00	94.30	91.80	82.50	76.10	72.10	65.50	49.80	73.20	71.40	66.80	53.40
	ELFS (DINO)	**95.50**	**95.20**	**93.20**	**87.30**	**76.80**	**73.60**	**68.40**	**54.90**	**73.50**	**71.80**	**67.20**	**54.90**
Doesn't Need Training Dynamics	Prototypicality	94.70	92.90	90.10	70.90	74.50	69.80	61.10	32.10	70.90	60.80	54.60	30.60
	Random	94.33	93.40	90.94	79.08	74.59	71.07	65.30	44.76	72.18	70.34	66.67	52.34
	$\text{Random}_{\text{FFCV}}$	-	-	-	-	-	-	-	-	73.37 ±0.08	71.71 ±0.10	67.85 ±0.04	51.29 ±0.20
	Ours-LF	**94.81** ±0.14	**93.93** ±0.13	**91.75** ±0.34	**84.02** ±0.44	**74.67** ±0.23	**72.07** ±0.58	**65.50** ±0.17	**49.91** ±0.96	**73.61** ±0.08	**71.99** ±0.05	**68.42** ±0.21	**53.21** ±0.06

such that 1) we reduce the samples that need to be labeled by humans and 2) we can improve the efficiency of model training by only training the model on a subset of data. Our concept-based score can also be effectively utilized for this task with a simple modification. Similar to previous works [36, 55, 74], we assume that we know the number of classes in the datasets. Additionally, we assume that we also know the names of the classes in the datasets. Previous works have demonstrated that VLMs such as CLIP [45] achieve excellent zero-shot performance without requiring fine-tuning on specific datasets. We leverage this capability of CLIP models to obtain pseudo-labels for the images in our unlabeled dataset and use them to obtain our difficulty score for each sample

$$\text{AUM}(x, y_{\texttt{pseudo}}) = \frac{1}{T}\sum_{t=1}^{T} M^t(x, y_{\texttt{pseudo}}), \qquad (5)$$

where for an image x in the dataset, $y_{\texttt{pseudo}} = \arg\max_{j \in \mathcal{Y}} \mathcal{V}_{enc}(x) \cdot \mathcal{W}^{\intercal}_{\texttt{zeroshot}}$ where $\mathcal{W}_{\texttt{zeroshot}} \in \mathbb{R}^{N \times d}$ is a matrix with columns defined as $\mathcal{T}_{enc}(s_j)$ and $s_j =$ "a photo of a $\{j^{th}$ `class name`$\}$" for each class $j \in \mathcal{Y}$ [45, 63]. We use these scores along with CCS to produce the coreset. Similar to [36, 74], this coreset is then assumed to be annotated by humans and downstream models (θ) are trained on this annotated coreset.

Evaluating performance on label-free CS: Table 4 shows the accuracy of models trained on coresets when the training set lacks labels (we report the numbers presented by [74] for all the methods and baselines). The results show that the random subsets are a competitive baseline and even it outperforms Prototypicality [55]. Our results also show that our coresets outperform the random subsets for all pruning rates with the improvements being the most significant at higher pruning rates. Compared to ELFS [74], our method provides competitive performance and even surpasses it for lower pruning rates on Imagenet, without using any information about the downstream model's architecture or its training dynamics. Thus, our method is effective for this problem with a simple modification of utilizing the zero-shot predictions from CLIP as pseudo-labels.

C. Transferability and Efficiency of CS

Here we evaluate the performance of training downstream models with three different architectures to highlight the effectiveness of our approach for model-agnostic CS. Table 5 shows superior performance than random for ResNet-18, ResNet-34, and ResNet-50 models trained on our coresets for standard and label-free CS for various pruning rates.

Next, we compare the efficiency of our approach at finding coresets compared to approaches relying on training the downstream model on the entire dataset to obtain the training dynamics-based score. Using two A-100 GPUs, our approach can find the coreset in approximately 30 minutes for the Imagenet dataset giving a 15x speed up over training dynamics-based approaches. We obtain a similar speed up for CIFAR-10/100 where our method finds the coreset in less than 2 minutes.

Moreover, since our method is agnostic to the architecture of the downstream model we do not need to repeat the CS step for different architectures, unlike other methods which need to train the downstream model with new model architecture to find the best coreset for it.

D. Visualizing easy/challenging samples based on concept-based score

Similar to Fig 2 in Sec. 4.2 of the main paper, we visualize easy and challenging examples in Fig. 3 for CIFAR-10 and subset of classes from CIFAR-100. As observed the easy images (the ones that get high scores in our approach) are more canonical images of the class labels whereas the challenging ones are images that can potentially be assigned another class in the same dataset or are mislabeled in the dataset. The clear distinctions between these images show

Table 5. Superior performance of downstream models with different architectures trained on our coresets for Imagenet compared to Random for standard (Ours) and label-free (Ours-LF) CS highlight our method's effectiveness for downstream model-agnostic CS.

Model Arch.	Method	Pruning Rates			
		30%	50%	70%	90%
RN-18	**Random**	$71.15_{\pm 0.23}$	$68.48_{\pm 0.10}$	$63.15_{\pm 0.19}$	$44.96_{\pm 0.50}$
	Ours-LF	$71.21_{\pm 0.09}$	$68.77_{\pm 0.13}$	$63.76_{\pm 0.09}$	$47.50_{\pm 0.10}$
	Ours	$70.94_{\pm 0.19}$	$69.30_{\pm 0.84}$	$65.16_{\pm 0.04}$	$49.57_{\pm 0.16}$
RN-34	**Random**	$73.37_{\pm 0.08}$	$71.71_{\pm 0.10}$	$67.85_{\pm 0.04}$	$51.29_{\pm 0.20}$
	Ours-LF	$73.61_{\pm 0.08}$	$71.99_{\pm 0.05}$	$68.42_{\pm 0.21}$	$53.52_{\pm 0.06}$
	Ours	$73.39_{\pm 0.12}$	$72.34_{\pm 0.13}$	$69.44_{\pm 0.17}$	$55.92_{\pm 0.02}$
RN-50	**Random**	$76.06_{\pm 0.11}$	$74.44_{\pm 0.04}$	$70.50_{\pm 0.02}$	$53.56_{\pm 0.13}$
	Ours-LF	$76.54_{\pm 0.08}$	$74.84_{\pm 0.03}$	$71.10_{\pm 0.09}$	$55.22_{\pm 0.92}$
	Ours	$76.29_{\pm 0.10}$	$75.12_{\pm 0.03}$	$72.05_{\pm 0.09}$	$58.26_{\pm 0.46}$

that our concept-based score aligns well with human intuition on the difficulty of the samples.

E. Ablation studies

Here, we present an ablation to study the effect of different methods for concept generation and the effect of keeping different numbers of concepts per class (k) in the bottleneck.

E.1. Comparing Concept Generation Techniques

Table 6 shows how the performance of models trained on our coresets change when different methodologies are used to generate the concept sets. Since our method uses LLaVA [34], which is a vision language model, we compare the performance of models trained on the random subsets and coresets obtained using class-wise concepts (only textual information) and concepts extracted using both visual and textual information. For concepts generated using only textual information, we consider two alternatives, namely **class-wise attributes** (CW-A) and **class-wise descriptions** (CW-D). While CW-A considers concepts formed by a single or a few words, CW-D consists of longer, more descriptive concepts (eg., a descriptive concept for the class butterfly is "*a beautiful insect with colorful wings*"). For CW-D, we use a subset of k concepts provided by [70], generated via the GPT-3 model. Our results show that CW-A performs better than CW-D for both the standard and label-free CS problems. Thus, we use attribute-level concepts in our work.

Next, for generating concepts using both visual and textual information, we consider two alternatives. The first is a **class-wise one-shot image attribute** approach where we first cluster all images of a class in the embedding space of the CLIP's visual encoder and identify the image whose embedding is the closest to the cluster center (for the label-free setting we use the pseudo-labels of the images during clustering), then we prompt LLaVA to generate attributes using this single image and the class name. Once generated we use k discriminative concepts to form the bottleneck. The second is **image-wise** attribute approach, where we use each image in the training set and prompt LLaVA to generate per image attributes describing the image. Once generated we sort the concepts based on their frequency of occurrence in a class and use the most frequently occurring discriminative concepts to form the bottleneck. While the image-level concepts lead to the best coresets, it is slow and costly to prompt LLaVA to generate attributes for all the images in a large dataset such as Imagenet. For CIFAR-100, this process took about nine hours to complete (in comparison CW-A can be extracted in 5 minutes for CIFAR-100, without parallel computation) which is very costly compared to the small performance gains it provides over other approaches. Lastly, while the one-shot approach is better than CW-A in most cases, the additional step of clustering can be costly for larger datasets such as Imagenet. Based on these results, we used CW-A for concept generation using LLaVA.

Table 6. Coreset performance on CIFAR-100 for 90% pruning rate for concepts generated using different methods. A random subset of CIFAR-100 achieves an accuracy of $44.76_{\pm 1.58}$ at this rate. (Class-wise attributes approach is used for in rest of the paper.)

	Concept generation methods			
	Prompt uses only class label		*Prompt uses class label + image(s)*	
	class-wise attributes	**class-wise descriptions**	**class-wise one-shot image attributes**	**image-wise attributes**
Ours	$51.85_{\pm 0.51}$	$51.05_{\pm 0.71}$	$51.68_{\pm 0.45}$	$52.47_{\pm 0.39}$
Ours-LF	$49.91_{\pm 0.96}$	$49.85_{\pm 0.83}$	$50.22_{\pm 0.16}$	$51.05_{\pm 0.93}$

E.2. Comparing Effect of Number of Concepts (k)

In Table 7, we show how the number of concepts extracted per class label affects the selection of coresets. Once the list of attribute-level concepts is generated by LLaVA, we can select k concepts per class either randomly or choose concepts unique to a class (discriminative). Our results show that using even k=1 is sufficient to surpass the performance using a random subset (Random baseline). This performance increases when we keep discriminative concepts in our concept bottleneck, with $k = 5$ achieving the best results. While the size of the concept bottleneck need not be very large to find good coresets, it is helpful to take a sample's visual similarity with a set of concepts rather than a single concept per class. Thus, we selected 5 discriminative concepts per class to form the concept bottleneck.

F. Details for concept set generation

Prompt Selection: To extract concepts for our approach, we only use the class labels in the prompt as can be seen in Figure 1. The prompt, "Can you give distinct attributes

Table 7. Effect of the number of concepts k (per class) and the method of selecting k attribute-level concepts from the concepts generated by LLaVA vs. the accuracy of models trained on the coresets of CIFAR-100 at 90% pruning rate. A random subset of CIFAR-100 achieves an accuracy of $44.76_{\pm 1.58}$ at this rate.

Concept selection	k=1	k=5	k=10
Random concepts	$48.78_{\pm 0.96}$	$50.15_{\pm 1.64}$	$50.39_{\pm 0.72}$
Discriminative	$51.42_{\pm 0.18}$	$\mathbf{51.58}_{\pm 0.51}$	$51.22_{\pm 0.72}$

for ⟨class name⟩. Give the output separated by a comma in the line." instructs the VLM not only to provide distinct keywords but also adds formatting instructions. However, despite the instructions included in the prompt, LLaVA outputs are not always formatted well, often containing duplicate entries, mismatched commas and braces, and sometimes having a detailed explanation before the keywords. To remedy this we run the LLaVA output through a simple post-processing script and use regular expressions to clean the LLaVA outputs. For our experiments where we perform ablation of various concept-bottleneck generation methods (Table 6), we also use two more concept generation methods, one is one-shot image-based class concepts and the second is image-level concept generation. For the former, where we select one representative image per class via clustering, we prompt LLaVA as follows, "⟨image⟩ Can you give distinct attributes for such an image of ⟨class name⟩. Give the output separated by a comma in the line." And, to get concepts for every image of a class, we use a similar prompt as follows, "⟨image⟩ Can you give distinct visual attributes for this image of ⟨class name⟩. Give the output separated by a comma in the line." Each LLaVA prompt request on a single A-100 GPU takes approximately 3 seconds.

Alternative VLMs for Concept Generation: We leverage LLaVA as our choice of VLM for concept generation, however in Table 6, we also compare against concepts extracted from GPT (column 2) [67]. We see comparable performance with those extracted from LLaVA. Moreover, LLaVA is an open-source model whereas GPT prompting is not. We also experimented with retrieving concepts via SpLiCE [4]. However, a major limitation of SpLiCE is that similar to image-level concepts it is a costly approach. SpLiCE uses a linear optimization for sparse concept decomposition and can take up to 3 hours for $50,000$ images which is slower than generating class-level concepts from LLaVA.

G. Experimental Details

G.1. Datasets, and Model Implementation Details

For CIFAR-10/CIFAR-100, we train a ResNet-18 model and for Imagenet we train ResNet-18, ResNet-34, and ResNet-50 models on the coresets for all pruning ratios.

Emotion Recognition: For emotion recognition, we use the Affectnet dataset [38] for our experiments. AffectNet is a large-scale facial expression dataset designed for training and evaluating affective computing models [61]. It contains facial images collected from the internet using web search queries for emotion-related keywords in multiple languages. Each image is manually annotated for eight discrete emotion categories: `neutral, happiness, sadness, surprise, fear, disgust, anger, contempt`. For our experiments, we utilize an openly available version of this dataset [23], containing roughly 16000 training and 14000 testing samples. According to our approach we first use LLaVA to extract concepts for the 8 emotion classes, using the following prompt, *"What are the facial features that distinguish emotion class name from other emotion types. Focus on changes in eyes, nose, lips, eyebrows, mouth. Give the output separated by commas in a line."*. We get $5-10$ distinctive facial feature concepts for every emotion, for instance for emotion class *happy*, we get the following concepts, *"wide open eyes", "sparking eyes", "smiling lips", "open mouth", "raised eyebrows", "flushed cheeks", "teeth barred"*.

Biomedical Image Recognition: For biomedical image recognition, we use the BloodMNIST [1] dataset from the MedMNIST [68, 69] which comprises of images of normal blood cells, captured from individuals without infection, hematologic or oncologic disease and free of any pharmacologic treatment at the moment of blood collection. It consists of a total of $17,092$ images and is organized into 8 classes (`basophil, eosinophil, erythroblast, immature granulocytes, lymphocyte,monocyte, neutrophil, platelet`).

For this dataset, we first extract concepts for the 8 blood cell types via GPT using the following prompt, *"What are the features that can distinguish blood cell class name from rest of the blood cell types on their size, shape, nucleus appearance, and the presence of granules in their cytoplasm"*. We obtain 10 concepts for every blood cell type, for instance, for *platelets*, we get the following concepts, *"Smallest blood component", "No nucleus", "Granules present", "Irregular shape", "Cytoplasmic fragments", "Variable granule distribution", "Oval to round shape", "Small dense granules", "Lacks chromatin", "Compact cytoplasmic body"*. To accelerate training on Imagenet, we utilize the training code based on FFCV [31]. We run CS for three trials with different random seeds for all experiments and report the average of these runs in our tables. For generating the concept annotation we use a recently proposed open source model LLaVA [33, 34]. Extracting the concepts for each class using this model takes a mere 3 sec-

onds per prompt.

For computing the concept similarity scores between the visual and concept bottleneck features we used the CLIP [45] model following the previous works [67, 70, 72] which showed its effectiveness for this task. Specifically, we used the ViT B-32 CLIP model [45]. For computing the pseudo-labels in Sec. B we used a ViT L-14 CLIP model trained on the DataComp-1B dataset [22]. The accuracies of the models trained on the entire training set are 95.44% and 78.74% for ResNet(RN)-18 on CIFAR-10/100 and 72.4% for RN-18, 75% for RN-34, and 78.4% for RN-50 on Imagenet.

For generating the importance score we pre-compute the concept similarity scores for the entire dataset and then train the concept-bottleneck layer (in block 2 of Fig. 1) for 100 epochs across all experiments. This training only requires 800 seconds for Imagenet which is significantly more efficient than training the ResNet-34 model on Imagenet (requires roughly 8 hours without FFCV and about 4 hours with FFCV on two A-100 GPUs).

After the coresets are selected, we use the setting and code from [73] for training a ResNet-18 model for 40000 iterations with a batch size of 256 on the coresets for all pruning ratios for CIFAR-10/CIFAR-100. For Imagenet, we train ResNet-18, ResNet-34, and ResNet-50 models for 100 epochs on the coresets identified by our method using the training code based on FFCV [31].

The performance of the label-free CS is dependent on the quality of the pseudo-labels. Compared to the clustering-based approach used by ELFS [74], our approach of using the zero-shot classification ability of CLIP models yields significantly better pseudo-label quality along with being simpler and more efficient to compute. Specifically, for CIFAR-10/100, pseudo-labels of the training set are computed using the CLIP L-14 model trained on the DataComp-1B dataset [22] yields an accuracy of 98.52% and 87.28% whereas for Imagenet it achieves an accuracy of 79.47% which are better than the best pseudo-label accuracy obtained by the clustering approach in ELFS (92.5% and 66.3% on CIFAR-10/100 and 58.8% on Imagenet).

G.2. CS baselines and methods

We compare our method against various baselines and SOTA methods proposed by previous works. 1) **Random:** Uniformly select samples from the datasets to form the coreset. $\text{Random}_{\text{FFCV}}$ denotes the performance of the models trained on random subsets of Imagenet using the training code based on FFCV [31]. 2) **Entropy [11]:** Selects samples based on entropy which is computed as the uncertainty in the model's prediction on a sample. 3) **Forgetting Score [58]:** Selects samples based on the forgetting score which is computed as the number of times an example is misclassified after being correctly classified earlier during training of the downstream model. A higher forgetting score indicates a more challenging sample. 4) **AUM [44]:** Selects samples based on their average margin during training of the downstream model i.e., the difference between the target class and the next highest class across the training epochs. Lower AUM indicates a more challenging sample. For the forgetting score, AUM, and our method, we use CCS [73] for sampling to form the coreset whereas for entropy we select the samples with the highest entropy in the coreset as done in previous works [11, 73].

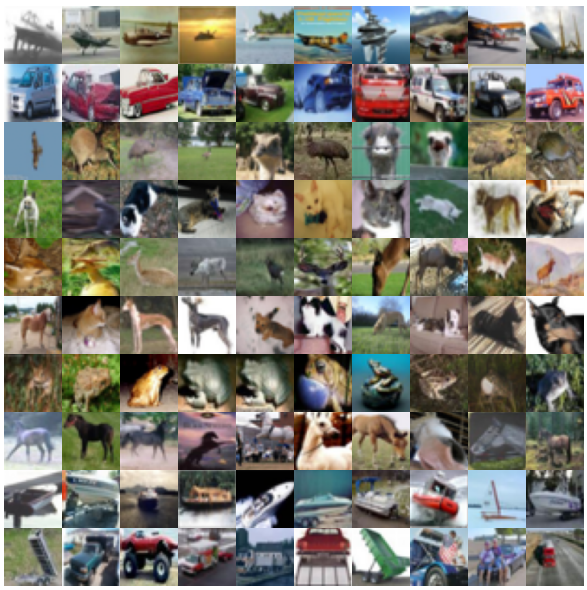
For label-free CS, we use 1) **Prototypicality [55]:** which first performs k-means clustering in the embedding space of SwAV [6] model and ranks samples based on their Euclidean distance to the cluster centers. Samples further away from the cluster center are then used to form the coreset. 2) **ELFS [74]:** estimates the pseudo-labels of the unlabeled samples using a deep clustering approach (using the embedding space of SwAV [6] and DINO [7]) and forms the coreset using the training dynamics of the downstream model trained on the pseudo-labeled data.

Crucially, SOTA methods such as forgetting score, AUM, and ELFS require training the downstream model on the entire dataset (with true/pseudo labels) first, for CS, unlike our method which is independent of the downstream model. While the Random and the Prototypicality don't require the downstream model for CS, we show in the following sections that our results are significantly better than these.

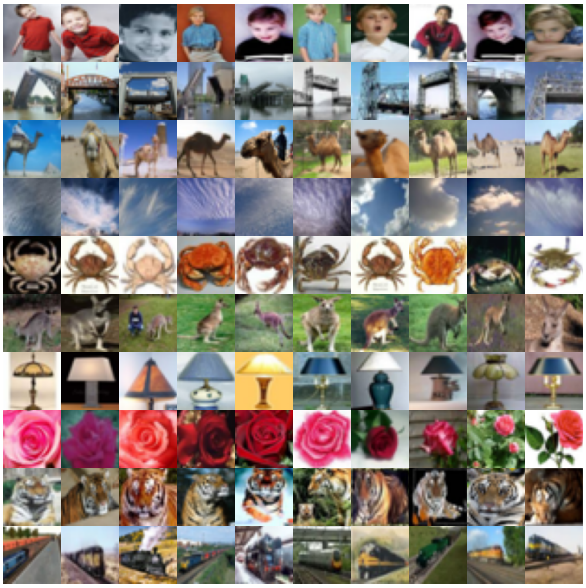
G.3. Algorithm for stratified sampling using CCS [73]

Here we present the algorithm for sampling the training examples to form the coreset based on the coverage-based selection methodology proposed by [73]. A crucial component of the algorithm is the cutoff rate β which controls how many challenging samples should be removed from consideration when selecting the coreset. This is done to eliminate misclassified samples from the dataset since they can hurt the performance of the model trained on coreset, especially at high pruning rates. Previous works [73, 74] ablate the values of this cutoff ratio by training the downstream model on a range of values. In our work, we simply use the values proposed by the previous works and find that they work well for our score as well. The cutoff rates β for different pruning rates α are as follows (α, β). For CIFAR-10: (30%, 0), (50%, 0), (70%, 10%), (90%, 30%), for CIFAR-100: (30%, 10%), (50%, 20%), (70%, 20%), (90%, 50%), for Imagenet: (30%, 0), (50%, 10%), (70%, 20%), (90%, 30%). We used CCS for label-free CS as well and the cutoff rates used were for CIFAR-10: (30%, 0), (50%, 0), (70%, 20%), (90%, 40%), for CIFAR-100: (30%, 0), (50%, 20%), (70%, 40%), (90%, 50%), for Imagenet: (30%, 0), (50%, 10%), (70%, 20%), (90%, 30%).

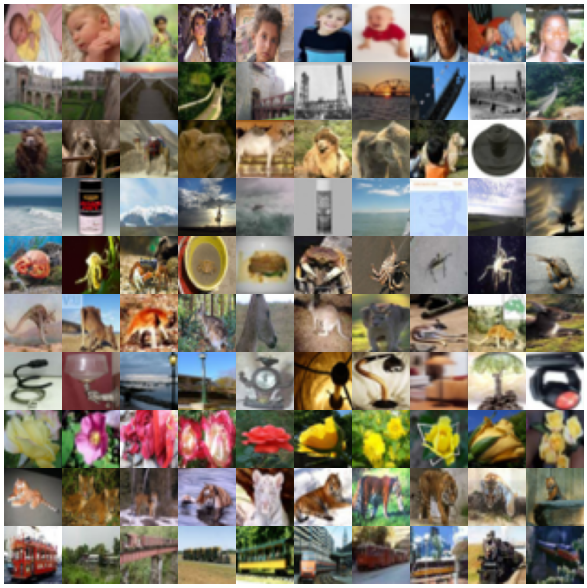
(a) Easy images from CIFAR-10



(b) Challenging images from CIFAR-10



(c) Easy images from CIFAR-100



(d) Challenging images from CIFAR-100

Figure 3. Class-wise easy and challenging images for the 10 classes (`airplane`, `car`, `bird`, `cat`, `deer`, `dog`, `frog`, `horse`, `ship`, `truck`) in CIFAR-10 and for a subset of 10 classes (`boy`, `bridge`, `camel`, `cloud`, `crab`, `kangaroo`, `lamp`, `rose`, `tiger`, `train`) from CIFAR-100. Similar to the results in Fig. 2, easy images (a,c) are more canonical images associated with the class labels whereas challenging images (b,d) are images that are confused between two or more classes in the dataset.

H. Limitations and Future Work

In our work, the concept extraction from LLM is treated as a pre-processing step which is independent of the data difficulty score computation and coreset selection. However, LLMs can produce very noisy and non-discriminative concepts for various classes (eg., a concept "blue" can be associated with several classes in the Imagenet dataset), leading to poor concept-similarity scores. Moreover, while class-wise concept extraction is efficient, image-level concepts could be much more informative for a sample's difficulty. Thus, improving the efficiency of concept extraction and

Algorithm 1 Coverage-centric Coreset Selection (CCS) [73]

Input: Dataset with difficulty scores: $\mathbb{D} = \{(x, y, s)\}_{i=1}^{n}$, pruning ratio: α, cutoff rate: β, number of bins: b.
Output: Coreset: $\mathcal{S}$

```
# Prune hardest examples
𝔻′ ← 𝔻 \ {⌊n × β⌋ hardest examples}
A_1, A_2, ··· , A_b ← Split scores in 𝔻′ into b bins.
ℬ ← {B_i : B_i consists of samples with scores in A_i for i = 1, ··· , b}.
# Define the size of the coreset
m ← n × α.

while ℬ ≠ ∅ do
  # Select the bin with the fewest examples
  B_min ← arg min_{B∈ℬ} |B|.
  # Compute the budgets for this bin
  m_B ← min{|B|, ⌊m / |ℬ|⌋}.
  𝒮_B ← randomly sample m_B samples from B_min.
  𝒞 ← 𝒞 ⋃ 𝒮_B.
  ℬ ← ℬ \ {B_min}.
  m ← m − m_B.
end while
return 𝒞.
```

tuning the prompt for LLMs/VLMs to incorporate the feedback from score computation or CS can help generate better concept bottlenecks leading to a better estimate of a sample's difficulty. While these are important research directions we leave them for future work.