Back to the Future: Models as Active Learning Surrogates for Next Generation ML Deployments

Lukas Frickenstein*[1], Moritz Thoma*[1], Pierpaolo Mori*[1,3], Shambhavi Balamuthu Sampath*[1], Nael Fasfous[1], Manoj-Rohit Vemparala[1], Alexander Frickenstein[1], Christian Unger[1], Claudio Passerone[3], Walter Stechele[2]
* indicates equal contributions

[1] BMW Autonomous Driving
[2] Technical University of Munich
[3] Politecnico di Torino
[1]firstname.lastname@bmw.de, [2]firstname.lastname@tum.de,
[3]firstname.lastname@polito.it

Abstract. Rapid development of hardware goes hand-in-hand with the advancement of modern computer vision (CV) algorithms. In a typical machine learning operations (MLOps) flow, this continuous evolution of hardware and software is coupled with an active growth in data collected for training. These three pillars of MLOps continue their parallel continuous integration and improvement after an iteration of the deployment has been released. Ideally, the data chosen to improve the next iteration of the deployment is tailored for the future software solution and the future hardware capabilities which enable it. However, here we have a causality problem, where data needs to be collected for a future algorithm from a fleet of deployments which are still running the last iteration of software and hardware. In this paper, we prove that models of previous MLOps iterations are capable surrogates for choosing data for future network architectures running on more capable hardware. We show that surrogate models for the DeepLabv3+ architecture using a ResNet-50 backbone provide a +3.2 p.p. mIoU improvement on average using uncertainty scores over randomly selecting data to train the deployment model on the CityScapes dataset. Further, we show that the type of surrogate has a huge impact on the prediction capability of the deployment model. For instance, the prediction capability of a deployment model, DeepLabv3+, using a MobileNetV3 backbone, can vary by up to +2.4 p.p. on the CityScapes dataset.

Keywords: Active Learning, Machine Learning Operations, Computer Vision

1 Introduction

The industrialization of machine learning (ML) algorithms has led to disruptive changes in several application domains, such as robotics and autonomous driving. Customer expectations have risen to all time highs, resulting in an explosive

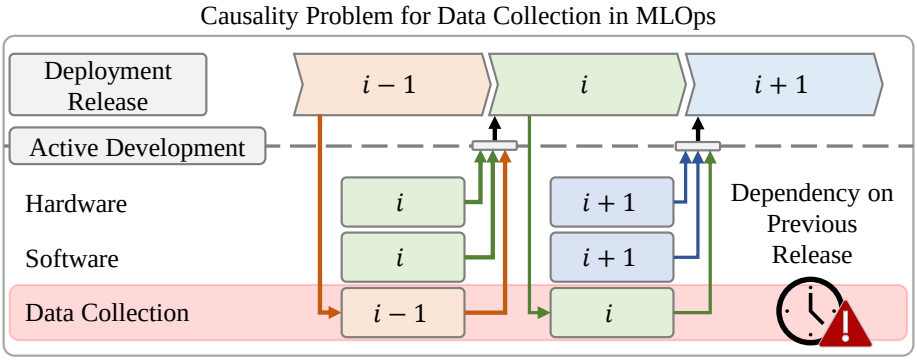

Fig. 1: Back to the future: Hardware and software are co-developed for future deployment releases i, while data collection is tied to the fleet of deployment from past releases $i-1$.

growth in both hardware development and deep neural network (DNN) complexity. As ML engineers develop new methods of training ever larger networks, new layer types, and complex network hierarchies [3,6], hardware engineers had to keep up the pace by providing more compute performance, higher memory density, and lower power consumption accelerators [12–14]. This fast pace game of catch-up often led to hardware being rendered obsolete in a mere year or two. Along with the co-development/co-evolution of DNNs and the hardware which they are executed on, a rapid uptake of ML-based products meant that an ocean of data processed by the end-users had untapped potential to drastically improve the training and generalization of the ML algorithms in future generations of the product. This life-cycle of DNN, hardware, and data development is encapsulated in the function of machine learning operations (MLOps).

Within MLOps, researchers in the domain of active learning develop methods to carefully curate and label the most relevant data to be learned by a DNN from a pool of unlabeled data [7,17,20]. Here, the DNN typically chooses data it wants to learn for itself, for example, based on its own uncertainty of the data. However, ideally, the data chosen to improve the next iteration of the deployment should be tailored for the future DNNs and the future hardware capabilities which enable it. Here we run into a causality problem, where data needs to be collected for a future algorithm from a fleet of deployments which are still running the last iteration of software and hardware. Fig. 1 illustrates the causality problem, i.e. that the development of hardware and software for a future release i (highlighted in green) is carried out concurrently, while data collection remains dependent on the fleet of deployments from past releases $i-1$ (highlighted in orange).

In line with these challenges, we identify two shortcomings of current active learning literature: (1) The large pool of unlabeled fleet data is expected to have already been collected and stored on a server. This leaves the underlying method of collection from the fleet undefined, and the usage of upload bandwidth

and server capacity sub-optimal. (2) The techniques enable the DNN to choose data for itself, from the pre-existing unlabeled pool of data. This disregards the fact that DNNs inevitably become outdated and will have successor networks, potentially with vastly different architectures.

In this paper, we investigate active learning in the context of an evolving MLOps flow and view DNNs of past iterations as *surrogate* data curators for their next-generation, successor DNNs. This demystifies the method of data collection of the initial unlabeled pool by the surrogate, which is capable of running on the past iteration's hardware fleet. Furthermore, we analyze the relevance of the data chosen by the past surrogate with respect to a future network, which will be trained and deployed on next-generation hardware, not yet available in the fleet. The contributions of this work are summarized as follows:

- We investigate the challenges of active learning in the context of real-world development, where the model becomes more advanced over time while relying on data collected from previous generations of hardware and software. We view past models as active learning surrogates to pick data for future generation DNN models.
- We perform large-scale experiments to empirically show that surrogate models are capable of collecting data for future DNNs, allowing the reuse of training data of previous MLOps iterations in next generations. Exemplary, when training a future DeepLabv3+-ResNet-50 on data collected by a past generation model, the prediction capability improves by +3.2 p.p. when compared to training on random data of CityScapes.
- We analyze the impact of the surrogate type on the prediction capability of a deployment model, where the prediction capability of a deployment model, DeepLabv3+, using a MobileNetV3 backbone, can vary up to +2.4 p.p. on the CityScapes dataset. We analyze the entropy scores of different models and present arguments for characteristics of good surrogates based on empirical evidence. The mean entropy of DeepLabv3+ using MobileNetV3 as a backbone is $\times 1.9$ larger than entropy values of Xception-65, leading to better separation of samples.

This work is structured as follows. Sec. 2 discusses relevant methods in the field of active learning using static models or proxy models for data collection. Sec. 3 benchmarks representative CNNs of recent years on three common hardware accelerators, emphasizing the necessity of models as active learning surrogates for next generation ML deployments. Subsequently, the classical active learning cycle that uses static models for data collection and common active learning policies are revisited to showcase the dependency of the selection process on learned features of the static DNN. Sec. 4 introduces the method for viewing DNNs from previous iterations in an MLOPs flow as surrogate data curators for successor DNNs. Next, in Sec. 5, large-scale experiments are presented to demonstrate the capability of surrogate models to collect data for future DNNs. Finally, Sec. 6 summarizes the presented work and provides an outlook for future research.

2 Related Work

2.1 Active Learning

Active learning techniques aim to reduce the amount of labeled data needed to train a model by strategically selecting the most informative instances for labeling [20]. The approaches in active learning can be categorized into (1) uncertainty-based, (2) diversity-based, and (3) hybrid methods.

Uncertainty-based active learning methods focus on selecting samples that the model is uncertain about, exploiting the ambiguity within its predictions. There are various methods to estimate uncertainty, which can be categorized as single- or multi-pass approaches. Single-pass approaches estimate uncertainty in a single forward pass. LLAL [20], introduces loss modules to the model architecture which learn the loss behavior during training based on its internal activations. Subsequently, the predicted loss on unlabeled data serves as a measure of uncertainty to indicate the relevance of the sample to be labeled by an oracle. Shannon entropy [7], which we will refer to as entropy, calculates the Shannon entropy on the model's output to determine uncertainty. In contrast, multi-pass approaches estimate uncertainty by performing multiple forward passes. QBC [18] uses disagreement between multiple models as an uncertainty indicator. Noise Stability [9], perturbs the model weights for every pass and uses the output variance as an uncertainty estimate.

Diversity-based active learning methods aim to select samples that are diverse and representative of the entire dataset by comparing sample representations within the dataset. K-means [2] selects samples based on their distance to the average representation of known samples. Core-Set [17] chooses samples so that their representations have a maximal coverage of the representation space of all samples.

Hybrid active learning methods combine both uncertainty and diversity-based approaches to select the most informative samples for labeling. BADGE [1] does k-means clustering on pseudo gradients to select data. Alfa-Mix [15] selects data based on confusion between classes in the output that arises when interpolating intermediate features towards other known features. More recently, hybrid approaches dedicated to 2D and 3D object detection have emerged [10, 11].

While all presented works demonstrate the ability of their respective approach to select informative data, they all assume a *static model* for data collection and deployment, i.e. the model chooses its data for itself. However, this assumption does not align with the dynamic nature of the MLOps flow, where model architectures may change over development iterations, as highlighted in Sec. 1.

2.2 Active Learning with Proxy Models

Very few works have addressed the usage of active learning in a setting, where the model for data selection is not the model used for deployment. General-and-Efficient-Active-Learning [19] proposes an active learning method, that uses a large, pre-trained model for selecting data for another model. However, while

Table 1: Classification of related work explorations of active learning (AL) methods using static data collection models, proxy models and surrogate models for next generation machine learning deployments for data collection.

AL using:	[1, 2, 7, 15, 17, 18, 20]	[4, 19]	[This Work]
- Static Models	✓	✓	✓
- Proxy Models	✗	✓	✓
- Surrogate Models for Next-Gen	✗	✗	✓

they do not use the same model for selection and deployment, the models still remain static. Selection-via-proxy [4] uses simple models as a first proof of transferability of active learning data with the main goal of reducing the number of training epochs and GPU hours for data selection. However, the paper does not investigate different model architectures in the context of past and future releases, nor do they give an indication to what makes for a good proxy model. Furthermore, proxy models only refer to simpler and faster models of the same architecture which save GPU hours during data selection. This is different to the concept of surrogates introduced by this work, which are temporally older releases of potentially different architectures, that enable the reuse of data collected for future releases.

We summarize the explorations and investigations conducted in existing literature in Tab. 1. As with other studies, we report the effectiveness of AL methods using static models that choose data for themselves (see diagonal entries of Tab. 3, 4 and 5). Other than existing works that use static model as a proxy to collect data for another model, we introduce the concept of models as active learning surrogates for next generation ML deployments.

3 Background

3.1 Evolution of Latency

The following emphasizes the ongoing competition between HW engineers, who strive to create more powerful and energy-efficient accelerators with increased compute performance and memory density, and ML engineers, who continually develop new training methods for ever larger networks, new layer types and complex network hierarchies. To this end, we compute the theoretical latency of different ML algorithms on three common edge accelerators. On one hand, we select three widely-used HW accelerators that represent the advancements in the field over recent years with increased available peak tera-operations-per seconds (TOPS). On the other hand, we report the number of giga-operations (GOPs) required by representative DNNs from recent years. Tab. 2 reports the theoretical latency (ms) of a DeepLabv3+ [3] using various backbones such as ResNet-20/18/50 [6], MobileNetV3 [8] and Xception-65 [3] running on three example edge accelerators, namely Nvidia Jetson TX2 [13], Xavier [14] and Orin [12].

Table 2: Theoretical latency (ms) of a DeepLabv3+ (DLv3+) using various backbones such as ResNet-20/18/50 (Res-20/18/50), MobileNetV3 (MNV3) and Xception-65 (Xc-65) running on three example edge accelerators.

Hardware (TOPS)	**DLv3+ Latency (ms)**				
	Res-20 60.3 GOPs	MNV3 61.1 GOPs	Res-18 181.8 GOPs	Res-50 404.1 GOPs	Xc-65 512.8 GOPs
TX2 [13] (1.3*)	46.4	47	139.8	310.8	394.5
Xavier [14] (21)	2.9	2.9	8.7	19.2	24.4
Orin [12] (50)	1.2	1.2	3.6	8.1	10.3

[*] Only supports float operations.

We use a color coding (applied to all entries) to indicate increased theoretical latency, from high to low latency. We observe an increased theoretical latency for larger models requiring a larger number of GOPs running on HW accelerators with limited TOPS. This empirically renders potential future ML algorithms not suitable for older generation HW.

As DNNs and the HW they run on continue to evolve together, a rapid uptake of ML-based products meant that an ocean of data processed by the end-users had untapped potential to drastically improve the training of the ML algorithms in future generations of the product. However, using this untapped potential is constrained by the causality problem. In order to collect data for future algorithms, it is necessary to collect data from a fleet of deployments that are still running the previous iteration of software and hardware (recall Fig. 1). This limitation is further supported by the observed increase in theoretical latency of potential future ML algorithms when executed on older generation hardware (recall Tab. 2).

3.2 Active Learning Baselines

To investigate the effectiveness of models as active learning surrogates for next generation DNNs, we reintroduce two aspects. First, we explain the classical active learning cycle using static models. Second, we present three common representatives of active learning policies to showcase the dependency of the selection process on the learned features of a DNN.

Classical Active Learning Cycles using Static Models: Without loss of generality, pool-based active learning aims to select samples from an unlabeled data pool D_U. The active learning policy defines a selection function $\mathcal{S}$ to identify the most informative data samples in D_U, forming a subset D_S. A human annotator provides labels Y for the data samples X in D_S. In this way, the initial, randomly selected training dataset D_{rand} is actively extended in cycles $c \in C$ to generate the training dataset D_T used to train a DNN θ. This iterative process of selection, labeling and training is repeated until a given labeling budget B is exhausted.

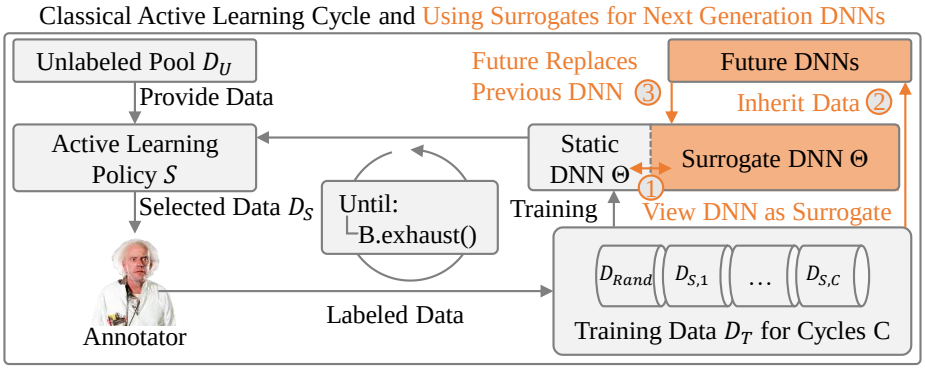

Fig. 2: We extend the classical active learning cycle by (1) viewing DNNs as surrogate data curators to (2) allow inheritance of their training data for future, next generation DNNs that (3) become themselves surrogates for future DNNs.

Here, the architecture of the model θ is expected to remain constant to choose the samples it wants to learn for itself, i.e. *static* (see Fig. 2).

Active Learning Policies: Many active learning methods use learned features from a DNN to asses the informativeness of unlabeled samples [7, 17, 20]. They differentiate in their selection function S to choose samples for labeling. This work investigates the effectiveness of models as active learning surrogates for next generation models on two uncertainty-based and one diversity-based active learning method. **Uncertainty-based** methods exploit the ambiguity within a DNN's prediction. The selection function of entropy [7] $S_{entropy}$ computes the entropy of the logits of a DNN θ for a given data sample x_i drawn from the unlabeled data pool D_U and chooses the samples with the highest values (topk) to satisfy a given budget B, as shown in Eq. 1.

$$\mathcal{S}_{entropy}(B, \theta, D_U) = \underset{x_i \in D_U; k=B}{\text{topk}} \Big[-\sum \theta(x_i) \cdot \log \theta(x_i) \Big] \tag{1}$$

The selection function of learning loss for active learning [20] (LLAL) S_{LLAL} predicts the loss $\mathcal{L}$ of a given DNN θ for a given sample x_i of D_U using an additional module θ_{LLAL} and uses it as a proxy for the uncertainty of θ. Again, samples with the highest predicted loss value (topk) are chosen until the budget B is met, see Eq. 2.

$$\mathcal{S}_{LLAL}(B, \theta, \theta_{LLAL}, D_U) = \underset{x_i \in D_U; k=B}{\text{topk}} \big[\hat{\mathcal{L}} := \theta_{LLAL}(\theta, x_i) \big] \tag{2}$$

Diversity-based active learning aims to curate a diverse and representative training data set from a pool of unlabeled data. The Core-Set [17] selection picks data samples such that the new data pool minimizes the distance Δ between the sample representation $\theta(x_i)$ of the unlabeled pool $x_i \in D_U$ and the nearest

sample representation $\theta(x_j)$ of the existing selection and the training data pool $x_j \in D_S \cup D_T$, see Eq. 3.

$$\mathcal{S}_{core}(B, \theta, D_U, D_T) = \min_{|D_S| \leq B} \max_{x_i} \min_{x_j} \Delta[\theta(x_i), \theta(x_j)] \tag{3}$$

To show the effectiveness of different selection functions as introduced above, choosing data samples uniformly at random from the unlabeled data pool D_U or using the model's actual loss serve as a **reference** point. In the research context, using the ground truth label Y to compute the true prediction error $\mathcal{L}$ of the model allows to select data from the data pool D_U based on the networks performance (Eq. 4).

$$\mathcal{S}_{True\text{-}Loss}(B, \theta, D_U, Y) = \underset{x_i \in D_U; y_i \in Y; k=B}{\text{topk}} \left[\mathcal{L}(\theta(x_i), y_i)\right] \tag{4}$$

Lastly, randomly selecting samples to label from the unlabeled pool D_U is a crucial lower-bound for active learning aiming to carefully curate and label the most relevant data.

4 Models as Active Learning Surrogates

In this work, we view DNNs of past iterations in an MLOps flow as surrogate data curators for their next-generation, successor DNNs, i.e. viewing static models in classical active learning cycles as *surrogates*. This view allows future DNNs to inherit the training data selected from past generations, thus removing the need to restart the active learning cycle embedded in an MLOps flow. Ultimately, the introduced future DNN becomes itself a surrogate of its successor DNNs. Fig. 2 highlights the re-definition of the classical active learning cycle in (orange) by (1) viewing DNNs as surrogate data curators to (2) allow inheritance of their training data for future, next generation DNNs, which (3) ultimately become themselves surrogate models for their successor models in an MLOps flow. We show the capabilities of models as active learning surrogates for next generation ML deployments by exploring the following sequence:

- First, in Sec. 5.2 we consider the classic active learning cycle using static models. Here, we investigate the influence when switching DNNs of various types and sizes, feeding their learned features to three selection functions, thus choosing data for itself.
- Second, in Sec. 5.3 we view the models selecting data for themselves as surrogate data curators for their next-generation, successor DNNs. Training data is inherited to DNNs of various types and sizes, thus train models on data collected by surrogate models.
- Lastly, in Sec. 5.4 we support the empirical evaluations of surrogate data curators by in-depth investigations of their entropy-based sample assessment.

5 Experiments

5.1 Setup

Datasets. Data and its associated task are a pivotal aspect for active learning. Therefore, experiments are carried on the task of image classification and semantic segmentation. For image classification, the large-scale dataset of ImageNet [16] is used, containing $\sim$ 1.28M training and 50K validation images with 1000 classes. For the task of semantic segmentation, experiments are carried out on the ADE20K [21] and CityScapes [5] dataset. For ADE20K, 20210 train and 2000 validation images of different aspect ratios and sizes are used and contain 150 classes. CityScapes [5] consists of 2975 train and 500 validation images and presents 19 classes. We want to highlight the notoriously more costly labeling process for the task of semantic segmentation due to its pixel-level annotations when compared to image classification. This increases the significance of finding the most relevant data.
Models. For image classification we train and evaluate two variants of ResNet [6], namely ResNet-18/34 (Res-18/34). The architecture of DeepLabv3+ [3] (DLv3+) is employed for the task of semantic segmentation using a variety of different backbone types including MobileNetV3-Small [8] (MNV3), ResNet-20/18/50 (Res-20/18/50) and Xception-65 [3] (Xc-65). All backbones are pre-trained on the ImageNet dataset, and are trained and evaluated at output stride 16 using dilated convolutions, and share the same DeepLabv3+ head.
Training Parameters. If not otherwise mentioned, the augmentations and training parameters are adopted from the baseline implementations. Further, for semantic segmentation experiments we use the SGD optimizer with a momentum of 0.9 and weight decay of 4e-4 for all layers except the batch normalization layers. For CityScapes we train for 220 epochs while using batch size 4 for all models except Xception-65, where we use batch size 6. For ADE20K we train for 64 epochs using batch size 16 for all models.
Active Learning. We evaluate three active learning methods, namely Entropy, LLAL and Core-Set and two reference methods, as described in Sec. 3.2. We calculate the entropy from the model's output layers for both ImageNet and CityScapes. For LLAL, we follow the specifications from the original paper [20] for the task of image classification. When switching to the task of semantic segmentation, the dense layers of the prediction module θ_{LLAL} are replaced by convolution layers and global average pooling layers are omitted. Further, we adjust the four connection points of θ_{LLAL} to the low- and high-level features, as well as one feature after its ASPP block and one before the last convolution layer of the DeepLabv3+ head. Note that the active learning methods entropy, LLAL and true loss generate pixel-level scores which we aggregate to scalar values allowing an assessment on sample-level. Core-Set uses the outputs of the last convolution layer and the low-level feature of DeepLabv3+ as feature representations for image classification and semantic segmentation respectively. We adopt an active learning cycle to select 2 % samples from the unlabeled pool each time over a total of 10 cycles, resulting in a labeling budget of 20 % of the

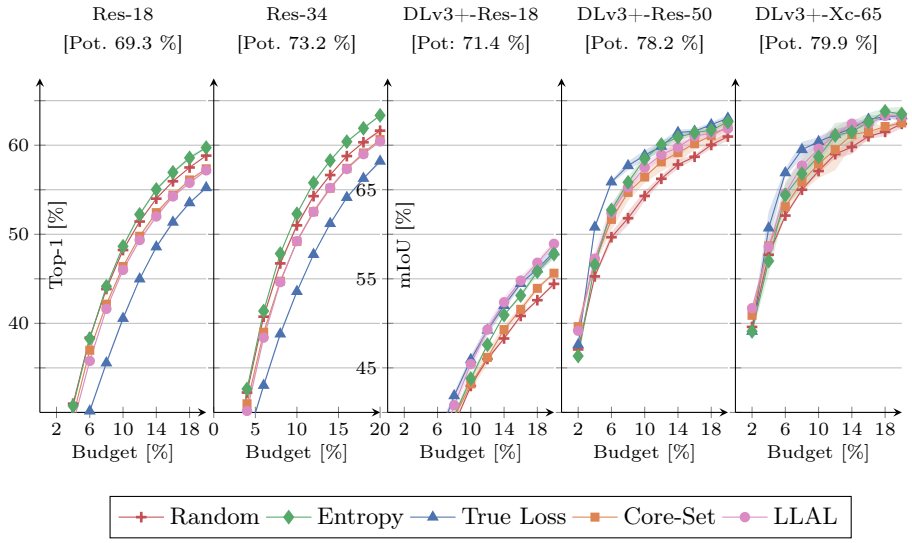

Fig. 3: Active learning baselines using learned features from ResNet-18 (Res-18) and ResNet-34 (Res-34) for ImageNet, and DeepLabv3+-ResNet-18 (DLv3+-Res-18), DeepLabv3+-ResNet-50 (DLv3+-Res-50) and DeepLabv3+-Xception-65 (DLv3+-Xc-65) for CityScapes. Corresponding performances of model potentials are indicated above the individual plots in square brackets.

total unlabeled pool. As data selected from previous cycles affects the selection process of future cycles, all active learning experiments start with the same initial selection of randomly chosen data samples. It is worth mentioning that we report the average and standard deviations over three runs for all experiments.

5.2 Active Learning Baselines Using Various Models

In this section, we consider the classic active learning cycle using static models. Here, we perform extensive experiments by switching to DNNs of various types and sizes, thus offering different learned features to the selection function. Fig. 3 visualizes the prediction capability of a model over an increasing training data budget when using five different active learning methods, namely Entropy [♦], LLAL [●], Core-Set [■], True-Loss [▲] and Random [+]. We feed the selection functions with learned feature representations of a ResNet-18 and ResNet-34 for ImageNet, and features of a DeepLabv3+ architecture using ResNet-18, ResNet-50 and Xception-65 backbones for CityScapes. Prediction capabilities of different models can be observed from left to right.

In general, models of various types and sizes offer different prediction capabilities when training on the complete dataset, which we will refer to as *potential* (see brackets in the plot titles of Fig. 3). For example, the DeepLabv3+ architecture using a Xception-65 as backbone achieves 79.9 % mIoU when trained

Table 3: Prediction capability (mIoU [%]) of deployment models (rows) when trained on CityScapes data selected through surrogate models (columns).

Depl. Model	**Surrogate Data Collection Model**					random
	Res-20	MNV3	Res-18	Res-50	Xc-65	
Res-20	59.6 ± 0.5	60.1 ± 0.3	59.7 ± 0.3	58.6 ± 0.2	58.4 ± 0.6	57.8 ± 0.5
MNV3	63.8 ± 0.3	64.2 ± 0.2	63.6 ± 0.3	63.7 ± 0.4	61.8 ± 0.3	61.0 ± 0.4
Res-18	57.2 ± 0.4	57.2 ± 0.7	57.6 ± 0.6	56.4 ± 0.3	56.6 ± 0.2	54.8 ± 0.5
Res-50	73.5 ± 0.4	74.4 ± 0.3	74.4 ± 0.1	73.7 ± 0.1	73.8 ± 0.3	71.2 ± 0.3
Xc-65	74.8 ± 0.3	75.3 ± 0.2	74.9 ± 0.3	75.5 ± 0.2	74.6 ± 0.3	72.4 ± 0.6

on CityScapes. We observe that both actively learned ResNet-18 and ResNet-34 achieve approximately $\times$0.80-0.87 of their respective potential. The same holds for the task of semantic segmentation when actively training a DeepLabV3+ using a ResNet-50 and Xception-65 backbone, achieving approximately $\times$0.91-0.93 of their potential. However, when actively training a DeepLabV3+ using a ResNet-18 backbone, not only the prediction capability drops, but also only achieves approximately $\times 0.76 - 0.82$ of its already low potential. Intuitively, one would pick feature representations of a model proven to be effective on the task for the selection method. However, in the next sections, we will show that this might not always be the optimal choice.

5.3 Effectiveness of Surrogate Models

In this section, we proceed by allowing different models to act as *surrogates* to select and curate a training dataset for next generation models of varying sizes and types. As entropy-based active learning selection has proven to be effective across tasks, datasets, and model sizes (see Fig. 3), we use it for the following experiments. Tab. 3, 4 and 5 report the prediction capability (Top-1/mIoU) of various deployment models (rows) when trained on data selected through models as active learning surrogates (columns) for the CityScapes, ADE20K and ImageNet dataset respectively. We use a color coding (applied to every row) to indicate the effectiveness of a surrogate model to choose data for a given deployment model, from worst to best.

Surrogate vs. Random Selection. We show the capability of models as active learning surrogates to select data for future deployment models across a variety of tasks, datasets, model sizes and types when compared to training future deployment models on randomly curated datasets. This manifests in an improvement of +3.2 p.p. when the surrogate DeepLabv3+-ResNet-18 chooses data for the deployment model DeepLabv3+-ResNet-50. Counterintuitively, recall that DeepLabv3+-ResNet-18 had the worst baseline accuracy potential (see Fig. 3). This is the first indication, that smaller models as surrogates do not necessarily have to result in a compromise in the active learning cycle, when their successor is a more complex larger model. We analyze this aspect further

Table 4: Prediction capability (mIoU [%]) of deployment models (rows) when trained on ADE20K data selected through surrogate models (columns).

Depl. Model	**Surrogate Data Collection Model** Res-18	Res-50	Xc-65	random
Res-18	11.30 ± 0.01	11.26 ± 0.11	10.30 ± 0.15	11.49 ± 0.14
Res-50	31.17 ± 0.01	30.99 ± 0.12	29.84 ± 0.04	30.36 ± 0.36
Xc-65	31.36 ± 0.02	31.18 ± 0.12	30.98 ± 0.36	31.02 ± 0.02

Table 5: Prediction capability (Top-1 [%]) of deployment models (rows) when trained on ImageNet data selected through surrogate models (columns).

Depl. Model	**Surrogate Data Collection Model** Res-18	Res-34	random
Res-18	59.19 ±0.12	59.27 ±0.19	58.47 ±0.01
Res-34	62.16 ±0.08	62.22 ±0.26	60.83 ±0.10

in Sec. 5.4. Similarly, the deployment model ResNet-34 achieves an improvement of +1.33 p.p. when its data is collected through a surrogate ResNet-18 on the ImageNet dataset. For ADE20K, DeepLabv3+-ResNet-50 achieves an improvement of +0.81 p.p. when training on data selected by the surrogate DeepLabv3+-ResNet-18. However, when relying on data collected through a DeepLabv3+-Xception-65, deployment models fall short up to −1.19 p.p. when compared to random selection. This is the second indication, that smaller models as surrogates do not necessarily have to result in a compromise and can actually outperform a more complex data selection model. In total, we formed 38 model pairs, of which 28 are surrogate/deployment pairs (excluding models picking data for themselves, i.e. diagonal entries in the tables). 25 out of the 28 surrogate/deployment pairs achieved better prediction capability when training the deployment model on data selected by its surrogate when compared to training the deployment model on randomly selected data.

Surrogate vs. Surrogate. We compare the prediction capability of the deployment models when trained on data selected through different surrogate models (arranged in rows). Consider a DeepLabv3+ using Xception-65 as backbone; on the one hand, it has proven to be effective in its potential on both the CityScapes and ADE20K dataset, but on the other hand it performs poorly as a surrogate for selecting data for other deployment models (see red cells in Tab. 3 and 4). Contrary, using a MobileNetV3 backbone selects three times the best and two times the second best training dataset for other deployment models from the CityScapes dataset (see green cells in Tab. 3), while offering less learning potential when compared to the Xception-65 backbone variant in

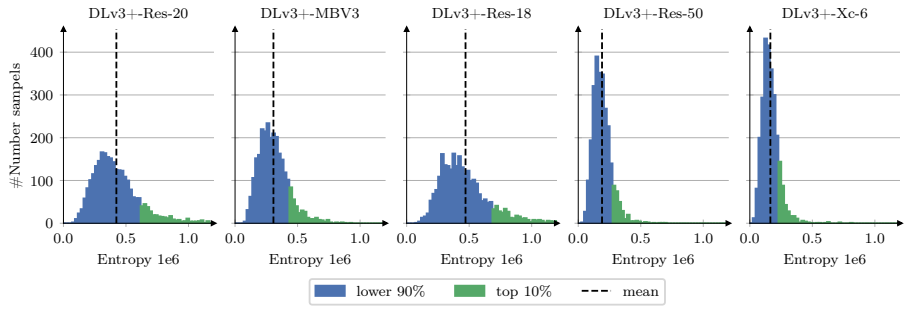

Fig. 4: Prediction entropy of different DeepLabv3+ variants after training them on the same 10% random CityScapes train image subset and evaluated on the 90% unseen images from the set; Investigated DeepLab variants are DeepLabv3+-MobileNetV3 (DLv3+-MBV3), DeepLabv3+-ResNet-{20, 18, 50} (DLv3+-Res-{20, 18, 50}), and DeepLabv3+-Xception-65 (DLv3+-Xc-65).

Sec 5.2. This manifests in an improvement of +1.7 p.p. for the deployment model DeepLabv3+-ResNet-20.

Discussion. The large-scale empirical results lead to the following findings: (1) Surrogate models showed the capability to collect data for future DNNs for the task of semantic segmentation. This allows training data of previous MLOps iterations to be reused and kick-start active learning cycles in next iterations (surrogate data collection outperforms a random selection in 25 out of 28 cases). (2) The type of surrogate has a huge impact on the prediction capability of the deployment model. Counterintuitively, models that have proven to be effective on the task of semantic segmentation (high potential), do not necessarily result in good training data selection for different types of deployment models and, in contrary, are worse in most cases. In the following section, we investigate this point further.

5.4 Model-Dependent, Entropy-based Selection

To gain more insights into the different selection behavior of models, we examine the entropy score of various model variants on the Cityscapes dataset. The experimental setup trains the DeepLabv3+ architecture using five backbones, specifically, MobileNetV3, ResNet-20/18/50 and Xception-65 on 275 randomly selected samples (10 %), to ensure learned model features. Fig. 4 plots the entropy histogram over the remaining, unseen 2678 samples (90 %) for all model variants. Further, samples with the top 10% entropy values are highlighted in green.

As is clear when looking at the histograms, both the absolute entropy values and the mean entropy are much smaller for the bigger networks. This indicates that larger networks which exhibit reduced mean entropy have more capacity to learn the task at hand with less uncertainty on the unlabled dataset. However,

smaller distribution of entropy values decreases the separation between the different bins of the histogram, which is evident in the much wider distributions of the entropy values of the smaller networks. Considering the high selection quality of smaller models as found by our experiments in Sec. 5.3, we hypothesize that the wider gaps in entropy values could lead to better differentiation between important and unimportant images, explaining the improved selection performance of the small networks.

6 Conclusion

In this work, we investigate active learning in the context of an evolving MLOps cycle and show that the selected/labeled data of previous MLOps iterations with a surrogate model should be reused in the next-generation for training successor DNNs. This demystifies the method of data collection of the initial unlabeled pool by the surrogate, which is capable of running on the current iteration's hardware fleet. We show that a surrogate model for DeepLabv3+-ResNet-50 provides an +3.2 p.p. mIoU improvement on average, using uncertainty scores over randomly selecting data to train the deployment model on the CityScapes dataset. Counter-intuitively, we show that in almost all cases using the model with the best prediction capabilities as a surrogate data collection model to curate a training dataset results in bad prediction capabilities of the deployment models for the task of semantic segmentation. For example, this can lead to a loss of prediction capability up to -2.4 p.p. on the CityScapes. With these results, we want to motivate the domain to further investigate important aspects and characteristics of a good data picker, beside improving scoring functions for the selection process. For future work, the concept of models as active learning surrogates should be investigated across more diverse architectures, such as CNNs as surrogates to vision transformers. Furthermore, surrogates in active learning and MLOps should be investigated with compression in the loop, to identify limits of the learning capacity of deployed models, acting as surrogates for future releases.

References

1. Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations (ICLR)*, 2020.
2. Zalán Bodó, Zsolt Minier, and Lehel Csató. Active learning with clustering. In *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, Proceedings of Machine Learning Research. PMLR, 2011.
3. Liang-Chieh Chen, Yukun Zhu, George Papandreou, et al. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In *ECCV*, 2018.
4. 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. In *International Conference on Learning Representations (ICLR)*, 2020.

5. Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
6. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
7. Alex Holub, Pietro Perona, and Michael C. Burl. Entropy-based active learning for object recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2008.
8. Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.
9. Xingjian Li, Pengkun Yang, Tianyang Wang, Xueying Zhan, Min Xu, Dejing Dou, and Chengzhong Xu. Deep active learning with noise stability. *arXiv preprint arXiv:2205.13340*, 2022.
10. Yadan Luo, Zhuoxiao Chen, Zijian Wang, Xin Yu, Zi Huang, and Mahsa Baktashmotlagh. Exploring active 3d object detection from a generalization perspective. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.
11. Mengyao Lyu, Jundong Zhou, Hui Chen, Yijie Huang, Dongdong Yu, Yaqian Li, Yandong Guo, Yuchen Guo, Liuyu Xiang, and Guiguang Ding. Box-level active detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.
12. NVIDIA Corporation. Nvidia jetson orin nx series. In *https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/*.
13. NVIDIA Corporation. Nvidia jetson tx2 nx module. In *https://developer.nvidia.com/embedded/jetson-tx2-nx*.
14. NVIDIA Corporation. Nvidia jetson xavier nx series. In *https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-xavier-series/*.
15. Amin Parvaneh, Ehsan Abbasnejad, Damien Teney, Gholamreza Reza Haffari, Anton Van Den Hengel, and Javen Qinfeng Shi. Active learning by feature mixing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
16. Olga Russakovsky, Jia Deng, Hao Su, et al. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015.
17. Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations (ICLR)*, 2018.
18. H Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, 1992.
19. Yichen Xie, Masayoshi Tomizuka, and Wei Zhan. Towards general and efficient active learning, 2022.
20. Donggeun Yoo and In So Kweon. Learning loss for active learning. In *IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2019.
21. Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision (IJCV)*, 2019.