Companion Classification Losses for Regression Problems

Aitor Sánchez-Ferrera and Jose R. Dorronsoro Ibero

January 30, 2025

Companion Classification Losses for Regression Problems

Aitor Sánchez-Ferrera[1][0000−0001−6127−0686] and Jose R. Dorronsoro[2,3][0000−0002−5271−0616]

[1] Intelligent Systems Group, Department of Computer Science and Artificial Intelligence, University of the Basque Country UPV/EHU
[2] Department of Computer Engineering, Universidad Autónoma de Madrid, Madrid, Spain
[3] Inst. Ing. Conocimiento, Universidad Autónoma de Madrid, Madrid, Spain

Abstract. By their very nature, regression problems can be transformed into classification problems by discretizing their target variable. Within this perspective, in this work we investigate the possibility of improving the performance of deep machine learning models in regression scenarios through a training strategy that combines different classification and regression objectives. In particular, we train deep neural networks using the mean squared error along with categorical cross-entropy and the novel Fisher loss as companion losses. Finally, we will compare experimentally the results of these companion loss methods with the ones obtained using the standard mean squared loss.

Keywords: deep neural networks · companion losses · mean squared error · categorical crossentropy · Fisher loss · representation learning

1 Introduction

Learning the underlying structure of data is crucial to achieve good results in regression and classification tasks using machine learning methods. In this context, several studies have been conducted in the field of *representation learning*, which aims to learn good representations of data to enhance the performance of deep learning methods as supervised predictors [2]. Moreover, the advent of modern and flexible DNN environments such as Keras [6] and its Tensorflow [1] backend or PyTorch [14] has enabled the design of more complex architectures that facilitate the undertaking of supervised tasks. In this sense, the capability of creating DNNs with diverse architectures has led researchers to experiment with models which are trained by learning different tasks in parallel while they use a shared data representation. The assumption behind this methodology is close to the standard one in multitask learning, i.e., that what is learned for each task can help learning others [3].

Building upon this notion, [9] proposed the application of companion losses to improve the performance of classification models. In particular, they proved that the use of training objectives that combined different classification losses (categorical cross-entropy, the Hinge loss, and a novel Fisher loss) can be beneficial in tackling classification problems. Encouraged by these promising results, [10] used companion losses to train DNN models to tackle tasks related to ordinal regression, where the class labels contain ranking information about the underlying samples [7]. In this case, they considered the cross-entropy loss, the mean-squared error, and the Fisher loss. Note that the mean-squared error is a loss function related to regression problems, so this last paper mixes classification and regression losses in its experiments. Nevertheless, there are other works that propose multi-task learning mixing classification and regression objectives in artificial neural networks. For instance, [5] combined the cross-entropy and the mean squared error for age prediction. In addition, a deep multi-task multi-channel learning framework for disease classification and clinical score regression was proposed by [12]. Finally, [13] trained their models to perform regression and classification jointly to model Alzheimer's disease diagnosis. Besides these works, other contributions extend the combination of classification and regression training objectives to random forests. Concretely, some contributions proposed to train random forests by means of a joint objective function of classification and regression for computer vision tasks, such as multi-object segmentation [11], object detection [16] and pose estimation [15].

This work follows this line of research to improve the performance of deep learning models in regression. Specifically, we propose to combine the mean-squared error with the categorical cross-entropy and the novel Fisher loss to reduce the errors of the predictions of DNNs in regression scenarios. However, unlike in the related works we have mentioned earlier, the regression datasets we consider in our experimentation only consist of a single target variable. Thus, we will have to discretize our original regression datasets into classification problems to train our models jointly for both classification and regression. More precisely, our contributions here are:

– The proposal of deep neural networks for regression that combine classification and regression losses during their training.
– A series of experiments comparing the performance of our companion models against those using only the mean-squared error during its training.
– A statistical analysis of the results obtained by the models used in the experiments.

The remaining of this paper is structured as follows. In Section 2 we provide an overview of the losses we consider for our companion models and present the formulation of the combined objectives. Section 3 describes the details of the experiments performed and, finally, Section 4 presents the conclusions of this work and the lines of research for further development of this methodology.

2 Companion Losses for Regression

We work with DNN architectures that have a sample x as input and whose output is denoted by $F(x, \mathcal{W})$, where $\mathcal{W}$ represents the parameters of the model, i.e., the set of weight matrices and bias vectors. We denote the ground truth target values as y, which are continuous in regression problems and categorical in classification problems. Moreover, the output of the last hidden layer of the networks is computed as $z = \Phi(x, \widetilde{\mathcal{W}})$, where $\widetilde{W}$ are the weights and biases of the model up to the last hidden layer. Thus, the output of regression networks using a linear activation function is computed as $\hat{y} = Wz + B$, where W and B are the weights and bias, respectively, of the output layer. Given this, a neural network used to solve regression problems can be trained using the mean squared error loss function by minimizing

$$\ell_{mse}(\mathcal{W}) = \frac{1}{2N} \sum_{n=1}^{N} (y^n - W\Phi(x^n, \widetilde{\mathcal{W}}) - B)^2, \tag{1}$$

where N is the total number of patterns in the training set.

In classification DNNs, unlike in regression networks, the output layer generally uses the Softmax activation function, which converts the output into a probability distribution over K possible classes. This enforces the output values to verify that $\sum_k F_k(x; \mathcal{W}) = 1$ and we assume that $P(k|x) \approx F_k(x; \mathcal{W})$. Then, we can train a DNN for classification tasks using the categorical cross-entropy, that is, minimizing the negative log likelihood:

$$\ell_{ce}(\mathcal{W}) = -\sum_{n=1}^{N} \sum_{k=0}^{K-1} y_k^n \log F_k(x^n; \mathcal{W}), \tag{2}$$

where K is the number of classes considered in the classification problem and y is a one-hot encoded vector containing the ground truth classes of the patterns in the training set. Once the model is trained, the class of an instance is determined according to $\arg\max_k F_k(x; \mathcal{W}^*)$, where $\mathcal{W}^*$ is the set of optimal weights.

Finally, we also consider a novel Fisher loss for DNNs. In the linear case, if we denote the between-class and total covariance matrices of the sample patterns as S_B and S_T respectively, together with their respective counterparts s_B and s_T for the projections $z = Ax$, the Fisher criterion maximizes the trace criterion $g(A) = \text{trace}(s_T^{-1} s_B) = \text{trace}((A^T S_T A)^{-1}(A^T S_B A))$. According to [18], this can be achieved by solving the following least squares problem

$$\min \frac{1}{2} ||y^f - XW - \mathbf{1}_N B||^2, \tag{3}$$

where W is a $d \times K$ matrix of weights, d is the dimension of the inputs, B is a $1 \times K$ bias vector and $\mathbf{1}_N$ is an all-ones vector. In addition, y^f is the Fisher target matrix, where we have $y^f_{nk} = \frac{N - N_k}{N\sqrt{N_k}}$ when the n-th row is associated with a pattern x^n corresponding to class k, and $y^f_{nm} = -\frac{\sqrt{N_m}}{N}$ for $m \neq k$, where N_k is the number of instances belonging to class k. Building on this, [8] proposed to extend the use of the Fisher loss to deep neural networks by formulating it as

$$\ell_f(y_f, \hat{y}_f) = \frac{1}{2} ||y^f - \Phi(X; \widetilde{\mathcal{W}})W - \mathbf{1}_N B||^2, \tag{4}$$

where $\hat{y}_f$ denotes the output of the network. In this way, the Fisher loss forces the last hidden layer of the model to produce a projection of data that maximizes the Fisher criterion and, hence, concentrates the patterns belonging to the same class while pushing aside the ones from different classes.

As mentioned earlier, the goal of this work is to investigate whether the use of companion classification losses improves the performance of DNN in regression scenarios. Therefore, we will train our models using the mean squared error as the main loss of our training procedure, while using the categorical cross-entropy and the Fisher loss as auxiliary losses. As an example, following the formulation suggested in [9,10], a model that combines the mean squared error and the Fisher loss requires multiple outputs and targets to minimize the following companion loss:

$$\ell(y, y_f, \hat{y}, \hat{y}_f) = \ell_{mse}(y, \hat{y}) + \lambda \ell_f(y_f, \hat{y}_f), \tag{5}$$

where $\hat{y}$ and $\hat{y}_f$ are the predictions for minimizing the MSE and the Fisher loss, respectively, and λ is a hyperparameter that must be appropriately chosen and represents the weight of the Fisher companion loss in the general training objective of the model. Note that the Fisher loss in this case is only used to learn a good representation of data that may improve the performance of the DNN in regression problems, i.e., minimizing the scoring function of the model when solving a regression problem. Finally, and although our companion models will have several output values, we will consider only as the predictions of the model the single outputs corresponding to the mean squared error predictor.

3 Experimental Results

In this section we describe the design of the proposed models, report the datasets we will use and the strategies to transform them into classification problems, present the proposed experimental methodology, and describe the results along with a final brief discussion in this regard.

3.1 Models Proposed

We will consider three different models based on different combinations of the loss functions described in Section 2. Specifically, the configurations of the models proposed are the following:

- `mse`: the model uses a single linear output and the mean squared error loss function. The predictions are equal to the unique output of the model.
- `mse_ce`: the model uses two different outputs. The first one is linear and it is used to minimize the mean squared error loss, while the second one corresponds to the result of the softmax activation function and it minimizes the categorical cross-entropy loss. The predictions are again equal to the linear output.

Table 1. Training and (when available) test patterns and number of features.

	n. patterns train	n. patterns test	dimension
abalone	4177	-	8
bodyfat	252	-	14
cadata	20640	-	8
cpusmall	8192	-	12
housing	506	-	13
mg	1385	-	6
mpg	392	-	7
sotavento	17544	8760	200
space_ga	3107	-	6

Table 2. Discretization method used for the discretization of the original regression problems and number of classes considered.

	discretization method	n. classes considered
abalone	automatic	$K = 2, K = 3, K = 4, K = 5$
bodyfat	automatic	$K = 2, K = 3, K = 4, K = 5$
cadata	automatic	$K = 2, K = 3, K = 4, K = 5$
cpusmall	automatic	$K = 2, K = 3, K = 4, K = 5$
housing	manual	$K = 4, K = 5$
mg	automatic	$K = 2, K = 3, K = 4, K = 5$
mpg	automatic	$K = 2, K = 3, K = 4, K = 5$
sotavento	manual	$K = 3, K = 4$
space_ga	automatic	$K = 2, K = 3, K = 4, K = 5$

– `mse_fisher`: the model uses two linear outputs, the first one to minimize the mean squared error and the second one corresponding to the minimization of the Fisher loss. The predictions are once more equal to the first output of the model.

3.2 Datasets and their Discretizations

We will be using nine different datasets, concretely `abalone, bodyfat, cadata, cpusmall, housing, mg, mpg, sotavento` and `space_ga`. All have been taken from the LIBSVM data repository [4], except for Sotavento, which was provided by the UAM–ADIC Chair for Data Science and Machine Learning. Table 1 shows the statistics of the datasets, including their training and, when available, test sample sizes and dimensions.

As described before, we will use the categorical cross-entropy and the Fisher loss as companion losses in our proposed companion models. Since these two functions correspond to classification contexts, we need to transform the original regression problems into classification ones. In our case, we will perform this transformation by discretizing the target variables of our datasets. The goal of this procedure is to group the instances into different classes based on their target values. Note that we can define multiple classification problems from the same regression problem, depending on the number of artificial classes we want to consider. Thus, the resulting number of classes K of the generated classification problems can be considered a hyperparameter that must be adjusted during the experimentation phase of this paper.

1. The manual definition of the classes according to the context of the target and,
2. The automatic definition of the classes according to a statistical criterion.

In the first method, the classes of the instances are set by analyzing the information (e.g., range and context) of the target variables. This has been the procedure to discretize the `housing` and `sotavento` datasets. More precisely, the target values in the `housing` dataset describe house prices and these are delimited withing the range $[5, 50]$, so we have considered two discretizations based on using $K = 4$ and $K = 5$ classes and establishing the following categories:

– Categories when $K = 4$: "very low" ($[5, 20)$), "low" ($[20, 30)$), "high" ($[30, 40)$), and "very high" ($[40, 50]$).
– Categories when $K = 5$: "very low" ($[5, 10)$), "low" ($[10, 20)$), "normal" ($[20, 30)$), "high" ($[30, 40)$) and "very high" ($[40, 50]$).

Similarly, the target values in `sotavento` represent the production of the electriticy generated by the Sotavento wind farm normalized to the range $[0, 1]$. Thus, in this case we have have considered two discretizations based on $K = 3$ and $K = 4$ classes establishing the following categories:

– Categories when $K = 3$: "low" ($[0, 0.3)$), "normal" ($[0.3, 0.6)$) and "high" ($[0.6, 1]$).
– Categories when $K = 4$: "very low" ($[0, 0.2)$), "low" ($[0.2, 0.4)$), "normal" ($[0.4, 0.6)$) and "high" ($[0.6, 1]$).

In the second method, we set the classes of the instances by dividing their ranges according to the percentiles of the target variables. It is clear that the manual approach to transform the original regression problems into classification problems is more expensive, since special knowledge about the problem must be used. Table 2 describes the method used for each transformation and the number of classes considered in the generated classification problems.

3.3 Experimental Methodology

The models considered for the experimentation include an L_2 regularization term that involves the use of an α hyperparameter whose optimal value must be found. Moreover, the models that use multiple losses have an additional hyperparameter λ that represents the weight of the companion losses in their training objectives. Finally, as stated in Section 3.2, we propose different discretizations of the original regression problems for each of the considered datasets, which entails another additional hyperparameter K representing the number of classes of the generated classification problems in the case of the companion models.

All in all, we have up to three hyperparameters to fix in our experiments, which are adjusted using a 5-fold cross-validation (CV) grid search. In particular, the α values will be of the form 10^k, with k in the range $-5 \leq k \leq 2$. Similarly, we consider λ values within the range $[0, 1]$, as we restrict the companion losses not to have a higher weight than the main mean-squared error loss. Thus, to find the optimal λ, we will examine values of the form $0.05 * k$, with k now in the range $0 \leq k \leq 20$. Finally, we find the optimal K among the discretizations presented in Table 2. In this way, we will find the optimal values α^*, λ^* and K^* for each of the hyperparameters of the models considered.

With respect to the evaluation of the models, we will test their performance by the following procedure:

1. We generate 5-fold cross-validated predictions estimates five times for each of the input data points using different seeds from the ones we use during hyperparametrization.
2. We compute the averages of these five predictions.
3. We finally compute the mean absolute error between these average predictions and the ground truth values of each problem

Remember that in the `Sotavento` dataset we have different data samples for training and test, namely, the years 2016 and 2017 for train and validation, and 2018 for test. In that case, we will use a 2-fold CV grid search procedure jointly on data from 2016 and 2017 to find the optimal α^*, λ^* and K^* hyperparameters. Once set these optimal values, we will fit the models five times in the training set using different seeds for weight initialization and minibatch handling, generate their predictions per each data data point in the test set and compute the mean absolute error among their averages and the ground truth values.

3.4 Results and Discussion

We analyze now the results for each of the datasets considered. Note that we define three different architectures for our artificial neural networks based on the use of one/two (based on the complexity of the problem, one for `cadata`, `cpusmall`, `housing`, `mg`, `mpg`, and `space_ga`, while two for `abalone`, `bodyfat`, and `sotavento`), three and five hidden layers. Tables 3, 4 and 5 show the MAE of the predictions of the optimal models for the one/two, three and five hidden-layer architectures, respectively. The last column of each table contains the p-values obtained by the Wilcoxon signed rank test [17] when comparing the absolute errors of the `mse` model and the ones generated by the companion model that obtained the lowest errors among the `mse_ce` and `mse_fisher` models. Finally, smaller values are highlighted in boldface when the absolute error distributions of the `mse` and the best companion model are statistically different at the $p = 0.10$ level or below.

Table 3. Mean absolute errors and p values for `mse`, `mse_ce` and `mse_fisher` models with one-two (depending on the dataset) hidden layers.

	`mse`	`mse_ce`	`mse_fisher`	p-value
abalone	**1.458**	1.463	1.466	**0.05**
bodyfat ($\times 10^2$)	0.107	0.107	0.107	1.00
cadata ($/10^4$)	**3.507**	3.519	3.525	**0.05**
cpusmall	2.023	2.023	2.044	1.00
housing	2.065	2.065	2.061	0.80
mg ($\times 10^2$)	8.981	**8.911**	8.996	**0.03**
mpg	1.869	1.891	1.870	0.80
sotavento ($\times 10^2$)	6.616	6.678	**6.541**	**0.00**
space_ga ($\times 10^2$)	7.442	7.442	7.462	1.00

Table 4. Mean absolute errors and p values for `mse`, `mse_ce` and `mse_fisher` models with three hidden layers.

	`mse`	`mse_ce`	`mse_fisher`	p-value
abalone	1.463	1.466	1.460	0.69
bodyfat ($\times 10^2$)	0.113	0.113	0.133	1.00
cadata ($/10^4$)	3.343	3.354	**3.239**	**0.00**
cpusmall	2.025	2.025	2.029	1.00
housing	2.085	2.056	**1.978**	**0.01**
mg ($\times 10^2$)	9.197	9.100	9.120	0.40
mpg	1.899	1.899	1.908	1.00
sotavento ($\times 10^2$)	6.698	**6.639**	6.842	**0.00**
space_ga ($\times 10^2$)	7.044	7.107	7.117	0.40

Table 5. Mean absolute errors and p values for `mse`, `mse_ce` and `mse_fisher` models with five hidden layers.

	`mse`	`mse_ce`	`mse_fisher`	p-value
abalone	1.462	1.467	1.457	0.18
bodyfat ($\times 10^2$)	0.122	0.122	0.120	0.59
cadata ($/10^4$)	3.264	**3.165**	3.225	**0.00**
cpusmall	2.017	2.017	2.050	1.00
housing	2.080	2.080	**1.970**	**0.06**
mg ($\times 10^2$)	9.281	**9.155**	9.172	**0.07**
mpg	1.916	1.957	1.986	0.21
sotavento ($\times 10^2$)	6.786	**6.528**	6.761	**0.00**
space_ga ($\times 10^2$)	**6.978**	7.169	7.149	**0.02**

Among the models with one and two hidden layers, we can reject the null hypotheses at level $p = 0.05$ level in favor of the `mse` network when comparing it to our companion models in the `abalone` and `cadata` datasets. However, our companion models outperform the mean-squared error-based model in the `mg` (in the case of `mse_ce`) and `sotavento` (in the case of `mse_fisher`) regression problems. Regarding the errors given by the models that use three hidden lay-

Table 6. Mean absolute errors and p values for `mse`, `mse_ce` and `mse_fisher` models using the optimal architecture for the `mse` network. Smaller values in bold face when absolute error distributions of the mse and the best companion model are statistically different.

	n_hidden_layers	`mse`	`mse_ce`	`mse_fisher`	p-value
abalone	2	**1.458**	1.463	1.466	**0.05**
bodyfat ($\times 10^2$)	2	0.107	0.107	0.107	1.00
cadata ($/10^4$)	5	3.264	**3.165**	3.225	**0.00**
cpusmall	5	2.017	2.017	2.05	1.00
housing	2	2.065	2.065	2.061	0.80
mg ($\times 10^2$)	1	8.981	**8.911**	8.996	**0.03**
mpg	1	1.869	1.891	1.870	0.80
sotavento ($\times 10^2$)	5	6.786	**6.528**	6.761	**0.00**
space_ga ($\times 10^2$)	5	**6.987**	7.169	7.149	**0.02**

ers, with a 1% significance level, the `mse_fisher` is better than the `mse` when tackling the `cadata` and `housing` problems. Moreover, the `mse_ce` outperforms the `sotavento` problem with a 0% significance. Finally, when using five hidden layers, `mse` produces lower errors in the `space_ga` dataset with a 2% significance, while the `mse_ce` network outperforms the `cadata` and `sotavento` problems with p values below 0.005. Furthermore, we can reject the null hypothesis in favor for the `mse_fisher` and `mse_ce` models against the `mse` in `housing` and `mg`, respectively, but now with a 10% significance. Model differences do not appear to be significative in all other cases not mentioned above.

Note that there are some cases where the errors generated by the `mse` model and our companion networks are exactly the same. This is because the CV procedure found $\lambda = 0$ as the optimal hyperparameter value in our companion models. As a result, since λ represents the weight of the companion loss of our combined models, when $\lambda = 0$ we obtain models that perform exactly the same as the `mse` network. An example of this can be seen in Table 3, where the three models get the same errors for the `bodyfat` problem. Finally, notice that the `mse` model can still produce lower errors than a companion model even when the training CV procedure sets $\lambda > 0$, since the latter may have a lower generalization ability. All in all, the models trained using only the mean-squared error perform similar to the companion models when using few hidden layers. Nevertheless, the use of deeper architectures led to a slightly better performance of the networks that combine regression and classification objectives to address regression problems.

Finally, for a more concise discussion, Table 6 depicts a brief summary of the experiments that we have carried out in this work. Specifically, to make an objective comparison among the models considered, we have gathered the results of the experiments in which the `mse` achieved its lowest errors; in other words, we will consider the best `mse` architectures against the competing `mse_ce` and `mse_fisher` models with the same architecture, giving thus a slight edge to the mean squared error based models. In this regard, the `mse` network outperforms the companion models rejecting the null hypothesis of the Wilcoxon signed rank test in two out of nine experiments. Conversely, the `mse_ce` model shows a better performance in three. Finally, in the remaining four experiments the null hypotheses cannot be rejected since $p > 0.10$, so we conclude that the corresponding models give similar results.

4 Conclusions and Further Work

In this work we have proposed the use of classification losses (i.e., the categorical cross-entropy and the novel Fisher loss) as companion objectives for neural networks to tackle regression problems. Our motivation for this is that previous studies showed that the combination of classification and regression training objectives improved the performance of machine learning models in different tasks in the field of supervised learning.

Overall, we have carried out 27 experiments (taking into account that we have used three losses for each of the datasets considered) in which we have compared the performance of neural networks trained using only the mean-squared error against models that use companion classification losses during their training. In a brief summary, the `mse` model outperformed in three out of 27 experiments, while the companion models got better results in nine out of 27. Finally, in the rest of experiments the results were similar, as the error distributions were not significantly different. This allows us to conclude that using companion classification losses for regression problems may be useful depending on the problems that we want to tackle. Also, we have applied manual (in the case of `housing` and `sotavento` datasets) and automatic methods (in the remaining problems) for discretizing regression problems in order to employ our companion models. In this regard, the results show that both strategies can be useful for implementing the use of companion losses in deep neural networks.

As lines for future work, we propose to extend this methodology to other companion classification losses. In this work we have only considered the categorical cross-entropy and the Fisher loss. However, there are other functions, such as the Hinge loss, that may lead to an improvement of the representations learned for regression problems. Additionally, it would also be interesting to explore the design of a criterion to find optimal discretizations that improve our companion models. Another point of interest would be to extend the search space of the λ hyperparameter, which represents the weight of the companion losses in the general training objective of our models. In this work we have limited its values within the range $[0, 1]$, but we think that letting this hyperparameter to take values bigger than 1 may be benefitial for our companion models, since there were some experiments in which the optimal λ value was found to be $\lambda = 1.0$, hinting that possibly better results could be obtained using larger λ ranges.

Apart from that, we point out that in this work we have used the mean absolute error between the regression targets and outputs as the scoring function in order to choose the optimal hyperparameters of our models; in other words, when choosing these optimal hyperparameters we have not considered the performance of the classification component of our companion models. This suggests that, alternatively, it would be valuable to try to use other scorings that mix a regression criterion with a measure that takes into account the classification performance of our companion models, as this may lead to better results. We are currently pursuing these lines of work as well as other related venues in order to improve the use of companion models for regression.

Acknowledgements The authors acknowledge financial support from the European Regional Development Fund and the Spanish State Research Agency of the Ministry of Economy, Industry, and Competitiveness under the project PID2019-106827GB-I00. They also thank the support of the UAM–ADIC Chair for Data Science and Machine Learning and gratefully acknowledge the use of the facilities of Centro de Computación Científica (CCC) at UAM. They finally acknowledge the financial support of the Department of Education of the Basque Government under the grant PRE_2022_1_0103.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015), https://www.tensorflow.org/, software available from tensorflow.org
2. Bengio, Y., Courville, A., Vincent, P.: Representation learning: A review and new perspectives. IEEE transactions on pattern analysis and machine intelligence **35**(8), 1798–1828 (2013)
3. Caruana, R.: Multitask learning. Machine Learning **28**, 41–75 (1997)
4. Chang, C.C., Lin, C.J.: Libsvm: A library for support vector machines. ACM transactions on intelligent systems and technology (TIST) **2**(3), 1–27 (2011)
5. Chen, J., Cheng, L., Yang, X., Liang, J., Quan, B., Li, S.: Joint learning with both classification and regression models for age prediction. In: Journal of Physics: Conference Series. vol. 1168, p. 032016. IOP Publishing (2019)
6. Chollet, F., et al.: Keras (2015), https://github.com/fchollet/keras
7. Christensen, R.H.B.: ordinal—regression models for ordinal data. R package version **28**, 2015 (2015)
8. Diaz-Vico, D., Dorronsoro, J.R.: Deep least squares fisher discriminant analysis. IEEE transactions on neural networks and learning systems **31**(8), 2752–2763 (2019)
9. Díaz-Vico, D., Fernández, A., Dorronsoro, J.R.: Companion losses for deep neural networks. In: Hybrid Artificial Intelligent Systems: 16th International Conference, HAIS 2021, Bilbao, Spain, September 22–24, 2021, Proceedings. pp. 538–549. Springer (2021)
10. Díaz-Vico, D., Fernández, A., Dorronsoro, J.R.: Companion losses for ordinal regression. In: Hybrid Artificial Intelligent Systems: 17th International Conference,

HAIS 2022, Salamanca, Spain, September 5–7, 2022, Proceedings. pp. 211–222. Springer (2022)

11. Glocker, B., Pauly, O., Konukoglu, E., Criminisi, A.: Joint classification-regression forests for spatially structured multi-object segmentation. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) Computer Vision – ECCV 2012. pp. 870–881. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
12. Liu, M., Zhang, J., Adeli, E., Shen, D.: Deep multi-task multi-channel learning for joint classification and regression of brain status. In: Medical Image Computing and Computer Assisted Intervention- MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part III. pp. 3–11. Springer (2017)
13. Liu, M., Zhang, J., Adeli, E., Shen, D.: Joint classification and regression via deep multi-task multi-channel learning for alzheimer's disease diagnosis. IEEE Transactions on Biomedical Engineering **66**(5), 1195–1206 (2018)
14. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: Advances in Neural Information Processing Systems 32. pp. 8024—-8035. Curran Associates, Inc. (2019)
15. Schulter, S., Leistner, C., Wohlhart, P., Roth, P.M., Bischof, H.: Alternating regression forests for object detection and pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (December 2013)
16. Schulter, S., Leistner, C., Wohlhart, P., Roth, P.M., Bischof, H.: Accurate object detection with joint classification-regression random forests. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2014)
17. Wilcoxon, F.: Individual comparisons by ranking methods. Biometrics Bulletin **1**(6), 80–83 (1945), http://www.jstor.org/stable/3001968
18. Zhang, Z., Dai, G., Xu, C., Jordan, M.I.: Regularized discriminant analysis, ridge regression and beyond. The Journal of Machine Learning Research **11**, 2199–2228 (2010)