Universality Patterns in the Training of Neural Networks

Anonymous Authors[1]

Abstract

This paper proposes and demonstrates a surprising pattern in the training of neural networks: there is a one to one relation between the values of any pair of losses (such as cross entropy, mean squared error, $0/1$ error etc.) evaluated for a model arising at (any point of) a training run. This pattern is *universal* in the sense that this one to one relationship is identical across architectures (such as VGG, Resnet, Densenet etc.), algorithms (SGD and SGD with momentum) and training loss functions (cross entropy and mean squared error).

1. Introduction

Neural networks are state of the art models for various tasks in machine learning, especially those in computer vision and natural language processing. While there has been significant progress in designing and applying neural networks for various tasks, our understanding of most aspects of their behavior has not yet caught up with these advances. One significant challenge in furthering our understanding is the huge variation in deep learning models. In the context of image classification (which will be the context of the current paper), there are several well known models that have been developed by the machine learning community: VGG (Simonyan & Zisserman, 2014), Resnet (He et al., 2016), Densenet (Huang et al., 2017) to name a few; all of them having their own unique structure. Is it possible to understand the behavior of *all* of these models through a common lens?

The main contribution of the current paper is to propose and demonstrate that, despite the diversity in the structure of these different models, there are some striking resemblances in the behavior of all of these models. More concretely, the *training* curves across any two loss functions (such as cross entropy vs $0/1$ error or cross entropy vs mean squared error) essentially *overlap* across *all of the above* models. See Figure 1 for a pictorial description and Section 3 for a rigorous description.

This observation suggests that training of most (if not all) deep neural networks follows the same pattern. The existence of such universal patterns is quite exciting as it points to the possibility of understanding the behavior (in this instance training behavior) of different neural networks through a single approach. We also note that, while this similarity in behavior extends, to some extent, also to the *test* data, there are limitations (see Section 4.2).

Paper organization: In Section 2, we will present the setup and required definitions. Section 3 formally describes the phenomenon identified in this paper. Section 4 presents the main experimental results. We conclude in Section 5. More experimental results are presented in the appendix.

Notation: Vectors are written in bold small case letters (such as $\mathbf{p}$ and $\mathbf{x}$). $\|\mathbf{x}\|_p := \left(\sum_{i=1}^d x_i^p\right)^{1/p}$ denotes the ℓ_p-norm for vectors $\mathbf{x} \in \mathbb{R}^d$. The canonical basis for $\mathbb{R}^K$ is represented as $\{\mathbf{e}_i, \ldots, \mathbf{e}_K\}$ where $e_{ij} = 1$ if $i = j$ else $e_{ij} = 0$.

2. Preliminaries

For the task of learning a classifier, the goal is to learn a mapping from the input space $\mathcal{X} \subseteq \mathbb{R}^d$ to the set of class labels $[K] := \{1, \ldots, K\}$, where K is the total number of classes. Let $\Delta_K := \left\{\mathbf{p} \in \mathbb{R}_+^K \mid \sum_{k=1}^K p_k = 1\right\}$ denote the probability simplex on $\mathbb{R}^K$, then for a neural network $F_{\boldsymbol{\theta}} : \mathcal{X} \to \Delta_K$ parameterized by $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^D$, the classifier C can be obtained by choosing the class label for which the prediction is the maximum. The classifier $C_{\boldsymbol{\theta}}(\mathbf{x}) := \arg\max_{k \in [K]} [F_{\boldsymbol{\theta}}(\mathbf{x})]_k$ thus maps the input $\mathbf{x} \in \mathcal{X}$ to a class label contained in $[K]$. The training of the network F on observed data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ is a minimization problem over its parameters $\boldsymbol{\theta}$. One minimizes the empirical loss $\mathcal{L}(\boldsymbol{\theta}; F, l) := \frac{1}{n}\sum_{i=1}^n l(F_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i)$ where $l : \Delta_K \times [K] \to \mathbb{R}_+$ is a loss function. In practice, l is usually chosen to satisfy $l(\mathbf{e}_y, y) = 0 \; \forall \; y \in [K]$. In this paper, we consider the following commonly used training loss functions:

1. *Cross Entropy:* $l_{\text{CE}}(\mathbf{p}, y) := -\log p_y$, and
2. *Mean Squared Error:* $l_{\text{MSE}}(\mathbf{p}, y) := \|\mathbf{p} - \mathbf{e}_y\|_2^2$.

With the training performed on a fixed loss function, one can also look at other surrogate losses to evaluate the performance of the neural network. We consider the below few more loss functions

1. $0/1$ *Error:* $l_{0/1}(\mathbf{p}, y) := \mathbb{1}\left[\arg\max_{k\in[K]} p_k \neq y\right]$,
2. ℓ_p *Error:* $l_{\ell_p}(\mathbf{p}, y) := \|\mathbf{p} - \mathbf{e}_y\|_p$ for $p > 0$.

It is to note that

1. $l_{0/1}$ is not continuous in its first variable, therefore one cannot use gradient based optimization algorithms to directly minimize it, and
2. l_{MSE} & l_{CE} are fundamentally different loss functions. l_{CE} depends only on p_y, i.e., prediction on true class, whereas l_{MSE} depends on the full prediction vector $\mathbf{p}$.

In the paper, we will denote l as the *loss function* and the corresponding $\mathcal{L}(\boldsymbol{\theta}; F, l)$ as the *loss* of the network $F_{\boldsymbol{\theta}}$ when trained using the loss function l. We will also denote this loss by $\mathcal{L}(l)$ when we are arguing about multiple networks.

3. Description of the Phenomenon

In this section, we will rigorously describe the phenomenon that we demonstrate in this paper. Fix any two loss functions l_1 and l_2 (e.g., each of these could be l_{CE} or l_{MSE} or $l_{0/1}$ etc.). As a training run progresses, the value of these losses on the training data $\mathcal{L}(\boldsymbol{\theta}_t; F, l_1)$ and $\mathcal{L}(\boldsymbol{\theta}_t; F, l_2)$, in general, decrease. We posit that

- **Uniqueness**: For any two loss functions l_1 and l_2, there exists a unique function $f(\cdot)$ such that for any (intermediate) model $F_{\boldsymbol{\theta}_t}$ in the training run, its loss $\mathcal{L}(\boldsymbol{\theta}_t; F, l_2)$ on the training data is given by $f(\mathcal{L}(\boldsymbol{\theta}_t; F, l_1))$, where $\mathcal{L}(\boldsymbol{\theta}_t; F, l_i)$ is the loss evaluated on the training data using loss function l_i. In practice however, there are several sources of randomness (stochastic gradient descent, random initialization etc.) which mean that we expect that $\mathcal{L}(\boldsymbol{\theta}_t; F, l_2) \sim f(\mathcal{L}(\boldsymbol{\theta}_t; F, l_1))$ rather than exact equality.
- **Universality**: The function $f(\cdot)$ depends only on l_1 and l_2 and is identical for all network architectures (such as Resnet, VGG, Densenet etc.), training loss functions (such as l_{CE}, l_{MSE} etc.) and training algorithms (such as SGD and SGDm etc.).

  While the universality of $f(\cdot)$ is clearly surprising, we note that the existence of a unique function $f(\cdot)$, as well as our empirical observation that it is monotonic, from a theory point of view, are quite surprising.

4. Experimental Results

Before describing the observations, we first describe all the possible combinations of datasets, architectures, training losses and training algorithms we have used.

- **Architectures**: We consider the below architectures
  1. Multi Layer Perceptrons (MLP) - 1 and 2 hidden layered with 1024 neurons in each (`mlp1024x1` and `mlp1024x2` resp.) with batch normalization and ReLU activation,
  2. VGG (Simonyan & Zisserman, 2014)- VGG-11 and VGG-16 with and without batch normalization layers (`vgg##` and `vgg##wobn` resp.),
  3. Resnet (He et al., 2016) - Resnet-18 and Resnet-50 (`resnet18` and `resnet50` resp.),
  4. Densenet (Huang et al., 2017) - Densenet-121 (`densenet`) and,
  5. Fully Connected (`FC1`) - No hidden layers. The output is computed by softmax.
- **Datasets**: We consider 4 datasets tabulated in Table 1.

Table 1. Datasets used.

DATA SET	TRAIN IMAGES	CLASSES	DIMENSION
CIFAR-10 (KRIZHEVSKY & HINTON, 2009)	50,000	10	32×32
CIFAR-10 RANDOM (RANDOM LABELS)	50,000	10	32×32
RANDOM (RANDOM PIXELS)	50,000	10	32×32
IMAGENET-20 (DENG ET AL., 2009) (SUBSET OF 20 CLASSES)	26,000	20	224×224

- **Loss functions**: We consider 2 training loss functions
  1. l_{MSE} and, 2. l_{CE}
- **Algorithms**: We consider 2 training algorithms:
  1. Stochastic Gradient Descent (SGD)
  2. SGD with 0.9 momentum (SGDm)

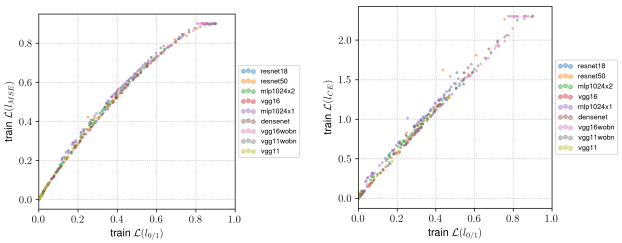
  Both the algorithms use a batch-size of 512 and a constant learning rate from $2^{\{-5,\ldots,-10\}}$ such that there is no divergence in training. The models are trained for $150 - 250$ epochs.

4.1. Universality in training surrogate loss and classification error on train data

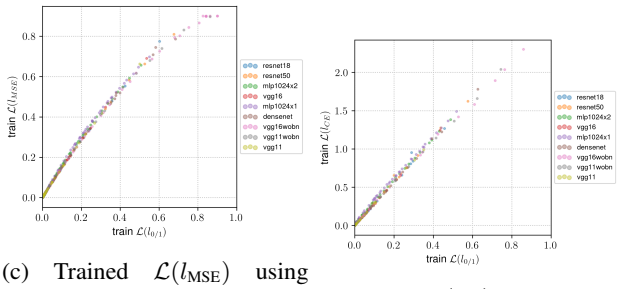
Fixing a dataset, one can learn the classifier by training the network on a suitable training surrogate loss function (like l_{MSE} and l_{CE}) using a variety of algorithms (like SGD and SGDm) and hope to minimize the empirical classification error $\mathcal{L}(l_{0/1})$. Across the training profile, we can as well see how their corresponding training surrogate losses behave with respect to each other.

From Figure 1 we observe that the trends of training loss ($\mathcal{L}(l_{\text{MSE}})$ or $\mathcal{L}(l_{\text{MSE}})$) and training $0/1$ error $\mathcal{L}(l_{0/1})$ overlap across all architectures for both the training algorithms SGD and SGDm for CIFAR-10. The corresponding plot for IMAGENET-20 (Figure 6) is presented in Appendix A.1.

For 1-hidden layered MLPs we show that this behavior also holds if we the number of neurons in the hidden layer is large. See Figure 7 in Appendix A.1 for the plots.

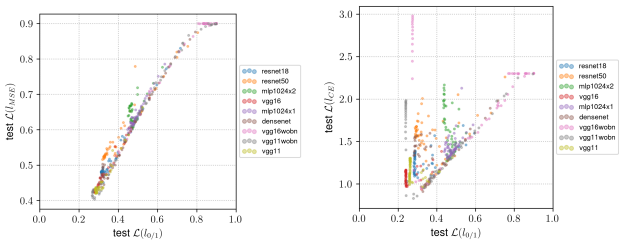


(a) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGD (b) Trained $\mathcal{L}(l_{\text{CE}})$ using SGD

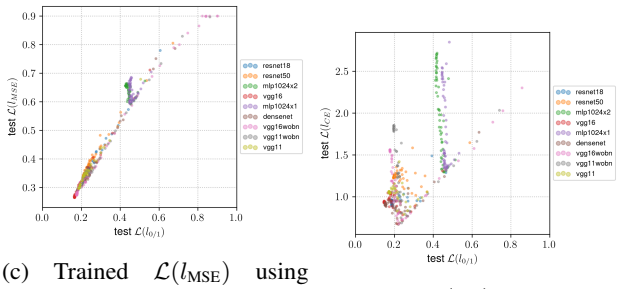


(c) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGDm (d) Trained $\mathcal{L}(l_{\text{CE}})$ using SGDm

Figure 1. Training CIFAR-10 on train loss functions l_{MSE} and l_{CE} using SGD and SGDm for all architectures.



(a) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGD (b) Trained $\mathcal{L}(l_{\text{CE}})$ using SGD



(c) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGDm (d) Trained $\mathcal{L}(l_{\text{CE}})$ using SGDm

Figure 2. Test losses for training CIFAR-10 on loss functions l_{MSE} and l_{CE} using SGD and SGDm for all architectures.

4.2. Observations on surrogate loss and classification error on test data

Interestingly, the overlapping behavior observed between training surrogate loss and training classification error does not quite extend to the test data for CIFAR-10. We show this in Figure 2 where we do not observe a significant overlap across all architectures for both the training algorithms SGD and SGDm on CIFAR-10. On the other hand, similar figure (Figure 8) for IMAGENET-20 exhibits a universal behavior. The plot for IMAGENET-20 is presented in Appendix A.2. These results motivate further exploration of possible universality patterns in the test data.

4.3. Universality over other surrogate losses

While a network $F_{\boldsymbol{\theta}}$ may be trained using a particular training loss function ($\mathcal{L}\left(l_{\text{MSE}}\right)$ or $\mathcal{L}\left(l_{\text{CE}}\right)$), one may still evaluate other surrogate losses that have not been used in training. For example, derived metrics like accuracy $(1-\mathcal{L}\left(l_{0/1}\right))$ are used to evaluate a model as a classifier. Other loss functions apart from the training loss are not involved in the training phase of the network, so there is no reason for picking only the $0/1$ loss function $l_{0/1}$. We also evaluate the performance of the model on other surrogate losses $\left\{l_{\text{MSE}}, l_{\text{CE}}, l_{0/1}, l_{\ell_p}\right\} \setminus \left\{l_{\text{training}}\right\}$ where $l_{\text{training}} \in \left\{l_{\text{MSE}}, l_{\text{CE}}\right\}$. Figures for several pairs of loss functions are presented in the Appendix A.3. We observe the universality phenomenon for all pairs of surrogate losses that we have considered.

4.4. Robustness to initializations

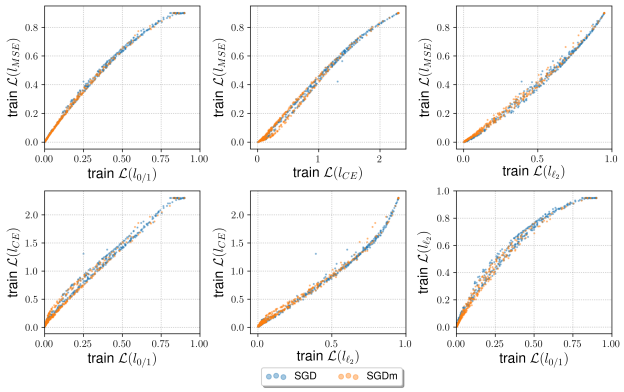
We also verify if the observed trends hold across different random initialization of the parameters of the networks. Figure 11 and Figure 12 in Appendix A.4 show that indeed the observations are robust to the randomness in initialization.

4.5. Universality across algorithms and training surrogate loss functions

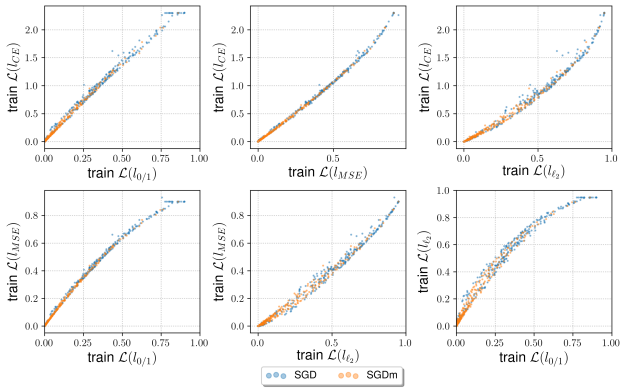
Similar trends are observed in Figure 3 for CIFAR-10 where we keep the initialization and the training loss function fixed, but look at different training algorithms SGD and SGDm for all architectures jointly. A similar figure (Figure 13) for IMAGENET-20 is presented in Appendix A.5. Coming to universality across loss functions, we again observe similar overlapping trends in Figure 4 for CIFAR-10 where we keep the initialization and the training algorithm fixed, and vary the training loss on the two loss functions (l_{MSE} and l_{MSE}) for all architectures jointly. The corresponding figure (Figure 14) for IMAGENET-20 is presented in Appendix A.5.

4.6. Observations across datasets

As long as the datasets have same number of classes, we also check if this phenomenon is also observed across different datasets. We compare CIFAR-10 with CIFAR-10 Random and Random datasets mentioned in Table 1. We do not observe similar significant overlapping behavior when we switch datasets as we present our observations in Figure 15 and Figure 16 in Appendix A.6.

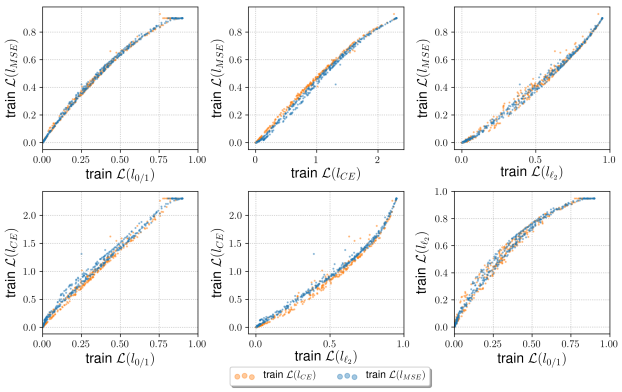


(a) Trained $\mathcal{L}(l_{\text{MSE}})$

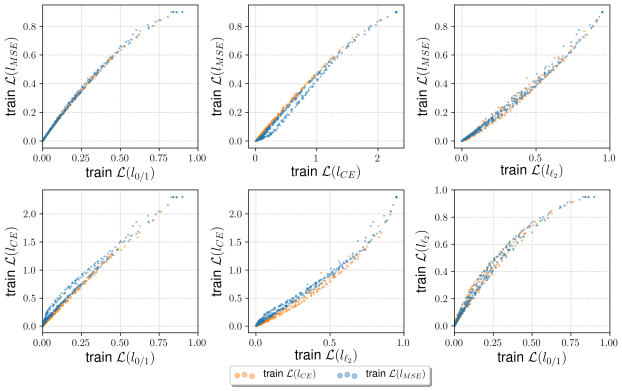


(b) Trained $\mathcal{L}(l_{\text{CE}})$

Figure 3. Pairs of surrogate losses for training CIFAR-10 on loss functions l_{MSE} and l_{CE} across SGD and SGDm for all architectures.



(a) Trained using SGD



(b) Trained using SGDm

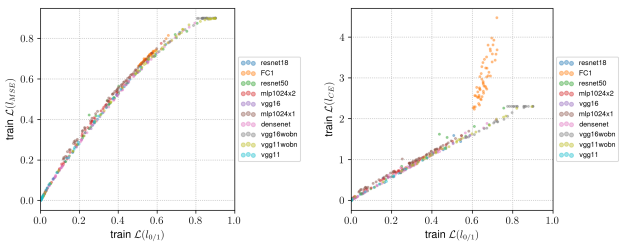
Figure 4. Pairs of surrogate losses for training CIFAR-10 using SGD and SGDm across loss functions l_{MSE} and l_{CE} for all architectures.

4.7. Comparison with shallow fully connected networks

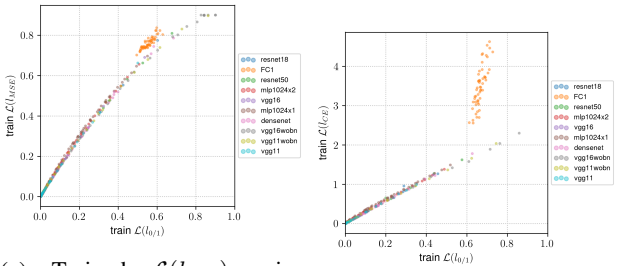
The universality phenomenon seems to arise only for large models but not for small models. For a shallow fully connected network with no hidden layers, the curves do not overlap in most of the cases as presented in Figure 5.

5. Discussion and future directions

We propose and demonstrate that there is a training dynamics which is universal for various neural networks. While similar behavior does not seem to hold for test data, preliminary results call for further investigation. One possible explanation is that distribution of predictions through the training process follows a universal law. However, measuring standard distances (e.g., Wasserstein) requires a large number of samples. One potential solution is to measure weaker distances such as neural net distances (Arora et al., 2017). We believe that our observations could lead to a new way of understanding neural networks in a unified manner.



(a) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGD (b) Trained $\mathcal{L}(l_{\text{CE}})$ using SGD



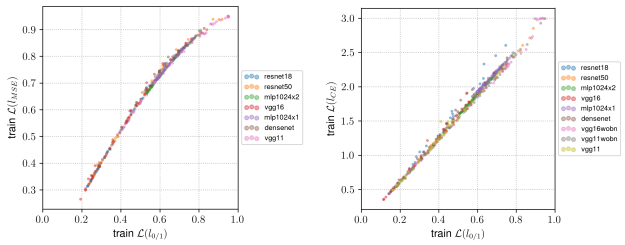
(c) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGDm (d) Trained $\mathcal{L}(l_{\text{CE}})$ using SGDm

Figure 5. Training CIFAR-10 on train loss functions l_{MSE} and l_{CE} using SGD and SGDm across deep and shallow architectures.

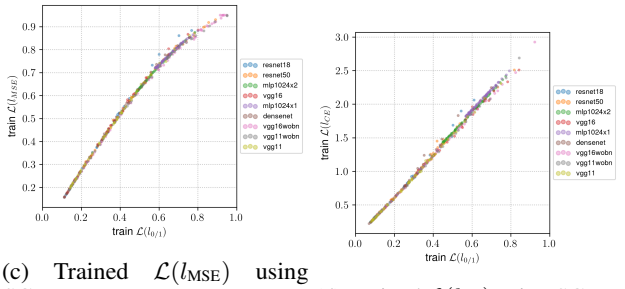
A. Appendix

A.1. Universality in training surrogate loss and classification error for train data

This section is an extension to Section 4.1 where we describe our observations on CIFAR-10 dataset. Here in Figure 6 we show similar results on IMAGENET-20 where we see that the overlapping trend of training loss and training $0/1$ error also holds on Imagenet-20.



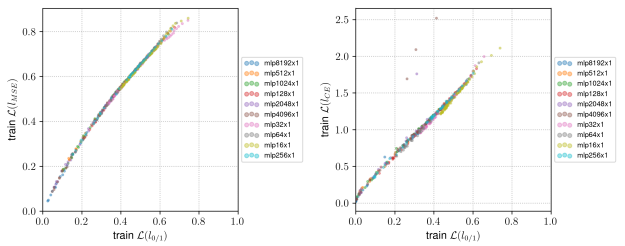
(a) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGD (b) Trained $\mathcal{L}(l_{\text{CE}})$ using SGD



(c) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGDm (d) Trained $\mathcal{L}(l_{\text{CE}})$ using SGDm

Figure 6. Training Imagenet-20 on train loss functions l_{MSE} and l_{CE} using SGD and SGDm for all architectures.

We observe similar behavior when we vary the number of hidden neurons in a 1-hidden layer MLP. Figure 7 shows that there is a significant overlap in the curves of training loss and classification error for MLPs with varying number of neurons in $2^{\{4,\ldots,13\}}$.

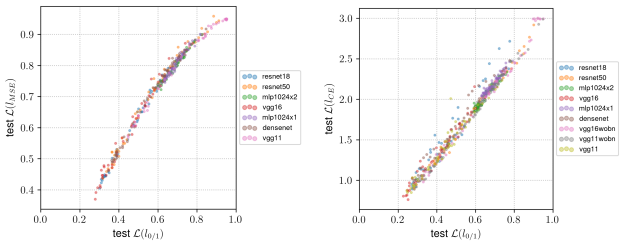


(a) Trained $\mathcal{L}(l_{\text{MSE}})$ (b) Trained $\mathcal{L}(l_{\text{CE}})$

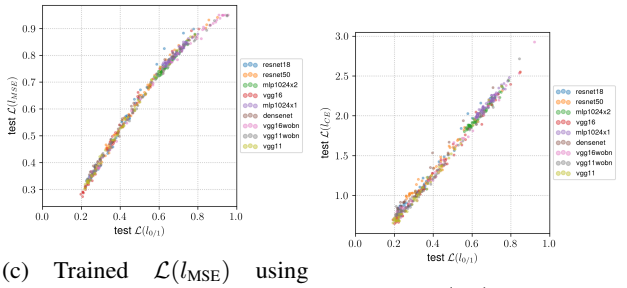
Figure 7. Training CIFAR-10 on train loss functions l_{MSE} and l_{CE} using SGD for MLPs with different number of neurons.

A.2. Observations on training surrogate loss and classification error for test data

This section is an extension to Section 4.2 where we describe our observations on CIFAR-10 dataset. Here in figure 8 we show the results on IMAGENET-20 where we observe a mild overlapping trend of training loss and $0/1$ error on the test data. This overlap observed is similar to that of CIFAR-10 if the over-fitting stage (non-monotonic part of the curves) is ignored.



(a) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGD (b) Trained $\mathcal{L}(l_{\text{CE}})$ using SGD



(c) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGDm (d) Trained $\mathcal{L}(l_{\text{CE}})$ using SGDm

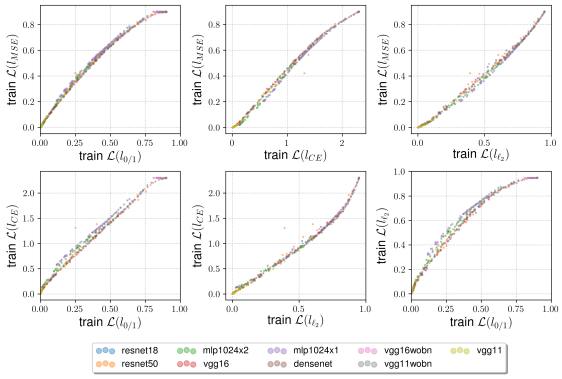
Figure 8. Test losses for training IMAGENET-20 on loss functions l_{MSE} and l_{CE} using SGD and SGDm for all architectures.

A.3. Universality across all pairs of surrogate losses

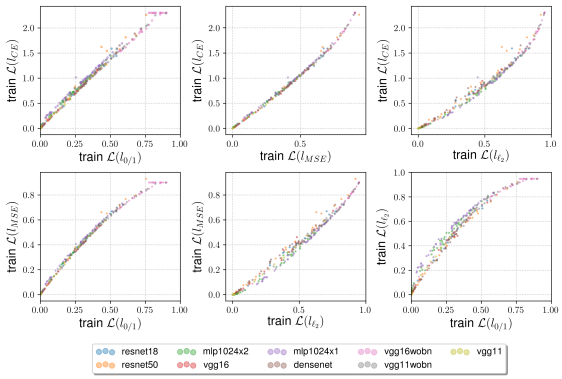
This section is an extension of Section 4.3 where we describe the universality phenomenon holding across all the architectures and all the pairs of surrogate losses. We show these observations in Figure 9 and Figure 10 for CIFAR-10 and IMAGENET-20 respectively.

A.4. Universality across multiple initializations

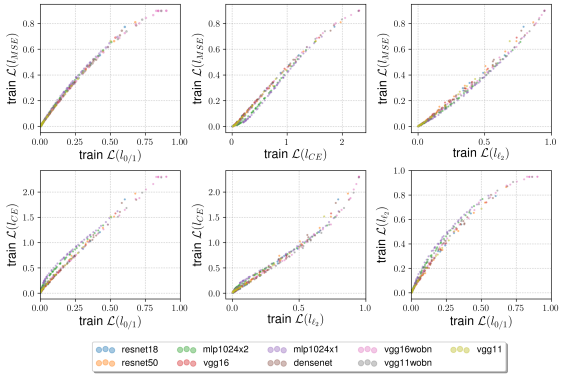
This section is an extension to Section 4.4 where we claim that our observations are robust to initializations from Figure 11 and Figure 12.



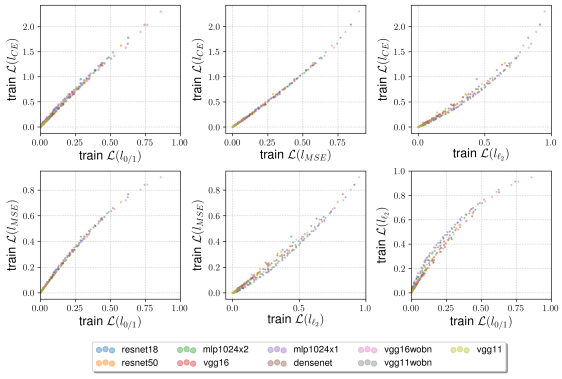
(a) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGD



(b) Trained $\mathcal{L}(l_{\text{CE}})$ using SGD

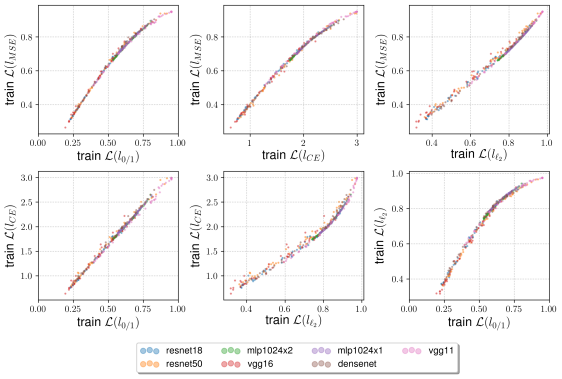


(c) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGDm

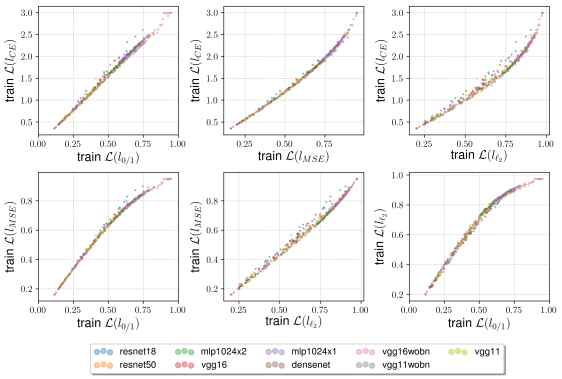


(d) Trained $\mathcal{L}(l_{\text{CE}})$ using SGDm

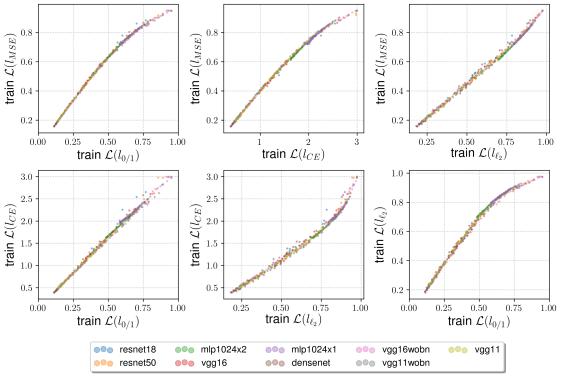
Figure 9. All pairs of surrogate losses for training CIFAR-10 on loss functions l_{MSE} and l_{CE} using SGD and SGDm across different architectures.



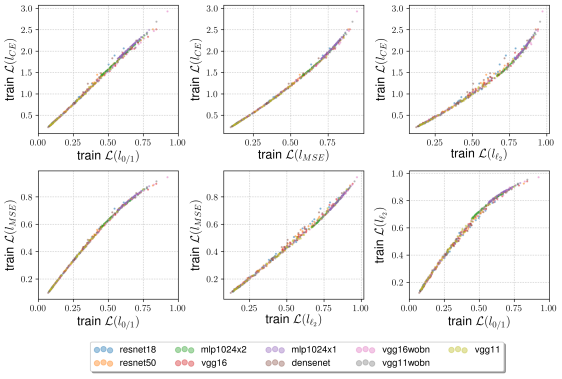
(a) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGD



(b) Trained $\mathcal{L}(l_{\text{CE}})$ using SGD

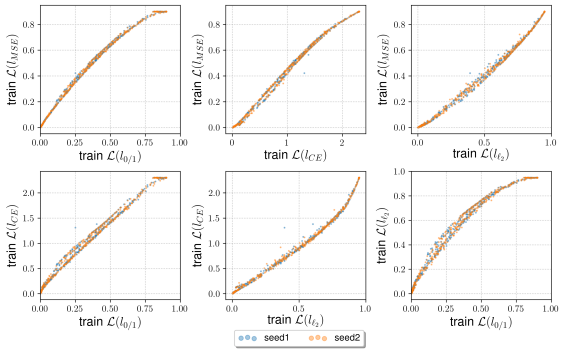


(c) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGDm

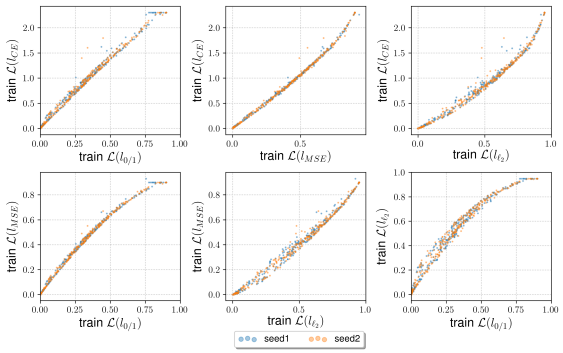


(d) Trained $\mathcal{L}(l_{\text{CE}})$ using SGDm

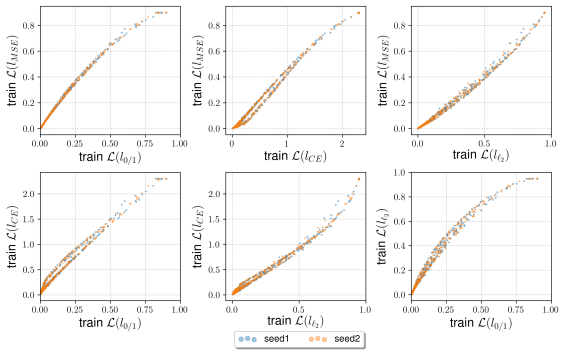
Figure 10. All pairs of surrogate losses across training IMAGENET-20 on loss functions l_{MSE} and l_{CE} using SGD and SGDm for different architectures.



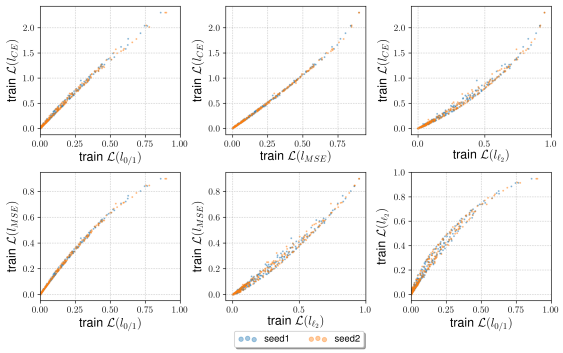
(a) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGD



(b) Trained $\mathcal{L}(l_{\text{CE}})$ using SGD

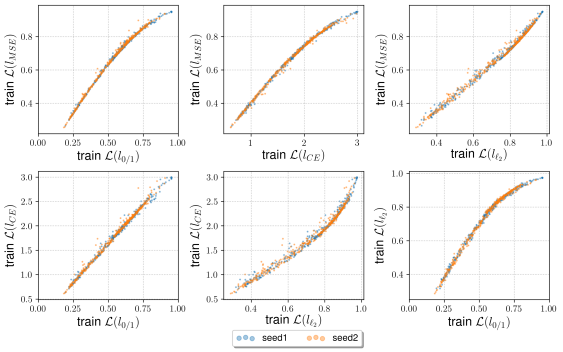


(c) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGDm

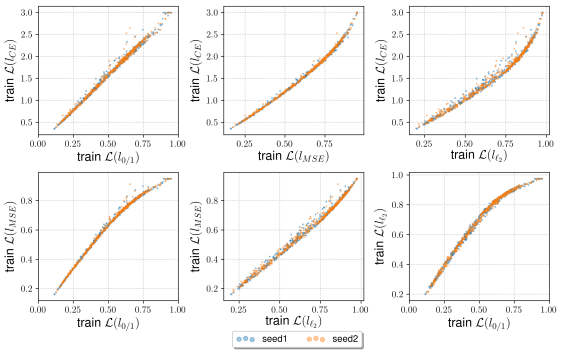


(d) Trained $\mathcal{L}(l_{\text{CE}})$ using SGDm

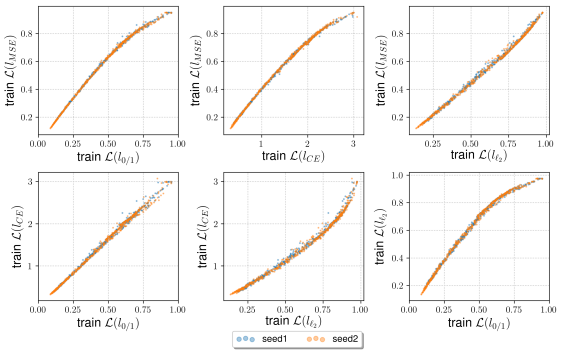
Figure 11. All pairs of surrogate losses for training CIFAR-10 on loss functions l_{MSE} and l_{CE} using SGD and SGDm across 2 random initializations for all architectures.



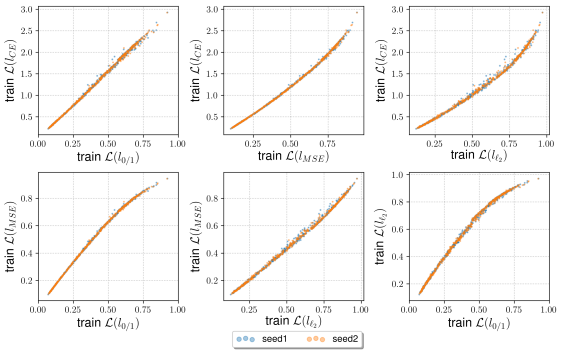
(a) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGD



(b) Trained $\mathcal{L}(l_{\text{CE}})$ using SGD

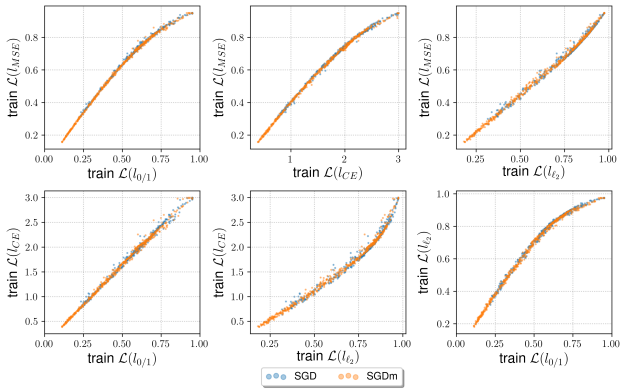


(c) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGDm

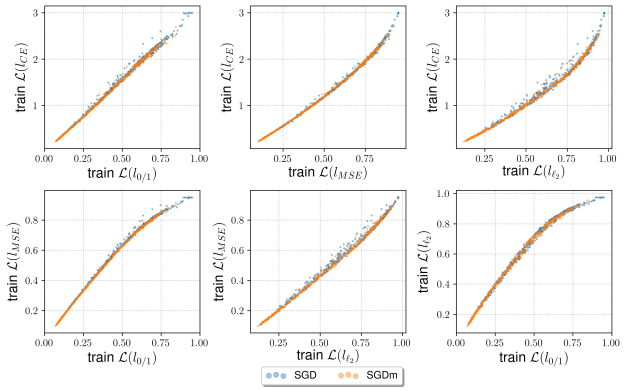


(d) Trained $\mathcal{L}(l_{\text{CE}})$ using SGDm

Figure 12. All pairs of surrogate losses for training IMAGENET-20 on loss functions l_{MSE} and l_{CE} using SGD and SGDm across 2 random initializations for all architectures.



(a) Trained $\mathcal{L}(l_{\text{MSE}})$



(b) Trained $\mathcal{L}(l_{\text{CE}})$

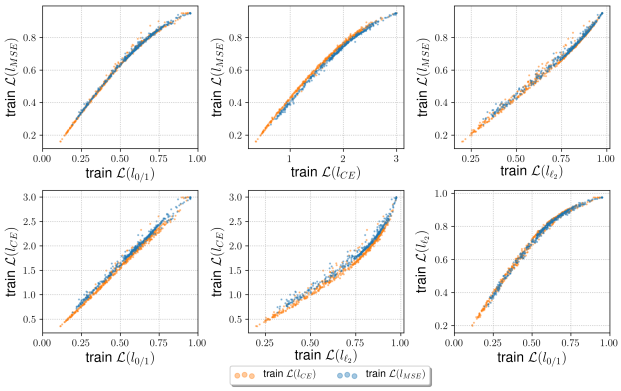
Figure 13. All pairs of surrogate losses for IMAGENET-20 on loss functions l_{MSE} and l_{CE} across SGD and SGDm for all architectures.

A.5. Universality across algorithms and training surrogate loss functions

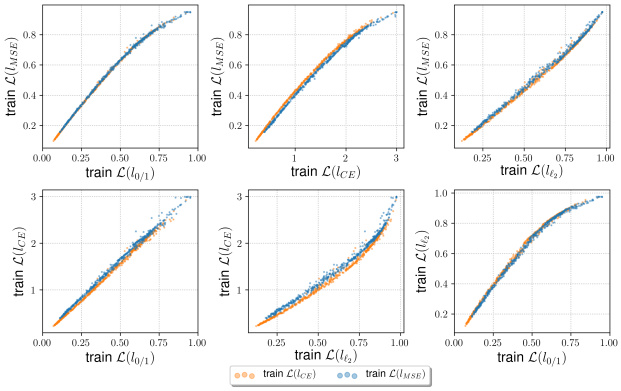
This section is in continuation to Section 4.5 where we describe our observations on CIFAR-10 dataset. Here in Figure 13 and Figure 14 we show similar results on Imagenet-20 where we see that the overlapping trend across different training algorithms and different surrogate loss functions respectively also holds on IMAGENET-20.

A.6. Observations across datasets

This section is in continuation to Section 4.6 where we describe our observations on the phenomenon across different datasets with same number of classes. As we see in Figure 15 and Figure 16, we do not observe the overlapping phenomenon when we switch datasets.

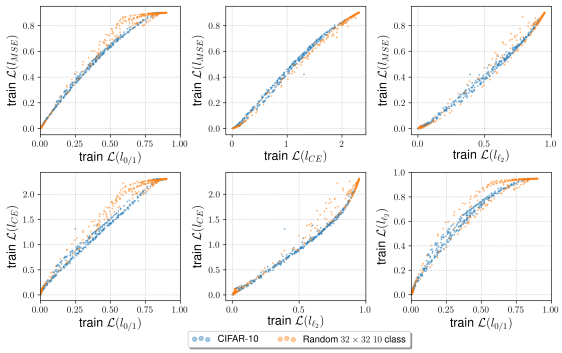


(a) Trained using SGD

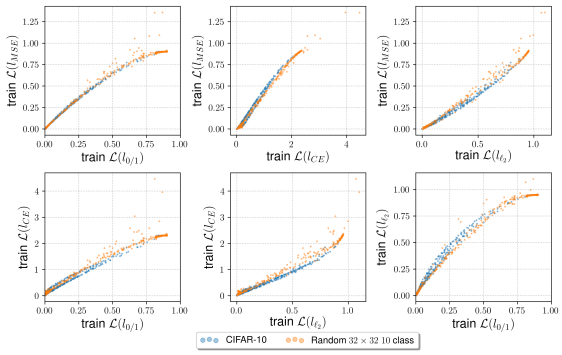


(b) Trained using SGDm

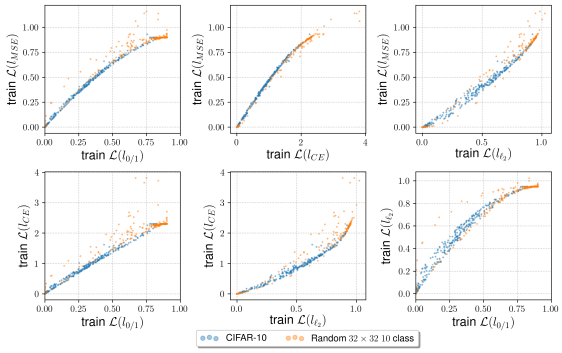
Figure 14. All pairs of loss functions for IMAGENET-20 using SGD and SGDm across loss functions l_{MSE} and l_{CE} for all architectures.



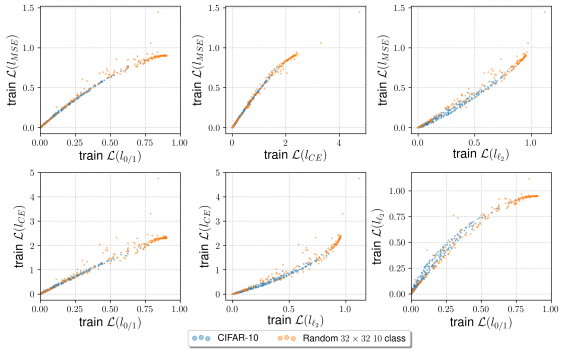
(a) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGD



(b) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGDm

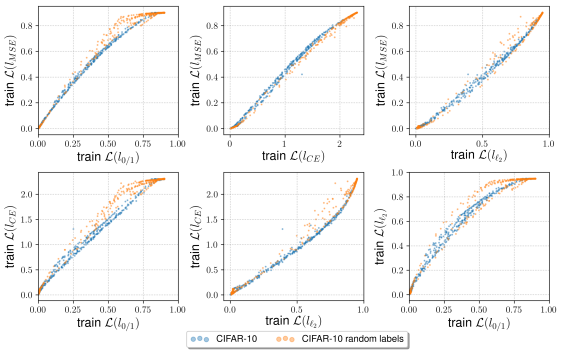


(c) Trained $\mathcal{L}(l_{\text{CE}})$ using SGD

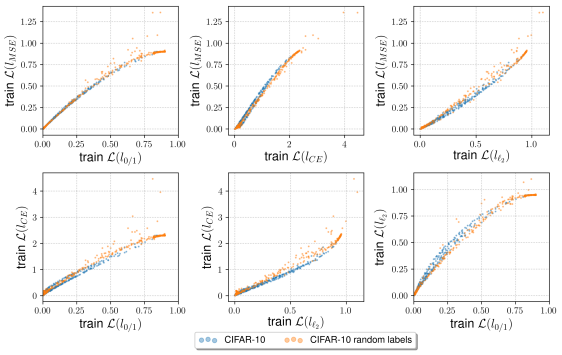


(d) Trained $\mathcal{L}(l_{\text{CE}})$ using SGDm

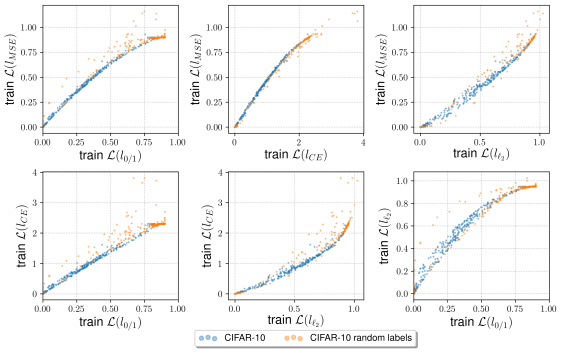
Figure 15. All pairs of surrogate losses for training using SGD and SGDm, on loss functions l_{MSE} and l_{CE} across CIFAR-10 and Random dataset, for all architectures.



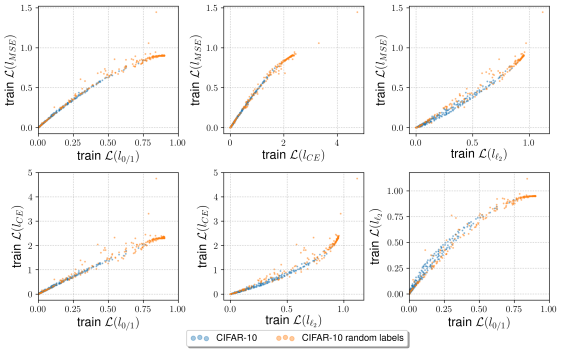
(a) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGD



(b) Trained $\mathcal{L}(l_{\text{MSE}})$ using SGDm



(c) Trained $\mathcal{L}(l_{\text{CE}})$ using SGD



(d) Trained $\mathcal{L}(l_{\text{CE}})$ using SGDm

Figure 16. All pairs of surrogate losses for training using SGD and SGDm, on loss functions l_{MSE} and l_{CE} across CIFAR-10 and CIFAR-10 Random, for all architectures.

References

Arora, S., Ge, R., Liang, Y., Ma, T., and Zhang, Y. Generalization and equilibrium in generative adversarial nets (gans). In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 224–232. JMLR. org, 2017.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.