A step towards understanding why classification helps regression

Silvia L. Pintea[1,2]   Yancong Lin[3]   Jouke Dijkstra[2]   Jan C. van Gemert[1]

[1] Computer Vision Lab, Delft University of Technology

[2] Division of Image Processing (LKEB), Leiden University Medical Center

[3] Intelligent Vehicles Group, Delft University of Technology

Abstract

A number of computer vision deep regression approaches report improved results when adding a classification loss to the regression loss. Here, we explore why this is useful in practice and when it is beneficial. To do so, we start from precisely controlled dataset variations and data samplings and find that the effect of adding a classification loss is the most pronounced for regression with imbalanced data. We explain these empirical findings by formalizing the relation between the balanced and imbalanced regression losses. Finally, we show that our findings hold on two real imbalanced image datasets for depth estimation (NYUD2-DIR), and age estimation (IMDB-WIKI-DIR), and on the problem of imbalanced video progress prediction (Breakfast). Our main takeaway is: for a regression task, if the data sampling is imbalanced, then add a classification loss.

Figure 1. To probe *"Why does classification help regression?"* we design a fully controlled dataset including the following scenarios: *Clean data* – $1D$ non-linear functions defined by the sum of two sine waves with different frequencies and amplitudes; *Noisy data* – uniform noise added to the outputs; *Out of distribution* – sampling different regions of the input space during training and during testing. (We show a single function here. The gray shading groups the function targets into 4 classes, as an example.)

1. Introduction

Regression models predict continuous outputs. In contrast, classification models make discrete, binned, predictions. For a continuous task, regression targets are a superset of the classification labels: they are more precise, taking values in-between the discrete classification bins. For regression, the error is only bounded by the precision of the measurements, for classification this also depends on the bin sizes: e.g. an age estimation classifier that can predict only young/old classes, cannot discriminate between middle-aged people. Additionally, when training a regression model, losses are proportional to the error magnitude, while for classification all errors receive an equal penalty: predicting bin 10 instead of 20, is just as incorrect as predicting bin 10 instead of bin 100. So classification cannot add anything new to regression; or can it?

Surprisingly, adding a classification loss to the regression loss [30, 44, 46, 51], or even replacing the regression loss with classification [10, 11, 38] is extensively used in practice when training deep models for predicting continu-
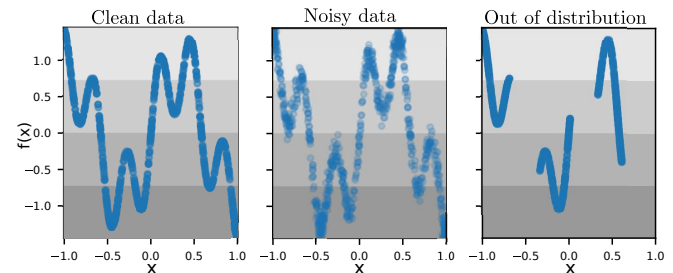
ous outputs. The classification is typically defined by binning the regression targets into a fixed number of classes. This is shown to be beneficial for tasks such as: depth estimation [11], horizon line detection [44], object orientation estimation [30, 51], age estimations [34]. And the reported motivation for discretizing the regression loss is that: it improves performance [44, 46], or that it helps in dealing with noisy data [44], or that it helps overcome the overly-smooth regression predictions [30, 40], or that it helps better regularize the model [22, 44]. However, none of these assumptions has been thoroughly investigated.

In this work, we aim to explore in the context of deep learning: *Why does classification help regression?* Intuitively, the regression targets contain more information than the classification labels. And adding a classification loss does not contribute any novel information. What is it really that a classification loss can add to a standard MSE (mean squared error) regression loss? And why does it seem beneficial in practice?

To take a step towards understanding why classification helps regression, we start the analysis in a fully controlled setup, using a set of $1D$ synthetic functions. We consider several prior hypothesis of when classification can help re-

gression: *noisy* data, *out-of-distribution* data, and the normal *clean* data case, as in Fig. 1. Additionally, we vary the sampling of the data from uniform to highly imbalanced as in Fig. 2. We empirically find out in which of these cases adding a classification loss improves the regression quality on the test set. Moreover, we explain these empirical observations by formulating them into a probabilistic analysis.

We urge the reader to note that our goal is not proposing a novel regression loss, nor do we aim to improve results with "superior performance" over state-of-the-art regression models. But rather, we aim to investigate a common computer vision practice: *i.e.* adding a classification loss to the regression loss, and we analyze for what kind of dataset properties and dataset samplings this practice is useful. Finally, we show experimentally that our findings hold in practice on two imbalanced real-world computer vision datasets: NYUD2-DIR (depth estimation) and IMDB-WIKI-DIR (age estimation) [47], and on the Breakfast dataset [23] when predicting video progression.

2. Why does classification help regression?

For an input dataset, $\mathcal{D}$, containing N samples of the form $(\mathbf{x}, y) \in \mathcal{D}$, we analyze what happens when we train a deep network with parameters $\boldsymbol{\omega}$ to predict a target $y^* = f(\mathbf{x}, \boldsymbol{\omega})$ for a sample $\mathbf{x}$, by minimizing NLL (negative log-likelihood) or equivalently, minimizing the MSE (mean squared error) regression loss:

$$\boldsymbol{\omega}^* = \arg\min_{\boldsymbol{\omega}} \sum_{(\mathbf{x}, y) \in \mathcal{D}} L(y, \mathbf{x}, \boldsymbol{\omega}) \tag{1}$$

$$= \arg\min_{\boldsymbol{\omega}} \left(\sum_{(\mathbf{x}, y) \in \mathcal{D}} -\log p(y|\mathbf{x}, \boldsymbol{\omega}) \right) \tag{2}$$

$$\equiv \arg\min_{\boldsymbol{\omega}} \lambda \frac{1}{N} \sum_{(\mathbf{x}, y) \in \mathcal{D}} (y - y^*)^2 \tag{3}$$

where $\boldsymbol{\omega}^*$ are the optimal parameters, and we can reinterpret the imbalanced likelihood as the mean of a Gaussian distribution with σ noise: $p(y|\mathbf{x}, \boldsymbol{\omega}) = \mathcal{N}(y; y^*, \sigma^2 I)$, in which case minimizing the NLL is equivalent to minimizing the MSE loss [3], and λ is a function of the noise σ.

We contrast Eq. (3) to the case when we discretize the targets y into a set of C classes and use a classification loss next to the regression loss:

$$L(y, \mathbf{x}, \boldsymbol{\omega}) = \lambda (y - y^*)^2 - \log p(y_c^*|\mathbf{x}, \boldsymbol{\omega}), \tag{4}$$

where y_k^*, $k \in \{1, .., C\}$ denotes the model predictions binned into classes, and specifically y_c^* is the prediction at the true class indexed by c, for the sample $\mathbf{x}$. For the classification term, we make the standard *softmax* distribution assumption.
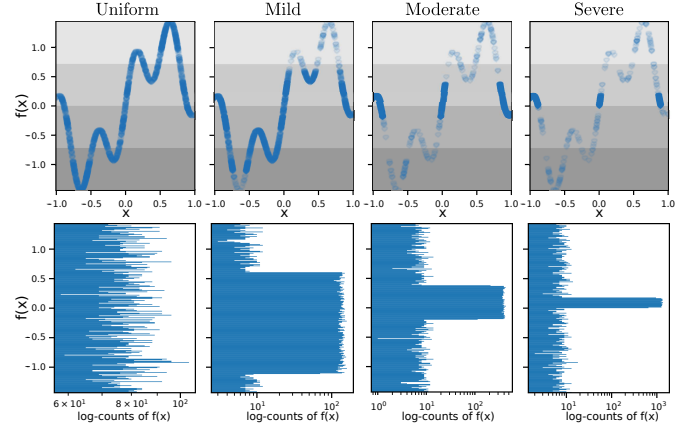

Figure 2. Data sampling. On the columns we increase the data imbalance from uniform (balanced) to severely imbalanced. On the first row we show the function $f(\mathbf{x})$ where darker datapoint colors visualize higher density. The gray shading on the first row groups the targets into 4 classes. On the second row we show the log-counts per function value, sampled for the training data. We sample the test data uniformly.

2.1. Controlled 1D analysis

To probe the question "*Why does classification help regression?*" we first want to know in which cases does classification help regression. We measure test-time MSE scores for each case in Fig. 1, and compare training with a regression loss as in Eq. (3), with training using an extra classification loss as in Eq. (4). We randomly sample 10 functions of the form: $f(x) = a \sin(cx) + b \sin(dx)$, where $f(x) \in [-1.5, 1.5]$ and $x \in [-1, 1]$. For every function we vary the dataset scenario as in Fig. 1, and we also vary the sampling of the data from uniform to severely imbalanced sampling, as in Fig. 2. Each dataset sampling is repeatedly performed with 5 different random seeds, where we always sample $\approx 30,000$ samples in total, and then randomly pick 1/3 for training, for validation, and for testing, respectively. In the *out-of-distribution* case, the training set misses certain function regions that are present in the validation/test set, and vice-versa; and there is an overlap of 1/4 between the function regions in the training set and the regions in the validation/test set. For the imbalanced sampling, we aim to sample a range of the targets y more frequently than other ranges. For this, we randomly select a location along the y-axis in each repetition: this defines the center of the peak in Fig. 2, second row. And depending on the sampling scenario, we use a fixed variance ratio around the peak (0.3, 0.1 and 0.03 for *mild*, *moderate* and *severe* sampling) to define the region from which we draw the frequent samples. We sample 75% of the samples from the peak region, and the rest uniformly from the other function areas.

We train a simple MLP (multi layer perceptron) with 3 linear layers ($[1 \times 6]$, $[6 \times 16]$, $[16 \times 1]$) and ReLU non-
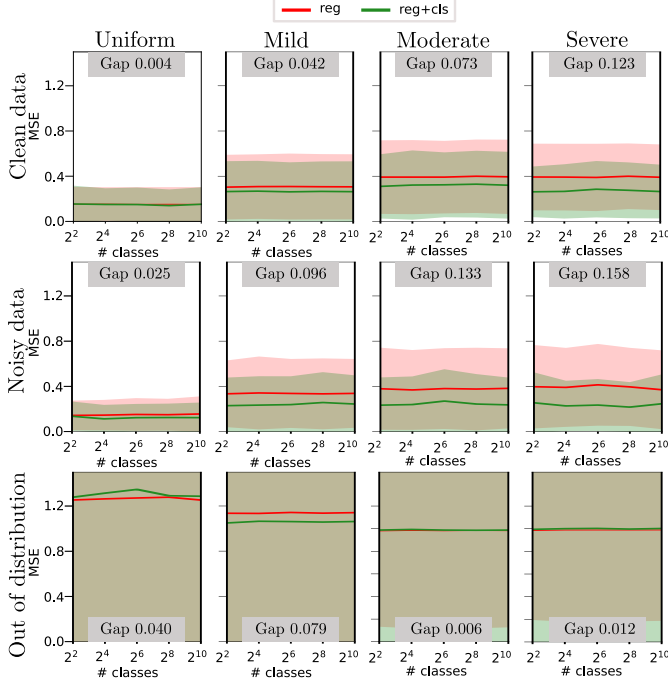
Figure 3. MSE per class, where we vary the number of classes from 4 to 1024. We evaluate on uniformly sampled test sets, across 5 repetitions. We plot means and standard deviations for each dataset (rows) and sampling variation (columns), where the shading represents the standard deviation. The red line, *reg*, should be constant across classes but it varies due to the sampling/training randomness. We also print the gap between the *reg* and *reg+cls* measured as absolute difference of MSE scores. The effect of the classification loss is present when the sampling of the data is imbalanced for the *clean* and *noisy* data.

linearities. For setting the hyperparameter λ, we perform a hyper-parameter search on the validation set. We find the best λ to be $1e+2$, $1e+3$ and $1e+4$ for the *clean*, *noisy* and *out-of-distribution*. We train for 80 epochs using an Adam optimizer with a learning rate of $1e-3$, $1e-2$ and $1e-4$ for *clean*, *noisy* and *out-of-distribution* respectively. We use a weight decay of $1e-3$. For classification we add at training-time a linear layer ($[16 \times C]$) predicting C classes. More details are in the supplementary material.

In Fig. 3 we show the MSE across all dataset and sampling variations, for $\{2^2, 2^4, 2^6, 2^8, 2^{10}\}$ classes. We define the class ranges uniformly. The test sets are uniformly sampled and we perform 5 repetitions. We plot the means and standard deviations. We print on every plot the gap between the *reg* and *reg+cls* measured as the average absolute difference of MSE scores. From this $1D$ analysis, we observe that the effect of the classification loss is visible when the training data is imbalanced for *clean* and *noisy* data.

2.2. Anchoring $1D$ experimental observations

In Section 2.1 we observe that classification has a more pronounced effect when the sampling of the data is imbalanced. Therefore, from here on we focus the analysis on imbalanced data sampling. We start from the derivations of Ren *et al.* [31] who define the relation between the NLL (negative log-likelihood) of imbalanced samples $-\log \widetilde{p}(y|\mathbf{x}, \boldsymbol{\omega})$ and the NLL of the balanced samples $-\log p(y|\mathbf{x}, \boldsymbol{\omega})$:

$$-\log \widetilde{p}(y|\mathbf{x}, \boldsymbol{\omega}) = -\log \frac{p(y|\mathbf{x}, \boldsymbol{\omega})\widetilde{p}(y)}{\int_{y'} p(y'|\mathbf{x}, \boldsymbol{\omega})\widetilde{p}(y')dy'} \quad (5)$$

where $\widetilde{p}(y)$ denotes the prior over imbalanced targets. Eq. (5) holds under the assumption that the data function remains unchanged, $\widetilde{p}(\mathbf{x}|y) = p(\mathbf{x}|y)$, which is the case for the *clean* and *noisy* data scenarios above. We decompose the $\log$ and rewrite the relation between the NLL of the balanced and the NLL of the imbalanced data:

$$-\log \widetilde{p}(y|\mathbf{x}, \boldsymbol{\omega}) + L_{\text{extra}}(y, \mathbf{x}, \boldsymbol{\omega}) = -\log p(y|\mathbf{x}, \boldsymbol{\omega}), \quad (6)$$

L_{extra} contains all the information about the imbalanced regression targets:

$$L_{\text{extra}} = \log \widetilde{p}(y) - \log \int_{y'} p(y'|\mathbf{x}, \boldsymbol{\omega})\widetilde{p}(y')dy', \quad (7)$$

$$= \log \widetilde{p}(y) - \log \int_{y'} \mathcal{N}(y'; y^*, \sigma^2 I)\widetilde{p}(y')dy', \quad (8)$$

where again y^* are the predicted targets.

To derive the link between optimizing a model on imbalanced data and using both a regression MSE loss and a classification loss, we assume the imbalanced regression targets y can be discretized into a set of classes, $k \in \{1, .., C\}$ such that $\sum_{k=1}^{C} p(y_k) = 1$. By going from continuous regression targets to discrete classes, we change the form of the log-likelihood from Gaussian to *softmax*:

$$L_{\text{extra}} \approx \log \widetilde{p}(y_c) - \log \sum_{k=1}^{C} p(y_k^*|\mathbf{x}, \boldsymbol{\omega})\widetilde{p}(y_k), \quad (9)$$

where we denote the true class label by y_c. Note that the regression targets y are imbalanced, but the classes y_k do not necessarily need to be imbalanced. We analyze in the experimental section the effect of defining balanced classes.

We make the observation that if we could optimize the class assignment, the L_{extra} term would disappear. If the classes are optimized, then the class likelihoods are close to 0 for all classes except the true class: $p(y_k^*|\mathbf{x}, \boldsymbol{\omega}) \approx 0$, $\forall k \neq c$, where c indexes the true class. Using this in the expression of L_{extra}, we obtain:

$$L_{\text{extra}} \approx \log \widetilde{p}(y_c) - \log p(y_c^*|\mathbf{x}, \boldsymbol{\omega})\widetilde{p}(y_c), \quad (10)$$

$$\approx -\log p(y_c^*|\mathbf{x}, \boldsymbol{\omega}), \quad (11)$$

where the $\widetilde{p}(y_c)$ terms cancel out when decomposing the second $\log$. Therefore, we can see that optimizing the class cross-entropy loss reduces the gap between between the NLL of imbalanced data and NLL of balanced data. (Note: we observe in practice that if the classifier fails to converge, adding a classification loss is detrimental to regression.)

2.3. Defining balanced classes in practice

Existing works show that optimizing imbalanced classes is problematic [26, 49]. In practice, researchers opt for using balanced classes in combination with regression [30, 44, 46, 51]. Here, the data is imbalanced, however we are free to define the class ranges such that we obtain balanced classes over imbalanced data sampling. To empirically test the added value of using balanced classes, we need a way to define balanced classes over imbalanced data sampling.

Given an imbalanced data sampling, we bin samples into classes, using uniform class ranges. This generates imbalanced classes, which we then re-balance by redefining the class ranges such that the class histogram is approximately uniform. To this end, we apply histogram equalization over the original classes:

$$q(k) = \left\lfloor \frac{C}{N} \sum_{j=1}^{k} \mathcal{H}_C(j) \right\rfloor, \tag{12}$$

where $\lfloor x \rfloor$ rounds x down to the nearest integer, and $\mathcal{H}_C(\cdot)$ computes the histogram of the samples per class, and $q(\cdot)$ is a mapping function that maps the old classes indexed by $k \in \{1, .., C\}$ to a new set of classes $\{1, .., \overline{C}\}$. Eq. (12) merges class ranges such that their counts are as close as possible. Thus, the number of equalized classes is lower or equal to the original number of classes, $\overline{C} \leq C$.

After class equalization, the new classes are not perfectly uniform. We further define a class-keeping probability $\rho(k)$, as the ratio between the minimum class count and the current equalized class count $\mathcal{H}_{\overline{C}}(k)$:

$$\rho(k) = \frac{\min_{j=1}^{\overline{C}} \mathcal{H}_{\overline{C}}(j)}{\mathcal{H}_{\overline{C}}(k)}, \tag{13}$$

where $\mathcal{H}_{\overline{C}}(\cdot)$ computes the histogram of equalized classes. Selecting training samples using only Eq. (13), without first equalizing the classes, will lead to never seeing samples from the most frequent classes. During training, for the regression loss we use all samples, while for the classification loss we pick samples $(\mathbf{x}, y_k)$ with a probability defined by $\rho(k)$. More details are in the supplementary material.

3. Empirical analysis

3.1. Hypothesis analysis on 1D data

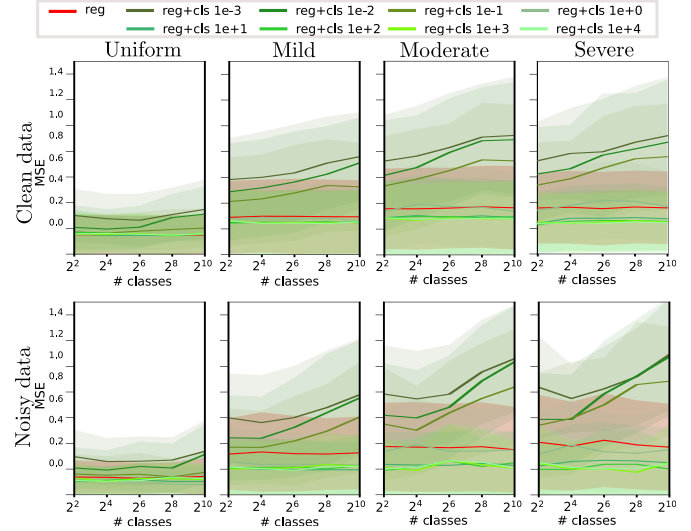We use the 10 randomly sampled $1D$ functions to further analyze the regression loss — *reg* from Eq. (3), and

Figure 4. **Effect of the λ hyperparameter on the validation:** We evaluate a range of values $\lambda \in \{1e{-}3, 1e{-}2, 1e{-}1, 1, 1e{+}1, 1e{+}2, 1e{+}3, 1e{+}4\}$ on the validation set. The shading represents the standard deviation of the MSE error across 3 repetitions. Setting λ correctly is essential when using a classification loss next to the regression loss.

regression with classification — *reg+cls* from Eq. (4). We start with the $1D$ data because it is easily interpretable and it offers a controlled environment to test the hypothesis that classification helps regression and to analyze the properties of the classes. [1]

Effect of the λ hyperparameter. We perform hyperparameter search on the validation set for setting the λ in Eq. (4). We vary $\lambda \in \{1e{-}3, 1e{-}2, 1e{-}1, 1, 1e{+}1, 1e{+}2, 1e{+}3, 1e{+}4\}$. Fig. 4 shows the MSE across 3 repetitions, when considering different number of classes, for *clean* and *noisy* data scenarios, across sampling variations. Higher values of λ typically perform better in this case. When the sampling of the data is imbalanced, there exists a value of λ such that the *reg* baseline is outperformed by the *reg+cls*. We use the best λ values found on the validation, when evaluating on the test set.

Effect of balancing the classes on imbalanced data. We numerically evaluate in Tab. 1 if using balanced classes (as defined in Eq. (12)-Eq. (13)) is less sensitive to the choice of λ. We perform 3 repetitions over all dataset scenarios and sampling cases, and vary $\lambda \in \{1e{-}3, 1e{-}2, 1e{-}1, 1, 1e{+}1, 1e{+}2, 1e{+}3, 1e{+}4\}$. For this we consider the percentage of runs (across different number of classes, random seeds, and values of λ) where classification helps regression — where *reg+cls* MSE is lower than *reg* MSE. Ideally this number should be close to 100%. Balancing the classes is more robust to the choice of λ, as on average there

---

[1] Our source code will be made available online, at the address: https://github.com/SilviaLauraPintea/reg-cls.

Dataset case	Imbalanced classes (↑)		Balanced classes (↑)	
	Clean	Noisy data	Clean	Noisy data
Uniform	55.42%	59.83%	59.00%	65.00%
Mild	54.17%	55.75%	57.67%	62.42%
Moderate	47.50%	56.58%	47.50%	56.00%
Severe	36.83%	49.08%	34.25%	46.25%
Avg	48.48%	55.31%	49.60%	57.42%

Table 1. **Effect of balancing the classes:** On the validation sets we test how sensitive *reg+cls* is to the choice of λ when balancing classes versus when using imbalanced classes. For this we vary $\lambda \in \{1e{-}3, 1e{-}2, 1e{-}1, 1, 1e{+}1, 1e{+}2, 1e{+}3, 1e{+}4\}$. And we measure the percentage of runs where adding a classification loss improves the regression predictions. Using balanced classes is less sensitive to the choice of λ.
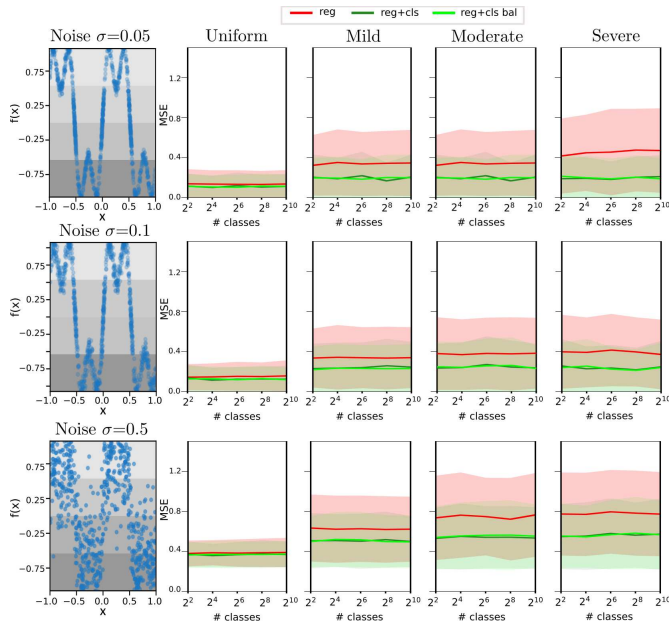

Figure 5. **Effect of noisy targets:** We vary the amount of noise in the targets, along the y-axis. We plot both using imbalanced classes (green), and using balanced classes (lime) compared to the *reg* baseline (red). We report MSE on the test sets across 5 repetitions. Despite the the noise level increasing drastically, the benefits of adding a classification loss to the regression loss remain visible.

are more runs where classification helps regression.

Effect of the noisy targets. In Fig. 3 we considered a single noise level for the *Noisy data* scenario. Here, we further analyze the effect on the MSE scores when varying the noise level in the targets for the baseline *reg* (in red) and the *reg+cls* with imbalanced classes (in green) as well as *reg+cls bal* where the classes are balanced (in lime color). We vary the level of noise σ of the targets on the y-axis $\sigma \in \{0.05, 0.1, 0.5\}$, and plot mean MSE and standard deviation on uniform test sets across 5 repetitions. For the balanced case, we indicate the original number of classes on the x-axis, while in practice the number of balanced classes

is typically $2\times$ lower than the initial one. Fig. 5 shows that despite severely increasing the noise level, adding a classification loss remains beneficial when the data is imbalanced.

3.2. Realistic image datasets

Imbalanced realistic image datasets. We run experiments on two realistic imbalanced datasets from Yang *et al.* [47]: depth estimation on the *NYUD2-DIR* dataset, and age estimation on the *IMDB-WIKI-DIR*. The supplementary material plots dataset statistics. For both datasets we use the Adam optimizer and set the learning rate to $1e{-}4$ for *NYUD2-DIR* with 5 epochs[1] and batch size 16 accumulated over 2 batches (to mimic batch size 32 on 2 GPUs), while for *IMDB-WIKI* we use a learning rate of $1e{-}3$ for 90 epochs and batch size 128. When comparing with Ren *et al.* [31] we use their best results (their GAI method). We also use the evaluation code provided in Yang *et al.* [47] and we report RMSE (root mean squared error) on *NYUD2-DIR* and MAE (mean absolute error)/MSE (mean squared error) for *IMDB-WIKI-DIR* as also done in [31, 47]. When re-running the baseline *reg* results we observed a large variability across different runs by varying the random seed, especially on the *NYUD2-DIR* dataset, therefore we report results averaged over 3 random seeds. We use the architecture of Yang *et al.* [47]: ResNet-50 [17] for *IMDB-WIKI-DIR*, and the ResNet-50-based model from [19] for *NYUD2-DIR*.

For adding the classification loss on *IMDB-WIKI-DIR* we append, only during training, a linear layer of size $[F, C]$ followed by a softmax activation and a cross-entropy loss, where F is the number of channels in the one-to-last layer. For the *NYUD2-DIR* the predictions are per-pixel, thus we use the segmentation head from Mask R-CNN [16] composed of a transposed convolution of size 2×2, ReLU, and a 1×1 convolution predicting the number of classes, C. At test time the classification branch is not used. We estimate the λ hyperparameter using the validation set provided in [47] on *IMDB-WIKI-DIR*. For *NYUD2-DIR* we define a validation set by randomly selecting 1/5 of the training directories with a seed of 0, and we use the same training/validation/test split for all our results. For *NYUD2-DIR* we use $\lambda{=}1.0$ for 100 classes and $\lambda{=}0.1$ for 10 and 2 classes. For *IMDB-WIKI-DIR* we set $\lambda{=}0.1$ for 100 classes and $\lambda{=}1.0$ for 10 and 2 classes. We compare the *reg* results with *reg+cls* adding the classification loss, and *reg+cls bal.* with balanced classes, where we consider 2, 10 and 100 classes.

Tab. 2 shows the RMSE results on *NYUD2-DIR* when training with the standard MSE loss compared to adding a classification loss. We report RMSE on the test, where the best model is selected on the validation set across epochs — *RMSE-val.* To compare with previous work who selects the best model on the test, we also report this as *RMSE-test*

---

[1]Using more epochs on *NYUD2-DIR* seems to lead to overfitting.

Samples	NYUD2-DIR RMSE-val (↓)				NYUD2-DIR RMSE-test (↓)			
	All	Many	Med.	Few	All	Many	Med.	Few
Kernel [47]	—	—	—	—	1.338	0.670	0.851	1.880
Balanced [31]	—	—	—	—	1.251	0.692	0.959	1.703
reg (MSE)	1.614 (±0.051)	0.554 (±0.002)	0.934 (±0.042)	2.360 (±0.081)	1.499 (±0.083)	0.578 (±0.010)	0.896 (±0.043)	2.171 (±0.136)
reg+cls (2 cls)	1.587 (±0.026)	0.618 (±0.003)	1.062 (±0.025)	2.278 (±0.041)	1.532 (±0.082)	0.624 (±0.036)	0.946 (±0.032)	2.204 (±0.141)
reg+cls (10 cls)	1.576 (±0.063)	0.585 (±0.008)	0.982 (±0.058)	2.282 (±0.095)	1.509 (±0.022)	0.582 (±0.013)	0.947 (±0.046)	2.178 (±0.047)
reg+cls (100 cls)	1.536 (±0.090)	0.569 (±0.018)	0.966 (±0.041)	2.222 (±0.146)	1.488 (±0.028)	0.578 (±0.015)	0.971 (±0.049)	2.141 (±0.045)
reg+cls bal. (2 cls)	1.599 (±0.020)	0.616 (±0.026)	1.033 (±0.058)	2.304 (±0.044)	1.522 (±0.060)	0.665 (±0.033)	1.003 (±0.059)	2.166 (±0.118)
reg+cls bal. (10 cls)	1.454 (±0.044)	0.607 (±0.041)	0.965 (±0.023)	2.077 (±0.087)	1.454 (±0.044)	0.607 (±0.041)	0.965 (±0.023)	2.077 (±0.087)
reg+cls bal. (100 cls)	1.553 (±0.117)	0.563 (±0.033)	0.897 (±0.043)	2.263 (±0.193)	1.487 (±0.051)	0.574 (±0.014)	0.869 (±0.019)	2.156 (±0.084)

Table 2. **Imbalanced realistic image data: NYUD2-DIR depth estimation.** We evaluate the baseline *reg* trained with MSE, and the *reg+cls* variants. We report RMSE when the best model is selected on the validation (*RMSE-val*), or as in [47] on the test set (*RMSE-test*). We average over 3 different random seeds. Adding a classification loss helps regression, which validates our hypothesis.

Samples	IMDB-WIKI-DIR MAE(↓)				IMDB-WIKI-DIR MSE(↓)			
	All	Many	Med.	Few	All	Many	Med.	Few
Focal [27]	7.97	7.12	15.14	26.96	136.98	106.87	368.60	1002.90
Kernel [47]	7.78	7.20	12.61	22.19	129.35	106.52	311.49	811.82
Balanced [31]	8.12	7.58	12.27	23.05	—	—	—	—
reg (MAE)	8.09 (±0.01)	7.23 (±0.02)	15.48 (±0.15)	26.81 (±0.48)	138.53 (±1.17)	107.82 (±0.96)	375.27 (±6.97)	1017.59 (±27.51)
reg+cls (2 cls)	7.95 (±0.05)	7.11 (±0.03)	15.08 (±0.27)	26.15 (±0.06)	135.15 (±0.40)	105.86 (±0.27)	361.02 (±7.00)	973.53 (±11.52)
reg+cls (10 cls)	7.93 (±0.06)	7.12 (±0.06)	14.93 (±0.17)	25.91 (±0.27)	135.69 (±1.65)	106.58 (±1.42)	359.27 (±5.70)	975.37 (±34.09)
reg+cls (100 cls)	7.61 (±0.02)	6.90 (±0.03)	13.46 (±0.44)	25.04 (± 0.82)	129.73 (±1.33)	103.25 (±1.12)	328.17 (± 10.40)	933.33 (± 9.16)
reg+cls bal. (2 cls)	7.94 (±0.06)	7.14 (±0.09)	14.71 (±0.51)	26.20 (±0.73)	134.91 (±1.20)	106.36 (±1.68)	351.48 (±14.82)	980.61 (±39.44)
reg+cls bal. (10 cls)	7.92 (±0.06)	7.10 (±0.05)	14.99 (±0.21)	25.58 (±0.39)	134.53 (±0.75)	105.40 (±0.43)	362.48 (±4.41)	940.91 (±18.70)
reg+cls bal. (100 cls)	7.59 (±0.09)	6.86 (±0.08)	13.61 (±0.35)	25.30 (±0.19)	127.87 (±1.99)	101.34 (±1.74)	327.55 (±12.98)	925.65 (±21.96)

Table 3. **Imbalanced realistic image data: IMDB-WIKI-DIR age estimation.** We evaluate the baseline *reg* when trained with MAE, and the *reg+cls* variants. We report test MSE and MAE on the test set, averaged over 3 repetitions with different random seeds. Adding a classification loss next to the regression loss is specifically beneficial on this dataset, where this simple yet popular strategy performs on par with state-of-the-art. (We use "—" where the authors' results are missing).

.

Balanced NYUD2-DIR		
	All RMSE-val (↓)	All RMSE-test (↓)
reg (MSE)	1.442 (±0.077)	1.492 (±0.042)
reg+cls	1.456 (±0.033)	1.593 (±0.025)
Balanced IMDB-WIKI-DIR		
	All MAE (↓)	All MSE (↓)
reg (MSE)	7.74 (±0.04)	131.03 (±1.44)
reg+cls	7.71 (±0.06)	131.27 (±1.00)

Table 4. **Balanced realistic image data: NYUD2-DIR depth estimation and IMDB-WIKI-DIR age estimation.** We compare the *reg* and *reg+cls* (using 100 classes) when the data is balanced. We report RMSE and MAE/MSE, respectively, across 3 repetitions. There are no clear improvements when adding a classification loss to the regression on balanced data.

(despite this being a bad practice). Additionally, note that our training set is slightly smaller because of using a validation set, so the *reg* results are worse than in [47] (*i.e.* 1.477 *RMSE-test*). We observe that there is an inconsistency between the training and test set, as the best model on the test set does not correspond to the best model on the validation set (which is a subset of the training). Despite all these, adding a classification loss still improves across all class-options, when selecting the best model on the validation set,

and for 100 classes and 10 balanced classes, when selecting the best model on the test set. This may be due to the classifier overfitting on the training data for fewer classes.

Tab. 3 gives the MAE and MSE results on *IMDB-WIKI-DIR* when using the standard MSE loss during training compared to when adding the classification at training time. The best results are obtained using 100 balanced classes. Here, adding a classification loss not only improves over the regression baseline, but it is also on-par with state-of-the-art methods specifically designed for imbalanced regression, such as [32, 47]. Adding a classification loss is similar to [32, 47], who define smooth classes over the data.

Balanced realistic image datasets. On the same two datasets: *NYUD2-DIR* and *IMDB-WIKI*, we test the effect of adding a classification loss when we re-balance the data by binning the targets into 100 bins and selecting samples per batch during training as defined as in Eq. (12)-Eq. (13) for both *reg* and *reg+cls*. Because we mask samples per batch to balance the training data, for *IMDB-WIKI* here we use a batch size of 128 and learning rate $1e-4$ for 90 epochs. While for *NYUD2-DIR* we use a learning rate of $1e-4$ and batch size of 8, accumulated over 4 batches (to mimic a batch of 32 on 1 GPU), for 5 epochs. Tab. 4 shows

Dataset split	Breakfast RMSE % (↓)				Breakfast RMSE frames (↓) – unnormalized			
	S1	S2	S3	S4	S1	S2	S3	S4
Random baseline	58.50 (±0.29)	58.37 (±0.07)	58.39 (±0.14)	58.38 (±0.12)	1394.48 (±992.50)	1511.04 (±1132.60)	1450.26 (±1085.56)	1420.21 (±1063.16)
[24]	31.24 (±0.80)	31.49 (±0.98)	30.85 (±0.66)	30.78 (±0.53)	1079.38 (±775.17)	1235.02 (±932.89)	1172.67 (±880.69)	1170.07 (±886.92)
reg (RMSE)	32.57 (±1.45)	33.30 (±1.69)	32.52 (±1.17)	33.08 (±1.49)	860.17 (±583.44)	891.39 (±641.58)	862.65 (±609.01)	845.48 (±605.46)
reg+cls (100 cls)	28.71 (±0.38)	28.84 (±0.55)	28.46 (±0.48)	28.44 (±0.50)	837.59 (±573.15)	870.55 (±630.89)	845.65 (±601.73)	809.76 (±595.37)
reg+cls bal. (100 cls)	—	—	—	—	837.61 (±573.17)	870.54 (±630.88)	845.65 (±601.72)	809.76 (±595.37)
	(a) Percentage prediction (normalized).				(b) Frame prediction (unnormalized).			

Table 5. **Imbalanced video progress prediction on Breakfast.** We report mean RMSE and standard deviations averaged over all 10 cooking tasks of the Breakfast dataset. (a) Progress prediction in video percentages. (b) Progress prediction in frame numbers. The overall progress prediction results leave space for improvements for all methods, because of the dataset challenges. However, also for this regression problem adding a classification loss has benefits.

the results across 3 repetitions. When the data is already balanced, adding a classification loss has limited effect.

3.3. Imbalanced video progress prediction

As an additional investigation on imbalanced data, we explore videos which are naturally imbalanced in the number of frames. We perform video progress prediction on the Breakfast video dataset [23] containing 52 participants performing 10 cooking activities. We follow the standard dataset split (S1, S2, S3, S4) [23] and use the corresponding train/test splits. We adopt the method in [24] and train a simple MLP on top of IDT (improved dense trajectory) features [41] of dimension 64 over trajectories of 15 frames. We evaluate RMSE when predicting either video progress percentages, or absolute frame numbers. Kukleva *et al*. [24] use an MLP with 3 linear layers and sigmoid activations. We change the sigmoid activations into ReLU activations for the *reg* and *reg+cls* models, since it works better when predicting absolute frame numbers. For all methods we keep the training hyperparameters from [24]: learning rate $1e{-}3$, Adam optimizer, and training for 40 epochs with the learning rate decreased by 0.1 at 30 epochs. We also report the random baseline results when using the untrained model. The data is imbalanced, in the sense that video lengths vary widely (see supplementary material for data distribution). We test again if adding a classification loss can benefit the regression predictions when using 100 classes in the *reg+cls*. We search for λ on a validation set created by randomly selecting 1/3 of the training videos with a seed of 0. We use $\lambda{=}100$ when predicting percentages, and $\lambda{=}10$ when predicting frame numbers. At training time, we add the classification loss via a linear layer on top of the one-to-last layer, followed by *softmax*. We train one model per task and report mean and standard deviations over all 10 tasks.

Tab. 5 depicts the results across all 4 splits. We report RMSE scores when predicting video progress in terms of percentages in Tab. 5(a), and when predicting video progress in terms of frames in Tab. 5(b). Even if we predict video progress percentages between [0,100]% in Tab. 5(a), because the video length varies widely, for some videos we

will have a lot more frame than for others, causing the data sampling to still be imbalanced. In both cases there is gain from adding a classification loss to the regression loss.

4. Discussion and limitations

Relation to nonlinear ICA. Hyvärinen *et al*. [21] show that there is a relation between learning to bin a continuous function and performing nonlinear ICA. Hyvärinen *et al*. [21] start from a continuous signal $\mathbf{x}$ generated by a non-linear combination of source-signals: $\mathbf{x}{=}f(\mathbf{s})$, where $f(\cdot)$ is a non-linear function and $\mathbf{s}$ are the independent and non-stationary sources. They split the signal into C temporal segments $\mathbf{x}{=}\{\mathbf{x}_1, \mathbf{x}_2, .., \mathbf{x}_C\}$ and train an MLP to predict for every sample $\mathbf{x}_{k_i}$ the segment it belongs to: $g(\mathbf{x}_{k_i}, \theta){=}k, k{\in}\{1, .., C\}$ and $\mathbf{x}_{k_i}{\subset}\mathbf{x}_k$, and θ are the MLP parameters. Hyvärinen *et al*. [21] prove that the last hidden layer of the MLP $h_g(\cdot, \theta)$ recovers the original sources $\mathbf{s}$ within a linear transformation: $h_g(\mathbf{x}_{k_i}, \theta) = \mathbf{w}\mathbf{s}_{k_i} + \mathbf{b}$, where $\mathbf{w}, \mathbf{b}$ define a linear transformation. Intuitively, Hyvärinen *et al*. [21] discretize the signal into segments and classify the segments, thus performing classification on a continuous function. However, they do not focus on combining the binning with optimizing a regression problem.

Similar to Hyvärinen *et al*. [21], predicting discrete signal bins has been successfully used for unsupervised video representation learning by discriminating between video segments [9] or classifying video speed [42]. Up to the point which the underlying continuous regression function (*e.g*. speed, time, age, depth) can be assumed to be generated by a non-linear combination of non-stationary source (whose statistics change with the function), adding a classification loss decorrelates the independent sources. Additionally, we hypothesize that there may be a relation between nonlinear ICA and imbalanced sampling: the independent sources $\mathbf{s}$ are also continuous and shared across samples. And having the hidden representation of the MLP constrained to be independent across dimensions may lead to a better use of sparse samples in certain areas of the target-space. However, we leave this for future research.

Limitations of analysis. The analysis performed here is still elementary and only aims to scratch the surface on the

usefulness of adding a classification loss when performing regression. A number of things have been disregarded here such as: the effect of the model depth, while keeping the model size fixed. Additionally, the choice of the optimizer and the loss function during training may also play an important role. Finally, delving more into the relation between nonlinear ICA and adding a classification loss to the regression, may be an interesting future research avenue.

5. Related work

5.1. Improved deep regression

A thorough analysis of the effect of deep architecture choices on regression scores is performed in [25]. Rather than considering architecture choices, other works focus on regression robustness – defined as less influence from outliers [15, 20]. The benefits of having a weighted regression loss has been extensively analyzed in classic work such as [6, 7]. With a similar goal, Barron [1] proposes a general regression loss that can be adapted to well-known functions. Minimizing a Tukey's biweight function also offers robustness to outliers [2]. Similarly, a smooth L_1 loss is used in [12] for improved object bounding-box regression. Mapping regression targets to the hyper-sphere can also aid regression [29]. Instead of focusing on the loss function, a smooth adaptive activation function is used in [18]. From a different direction, ensemble networks have been successfully used for improved regression estimates [8, 14, 39]. Deep negative correlation learning is proposed in [48] to learn diversified base networks for deep ensemble regression. Dissimilar to these works, our goal is not proposing a new network architecture or a new regression loss function, but rather analyzing why a combination between a classification and a regression loss can lead to improvements.

5.2. Discretized regression targets

Instead of optimizing continuous object pose angles, [38, 36] discretize them into classes and perform classification. Continuous prediction can be obtained back from discretized targets, by using *soft-min* for disparity learning [22]. Similarly, ages are discretized in classes and the final prediction is computed as an expectation over class probabilities in [34]. Rather than using predefined classes, clusters can be defined and, again, the final predication is a weighted sum over clusters [40, 43]. In a similar manner, a weighted combination of clustered regression targets is used for finding surface normals in [43]. Popular object detectors [28, 32, 33, 45] also rely on a set of predefined box locations that can be seen as bin centers, and regress the final box locations with respect to these centers. An explicit combination of regression and classification losses has been shown to improve results for object orientation regression [30, 50, 51] by first splitting orientations into bins and then regressing the precise value in each bin. Similarly, a joint classification loss over discretized targets and a regression loss is effective for horizon line detection [44]. Here, we want to analyze why these prior works opt for discretizing regression targets. More specifically, we analyze why and when a combination of a regression loss and a classification loss over discretized continuous targets improves results.

5.3. Regression on imbalanced data

Prior work has shown that imbalanced sampling can negatively affect the regression scores, and proposed ways to mitigate this by designing better data sampling techniques [4, 37]. How rare a data point is in the training set, can be modeled with kernel density estimation [35]. Not only the data sampling but also the target distribution can be fixed by smoothing the distribution of both labels and features using nearby targets [47]. Similarly, the learning model can be regularized such that samples that are close in label space are also close in the feature space [13]. While focusing on the learning model, ensemble methods are a viable solution when working with imbalanced regression problems [5]. Instead of focusing on the data sampling or the model, imbalanced regression estimates can be improved by adapting the MSE (mean squared error) loss [31]. Most similar to us are [32, 47] whose methods can be seen as defining smooth classes over the data. However, they aim to improve regression on imbalanced data, while we set off to analyze in which cases adding a classification loss can help regression, and what is the motivation behind this.

6. Conclusion

Here, we present a preliminary analysis on the effect of adding a classification loss to the regression loss. We make the observation that adding a classification loss to the regression loss has been used in computer vision for deep regression [30, 44, 46]. And we empirically test across data variations and data samplings on a set of 1D functions, the effect of adding a classification loss to a regression loss. We find that for imbalanced regression, adding a classification loss helps the most. Furthermore, we present an attempt at formalizing this observation starting from the derivations of Ren *et al*. [31] for imbalanced regression.

Additionally, we validate that adding a classification loss to the regression loss is beneficial on imbalanced real data, where we evaluate on imbalanced image data on NUYD2-DIR and IMDB-WIKI-DIR datasets [47], and imbalanced video progress prediction on the Breakfast dataset [23].

References

[1] Jonathan T Barron. A general and adaptive robust loss function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4331–4339, 2019. 8

[2] Vasileios Belagiannis, Christian Rupprecht, Gustavo Carneiro, and Nassir Navab. Robust optimization for deep regression. In *Proceedings of the IEEE international conference on computer vision*, pages 2830–2838, 2015. 8

[3] Christopher M Bishop. *Pattern recognition and machine learning*, volume 4. Springer, 2006. 2

[4] Paula Branco, Luís Torgo, and Rita P Ribeiro. Smogn: a pre-processing approach for imbalanced regression. In *First international workshop on learning with imbalanced domains: Theory and applications*, pages 36–50. PMLR, 2017. 8

[5] Paula Branco, Luis Torgo, and Rita P Ribeiro. Rebagg: Resampled bagging for imbalanced regression. In *Second International Workshop on Learning with Imbalanced Domains: Theory and Applications*, pages 67–81. PMLR, 2018. 8

[6] William S Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American statistical association*, 74(368):829–836, 1979. 8

[7] William S Cleveland and Susan J Devlin. Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American statistical association*, 83(403):596–610, 1988. 8

[8] Corinna Cortes, Mehryar Mohri, and Umar Syed. Deep boosting. In *International conference on machine learning*, pages 1179–1187. PMLR, 2014. 8

[9] Ishan Dave, Rohit Gupta, Mamshad Nayeem Rizve, and Mubarak Shah. Tclr: Temporal contrastive learning for video representation. *Computer Vision and Image Understanding*, 219:103406, 2022. 7

[10] Xin Ding, Qiong Zhang, and William J Welch. Classification beats regression: Counting of cells from greyscale microscopic images based on annotation-free training samples. In *CAAI International Conference on Artificial Intelligence*, pages 662–673, 2021. 1

[11] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2002–2011, 2018. 1

[12] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 8

[13] Yu Gong, Greg Mori, and Frederick Tung. Ranksim: Ranking similarity regularization for deep imbalanced regression. *ICML*, 2022. 8

[14] Shizhong Han, Zibo Meng, Ahmed-Shehab Khan, and Yan Tong. Incremental boosting convolutional neural network for facial action unit recognition. *Advances in neural information processing systems*, 29, 2016. 8

[15] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. Statistical learning with sparsity. *Monographs on statistics and applied probability*, 143:143, 2015. 8

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 5

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[18] Le Hou, Dimitris Samaras, Tahsin Kurc, Yi Gao, and Joel Saltz. Convnets with smooth adaptive activation functions for regression. In *Artificial Intelligence and Statistics*, pages 430–439. PMLR, 2017. 8

[19] Junjie Hu, Mete Ozay, Yan Zhang, and Takayuki Okatani. Revisiting single image depth estimation: Toward higher resolution maps with accurate object boundaries. In *2019 IEEE winter conference on applications of computer vision (WACV)*, pages 1043–1051, 2019. 5

[20] Peter J Huber. Robust statistics. In *International encyclopedia of statistical science*, pages 1248–1251. Springer, 2011. 8

[21] Aapo Hyvarinen and Hiroshi Morioka. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. *Advances in neural information processing systems*, 29, 2016. 7

[22] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *Proceedings of the IEEE international conference on computer vision*, pages 66–75, 2017. 1, 8

[23] H. Kuehne, A. B. Arslan, and T. Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of Computer Vision and Pattern Recognition Conference (CVPR)*, 2014. 2, 7, 8

[24] Anna Kukleva, Hilde Kuehne, Fadime Sener, and Jurgen Gall. Unsupervised learning of action classes with continuous temporal embedding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'19)*, 2019. 7

[25] Stéphane Lathuilière, Pablo Mesejo, Xavier Alameda-Pineda, and Radu Horaud. A comprehensive analysis of deep regression. *IEEE transactions on pattern analysis and machine intelligence*, 42(9):2065–2081, 2019. 8

[26] Yu Li, Tao Wang, Bingyi Kang, Sheng Tang, Chunfeng Wang, Jintao Li, and Jiashi Feng. Overcoming classifier imbalance for long-tail object detection with balanced group softmax. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10991–11000, 2020. 4

[27] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 6

[28] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37, 2016. 8

[29] Pascal Mettes, Elise van der Pol, and Cees Snoek. Hyperspherical prototype networks. *Advances in neural information processing systems*, 32, 2019. 8

[30] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7074–7082, 2017. 1, 4, 8

[31] Jiawei Ren, Mingyuan Zhang, Cunjun Yu, and Ziwei Liu. Balanced mse for imbalanced visual regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7926–7935, 2022. 3, 5, 6, 8

[32] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 6, 8

[33] Gregory Rogez, Philippe Weinzaepfel, and Cordelia Schmid. Lcr-net: Localization-classification-regression for human pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3433–3441, 2017. 8

[34] Rasmus Rothe, Radu Timofte, and Luc Van Gool. Dex: Deep expectation of apparent age from a single image. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 10–15, 2015. 1, 8

[35] Michael Steininger, Konstantin Kobs, Padraig Davidson, Anna Krause, and Andreas Hotho. Density-based weighting for imbalanced regression. *Machine Learning*, 110(8):2187–2211, 2021. 8

[36] Hao Su, Charles R Qi, Yangyan Li, and Leonidas J Guibas. Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *Proceedings of the IEEE international conference on computer vision*, pages 2686–2694, 2015. 8

[37] Luís Torgo, Rita P Ribeiro, Bernhard Pfahringer, and Paula Branco. Smote for regression. In *Portuguese conference on artificial intelligence*, pages 378–389. Springer, 2013. 8

[38] Shubham Tulsiani and Jitendra Malik. Viewpoints and keypoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1510–1519, 2015. 1, 8

[39] Elad Walach and Lior Wolf. Learning to count with cnn boosting. In *European conference on computer vision*, pages 660–676, 2016. 8

[40] Jacob Walker, Abhinav Gupta, and Martial Hebert. Dense optical flow prediction from a static image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2443–2451, 2015. 1, 8

[41] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE international conference on computer vision*, pages 3551–3558, 2013. 7

[42] Jiangliu Wang, Jianbo Jiao, and Yun-Hui Liu. Self-supervised video representation learning by pace prediction. In *European conference on computer vision*, pages 504–521. Springer, 2020. 7

[43] Xiaolong Wang, David Fouhey, and Abhinav Gupta. Designing deep networks for surface normal estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 539–547, 2015. 8

[44] Scott Workman, Menghua Zhai, and Nathan Jacobs. Horizon lines in the wild. *BMVC*, 2016. 1, 4, 8

[45] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio Savarese. Subcategory-aware convolutional neural networks for object proposals and detection. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 924–933. IEEE, 2017. 8

[46] Yang Xiao, Xuchong Qiu, Pierre-Alain Langlois, Mathieu Aubry, and Renaud Marlet. Pose from shape: Deep pose estimation for arbitrary 3d objects. *CoRR*, 2019. 1, 4, 8

[47] Yuzhe Yang, Kaiwen Zha, Yingcong Chen, Hao Wang, and Dina Katabi. Delving into deep imbalanced regression. In *International Conference on Machine Learning*, pages 11842–11851. PMLR, 2021. 2, 5, 6, 8

[48] Le Zhang, Zenglin Shi, Ming-Ming Cheng, Yun Liu, Jia-Wang Bian, Joey Tianyi Zhou, Guoyan Zheng, and Zeng Zeng. Nonlinear regression via deep negative correlation learning. *IEEE transactions on pattern analysis and machine intelligence*, 43(3):982–998, 2019. 8

[49] Yifan Zhang, Bingyi Kang, Bryan Hooi, Shuicheng Yan, and Jiashi Feng. Deep long-tailed learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 4

[50] Chenchen Zhao, Yeqiang Qian, and Ming Yang. Monocular pedestrian orientation estimation based on deep 2d-3d feedforward. *Pattern Recognition*, 100:107182, 2020. 8

[51] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *CoRR*, 2019. 1, 4, 8