A step towards understanding why classification helps regression

Supplementary material

A. 1D experimental details

For the $1D$ synthetic experiments we use the function $f(\cdot)$ defined by a sum of two sines with different frequencies and amplitudes:

$$f(x) = a\sin(cx) + b\sin(dx), \tag{14}$$

where $f(x) \in [-1.5, 1.5]$ and $x \in [-1, 1]$. We show here in Fig. 6 the 10 functions we have sampled from this function for the $1D$ experiments presented in the paper.

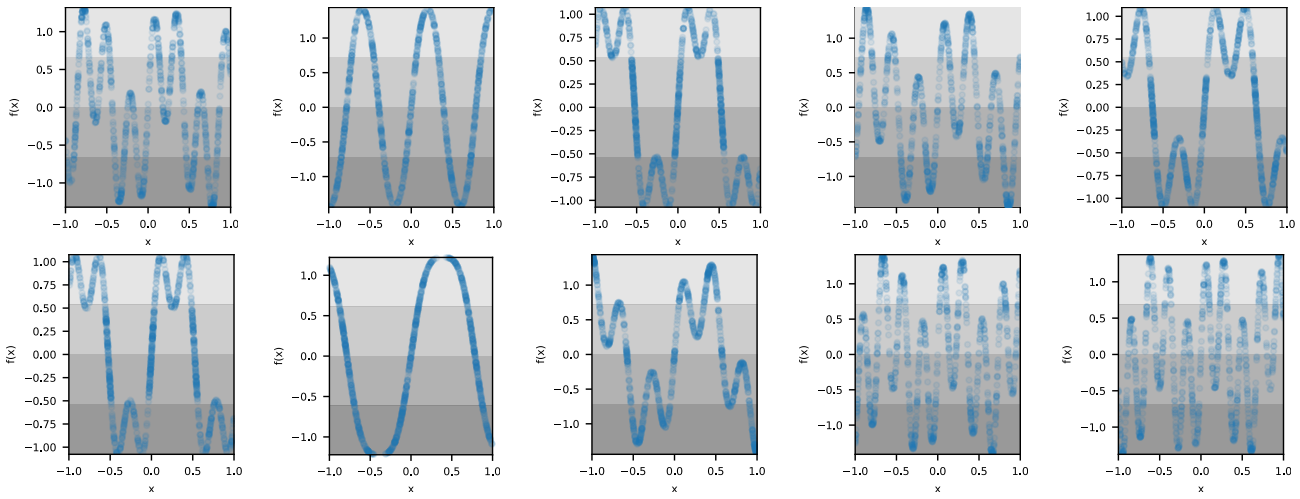


Figure 6. The 10 sampled $1D$ functions used in the experiments in the main paper. Over these 10 functions we vary the sampling into: 4 sampling scenarios (*uniform*, *mild*, *moderate* and *severe*), 3 data scenarios (*clean*, *noisy*, and *ood*), with 5 different seeds and 5 classes $(2^2, 2^4, 2^6, 2^8, 2^{10})$ resulting in 3000 experiments on the $1D$ data.

For the validation set we run the experiments over 3 repetitions with seeds: $\{0, 421, 8125\}$, while for testing we run the experiments by setting the random seed everywhere in the code to $\{0, 421, 8125, 2481, 849\}$. For each dataset scenario (*clean*, *noisy data*, *ood*) and sampling (*uniform*, *moderate*, *mild*, *severe*) we create 5 dataset versions. During training for every random seed we pick a dataset version, giving rise to 5 repetitions per function, over which we report results in the Fig. 3 in the paper. We print here in Tab. 6 the model architecture for the regression-only and regression and classification tasks.

Layer	Output Shape	Param
Linear1	[-1, 6]	12
ReLU1	[-1, 6]	—
Linear2	[-1, 16]	112
ReLU2	[-1, 16]	–
Linear3	[-1, 1]	17

(a) Regression only

Layer	Output Shape	Param
Linear1	[-1, 6]	12
ReLU1	[-1, 6]	—
Linear2	[-1, 16]	112
ReLU2	[-1, 16]	–
Linear3a	[-1, 1]	17
Linear3b	*[-1, C]*	*16C*

(b) Regression and classification

Table 6. MLP network architectures for the 1D experiments. When adding the classification loss during training, we use an additional linear layer (emphasized here) which maps the features of the one-to-last layer to the desired number of classes, C. During testing this layer is discarded.

Fig. 7 shows examples of predictions when using balanced and imbalanced classes on 3 severe sampling scenarios. The classification methods can make different mistakes, but overall the predictions are more accurate than for the regression alone.

B. Re-balancing the 2D imbalanced training data

For re-balancing the classes, we first redefine the class ranges to have equalized class counts using Eq. (12). For the regression loss we use all samples in each batch during training. For the classification loss we subsample the samples in the batch by considering the current

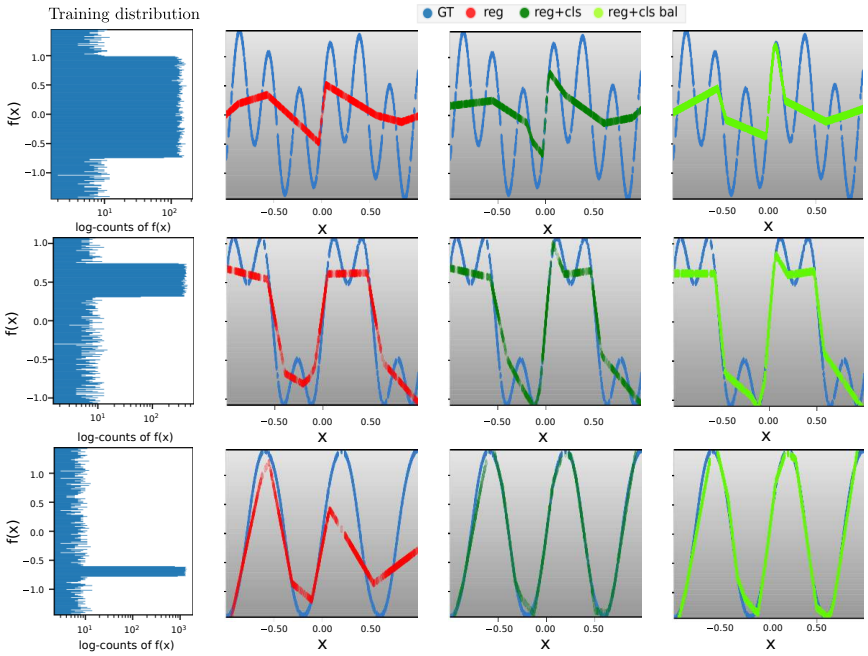
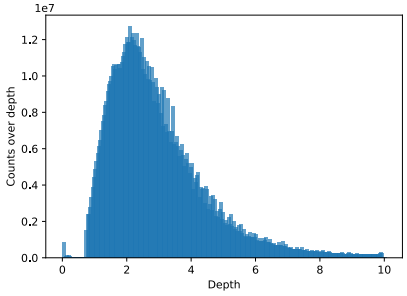
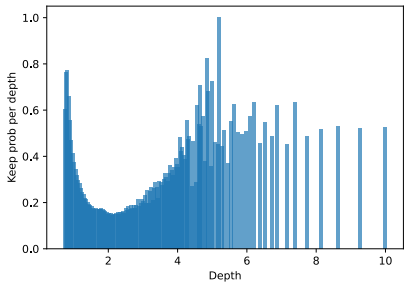


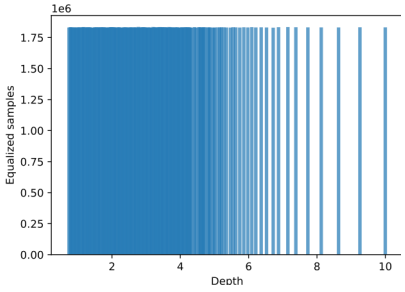
Figure 7. **Examples of 1D predictions:** For moderate/mild/severely imbalanced data sampling, on clean data scenario, when using 64 classes. The classification variants: with balanced classes (lime) and with imbalanced classes (green) make more accurate predictions than the regression baseline (red).



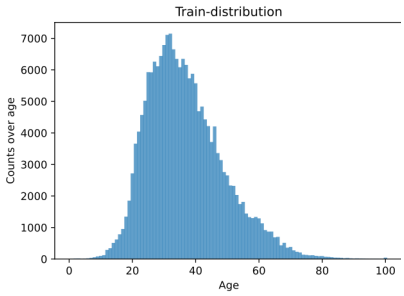
(a) Original NYUD2-DIR targets distribution.



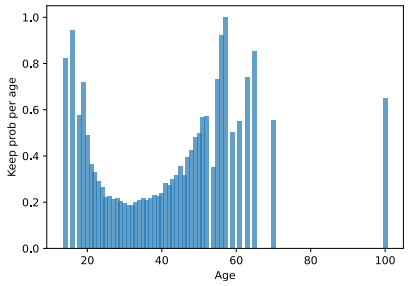
(b) NYUD2-DIR probabilities of keeping samples per class.



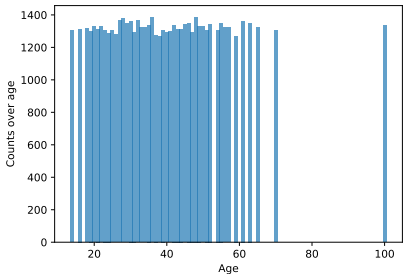
(c) NYUD2-DIR re-balanced samples.



(c) Original IMDB-WIKI-DIR targets distribution.



(d) IMDB-WIKI-DIR probabilities of keeping samples per class.



(e) IMDB-WIKI-DIR re-balanced samples.

Figure 8. **Re-balancing the training distribution in the classification loss.** (a) The initial distribution per class for the NYUD2-DIR dataset. (b) The probabilities of samples being retained per class, as described in Eq. (13). (c) The NYUD2-DIR samples re-balanced per class as described in Section 2.3. (d) The initial distribution per class for the IMDB-WIKI-DIR dataset. (e) The probabilities of samples being retained per class, as described in Eq. (13). (f) The IMDB-WIKI-DIR samples re-balanced per class as described in Section 2.3. Here we consider 100 classes for binning the dataset targets. Equalizing the class ranges using Eq. (12) and sampling samples per class over the training epochs using Eq. (13) is effective at balancing the classes.

training sample $(\mathbf{x}, y_k)$ in the classification loss with probability $\rho(k)$, where $\rho(k)$ is defined in Eq. (13). Specifically, for every training sample $(\mathbf{x}, y_k)$ we sample a random variable u from the uniform distribution and use this sample in the classification loss if the random value is smaller than $\rho(k)$:

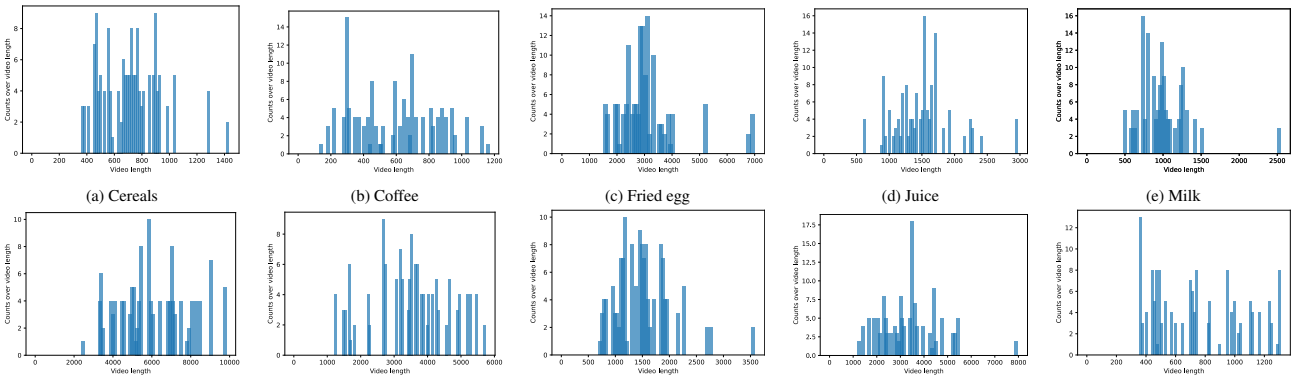
$$u \sim \mathcal{U}(0, 1), \tag{15}$$

$$\text{we classify } (\mathbf{x}, y_k) \text{ if } u \leq \rho(k). \tag{16}$$

This procedure is useful, because the complete dataset will be visited during training for classification (provided sufficient epochs), yet the samples seen per class will be equal. To illustrate this, we show in Fig. 8(a) the original data distribution on NYUD2-DIR, and IMDB-WIKI-DIR, the probabilities per class of keeping a sample for 100 equalized classes in Fig. 8(b), and the re-balanced training set distribution in Fig. 8(c). Here we consider 100 classes to bin the training targets. We see that after class equalization and re-balancing the class distribution is uniform. We could have only used the second option to balance the classes by sampling uniform samples per class using Eq. (13) without first equalizing the classes with Eq. (12). However, if the classes are severely imbalanced (as in the case of our data), some class probabilities are extremely low which leads to never selecting samples from those classes, and thus having the classification fail to converge.

C. Breakfast dataset task variations

Fig. 9 shows the large variation in video lengths across the 10 tasks on the Breakfast dataset on the training data. This large variability makes predicting video progression extra challenging, even when predicting video progression in percentages. This is due to having a most-frequent task length, and the model learns to predict better for the videos belonging to the most-frequent task length, and it makes mistakes when predicting progression on the videos that are a lot shorter or a lot longer than the average.



(a) Cereals (b) Coffee (c) Fried egg (d) Juice (e) Milk

(a) Pancake (b) Salad (b) Sandwich (c) Scrambled egg (d) Tea

Figure 9. **Breakfast task length variations.** Here we consider 100 classes for binning the video lengths. There is a considerable variation in the video lengths for the same cooking task. Because of this imbalance in the training data, the model makes more errors when predicting video progression on videos that are either considerably longer or shorter than the average. Next to this, the large variety in appearance of the videos makes the video progress prediction challenging.