We thank all the reviewers for their valuable suggestions and feedback. We kindly appeal to all the reviewers that they will reconsider and improve their scores, because as shown again in the new results provided in this rebuttal our method achieved the best performance in comparison to all the benchmarks across the real-world datasets. We believe our work presents a valuable and general regularization method for supervised learning models.

Table A: Additional Experiments (real data).

Data	Mixup	Mixup-Man
Regression (MSE)		
BH	0.134 ± 0.019	0.130 ± 0.023
WQ	0.717 ± 0.030	0.712 ± 0.028
FB	0.329 ± 0.184	0.387 ± 0.316
BC	0.339 ± 0.025	0.325 ± 0.039
SP	0.208 ± 0.040	0.282 ± 0.023
CM	0.333 ± 0.025	0.386 ± 0.031
Classification (AUROC)		
CC	0.763 ± 0.008	0.772 ± 0.007
PD	0.814 ± 0.018	0.808 ± 0.009
BC	0.719 ± 0.020	0.728 ± 0.013
LV	0.586 ± 0.041	0.571 ± 0.025
SH	0.904 ± 0.015	0.916 ± 0.011
RP	0.798 ± 0.016	0.804 ± 0.018

[[Reviewer 1]] ■ **Strength of empirical results:** We would like to point out that the benefit of our method is not only (the magnitude of) the improved performance; we also show that our method is consistently providing the best regularization (across every dataset - both synthetic and real-data). For example, in the encircled portion of Figure A in this response, our method consistently achieved the best MSE in comparison to all the benchmarks across the real-world datasets. This consistency is not seen in the other benchmark regularizers, which exhibit higher variance in their average rank across all the datasets. This notion is also conveyed in the synthetic experiments in Figure 3 and for real-data (classification and regression) in Figure 5 in the original manuscript. ■ **Experiments with Mixup:** We appreciate the recommendation for additional benchmarks. We have conducted the recommended experiments for Mixup and Manifold Mixup and show that CASTLE still outperforms the other benchmarks as shown in Table A (please compare with CASTLE in Table 3 in the original manuscript) and Figure A for the real datasets.

[[Reviewer 2]] ■ **Notation clarification:** You are assuming correctly - $V(W)$ is the ℓ_1 norm in our methodology. We will clarify this in the revised manuscript. ■ **Generalization bounds:** The primary goal of our method is improving out-of-sample prediction performance which we use a generalization bound as justification. We did not prove the consistency of using a reconstruction loss and a norm-based regularizer in DAG learning which has already been proven in [49] and [50], respectively. ■ **CNN Limitations:** We agree with you and will clarify the limitations of our method as we did for CNNs in the Broader Impact statement. ■ **Additional results:** We reinforce the superiority of our proposed method by providing additional results in Table A and Figure A in the response to Reviewer 1.

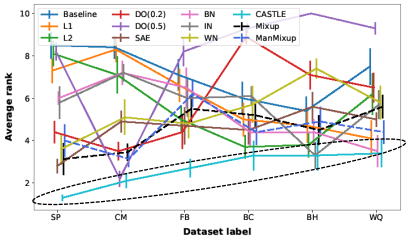

Figure A: Comparison in terms of average rank (in terms of MSE - lower is better). CASTLE has the best and most stable performance across all datasets.

[[Reviewer 3]] ■ **Experimental hyperparameters:** As mentioned in lines 255-256, we performed a grid-search over a wide range of hyperparameters. We believe that this is fairly done for each benchmark, as we conducted the same grid-search for our model that we did for the other benchmark methods (lines 257-258) and applied early stopping for each. The performance gain from our regularizer is not due to improper hyperparameter tuning. ■ **L2 missing:** Table 2 contains L2 regularization. ■ **Definition of $\mathbf{W}$:** We are not defining $\mathbf{W}$ twice. The adjacency matrix can be represented by a matrix containing negative values - see NOTEARS [39] and Non-parametric DAGs [40]. Because of this, we can embed the adjacency matrix in the input layers, $\mathbf{W}$'s, of the proposed neural network (Section 3.3). ■ **Prediction using causal parents:** Each feature is constructed using every other feature based on the DAG structure embedded in the neural network input layers. When a DAG is learned, the parent features (non-zero weights $\mathbf{W}$) are obligated to construct each child (see Figure 2 and lines 166-168). ■ **Variable u:** In Def. 1, each variable u_i is specific for a feature X_i in the DAG, they are not hidden confounders, but the random noise to generate the feature X_i. ■ **Causal neighbors:** Consider the case where a variable is just noise, and therefore does not have any causally adjacent nodes (neighbors). Reconstruction methods, such as SAE, naively (and inefficiently) learns to reconstruct noise variables that have no causal implications on the target variable. Through DAG learning, our method does not reconstruct these variables as the input weight matrices get forced to zero (see Figures 3, 6, and 7). ■ **Sibling variables:** We mean the function generating the sibling variables may share the some similarities. We will elaborate this point with concrete examples in the revised submission. ■ **$\mathbf{X}$ vs $\boldsymbol{X}$:** We define the random variables as $\boldsymbol{X}$ and the corresponding data matrix as $\mathbf{X}$. This is standard notation in machine learning. ■ **Regularization terms:** The description of the regularization terms is given in lines 162-168. We will describe them in more detail in the revised paper. ■ **Layer size:** We provided the layer size in Section 4 on line 264. ■ **Acyclicity constraint:** Starting on line 165, we introduce and describe Theorem 1 from [39]. We describe Theorem 1 by saying, "the graph given by $\mathbf{W}$ is a DAG if and only if $R_{\mathbf{W}} = 0$." ■ **Writing quality:** Although the other reviewers have positively acknowledged our exposition, we will work to improve the writing quality.

[[Reviewer 4]] ■ **Scalability:** For a typical dataset with hundreds of features or less, the computational training time does not differ significantly between the regularizers. For example, on simulated data, an experimental run with 200 features, 2000 samples, and 200 epochs had an average training time of ∼55s and ∼64s for SAE and CASTLE, respectively, on an Intel i7-6850K CPU at 3.60GHz. We will incorporate a computational complexity analysis of our method as well as a demonstration of the computational trade-offs between our method and improved performance in the revised manuscript. ■ **Notations:** We will correct the suggested typos in the revision.