Characterization of AI Model Configurations For Model Reuse

Anonymous ECCV submission

Paper ID 100

Abstract. With the widespread use of artificial intelligence (AI) models in biosciences, researchers need the ability to characterize AI models trained for specific tasks to support model reuse and dissemination. This work is motivated by characterizing AI models based on metrics derived from optimization curves captured during model training. Such AI model characterization can aid future model accuracy refinement, inform users about model hyper-parameter sensitivity, and assist in model reuse according to multi-purpose objectives. The challenges lie in designing quantitative AI model metrics, validating them, and disseminating them with shared pre-trained AI models. We approach these challenges by designing nine metrics derived from optimization curves collected during model training and by evaluating them on image segmentation tasks. The results demonstrate the value of quantitative metrics for multi-purpose reuse of AI models, as well as a use-case for recommending AI model architectures.

Keywords: Efficient training and inference methods; Fairness, accountability, transparency, and ethics in vision; Optimization and learning methods; Medical, biological, and cell microscopy

1 Introduction

The problems of reusing artificial intelligence (AI) models range from defining a standard AI model file format to sharing the code and AI models via repositories [7]. Multiple communities come together to define a standard file format, such as Open Neural Network Exchange (ONNX)[4], and agree on sharing application code, installation software dependencies, AI frameworks, and packaging via open framework projects (e.g., Conda, Colaboratory, PyTorch, TensorFlow), code and model repositories (e.g., GitHub, BitBucket, Model Zoo, Model Depot, TensorFlow Hub), and software packaging and distribution solutions (e.g., Docker, Apache Zookeeper, Apache Kafka) [7]. In the broad range of AI model reusability problems, our focus is on specific sub-problems related to characterizing AI models for the purpose of value added to parties reusing the models.

The need within the scientific imaging community for AI model characterization is driven by several factors. First, the scientific community strives for reproducible research results. Second, domain-specific applications with focus on special objects of interest acquired by unique imaging modalities struggle with insufficient training data (in comparison to typical imaging modalities and objects in computer vision datasets,

e.g., ImageNet or Microsoft Common Objects in Context (COCO)). Finally, the sciences struggles with a general lack of computational resources for AI model training compared to the resources available to large companies. We listed several example tasks with needed inputs for AI model reuse in Table 1. Characterizing AI models improves the input metadata about AI models for reuse and reproducibility. Additionally, model reuse saves computational resources and time while providing higher final model accuracy.

Table 1. Example tasks for AI model reuse

Task	Needed
Inference on new hardware	Trained model GPU utilization
Reproduce training	Training data Model architecture Optimal configuration Speed of training
Train more to improve model accuracy	Training data Trained model Convergence prediction Explored configurations
Establish model robustness	Training datasets Trained model Explored configurations
Select architecture for transfer learning	Training datasets Explored architectures Explored configurations

Our specific problem is illustrated in Figure 1 where the *recorded metadata about training* box contains the optimization curves and the *extraction of metrics about configurations* box computes the metrics to accompany an AI model. Our assumption is that data collected during training sessions are common to all modeling tasks including image classification and segmentation (tasks of our interest) and, therefore, the characteristics can be applied to a general set of AI models. Our approach to designing characteristics of AI models consists of three steps: (1) Develop experimental framework for collecting optimization curves from AI model training sessions; (2) Design and extract characteristics of AI models from optimization curves; (3) Visualize and validate AI model characteristics Our approach derives AI model characteristics from optimization curves collected during model training sessions in which researchers executes multiple training runs with different hyper-parameter configurations (e.g., AI architecture, model initialization, learning rate, batch size, and training datasets) to create an optimal model. The optimization curves are typically pairs of AI model accuracy metrics collected over many epochs from train and validation/test/holdout datasets (denoted in this work as train and test datasets).

The overarching challenges lie in (a) computational resources needed to simulate a large number of optimization curves generated by production-level training of AI mod-

els, (b) limited information content in optimization curves that combine contributions from model architecture, training hyper-parameters, and training dataset, and (c) limited a priori knowledge about relationships among parts of AI solutions (see Table 3) that could be used for validation of quantitative AI model metrics. Our technical challenges are in designing quantitative AI model characteristics (metrics) for ranking AI model configurations (datasets, model, training process), validating them, and integrating the metrics into on-going efforts to generate accompanying metadata for each disseminated pre-trained AI model [12].

Relation to prior work: The concept of describing AI models has already been discussed in the past (Datasheets for datasets [5], The Dataset Nutrition Labels [8,18], Google AI Model Cards [12]). The published work on Datasheets for datasets and Dataset Nutrition Labels has been focused mainly on training datasets from the perspective of fairness. The fairness aspect is documented via data attributes, motivation for collection, data composition, collection process, and recommended uses in [5], as well as via design of ranking widgets in [8]. In contrast to [8,18,5], our work is focused on documenting lessons-learned from the optimization curves collected during training sessions. While a placeholder for model performance measures has been designated in the AI model cards [12] (i.e., under Metrics heading), the metrics have not been defined yet, which is the gap our work is trying to address. In addition, our work aims at utilizing the information that is not preserved with disseminated AI models for the multi-purpose reuse of the AI models right now, although multiple platforms for optimizing AI model configurations have been designed, such as TensorBoard [1] or Experiment Manager [11], and many optimization curves are generated over a range of hyper-parameters.

Our contributions are (a) in leveraging information from log files of hyper-parameter AI model optimization campaigns performed by the entity sharing a pre-trained AI model for reuse and (b) in defining specific characteristics of AI models for model accompanying metadata, for example, metadata in AI model cards. The novelty of this work lies in defining computable metrics that characterize AI models and provide cost savings for further reuse of AI models.

2 Methods

Our approach to defining AI model characteristics from pairs of train and test optimization curves consists of three steps: (a) classify parts of an AI solution and their attributes, (b) design mathematical functions used in analytical and statistical analyses that characterize each part, and (c) compute quantitative characteristics over a range of AI model hyper-parameter configurations and several training datasets. The mathematical functions are applied to accuracy or error measurements collected during training, for instance, to cross entropy loss H^{CE} computed for training and testing datasets according to Table 2 and Equations 1-9. The output quantitative AI model characteristics serve multiple purposes: as entries under *Metrics* in the AI model card definition [12] and as inputs to ranking of AI models according to a variety of objectives (e.g., model accuracy refinement, model architecture recommendation).

AI model parts: We divided the AI model solution into three parts: model architecture, training process, and training dataset. The parts have associated attributes that

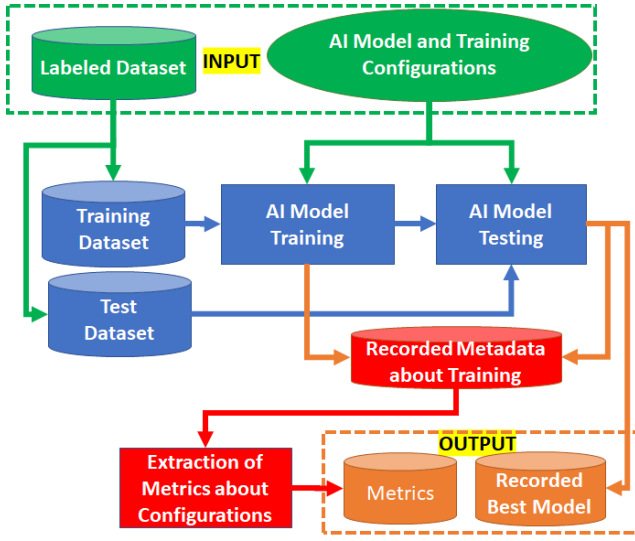


Fig. 1. A framework for extracting metrics about AI model configurations.

define feature engineering and input/output relationship (model architecture), optimization search strategy, and initial search point as well as computing limitations (training process), cross-validation distribution of information, and alignment of pre-training vs. training datasets (training dataset).

AI model characteristics: The AI model characteristics are defined as sums, deltas, correlations, and extreme points as well as least-squared fits of power and exponential models to optimization curves from a varying number of data points. Equations 1-9 denote the index of each epoch as ep, number of epochs as EP, the epoch for which a model achieves the minimum error as $ep^*(M_{er})$, the window around $ep^*(M_{er})$ as $\pm\delta$, initializations as $rand$ (random) or $pretrain$, execution time as T, correlation of two curves as ρ, and utilization of memory and processing power of a graphics processing unit as GPU^{mem} and GPU^{util}. In this work, we assume that the optimized error metric M_{er} per AI model is the cross entropy (CE) loss since it is widely used and supported by common AI libraries [15,1].

In Equation 5, the value of $H^{CE,fit}$ represents a predicted CE values from the first few epochs given power or exponential models for the least-squared fit approximation. In our analyses, we refer to the power model $a*x^b$ as PW and the exponential model $a*b^x$ as EXP for $a, b \in \mathbb{R}$ and $x = H^{CE}(ep)$. The metric $\Delta(fit)$ is a difference between predicted $H^{CE,fit}$ and measured H^{CE} cross entropy loss values. $\Delta(fit)$ is designed as an optimization cost function for finding the most accurate convergence prediction model ($\min \Delta(fit)$ constrained by the maximum number of measured epochs over two models $\{Model = PW, EXP\}$ and three sets of AI model optimization curves constrained by maximum of $\{10, 15, 20\}$ measured epochs.

AI model configurations: For each AI model configuration, one pair of train and test optimization curves is generated. Figure 3 shows an example of two pairs of op-

timization curves with fluctuations due to the optimization path in a high-dimensional variable space (i.e., the space is discontinuous and/or frequently flat without expected extrema, the objective evaluation fails due to sparse discrete objective space formed by integer and categorical variables and/or numerical difficulties and hardware failures [16]). Parties that share AI models for reuse have scripts or tools that automate sweeping a range of AI model parameters, such as, Optuna [2], Vizier [6], Autotune [9], or Experiment Manager [11], while following random, grid, simulated annealing, genetic algorithm, or Bayesian search strategies. The AI model characteristics can be derived from a set of pairs of optimization curves corresponding to many AI model configurations. We gather optimization curves from training sets of AI model configurations and from multiple image segmentation datasets to deliver parallel coordinate graph visualizations for multiple purposes of AI model reuse.

Table 2. Definition of AI model characteristics

Parts of AI solution	Measurement name & symbol	Eq.
Architecture	model error $M_{er}, ep^*(M_{er})$	1
	model stability M_{stab}	2
Training process	speed $T(M_{er})$	3
	initialization gain G_{init}	4
	predictability $\Delta(fit)$	5
	GPU utilization $\mathrm{GPU}^{MaxM}, \mathrm{GPU}^{AvgU}$	6
Training data	Compatibility D_{cm} (training data and architecture)	7
	Uniformity D_{unif} (training and testing data)	8
	Compatibility D_{init} (pre-trained data and domain data)	9

$$M_{er} = \min_{ep}(H^{CE}_{test}(ep)) \qquad ep^*(M_{er}) = \underset{ep}{\operatorname{argmin}} H^{CE}_{test}(ep) \tag{1}$$

$$M_{stab} = \sum_{ep=ep^*(M_{er})-\delta}^{ep^*(M_{er})+\delta} (H^{CE}_{test}(ep) - M_{er}) \tag{2}$$

$$T(M_{er}) = ep^*(M_{er}) * \frac{1}{EP} * \sum_{i=1}^{EP} T_i \tag{3}$$

$$G_{init} = M_{er}^{rand} - M_{er}^{pretrain} \tag{4}$$

$$\Delta(fit) = \sum_{ep=1}^{EP} (H_{test}^{CE}(ep) - H_{test}^{CE,fit}(ep)) \tag{5}$$

$$\text{GPU}^{MaxM} = \max_{ep}(\text{GPU}^{mem}(ep))$$
$$\text{GPU}^{AvgU} = \frac{1}{EP} * \sum_{ep=1}^{EP} \text{GPU}^{util}(ep) \tag{6}$$

$$D_{cm} = \sum_{ep=1}^{EP} (H_{train}^{CE}(ep) + H_{test}^{CE}(ep)) \tag{7}$$

$$D_{unif} = \rho(H_{train}^{CE}(ep), H_{test}^{CE}(ep)) \tag{8}$$

$$D_{init} = \sum_{ep=1}^{EP} (H_{test}^{CE,rand}(ep) - H_{test}^{CE,pretrain}(ep)) \tag{9}$$

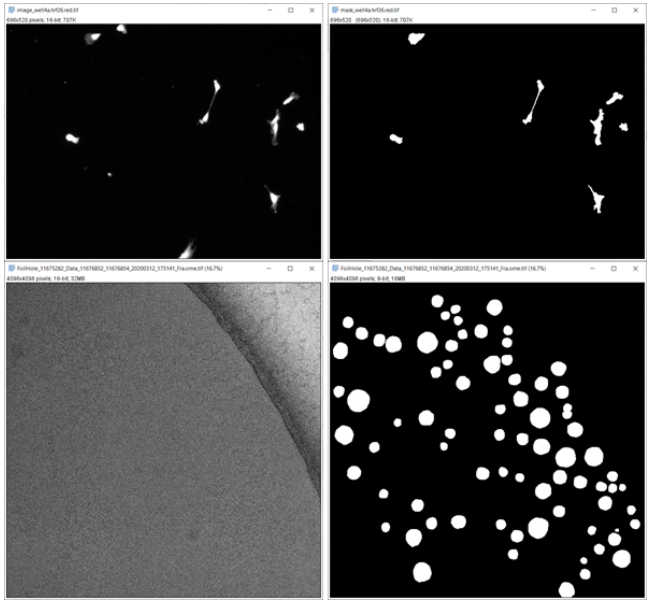


Fig. 2. Examples of training image pairs (intensity, segmentation mask) for A10 dataset (top row), and cryoEM dataset (bottom row).

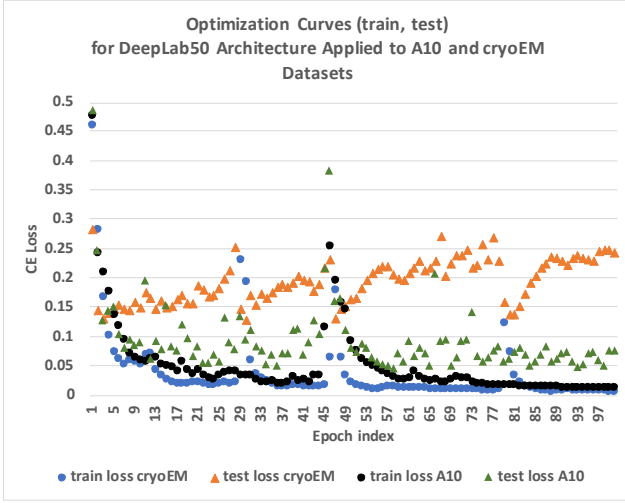


Fig. 3. Optimization curves (train, test) for DeepLab50 AI architecture trained on A10 and cryoEM datasets.

3 Experimental Results

We divided the experimental work into (1) building the framework for generating optimization curves over a range of AI model configurations, (2) validating the designed metrics based on a couple of datasets and their *prior* characterization of segmentation difficulties, and (3) describing multi-purpose reuses of AI models in Section 4.

Experimental framework: We explored the hyper-parameters in the AI model configurations shown in Table 3. All model architectures were implemented using the PyTorch library [15]. The model initialization using pre-trained coefficients was based on the COCO dataset [10] for object segmentation (1.5 million object instances). The objective was to sample a set of AI model configurations by collecting the optimization curves and then probing their accuracy, predictability, and compatibility with AI model architecture and training dataset.

We chose to train each model configuration for $EP = 100$ epochs. In general, this value will vary during hyper-parameter optimization runs depending on available computational resources, the definition of model convergence error, or the use of early stopping criterion (an increment observed in CE loss values over consecutive epochs is smaller than ϵ). We also set the value $\delta = 5$ epochs in Equation 2. All computations were performed on a compute node with a Quadro Ray Tracing Texel eXtreme (RTX) 4000 GPU card and Compute Unified Device Architecture (CUDA) 11.6.

Training datasets: We analyzed five datasets representing optical florescent, optical bright-field, electron, cryogenic electron, and neutron imaging modalities. The training image datasets are characterized in Table 4 in terms of the number of predicted classes

(#Classes), the number of pixels (#Pixels), and the average coefficient of variation ($\overline{CV}$) over all training images as defined in Equation 10.

$$\overline{CV} = \frac{1}{N}\sum_{i=1}^{N}\frac{\sigma_i}{\mu_i} \tag{10}$$

where μ_i and σ_i are the mean and standard deviation of each intensity image in the training collection of size N images. The A10 dataset denotes fluorescently labeled optical microscopy images of A10 cells [14]. The concrete dataset came from electron microscopy of concrete samples [3]. The cryoEM dataset was prepared by the authors using cryogenic electron microscopy of lipid nanoparticles. The infer14 dataset was prepared by the authors using data-driven simulations of porous concrete samples from measured neutron images [13]. The rpe2d dataset denotes time-lapse bright-field optical microscopy images of retinal pigment epithelial (RPE) cells published in [17].

Validation of AI characteristics: We selected two training image datasets labeled as A10 and cryoEM in Table 4 for validation. Examples of training image pairs are shown in Figure 2. The A10 dataset has a high contrast (the average coefficient of variation ($\overline{CV}$) per image is 1.34) while the cryoEM dataset has a low contrast (average image $\overline{CV}$ is 0.06) and a large heterogeneity in sizes and textures. The datasets were chosen based on the assumption that segmenting images with low contrast is a much harder task than segmenting images with high contrast.

Given the assumption about segmentation difficulty, one can validate AI model characteristics against expected inequalities to be satisfied by the values derived from these two datasets. The expected inequalities include model error M_{er}, uniformity of training and testing data D_{unif}, and convergence predictability $\Delta(fit)$ as shown in Equation 11. These inequalities are validated by comparing the values of M_{er} and D_{unif} in Table 5. Ideally, the correlation of train and test CE loss curves D_{unif} should be close to one.

Figure 5 shows the values in parallel coordinate plots including the values of $\Delta(fit, \text{A10})$ and $\Delta(fit, \text{CryoEM})$ on the right most vertical line denoted as $\min P(PW\_20)$. The values of $\min P(PW\_20)$ were calculated using the power model fit from the first 20 epochs. The sum of train and test optimization curves D_{cm}, as well as the convergence predictability $P(PW\_20)$, quantify the sensitivity of model training to hyperparameters (i.e., learning rate and initialization). In both A10 and CryoEM datasets, D_{cm} and $P(PW\_20)$ values for 2-3 architecture types indicate epoch-specific optimization divergence (as illustrated by the scattered black points in Figure 4(b) for the A10 dataset).

$$\begin{aligned} M_{er}(\text{A10}) &< M_{er}(\text{CryoEM}) \\ D_{unif}(\text{A10}) &> D_{unif}(\text{CryoEM}) \\ \Delta(fit, \text{A10}) &< \Delta(fit, \text{CryoEM}) \end{aligned} \tag{11}$$

Reuse of AI models with convergence predictability: The graphs in Figures 3 and 4 illustrate the tradeoffs in reusing AI models for several tasks in Table 1.. The configuration in Figure 4(b) achieves desirable lower cross entropy error M_{er} (vertically lowest black point) but undesirable higher training divergence $\Delta(fit)$ (large deviations of black points from predicted curves) than the configuration in Figure 4(a) for the same

Table 3. Explored hyper-parameters in AI model configurations

Hyper-parameters	Values of hyper-parameters
Architecture	DeepLab 50 DeepLab 101 MobileNetV3-Large LR-ASPP-MobileNetV3-Large FCN Resnet 50 FCN Resnet 101
Initialization	Random COCO pre-trained
Learning Rate	10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}
Optimizer	Adam
Optimization criterion	Cross entropy loss
Epochs	100
Batch size	2
Class balance method	Weighting by 1 – class proportion
Augmentation	None
Train-Test dataset split	80 : 20

Table 4. Training datasets. OF - optical fluorescent, EM - electron microscopy, OB - optical bright field, NI - neutron imaging

Dataset	Modality	#Classes	#Pixels [MPix]	$\overline{CV}$
A10	OF	2	5.79	1.34
concrete	EM	4	71.7	0.31
cryoEM	EM	2	117.44	0.06
infer14	NI	9	125.9	0.24
rpe2d	OB	2	53.22	0.84

Table 5. Summary of AI model characteristics per model architecture and per dataset where the models were optimized over the learning rates and pretraining options listed in Table 3.

Architecture	A10 M_{er}	A10 D_{unif}	cryoEM M_{er}	CryoEM D_{unif}
DeepLab101	0.0528	0.3513	0.1271	-0.0093
DeepLab50	0.0451	0.8356	0.1284	-0.2515
LR-ASPP	0.04	0.7978	0.1435	-0.3585
MobileNetV3	0.0568	0.5794	0.1602	0.3092
ResNet101	0.042	0.2256	0.1379	-0.0867
ResNet50	0.045	0.391	0.1369	-0.2269

dataset A10 and the same DeepLab50 AI architecture. A large divergence from the predicted optimization curves indicates that (a) it is not sufficient to predict the model training convergence using a few initial epochs (10, 15, or 20 epochs), (b) training and testing subsets might not have been drawn from the same distribution, and (c) the COCO dataset used for pretraining the AI model might not be compatible with the domain training dataset, and, hence, the test CE loss values vary a lot during the first few epochs. This is undesirable for researchers who would like to predict how many more epochs to run on the existing model while targeting a low test CE loss value.

Reuse of AI models with pretraining gain: Figure 6 shows the AI model initiation gain G_{init}, as defined in Equation 4, for DeepLab101 architecture and the five datasets listed in Table 4. The values of G_{init} are less than zero for all datasets except the concrete dataset. These values indicate that the objects in the COCO dataset are significantly different from the objects annotated in the five scientific microscopy datasets, and the pre-training on COCO does not yield better model accuracy.

Reuse of AI models with model stability: Figure 7 illustrates how stable each optimized AI model – per architecture and per training dataset – is over the hyperparameters listed in Table 3. If the test CE loss curve is close to constant within the neighborhood of $\delta = 5$ epochs, then the value of M_{stab}, as defined in Equation 2, is small indicating model stability. Based on Figure 7, all model architectures for the rpe2d dataset yielded highly-stable, trained models, while the stability of trained models for the infer14 dataset was low and varied depending on a model architecture.

4 Discussion

How to use AI model characteristics? The designed metrics are useful for the use cases listed in Table 1. The practical value of each metric is task dependent (e.g., convergence is of interest to improving accuracy but less important for inference on a new hardware). For instance, if the task is model portability to a new hardware or model-training reproducibility, then knowing maximum required GPU memory and GPU utilization would be very valuable.

The metrics are dataset-specific, but useful for matching datasets and AI model architectures. For example, if a researcher wants to reuse an existing AI model for the transfer learning task with a small set of optical microscopy images of A10 cells, then he/she would benefit from knowing what architectures and imaging modality datasets led to the most accurate shared AI model that was trained on an electron microscopy training dataset of RPE cells and transfer-learned on optical microscopy images of 3T3 cells.

Limitations of AI model characteristics We do not recommend drawing conclusions based on metric values when architectures, training parameters, and training datasets vary because the metrics are limited by their information content. The optimization curves entangle multiple sources of CE loss variability (as stated in the list of challenges in Section 1). Nevertheless, the metrics are useful if one has some apriori knowledge about the training datasets and the complexity of predicting image segmentation as illustrated in our validation.

5 Summary

This work presented design and validation of quantitative AI model metrics from a set of optimization curves. The designed metrics were evaluated on image segmentation tasks applied to image datasets selected based on their estimated segmentation level of difficulty. Our main results demonstrated use cases of scientists reusing pre-trained AI models for the purposes of (1) improving model accuracy by further training/optimization of model hyper-parameters constrained by computational resources (convergence predictability in Figure 4) and (2) selecting an AI model architecture that is best suited in terms of model accuracy, predictability of training convergence, and compatibility with training datasets (metric comparisons across architectures Figure 5).

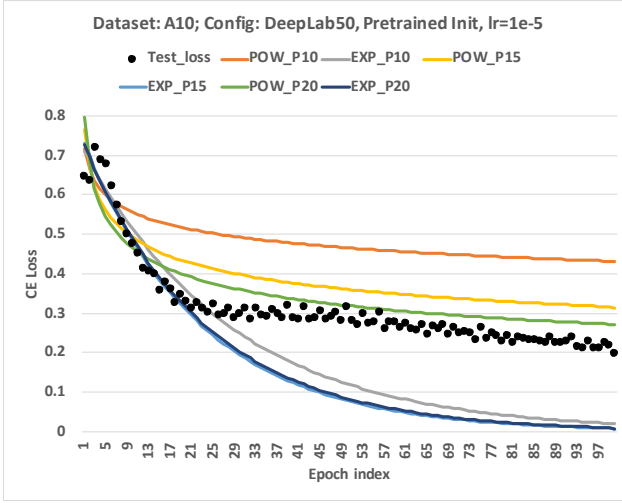
The impact of sharing AI models with presented metrics is significant for principal investigators limited by their grant budgets and small research labs limited by their own computational resources or the cost of cloud resources. A higher reuse of shared AI models can save not only cost and time to researchers but also advance their scientific goals more efficiently. The cost of achieving a higher reuse of AI models is the extra summarization of optimization sessions using transparent metrics and sharing them in AI model cards. In the future, we plan to research calibration protocols that would enable comparing AI model metrics across datasets and tasks.

6 Disclaimer

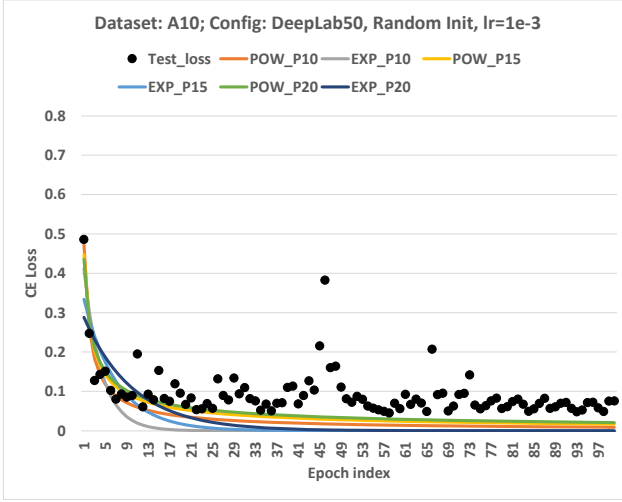
after blind review

7 Acknowledgement

after blind review

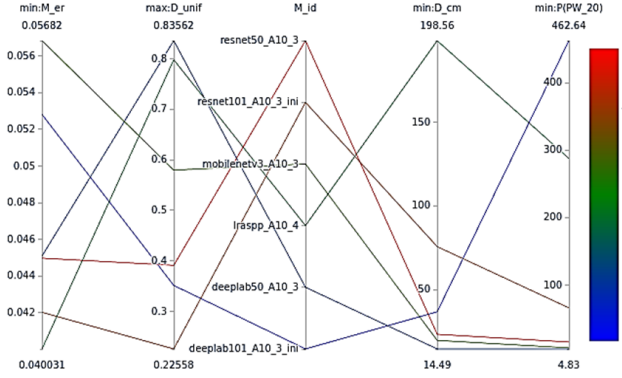


(a)

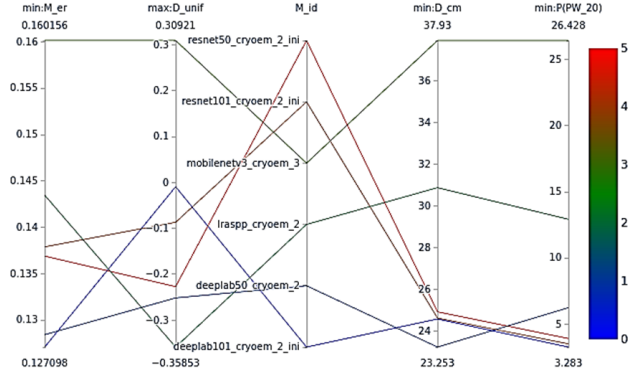


(b)

Fig. 4. Predictions of training model convergence from A10 dataset for two configurations. Top configuration: (DeepLab50, COCO pre-trained initialization, learning rate: 1e-5). Bottom configuration: (DeepLab50, random initialization, learning rate: 1e-3). Black dots are the measured test CE loss values. Color-coded curves are predictions for a set of fitted model parameters.



(a)



(b)

Fig. 5. Parallel coordinate plots for A10 (top) and CryoEM (bottom) datasets. The plots are intended to support decisions about which AI model architecture is the most accurate for the image segmentation tasks.

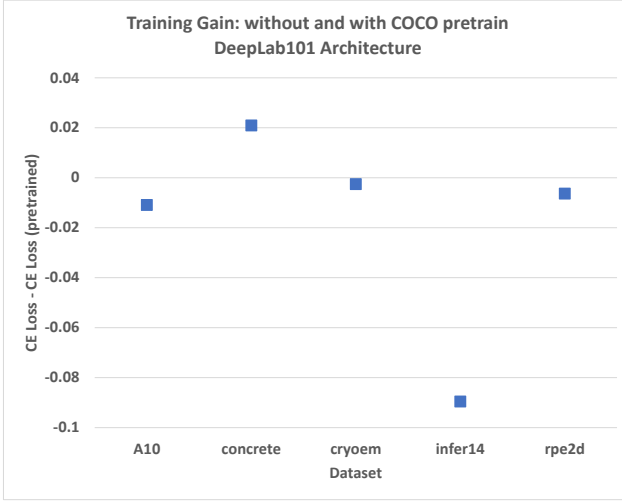


Fig. 6. Training gains from initializing DeepLab101 AI architecture using the COCO dataset.

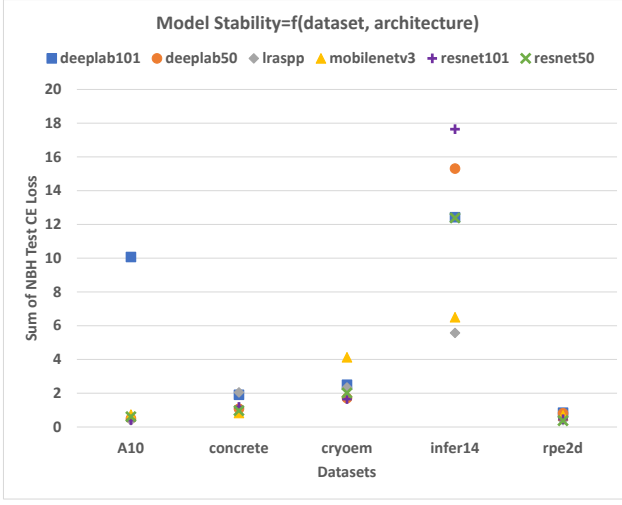


Fig. 7. Model stability of the most accurate AI model per architecture and per dataset.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015). https://doi.org/10.5281/zenodo.5898685, `https://www.tensorflow.org/`, software available from tensorflow.org
2. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework (2019)
3. Bajcsy, P., Feldman, S., Majurski, M., Snyder, K., Brady, M.: Approaches to training ai-based multi-class semantic image segmentation. Journal of Microscopy **279(2)**, 98–113 (2020). https://doi.org/http://dx.doi.org/10.1111/jmi.12906, `https://pubmed.ncbi.nlm.nih.gov/32406521/`
4. Community: Open neural network exchange (ONNX). https://onnx.ai/ (2022), `https://onnx.ai/`
5. Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J.W., Wallach, H., III, H.D., Crawford, K.: Datasheets for datasets. arXiv 1803.09010 (2021)
6. Golovin, D., Solnik, B., Moitra, S., Kochanski, G., Karro, J.E., Sculley, D. (eds.): Google Vizier: A Service for Black-Box Optimization (2017), `http://www.kdd.org/kdd2017/papers/view/google-vizier-a-service-for-black-box-optimization`
7. Haibe-Kains, B., Adam, G., Hosny, A., Khodakarami, F., Waldron, L., Wang, B., McIntosh, C., Goldenberg, A., Kundaje, A., Greene, C., Broderick, T., Hoffman, M., Leek, J., Korthauer, K., Huber, W., Brazma, A., Pineau, J., Tibshirani, R., Hastie, T., Ioannidis, J., Quackenbush, J., Aerts, H.: Transparency and reproducibility in artificial intelligence. Nature **586(7829)**, E14–E16 (2020). https://doi.org/10.1038/s41586-020-2766-y, `https://aclanthology.org/Q18-1041`
8. Holland, S., Hosny, A., Newman, S., Joseph, J., Chmielinski, K.: The dataset nutrition label: A framework to drive higher data quality standards. arXiv 1805.03677 (2018)
9. Koch, P., Golovidov, O., Gardner, S., Wujek, B., Griffin, J., Xu, Y.: Autotune. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (Jul 2018). https://doi.org/10.1145/3219819.3219837, `http://dx.doi.org/10.1145/3219819.3219837`
10. Lin, T.Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C.L., Dollár, P.: Microsoft COCO: Common objects in context. arXiv 1405.0312 (2014), `http://arxiv.org/abs/1405.0312`
11. Long, J., Shelhamer, E., Darrell, T.: Experiment manager. https://www.mathworks.com/help/deeplearning/ref/experimentmanager-app.html (2022)
12. Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I.D., Gebru, T.: Model cards for model reporting. Proceedings of the Conference on Fairness, Accountability, and Transparency (Jan 2019). https://doi.org/10.1145/3287560.3287596, `http://dx.doi.org/10.1145/3287560.3287596`
13. NIST: Data-driven simulations of measured neutron interferometric microscopy images (2022), `https://www.nist.gov/programs-projects/interferometry-infer-neutron-interferometric-microscopy-small-forces-and`

14. NIST: Fluorescent microscopy images of A-10 rat smooth muscle cells and NIH-3T3 mouse fibro-blasts. https://isg.nist.gov/deepzoomweb/data/dissemination (2022), https://isg.nist.gov/deepzoomweb/data/dissemination
15. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: PyTorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019), http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
16. Ranjit, M., Ganapathy, G., Sridhar, K., Arumugham, V.: Efficient deep learning hyperparameter tuning using cloud infrastructure: Intelligent distributed hyperparameter tuning with bayesian optimization in the cloud. In: 2019 IEEE 12th International Conference on Cloud Computing (CLOUD). pp. 520–522. IEEE Computer Society, Los Alamitos, CA, USA (jul 2019). https://doi.org/10.1109/CLOUD.2019.00097, https://doi.ieeecomputersociety.org/10.1109/CLOUD.2019.00097
17. Schaub, N., Hotaling, N., Manescu, P., Padi, S., Wan, Q., Sharma, R., George, A., Chalfoun, J., Simon, M., Ouladi, M., Simon, C.J., Bajcsy, P., K., B.: Deep learning predicts function of live retinal pigment epithelium from quantitative microscopy. J Clin Invest. **130(2)**, 1010–1023 (2020). https://doi.org/10.1172/JCI131187, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6994191/
18. Yang, K., Stoyanovich, J., Asudeh, A., Howe, B., Jagadish, H., Miklau, G.: A nutritional label for rankings. Proceedings of the 2018 International Conference on Management of Data (May 2018). https://doi.org/10.1145/3183713.3193568, http://dx.doi.org/10.1145/3183713.3193568