TANGOS: REGULARIZING TABULAR NEURAL NETWORKS THROUGH GRADIENT ORTHOGONALIZATION AND SPECIALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite their success with unstructured data, deep neural networks are not yet a panacea for structured tabular data. In the tabular domain, their efficiency crucially relies on various forms of regularization to prevent overfitting and provide strong generalization performance. Existing regularization techniques include broad modelling decisions such as choice of architecture, loss functions, and optimization methods. In this work, we introduce Tabular Neural Gradient Orthogonalization and Specialization (TANGOS), a novel framework for regularization in the tabular setting built on latent unit attributions. The gradient attribution of an activation with respect to a given input feature suggests how the neuron *attends* to that feature, and is often employed to interpret the predictions of deep networks. In TANGOS, we take a different approach and incorporate neuron attributions directly into training to encourage orthogonalization and specialization of *latent attributions* in a fully-connected network. Our regularizer encourages neurons to focus on sparse, non-overlapping input features and results in a set of diverse and specialized latent units. In the tabular domain, we demonstrate that our approach can lead to improved out-of-sample generalization performance, outperforming other popular regularization methods. We provide insight into *why* our regularizer is effective and demonstrate that TANGOS can be applied jointly with existing methods to achieve even greater generalization performance.

1 INTRODUCTION

Despite its relative under-representation in deep learning research, tabular data is ubiquitous in many salient application areas including medicine, finance, climate science, and economics. Additionally, it is a domain in which general purpose regularizers are of particular importance. Unlike areas such as computer vision or natural language processing, architectures for tabular data generally do not exploit the inherent structure in the input features (i.e. locality in images and sequential text, respectively) and lack the resulting inductive biases in their design. Consequentially, improvement over non-neural ensemble methods has been less pervasive. Regularization methods that implicitly or explicitly encode inductive biases thus play a more significant role. Furthermore, adapting successful strategies from the ensemble literature to neural networks may provide a path to success in the tabular domain (e.g. Wen et al., 2020). Recent work in Kadra et al. (2021) has demonstrated that suitable regularization is essential to outperforming such methods and, furthermore, a balanced *cocktail* of regularizers results in neural network superiority.

Regularization methods employed in practice can be categorized into those that prevent overfitting through data augmentation (Krizhevsky et al., 2012; Zhang et al., 2018), network architecture choices (Hinton et al., 2012; Ioffe & Szegedy, 2015), and penalty terms that explicitly influence parameter learning (Hoerl & Kennard, 1970; Tibshirani, 1996; Jin et al., 2020), to name just a few. While all such methods are unified in attempting to improve out-of-sample generalization, this is often achieved in vastly different ways. For example, $L1$ and $L2$ penalties favor sparsity and shrinkage, respectively, on model weights, thus choosing more parsimonious solutions. Data perturbation techniques, on the other hand, encourage smoothness in the system assuming that small perturbations in the input should not result in large changes in the output. Which method works best for a given task is generally not known *a priori* and considering different classes of regularizer is recommended in practice.

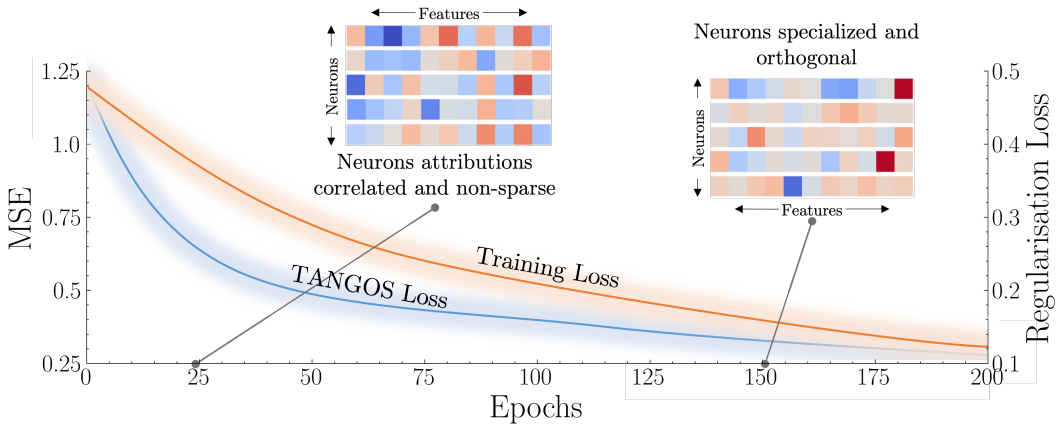


Figure 1: **TANGOS encourages specialization and orthogonalization.** TANGOS penalizes neuron attributions during training. Here, ■ indicates strong positive attribution and ■ indicates strong negative attribution, while interpolating colors reflect weaker attributions. Neurons are regularized to be *specialized* (attend to sparser features) and *orthogonal* (attend to non-overlapping features).

Furthermore, combining multiple forms of regularization simultaneously is often effective, especially in lower data regimes (see e.g. Brigato & Iocchi, 2021 and Hu et al., 2017).

Neuroscience research has suggested that neurons are both *selective* (Johnston & Dark, 1986) and have *limited capacity* (Cowan et al., 2005) in reacting to specific physiological stimuli. Specifically, neurons selectively choose to focus on a few chunks of information in the input stimulus. In deep learning, a similar concept, commonly described as a *receptive field*, is employed in convolutional layers (Luo et al., 2016). Here, each convolutional unit has multiple filters, and each filter is only sensitive to specialized features in a local region. The output of the filter will activate more strongly if the feature is present. This stands in contrast to fully-connected networks, where the all-to-all relationships between neurons mean each unit depends on the entire input to the network. We leverage this insight to propose a regularization method that can encourage artificial neurons to be more specialized and orthogonal to each other.

Contributions. (1) Novel regularization method for deep tabular models. In this work, we propose TANGOS, a novel method based on regularizing neuron attributions. A visual depiction is given in Figure 1. Specifically, each neuron is more *specialized*, attending to sparse input features while its attributions are more *orthogonal* to those of other neurons. In effect, different neurons pay attention to non-overlapping subsets of input features resulting in better generalization performance. We demonstrate that this novel regularization method results in excellent generalization performance on tabular data when compared to other popular regularizers. **(2) Distinct regularization objective.** We explore how TANGOS results in distinct emergent characteristics in the model weights. We further show that its improved performance is linked to increased diversity among weak learners in an ensemble of latent units, which is generally in contrast to existing regularizers. **(3) Combination with other regularizers.** Based upon these insights, we demonstrate that deploying TANGOS *in tandem* with other regularizers can further improve generalization of neural networks in the tabular setting beyond that of any individual regularizer.

2 Related Works

Neural Network Regularization. Regularization methods seek to penalize complexity and impose a form of smoothness on a model. This may be cast as expressing a prior belief over the hypothesis space of a neural network which attempts to aid generalization. ► **Categories.** A vast array of regularization methods have been proposed throughout the literature (for a comprehensive taxonomy see e.g. Kukačka et al., 2017). Modern nomenclature typically includes broad modelling decisions such as choice of architecture, loss function, and optimization method under the umbrella of regularization. Additionally, many regularization techniques augment the training data using methods such as input noise (Krizhevsky et al., 2012) or MixUp (Zhang et al., 2018). Dropout (Hinton et al., 2012) and

related approaches that augment a hidden representation of the input may also be included in this category. Possibly a more conventional category of regularization is that which adds explicit penalty term(s) to the loss function. These terms might penalize the network weights directly to shrink or sparsify their values as in L2 (Hoerl & Kennard, 1970) and L1 (Tibshirani, 1996), respectively. Alternatively, network outputs may be penalized to, for example, reduce overconfidence (Pereyra et al., 2017). ► **Weight Orthogonalization.** A number of works have studied the orthogonalization of network weights via various weight penalization methods (Bansal et al., 2018). More recent work in Liu et al. (2021) proposed to learn an orthogonal transformation of the randomly initialized incoming weights to a given neuron. In contrast, this work seeks to ensure that the gradients of different latent neurons with respect to a given input vector are orthogonal. ► **Combination.** Compositions of multiple regularization methods are extensively applied in practice. An early example in the regression setting is the elastic net penalty (Zou & Hastie, 2005) which attempts to combine sparsity with shrinkage in the coefficients. More recent work has demonstrated the effectiveness of combining several regularization terms on tabular data (Kadra et al., 2021), a domain in which neural networks superiority had previously been less convincing.

Gradient Attribution Regularization. A number of methods exist which incorporate a regularisation term to penalize the network gradients in some way. Penalizing gradient attributions is a natural approach for achieving various desirable properties in a neural network. Such methods have been in use at least since Drucker & Le Cun (1992), where the authors improve robustness by encouraging invariance to small perturbations in the input space. More recently, gradient attribution regularization has been successfully applied across a broad range application areas. Some notable examples include encouraging the learning of robust features in auto-encoders (Rifai et al., 2011), improving stability in the training of generative adversarial networks (Gulrajani et al., 2017), and providing robustness to adversarial perturbations (Moosavi-Dezfooli et al., 2019). While many works have applied a shrinkage penalty (L2) to input gradients, Ross et al. (2017a) explore the effects of encouraging sparsity by considering an L1 penalty term. Gradient penalties may also be leveraged to compel a network to *attend* to particular human-annotated input features (Ross et al., 2017b). A related line of work considers the use of gradient aggregation methods such as Integrated Gradients (Sundararajan et al., 2017) and, typically, penalizes their deviation from a given target value (see e.g. Liu & Avci (2019) and Chen et al. (2019)). In contrast to these works, we do not require manually annotated regions upon which we constrain the network to attend. Similarly, Erion et al. (2021) provide methods for encoding domain knowledge such as smoothness between adjacent pixels in an image. We note that while these works have investigated penalizing a predictive model's output attributions, we are the first to regularize attributions on latent neuron activations.

3 TANGOS

3.1 Problem Formulation

We operate in the standard supervised learning setting, with d_X-dimensional input variables $X \in \mathcal{X} \subseteq \mathbb{R}^{d_X}$ and target output variable $Y \in \mathcal{Y} \subseteq \mathbb{R}$. Let P_{XY} denote the joint distribution between input and target variables. The goal of the supervised learning algorithm is to find a predictive model, $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ with learnable parameters $\theta \in \Theta$. The predictive model belongs to a hypothesis space $f_\theta \in \mathcal{H}$ that can map from the input space to the output space.

The predictive function is usually learned by optimizing a loss function $\mathcal{L} : \Theta \rightarrow \mathbb{R}$ using *empirical risk minimization* (ERM). The empirical risk cannot be directly minimized since the data distribution P_{XY} is not known. Instead, we use a finite number of iid samples $(x, y) \sim P_{XY}$, which we refer to as the training data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$.

Once the predictive model is trained on $\mathcal{D}$, it should ideally predict well on out-of-sample data generated from the same distribution. However, overfitting can occur if the hypothesis space $\mathcal{H}$ is too complex and the sampling of training data does not fully represent the underlying distribution P_{XY}. Regularization is an approach that reduces the complexity of the hypothesis space so that more generalized functions are learned to explain the data. This leads to the following ERM:

$$\theta^* = \min_{\theta \in \Theta} \frac{1}{|\mathcal{D}|} \sum_{(x,y) \in \mathcal{D}} \mathcal{L}(f_\theta(x), y) + \mathcal{R}(\theta, x, y), \tag{1}$$

that includes an additional regularization term $\mathcal{R}$ which, generally, is a function of input x, the label y, the model parameters θ, and reflects prior assumptions about the model. For example, $L1$ regularization reflects the belief that sparse solutions in parameter space are more desirable.

3.2 Neuron Attributions

Formally, attribution methods aim at uncovering the importance of each input feature of a given input sample to the prediction of the neural network. Recent works have demonstrated that feature attributions methods can be incorporated into the training process (Lundberg & Lee, 2017; Erion et al., 2021). These *attribution priors* optimize attributions to have desirable characteristics, including interpretability as well as smoothness and sparsity in predictions. However, these methods have exclusively investigated *output* attributions, i.e., contributions of input features to the output of a model. To the best of our knowledge, we are the first work to investigate regularization of *latent attributions*.

We rewrite our predictive function f using function composition $f = l \circ g$. Here $g : \mathcal{X} \to \mathcal{H}$ maps the input to a representation $h = g(x) \in \mathcal{H}$, where $\mathcal{H} \subseteq \mathbb{R}^{d_H}$ is a d_H-dimensional latent space. Additionally, $l : \mathcal{H} \to \mathcal{Y}$ maps the latent representation to a label space $y = l(h) \in \mathcal{Y}$. We let $h_i = g_i(x)$, for, $i \in [d_H]$ denote the ith neuron in the hidden layer of interest. Additionally, we use $a^i_j(x) \in \mathbb{R}$ to denote the attribution of the ith neuron w.r.t. the feature x_j. With this notation, upper indices correspond to latent units and lower indices to features. In some cases, it will be convenient to stack all the feature attributions together in the attribution vector $a^i(x) = [a^i_j(x)]_{j=1}^{d_X} \in \mathbb{R}^{d_X}$.

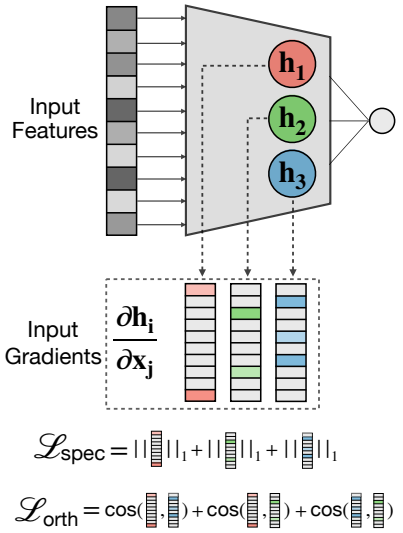


Figure 2: **Method illustration.** TANGOS regularizes the gradients with respect to each of the latent units.

Attribution methods work by using gradient signals to evaluate the contributions of the input features. In the most simplistic setting:

$$a^i_j(x) \equiv \frac{\partial h_i(x)}{\partial x_j}. \tag{2}$$

This admits a simple interpretation through a first order Taylor expansion: if the input feature x_j were to increase by some small number $\epsilon \in \mathbb{R}^+$, the neuron activation would change by $\epsilon \cdot a^i_j(x) + \mathcal{O}(\epsilon^2)$. The larger the absolute value of the gradient, the stronger the effect of a change in the input feature. We emphasize that our method is *agnostic* to the gradient attribution method, as different methods may be more appropriate for different tasks. For a comprehensive review of different methods, assumptions and trade-offs, see Ancona et al. (2017). For completeness, we also note another category of attribution methods is built around *perturbations*: this class of methods evaluates contributions of individual features through repeated perturbations. Generally speaking, they are more computationally inefficient due to the multiple forward passes through the neural network and are difficult to include directly in the training objective.

3.3 Rewarding Orthogonalization and Specialization

The main contribution of this work is proposing regularization on neuron attributions. In the most general sense, any function of any neuron attribution method could be used as a regularization term, thus encoding prior knowledge about the properties a model should have.

Specifically, the regularization term is a function of the network parameters θ and x, i.e., $\mathcal{R}(\theta, x)$, and encourages prior assumptions on desired behavior of the learned function. Biological sensory neurons are highly specialized. For example, certain visual neurons respond to a specific set of visual features including edges and orientations within a single receptive field. They are thus highly *selective* with *limited capacity* to react to specific physiological stimuli (Johnston & Dark, 1986; Cowan et al., 2005). Similarly, we hypothesize that neurons that are more specialized and pay attention to sparser

signals should exhibit better generalization performance. We propose the following desiderata and corresponding regularization terms:

- **Specialization.** The contribution of input features to the activation of a particular neuron should be sparse, i.e., $||a^i(x)||$ is small for all $i \in [d_H]$ and $x \in \mathcal{X}$. Intuitively, in higher-dimensional settings, a few features should account for a large percentage of total attributions while others are near zero, resulting in more *specialized* neurons. We write this as a regularization term for mini-batch training:
$$\mathcal{L}_{\text{spec}}(x) = \frac{1}{B}\sum_{b=1}^{B}\frac{1}{d_H}\sum_{i=1}^{d_H}\|a^i(x_b)\|_1,$$
where $b \in [B]$ is the batch index of $x_b \in \mathcal{X}$ and $\|\cdot\|_1$ denotes the l_1 norm.
- **Orthogonalization.** Different neurons should attend to non-overlapping subsets of input features given a particular input sample. To encourage this, we penalize the correlation between neuron attributions $\rho[a^i(x), a^j(x)]$ for all $i \neq j$ and $x \in \mathcal{X}$. In other words, for each particular input, we want to discipline the latent units to attend to different aspects of the input. Then, expressing this as a regularization term for mini-batch training, we obtain:
$$\mathcal{L}_{\text{orth}}(x) = \frac{1}{B}\sum_{b=1}^{B}\frac{1}{C}\sum_{i=2}^{d_H}\sum_{j=1}^{i-1}\rho\left[a^i(x_b), a^j(x_b)\right].$$
Here, C is the number of pairwise correlations, $C = \frac{d_H \cdot (d_H - 1)}{2}$, and $\rho[a^i(x_b), a^j(x_b)] \in [0, 1]$ is calculated using the cosine similarity $\frac{|a^{i\intercal}(x_b)\ a^j(x_b)|}{||a^i(x_b)||_2||a^j(x_b)||_2}$ where $\|\cdot\|_2$ denotes the l_2 norm.

These terms can be combined into a single regularization term and incorporated into the training objective. The resulting TANGOS regularizer can be expressed as:
$$\mathcal{R}_{\text{TANGOS}}(x) = \lambda_1\mathcal{L}_{\text{spec}}(x) + \lambda_2\mathcal{L}_{\text{orth}}(x),$$
where λ_1, $\lambda_2 \in \mathbb{R}$ act as weighting terms. As this expression is computed using gradient signals, it can be efficiently implemented and minimized in any auto-grad framework.

4 *How* AND *Why* DOES TANGOS WORK?

To the best of our knowledge, TANGOS is the only work to explicitly regularize latent neuron attributions. A natural question to ask is (1) *How is* TANGOS *different from other regularization?* While intuitively it makes sense to enforce *specialization* of each unit and *orthogonalization* between units, we empirically investigate if other regularizers can achieve similar effects, revealing that our method regularizes a unique objective. Having established that the TANGOS objective is unique, the next question is (2) *Why does it work?* To investigate this question, we frame the set of neurons as an ensemble, and demonstrate that our regularization improves diversity among *weak learners*, resulting in improved out-of-sample generalisation.

4.1 TANGOS REGULARIZES A UNIQUE OBJECTIVE

TANGOS encourages generalization by explicitly decorrelating and sparsifying the attributions of latent units. A reasonable question to ask is if this is unique, or if other regularizers might achieve the same objective implicitly. Two alternative regularizers that immediately come to mind are $L2$ weight regularization and $Dropout$. Like TANGOS, weight regularization methods implicitly and partially penalize the gradients by shrinking the weights in the neural network. Additionally, $Dropout$ trains an ensemble of learners by forcing each neuron to be more independent. In Figure 3, we provide these results on the UCI temperature forecast dataset (Cho et al., 2020), in which data from 25 weather stations in South Korea is used to predict next day peak temperature. We train a fully connected neural network for each regularization method. Specifically, we plot $\mathcal{L}_{spec}$ and $\mathcal{L}_{orth}$ for neurons in the penultimate layers and the corresponding generalization performance. We supply an extended selection of these results on additional datasets and regularizers in Appendix C.

First, we observe that TANGOS significantly decreases correlation between different neuron attributions while other regularization terms, in fact, increase them. For $L2$ weight regularization, this

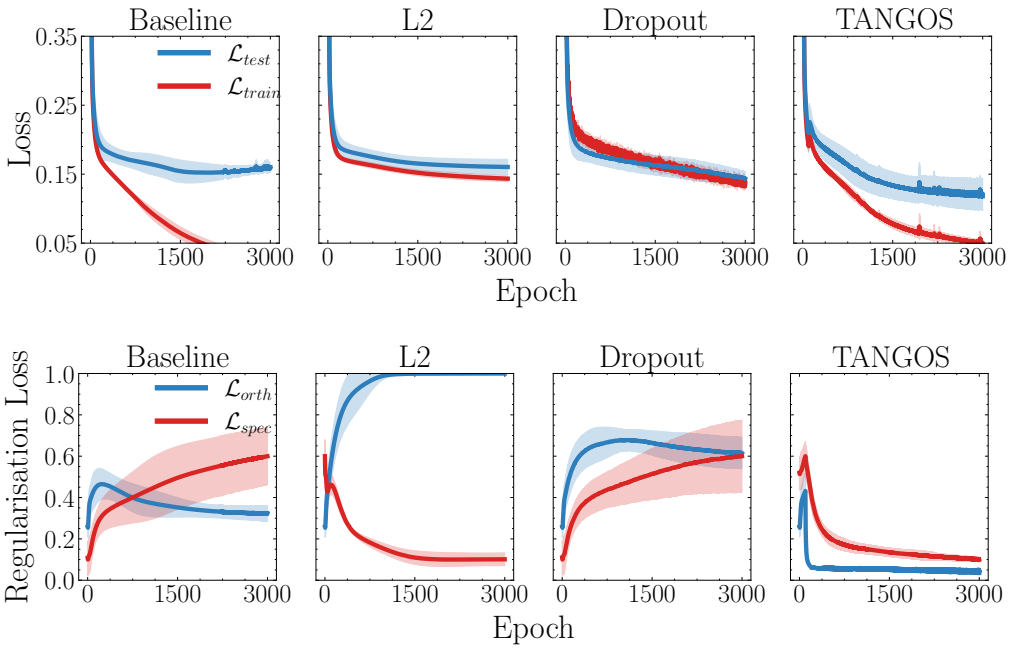


Figure 3: **Comparison of regularization objectives. (Top)** Generalization performance of key regularization techniques, **(Bottom)** corresponding neuron attributions evaluated on the test set. $L2$ and DO can reduce overfitting, but neuron attributions are in fact becoming more correlated. TANGOS achieves the best generalization performance by penalizing a different objective.

suggests that as the neural network weights are made smaller, the neurons increasingly attend to the same input features. A similar effect is observed for $Dropout$ - which has a logical explanation. Indeed, $Dropout$ creates redundancy by forcing each latent unit to be independent of others. Naturally, this encourages individual neurons to attend to overlapping features. In contrast, TANGOS aims to achieve specialization, such that neurons pay attention to sparse, non-overlapping features.

Additionally, we note that no alternative regularizers achieve greater attribution sparsity. This does not come as a surprise for $Dropout$, where the aim to induce redundancy in each neuron will naturally encourage individual neurons to attend to more features. While $L2$ does achieve a similar level of sparsity, this is paired with a high $\mathcal{L}_{orth}$ term indicating that, although the latent units do attend to sparse features, they appear to collapse to a solution in which they all attend to the same weighted subset of the input features. This, as we will discover in §4.2, is unlikely to be optimal for out-of-sample generalization.

Therefore, we conclude that the pairing of the specialization and orthogonality objectives in TANGOS regularizes a unique objective.

4.2 TANGOS Generalizes by Increasing Diversity Among Latent Units

Having established how TANGOS differs from existing regularization objectives, we now turn to answer *why* it works. In this section, we provide an alternative perspective on the effect of TANGOS regularization in the context of ensemble learning. A predictive model $f(x)$ may be considered as an ensemble model if it can be written in the form $f(x) = \sum_{T_k \in \mathcal{T}} \alpha_k T_k(x)$, where $\mathcal{T}$ represents a set of basis functions sometimes referred to as *weak learners* and the α_k's represent their respective scalar weights. It is therefore clear that each output of a typical neural network may be considered an ensemble predictor with every latent unit in its penultimate layer acting as a weak learner in their contribution to the models output. More formally, in this setting $T_k(x)$ is the activation of latent unit k with respect to an input x and α_k is the subsequent connection to the output activation. With this in mind, we present the following definition.

Definition 4.1. *Consider an ensemble regressor $f(x) = \sum_{T_k \in \mathcal{T}} \alpha_k T_k(x)$ trained on $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ where each (x, y) is drawn randomly from P_{XY}. Additionally, the weights are constrained such that $\sum_k \alpha_k = 1$. Then, for a given input-label pair (x, y), we define:*

(a) The overall ensemble error as: $\text{Err} = (f(x) - y)^2$.

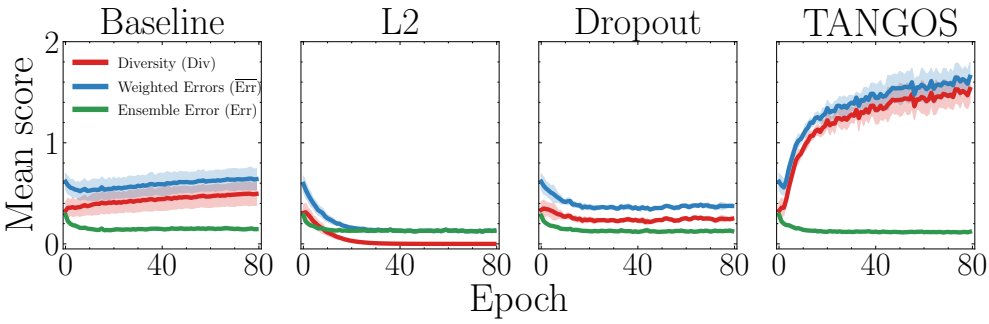


Figure 4: **Neuron Diversity.** Overall ensemble error and decomposition in terms of diversity and average error of the weak learners. Note while all methods achieve low overall error, TANGOS is the only method that does so by increasing the diversity among the latent units.

(b) The weighted errors of the weak learners as: $\overline{\text{Err}} = \sum_k \alpha_k (T_k(x) - y)^2$.

(c) The ensemble diversity as: $\text{Div} = \sum_k \alpha_k (T_k(x) - f(x))^2$.

Intuitively, $\overline{\text{Err}}$ provides a measure of the strength of the ensemble members while Div measures the diversity of their outputs. To understand the relationship between these two terms and the overall ensemble performance, we consider Proposition 1.

Proposition 1 ((Krogh & Vedelsby, 1994))**.** *The overall ensemble error for an input-label pair* (x, y) *can be decomposed into the weighted errors of the weak learners and the ensemble diversity such that:*

$$\text{Err} = \overline{\text{Err}} - \text{Div}. \tag{3}$$

This decomposition provides a fundamental insight into the success of ensemble methods: an ensembles overall error is reduced by decreasing the average error of the individual weak learners and increasing the diversity of their outputs. Successful ensemble methods explicitly increase ensemble diversity when training weak learners by, for example, sub-sampling input features (random forest, Breiman, 2001), sub-sampling from the training data (bagging, Breiman, 1996) or error-weighted input importance (boosting, Bühlmann, 2012).

Returning to the specific case of neural networks, it is clear that TANGOS provides a similar mechanism of increasing diversity among the latent units that act as weak learners in the penultimate layer. By forcing the latent units to attend to sparse, uncorrelated selections of features, the learned ensemble is encouraged to produce diverse learners whilst maintaining coverage of the entire input space in aggregate. In Figure 4, we demonstrate this phenomenon in practice by returning to the UCI temperature forecast regression task. We provide extended results in the Appendix C. We train a fully connected neural network with two hidden layers with the output layer weights constrained such that they sum to 1. We observe that regularizing with TANGOS drives up the diversity of the latent activations resulting in improved out-of-sample generalization. This is in contrast to other typical regularization approaches which also improve model performance, but exclusively by attempting to reduce the error of the individual ensemble members. This provides additional motivation for applying TANGOS in the tabular domain, an area where traditional ensemble methods have performed particularly well.

5 Experiments

In this section, we empirically evaluate TANGOS as a regularization method for improving generalization performance. We present our benchmark methods and training architecture, followed by extensive results on real-world datasets. There are a few main aspects that deserve empirical investigation, which we investigate in turn: ► **Stand-alone performance.** §5.1 compares performance of TANGOS, where the focus is on applying it as a stand-alone regularizer, with respect to a variety of benchmarks on a suite of real-world datasets. ► **In tandem performance.** §5.2 Motivated by our unique regularization objective and our analysis in §4, we demonstrate that applying TANGOS *in conjunction* with other regularizers can lead to even greater gains in generalization performance. All experiments were run on NVIDIA RTX A4000 GPUs.

Table 1: **Stand-Alone Regularization.** Comparison of regularizers on regression and classification in terms of test MSE and NLL. All models are trained on real-world datasets using 5-fold cross-validation and final evaluation reported on a held-out test set. **Bold** indicates the best performance. The average rank of each method across both regression and classification is included in the final row of the respective tables.

Dataset	Baseline	L1	L2	DO	BN	IN	MU	TANGOS
Regression (Mean Squared Error)								
FB	0.037	0.081	**0.029**	0.060	0.699	0.056	0.147	0.032
BH	0.192	0.197	0.183	0.241	0.190	0.218	0.286	**0.166**
WE	0.118	0.096	0.099	0.114	**0.090**	0.101	0.146	0.093
BC	0.323	0.263	0.277	0.282	0.294	0.308	0.323	**0.244**
WQ	0.673	0.641	0.644	0.666	0.639	0.669	0.713	**0.637**
SC	0.422	0.408	0.411	0.423	0.410	0.445	0.547	**0.387**
FF	1.274	1.280	1.274	1.266	1.330	**1.201**	1.289	1.276
PR	0.624	0.611	0.580	0.639	0.647	0.591	0.745	**0.573**
ST	0.419	0.416	0.418	0.390	0.461	0.539	**0.380**	0.382
AB	0.345	0.319	0.332	**0.312**	0.348	0.355	0.366	0.325
Avg Rank	5.3	3.7	3.3	4.5	5.0	5.3	7.1	**1.9**
Classification (Mean Negative Log-likelihood)								
HE	0.490	0.472	0.431	0.444	0.459	0.439	**0.416**	0.426
BR	0.074	0.070	0.070	0.078	0.080	0.071	0.095	**0.069**
CE	0.519	**0.395**	0.407	0.445	0.604	0.502	0.472	0.408
CR	0.464	0.405	0.402	0.457	0.460	0.481	0.448	**0.369**
HC	0.320	0.222	0.226	0.237	0.257	0.312	0.248	**0.215**
AU	0.448	0.442	0.385	0.405	0.549	0.479	0.478	**0.379**
TU	1.649	1.633	1.613	1.621	**1.484**	1.646	1.657	1.495
EN	1.040	1.040	1.042	1.058	1.098	1.072	1.065	**0.974**
TH	0.700	0.506	**0.500**	0.714	0.785	0.638	0.618	0.513
SO	0.606	**0.238**	0.382	0.570	0.484	0.540	0.412	0.371
Avg Rank	6.4	3.0	2.6	5.0	6.3	6.0	5.1	**1.7**

TANGOS. We train TANGOS regularized models as described in Algorithm 1 in the supplementary material. For the specialization parameter we search for $\lambda_1 \in \{1, 10, 100\}$ and for the orthogonalization parameter we search for $\lambda_2 \in \{0.1, 1\}$. For computational efficiency, we apply a sub-sampling scheme where 50 neuron pairs are randomly sampled for each input (for further details see Appendix E).

Benchmarks. We evaluate TANGOS against a selection of popular regularizer benchmarks. First, we consider weight decay methods **L1** and **L2** regularization, which sparsify and shrink the learnable parameters. For the regularizers coefficients, we search for $\lambda \in \{0.1, 0.01, 0.001\}$ where regularization is applied to all layers. Next, we consider Dropout (**DO**), with drop rate $p \in \{20\%, 50\%\}$, and apply DO after every dense layer during training. We also consider implicit regularization in batch normalization (**BN**). Lastly, we evaluate data augmentation techniques Input Noise (**IN**), where we use additive Gaussian noise with mean 0 and standard deviation $\sigma \in \{0.1, 0.01\}$ and MixUp (**MU**). Furthermore, each training run applies early stopping with patience of 30 epochs. In all experiments, we use 5-fold cross validation to train and validate each benchmark. We select the model which achieves the lowest validation error and provide a final evaluation on a held-out test set.

5.1 Generalization: Stand-Alone Regularization

For the first set of experiments, we are interested in investigating the individual regularization effect of TANGOS. To ensure a fair comparison, we evaluate the generalization performance on held-out test sets across a variety of datasets.

Datasets. We employ 20 real-world tabular datasets from the UCI machine learning repository. Each dataset is split into 80% for cross-validation and the remaining 20% for testing. The splits are standardized on just the training data, such that features have mean 0 and standard deviation 1 and

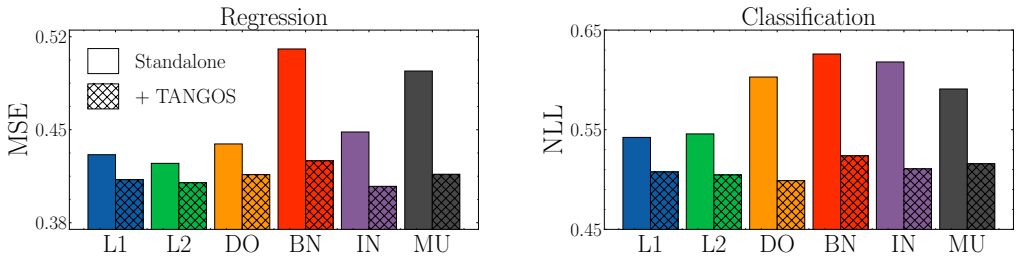


Figure 5: **In Tandem Regularization.** Aggregated errors across the 10 regression datasets (left) and the 10 classification datasets (right). In all cases the addition of TANGOS provides superior performance over the standalone regularizer.

categorical variables are one-hot encoded. Further details on the 20 datasets used can be found in the supplementary material.

Training and Evaluation. To ensure fair comparison, all regularizers are applied to an MLP with two ReLU-activated hidden layers, where each hidden layer has $d_H + 1$ neurons. The models are trained using Adam optimizer with a dataset dependent learning rate from $\{0.01, 0.001, 0.0001\}$ and are trained for up to a maximum of 200 epochs. For regression tasks, we report the average Mean Square Error (MSE) and on classification tasks, we report the average negative log-likelihood (NLL).

Results. In Table 1 we provide the benchmarking results for individual regularizers. We observe that TANGOS achieves best performance on $10/20$ of the datasets with no outstanding runner up. We also observe that on 6 of the remaining datasets, TANGOS ranks second. This is illustrated by a ranking plot which we provide in Appendix A. There we also provide a similar table displaying standard errors. Note that regularizers are seldom used on their own. Beyond TANGOS superiority as a stand-alone method, it remains to show that it brings additional value when used with other regularization methods. This is the purpose of next section.

5.2 Generalisation: In Tandem Regularization

Motivated by the insights described in §4, a natural next question is whether TANGOS can be applied in conjunction with existing regularization to unlock even greater generalization performance. In this set of experiments we investigate this question.

Setup. The setting for this experiment is identical to that of §5.1 except now we consider the six baseline regularizers *in tandem* with TANGOS. We now seek to examine if pairing our proposed regularizer with existing methods results in even greater generalization performance. We again run 5-fold cross-validation, searching over the same hyperparameters, with the final models evaluated on a held-out test set.

Results. We summarize the aggregated results over the datasets for each of the six baseline regularizers in combination with TANGOS in Figure 5. Consistently across all regularizers in both the regression and the classification settings, we observe that adding TANGOS regularization improves test performance. We provide the full table of results in the supplementary material. We also note an apparent interaction effect between certain regularizers (i.e. input noise for regression and dropout for classification), where methods that seemed to not be particularly effective as stand-alone regularizers become the best performing method when evaluated in tandem. The relationship between such regularizers provides an interesting direction for future work.

6 Discussion

In this work, we have introduced TANGOS, a novel regularization method which promotes specialization and orthogonalization among the gradient attributions of the latent units of a neural network. We show *how* this regularization objective is distinct from other popular methods by investigating their effects on model parameters. We also motivate *why* our method provides out-of-sample generalization by linking it's effect to increasing diversity in the latent units. Finally, we empirically demonstrate TANGOS utility as both a stand-alone approach as well as in tandem with other regularizers across a comprehensive set of real-world tabular datasets.

REPRODUCIBILITY STATEMENT

We have attempted to make our experimental results easily reproducible by both detailed description of our experimental procedure and providing the code used to produce our results. Experiments are described in Section 5 with further details in Appendices E and F. We have provided our code in the supplementary material and plan to release a public repository if accepted for publication. All datasets used in this work can be freely downloaded from the UCI repository (Dua et al., 2017) with specific details provided in Appendix F.

REFERENCES

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104*, 2017.

Pierre Baldi, Peter Sadowski, and Daniel Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5(1):1–9, 2014.

Nitin Bansal, Xiaohan Chen, and Zhangyang Wang. Can we gain more from orthogonality regularizations in training deep networks? *Advances in Neural Information Processing Systems*, 31, 2018.

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Lorenzo Brigato and Luca Iocchi. A close look at deep learning with small data. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 2490–2497. IEEE, 2021.

Peter Bühlmann. Bagging, boosting and ensemble methods. In *Handbook of computational statistics*, pp. 985–1022. Springer, 2012.

Jiefeng Chen, Xi Wu, Vaibhav Rastogi, Yingyu Liang, and Somesh Jha. Robust attribution regularization. *Advances in Neural Information Processing Systems*, 32, 2019.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

Dongjin Cho, Cheolhee Yoo, Jungho Im, and Dong-Hyun Cha. Comparative assessment of various machine learning-based bias correction methods for numerical weather prediction model forecasts of extreme air temperatures in urban areas. *Earth and Space Science*, 7(4):e2019EA000740, 2020.

Paulo Cortez and Aníbal de Jesus Raimundo Morais. A data mining approach to predict forest fires using meteorological data. 2007.

Paulo Cortez and Alice Maria Gonçalves Silva. Using data mining to predict secondary school student performance. 2008.

Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision support systems*, 47(4): 547–553, 2009.

Nelson Cowan, Emily M Elliott, J Scott Saults, Candice C Morey, Sam Mattox, Anna Hismjatullina, and Andrew RA Conway. On the capacity of attention: Its estimation and its role in working memory and cognitive aptitudes. *Cognitive psychology*, 51(1):42–100, 2005.

Jonathan Crabbé and Mihaela van der Schaar. Label-free explainability for unsupervised models. *arXiv preprint arXiv:2203.01928*, 2022.

Harris Drucker and Yann Le Cun. Improving generalization performance using double backpropagation. *IEEE Transactions on Neural Networks*, 3(6):991–997, 1992.

Dheeru Dua, Casey Graff, et al. Uci machine learning repository. 2017.

Gabriel Erion, Joseph D Janizek, Pascal Sturmfels, Scott M Lundberg, and Su-In Lee. Improving performance of deep learning models with axiomatic attribution priors and expected gradients. *Nature machine intelligence*, 3(7):620–631, 2021.

Kelwin Fernandes, Jaime S Cardoso, and Jessica Fernandes. Transfer learning with partial observability applied to cervical cancer screening. In *Iberian conference on pattern recognition and image analysis*, pp. 243–250. Springer, 2017.

Douglas H Fisher and Jeffrey C Schlimmer. Concept simplification and prediction accuracy. In *Machine Learning Proceedings 1988*, pp. 22–28. Elsevier, 1988.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34:18932–18943, 2021.

Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data? *arXiv preprint arXiv:2207.08815*, 2022.

Francesca Grisoni, Viviana Consonni, Sara Villa, Marco Vighi, and Roberto Todeschini. Qsar models for bioconcentration: is the increase in the complexity justified by more accurate predictions? *Chemosphere*, 127:171–179, 2015.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

Isabelle Guyon, Lisheng Sun-Hosoya, Marc Boullé, Hugo Jair Escalante, Sergio Escalera, Zhengying Liu, Damir Jajetic, Bisakha Ray, Mehreen Saeed, Michèle Sebag, et al. Analysis of the automl challenge series. *Automated Machine Learning*, pp. 177, 2019.

David Harrison Jr and Daniel L Rubinfeld. Hedonic housing prices and the demand for clean air. *Journal of environmental economics and management*, 5(1):81–102, 1978.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

Georg Hoffmann, Andreas Bietenbeck, Ralf Lichtinghagen, and Frank Klawonn. Using machine learning techniques to generate laboratory diagnostic pathways—a case study. *J Lab Precis Med*, 3:58, 2018.

Guosheng Hu, Xiaojiang Peng, Yongxin Yang, Timothy M Hospedales, and Jakob Verbeek. Frankenstein: Learning deep face representations using small data. *IEEE Transactions on Image Processing*, 27(1):293–303, 2017.

Sadiq Hussain, Rasha Atallah, Amirrudin Kamsin, and Jiten Hazarika. Classification, clustering and association rule mining in educational datasets using data mining tools: A case study. In *Computer Science On-line Conference*, pp. 196–211. Springer, 2018.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. PMLR, 2015.

Gaojie Jin, Xinping Yi, Liang Zhang, Lijun Zhang, Sven Schewe, and Xiaowei Huang. How does weight correlation affect generalisation ability of deep neural networks? *Advances in Neural Information Processing Systems*, 33:21346–21356, 2020.

William A Johnston and Veronica J Dark. Selective attention. *Annual review of psychology*, 37(1): 43–75, 1986.

Arlind Kadra, Marius Lindauer, Frank Hutter, and Josif Grabocka. Well-tuned simple nets excel on tabular datasets. *Advances in Neural Information Processing Systems*, 34, 2021.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. *Advances in neural information processing systems*, 7, 1994.

Jan Kukačka, Vladimir Golkov, and Daniel Cremers. Regularization for deep learning: A taxonomy. *arXiv preprint arXiv:1710.10686*, 2017.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Frederick Liu and Besim Avci. Incorporating priors with feature attribution on text classification. *arXiv preprint arXiv:1906.08286*, 2019.

Weiyang Liu, Rongmei Lin, Zhen Liu, James M Rehg, Liam Paull, Li Xiong, Le Song, and Adrian Weller. Orthogonal over-parameterized training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7251–7260, 2021.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks. *Advances in neural information processing systems*, 29, 2016.

Ryszard S Michalski, Igor Mozetic, Jiarong Hong, and Nada Lavrac. The multi-purpose incremental learning system aq15 and its testing application to three medical domains. In *Proc. AAAI*, volume 1986, pp. 1–041, 1986.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9078–9086, 2019.

Sérgio Moro, Paulo Rita, and Bernardo Vala. Predicting social media performance metrics and evaluation of the impact on brand building: A data mining approach. *Journal of Business Research*, 69(9):3341–3351, 2016.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

J. Ross Quinlan. Simplifying decision trees. *International journal of man-machine studies*, 27(3): 221–234, 1987.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Icml*, 2011.

Andrew Ross, Isaac Lage, and Finale Doshi-Velez. The neural lasso: Local linear sparsity for interpretable explanations. In *Workshop on Transparent and Interpretable Machine Learning in Safety Critical Environments, 31st Conference on Neural Information Processing Systems*, volume 4, 2017a.

Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. Right for the right reasons: Training differentiable models by constraining their explanations. *arXiv preprint arXiv:1703.03717*, 2017b.

W Nick Street, William H Wolberg, and Olvi L Mangasarian. Nuclear feature extraction for breast tumor diagnosis. In *Biomedical image processing and biomedical visualization*, volume 1905, pp. 861–870. SPIE, 1993.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pp. 3319–3328. PMLR, 2017.

Joseph J Thompson, Mark R Blair, Lihan Chen, and Andrew J Henrey. Video game telemetry as a critical tool in the study of complex skill learning. *PloS one*, 8(9):e75129, 2013.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Samuel George Waugh. *Extending and benchmarking Cascade-Correlation: extensions to the Cascade-Correlation architecture and benchmarking of feed-forward supervised artificial neural networks*. PhD thesis, University of Tasmania, 1995.

Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. *arXiv preprint arXiv:2002.06715*, 2020.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

Maciej Zikeba, Jakub M Tomczak, Marek Lubicz, and Jerzy 'Swikatek. Boosted svm for extracting rules from imbalanced data in application to prediction of the post-operative life expectancy in the lung cancer patients. *Applied Soft Computing*, 2013. doi: [WebLink].

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005.

A RANKING PLOT

In Table 1, we reported generalisation performance of TANGOS compared to other regularizers in a stand-alone setting. To gain a better understanding of relative performance, we visually depict the relative ranking of regularizers across all 20 datasets. Figure 6 demonstrates that TANGOS consistently ranks as one of the better performing regularizers, while performance of benchmark methods tend to fluctuate depending on the dataset.

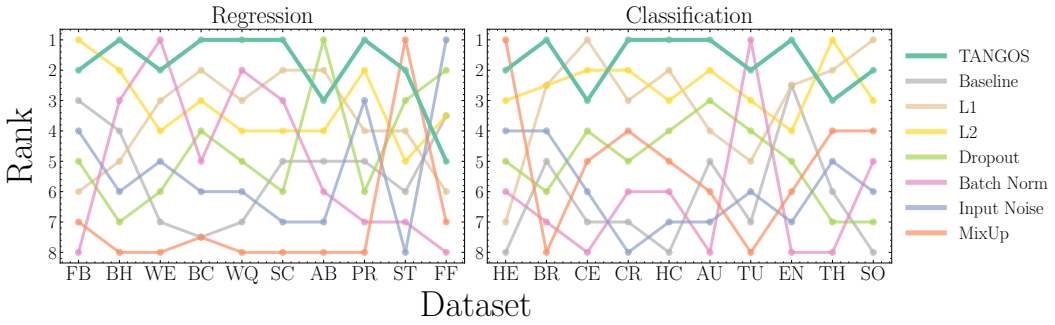


Figure 6: **Ranking of stand-alone regularizers.** Relative ranking of regularizer performance across the 20 datasets, as reported in Table 1. TANGOS consistently ranks among the best performing regularizers.

B STAND-ALONE UNCERTAINTY

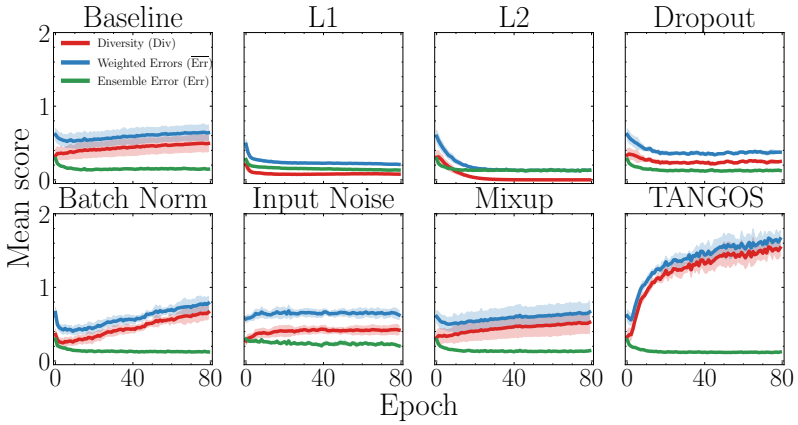
In Table 2, we report the standard deviation on generalization performance reported in Table 1. The standard errors are computed using 10 seeded runs.

Table 2: **Standard error on generalization performance.** Standard errors with respect to the random seed after retraining models from Table 1 experiments 10 times.

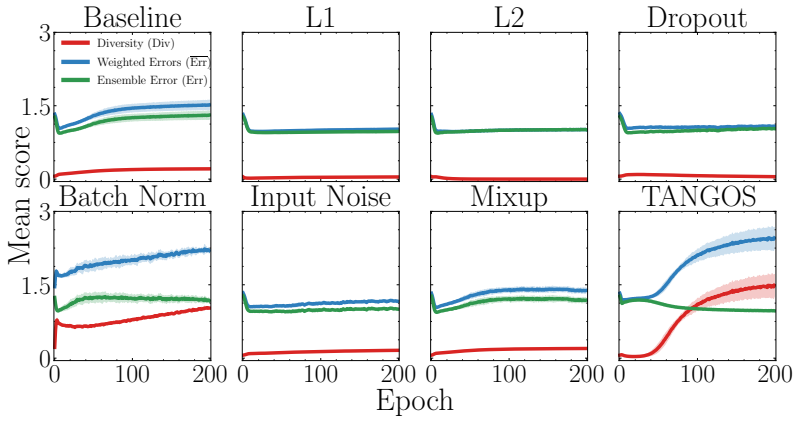
Dataset	Baseline	L1	L2	DO	BN	IN	MU	TANGOS
Regression (Mean Squared Error)								
FB	0.051	0.016	0.009	0.051	0.627	0.076	0.041	0.042
BH	0.023	0.021	0.029	0.022	0.025	0.023	0.011	0.028
WE	0.006	0.010	0.008	0.008	0.008	0.013	0.010	0.009
BC	0.013	0.007	0.005	0.009	0.020	0.024	0.012	0.009
WQ	0.016	0.019	0.005	0.014	0.021	0.015	0.019	0.008
SC	0.026	0.014	0.019	0.025	0.023	0.168	0.067	0.017
FF	0.034	0.029	0.035	0.033	0.041	0.036	0.031	0.035
PR	0.042	0.029	0.020	0.031	0.032	0.072	0.016	0.026
ST	0.090	0.084	0.085	0.064	0.076	0.080	0.082	0.029
AB	0.016	0.016	0.008	0.011	0.026	0.014	0.012	0.006
Classification (Mean Negative Log-likelihood)								
HE	0.057	0.049	0.009	0.033	0.163	0.038	0.067	0.032
BR	0.086	0.005	0.002	0.133	0.034	0.060	0.010	0.031
CE	0.060	0.007	0.008	0.043	0.051	0.044	0.056	0.023
CR	0.029	0.094	0.004	0.034	0.041	0.027	0.027	0.019
HC	0.091	0.014	0.019	0.026	0.054	0.111	0.024	0.014
AU	0.038	0.030	0.002	0.030	0.081	0.031	0.037	0.019
TU	0.075	0.075	0.087	0.077	0.087	0.096	0.078	0.064
EN	0.036	0.038	0.033	0.045	0.082	0.049	0.050	0.046
TH	0.048	0.021	0.002	0.065	0.096	0.047	0.039	0.001
SO	0.038	0.028	0.016	0.046	0.029	0.025	0.023	0.008

C INSIGHTS - EXTENDED RESULTS

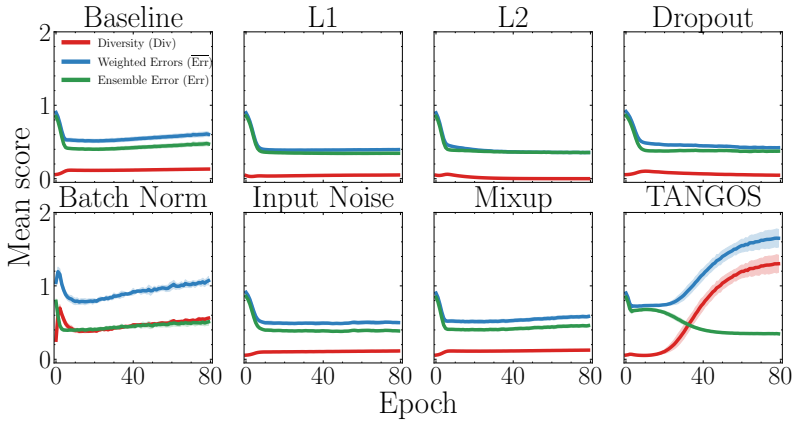
In this section, we present extended results of the decomposition of overall model error into diversity and weighted error among an ensemble of latent units from Section 4.2. We include all eight regularizers as described in Table 5 and three datasets (WE, ST and BC) as described in Table 4. The results are included in Figure 7.



(a) Weather (WE) dataset.



(b) Student (ST) dataset.



(c) Bioconcentration (BC) dataset.

Figure 7: **Neuron diversity.** Further examples of ensemble decomposition $\text{Err} = \overline{\text{Err}} - \text{Div}$ as discussed in Section 4.2.

D IN TANDEM RESULTS

In Table 3, we provide a detailed breakdown of Figure 5, specifically by reporting in tandem performance when benchmarks are paired with TANGOS across all datasets.

Table 3: **In tandem performance.** Mean $\pm$ standard deviation of generalization performance when each regularizer employed in tandem with TANGOS.

Dataset	L1	L2	DO	BN	IN	MU
Regression (Mean Squared Error)						
FB	0.033 $\pm$0.018	0.018 $\pm$0.009	0.023 $\pm$0.215	0.042 $\pm$0.268	0.028 $\pm$0.047	0.046 $\pm$0.073
BH	0.192 $\pm$0.022	0.176 $\pm$0.024	0.196 $\pm$0.024	0.220 $\pm$0.021	0.191 $\pm$0.023	0.178 $\pm$0.019
WE	0.093 $\pm$0.011	0.091 $\pm$0.009	0.092 $\pm$0.009	0.076 $\pm$0.009	0.081 $\pm$0.012	0.077 $\pm$0.013
WQ	0.637 $\pm$0.014	0.639 $\pm$0.006	0.628 $\pm$0.008	0.630 $\pm$0.015	0.644 $\pm$0.011	0.649 $\pm$0.018
BC	0.227 $\pm$0.011	0.236 $\pm$0.011	0.243 $\pm$0.013	0.275 $\pm$0.027	0.235 $\pm$0.009	0.260 $\pm$0.014
SC	0.407 $\pm$0.044	0.399 $\pm$0.072	0.370 $\pm$0.031	0.408 $\pm$0.049	0.391 $\pm$0.317	0.425 $\pm$0.061
AB	0.309 $\pm$0.028	0.312 $\pm$0.028	0.312 $\pm$0.026	0.311 $\pm$0.027	0.319 $\pm$0.037	0.308 $\pm$0.03
FF	1.281 $\pm$0.029	1.276 $\pm$0.043	1.268 $\pm$0.028	1.297 $\pm$0.027	1.203 $\pm$0.046	1.207 $\pm$0.011
PR	0.553 $\pm$0.047	0.572 $\pm$0.031	0.642 $\pm$0.093	0.561 $\pm$0.069	0.565 $\pm$0.025	0.568 $\pm$0.081
ST	0.392 $\pm$0.006	0.382 $\pm$0.006	0.388 $\pm$0.016	0.446 $\pm$0.018	0.417 $\pm$0.012	0.447 $\pm$0.011
Classification (Mean Negative Log-likelihood)						
HE	0.441 $\pm$0.046	0.427 $\pm$0.049	0.454 $\pm$0.095	0.407 $\pm$0.047	0.377 $\pm$0.067	0.397 $\pm$0.075
BR	0.074 $\pm$0.005	0.070 $\pm$0.002	0.068 $\pm$0.006	0.062 $\pm$0.003	0.065 $\pm$0.01	0.078 $\pm$0.011
CE	0.389 $\pm$0.007	0.396 $\pm$0.007	0.394 $\pm$0.033	0.446 $\pm$0.024	0.422 $\pm$0.075	0.422 $\pm$0.062
CR	0.362 $\pm$0.021	0.366 $\pm$0.015	0.364 $\pm$0.003	0.406 $\pm$0.023	0.367 $\pm$0.01	0.384 $\pm$0.036
HC	0.200 $\pm$0.056	0.179 $\pm$0.012	0.185 $\pm$0.009	0.186 $\pm$0.021	0.211 $\pm$0.008	0.181 $\pm$0.037
AU	0.368 $\pm$0.028	0.360 $\pm$0.138	0.352 $\pm$0.011	0.344 $\pm$0.013	0.379 $\pm$0.016	0.368 $\pm$0.041
TU	1.519 $\pm$0.082	1.506 $\pm$0.045	1.481 $\pm$0.048	1.506 $\pm$0.067	1.522 $\pm$0.049	1.503 $\pm$0.087
SO	0.227 $\pm$0.036	0.268 $\pm$0.071	0.233 $\pm$0.021	0.353 $\pm$0.052	0.257 $\pm$0.032	0.304 $\pm$0.054
EN	0.990 $\pm$0.024	0.971 $\pm$0.002	0.945 $\pm$0.001	1.004 $\pm$0.026	0.995 $\pm$0.001	1.007 $\pm$0.04
TH	0.506 $\pm$0.031	0.503 $\pm$0.008	0.513 $\pm$0.01	0.524 $\pm$0.008	0.512 $\pm$0.045	0.514 $\pm$0.03

E APPROXIMATION AND ALGORITHM

Calculating the attribution of the latent units with respect to the input involves computing the Jacobian matrix, which can be computed in $\mathcal{O}(1)$ time and has memory complexity $\mathcal{O}(d_H d_X)$. The computational complexity of calculating $\mathcal{L}_{orth}$ is $\mathcal{O}(d_H^2)$ (i.e. all pairwise computation between latent units). While the calculation can be efficiently parallelized, this still becomes impractically expensive with higher dimensional layers. To address this, we introduce a relaxation by randomly subsampling pairs of neurons to calculate attribution similarity. We denote by I denote the set of all possible pairs of neuron indices, $I = \{(i,j) \,\forall\, i,j \in [d_H] \text{ and } i \neq j\}$. Further, we let M denote a randomly sampled subset of I, $M \subseteq I$. We devise an approximation to the regularization term, denoted by $\mathcal{L}'_{orth}$, by estimating the penalty on the subset M, where the size of M can be chosen to balance computational burden with more faithful estimation:

$$\mathcal{L}'_{orth}(x) = \frac{1}{B}\sum_{b=1}^{B}\frac{1}{|M|}\sum_{(i,j)\in M}\rho[a^i(x_b), a^j(x_b)]$$

This reduces the complexity of calculating $\mathcal{L}_{orth}$ from $\mathcal{O}(d_H^2)$ to $\mathcal{O}(M)$. For our experimental results described in Tables 1, 2 and 3, we use $|M| = 50$. We empirically demonstrate that this approximation still leads to strong results in real-world experiments. The overall training procedure is described in Algorithm 1.

Additionally, we provide an empirical analysis of TANGOS designed to evaluate the effectiveness of our proposed subsampling approximation with respect to generalization performance and computational efficiency as the number of sampled neuron pairs M grows. Furthermore, we analyze the computational efficiency of our method as the number of latent units grows, evaluating the methods capacity to scale to large models.

All experiments are run on the BC dataset which we split into 80% training and 20% validation. We fix $\lambda_1 = 100$ and $\lambda_2 = 0.1$ throughout our experiments. We run each experiment over 10 random seeds and report the mean and standard deviation. All remaining experimental details are consistent

Algorithm 1 TANGOS regularization

Result: Learned parameters θ
Input: λ_1, λ_2, training data $\mathcal{D}$, learning rate η;
Initialise θ;
while not converged **do**
  Sample $\mathcal{D}_{mini}$ from $\mathcal{D}$;
  $\hat{\mathcal{L}}(f_\theta(x), y) = \mathbb{E}_{(x,y)\sim\mathcal{D}_{mini}}[\mathcal{L}(f_\theta(x), y)]$;
  $\hat{\mathcal{R}}(x) = \lambda_1 \mathbb{E}_{x\sim\mathcal{D}_{mini}}[\mathcal{L}_{spec}(x)] + \lambda_2 \mathbb{E}_{x\sim\mathcal{D}_{mini}}[\mathcal{L}'_{orth}(x)]$;
  $\theta \leftarrow \theta + \eta \nabla_\theta \left[\hat{\mathcal{L}}(f_\theta(x), y) + \hat{\mathcal{R}}(x)\right]$;
end while

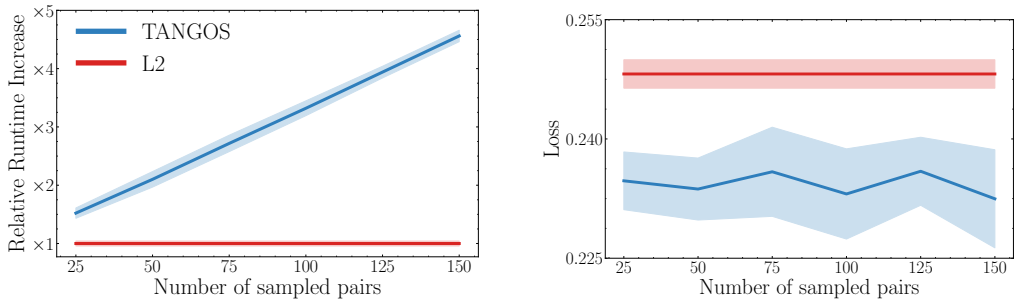


Figure 8: **Sampling efficiency.** Runtime increases linearly with number of sampled pairs (left) while better generalization performance is maintained even for low sampling rates (right). The benefits of TANGOS can be realized using our proposed sampling approximation with comparable runtime to even the most efficient existing regularization approaches.

with our experiments in Section 5. We note that our implementation of TANGOS is not optimized to the same extent as the Pytorch (Paszke et al., 2019) implementation of L2 to which we compare, and therefore we may consider our relative computational performance to be a loose upper bound on a truly optimized version.

In Figure 8 (left), we report the relative increase compute time per epoch as we increase the number of sampled pairs. As theory would suggest, this growth is linear. A natural follow up question is the extent to which model performance is affected by decreasing the number of sampled pairs. In Figure 8 (right), we observe that even very low sampling rates still result in excellent performance. Based on these results, our recommendation for practitioners is that while increasing the sampling rate can lead to marginal improvements in performance, relatively low sampling rates appear to be generally sufficient and do not require prohibitive computational overhead.

Given the results in Figure 8, we next wish to evaluate if the proposed sampling scheme enables TANGOS to scale to much bigger models. In order to evaluate this we vary the number of neurons in the relevant hidden layer while maintaining a fixed sampling rate of 50 pairs (consistent with our experiments in Section 5). Other experimental parameters are consistent with the previous experiment. The results are provided in Figure 9 where we observe a relatively slow increase in runtime as the model grows. These results demonstrate that TANGOS can efficiently be applied to much larger models by using our proposed sampling scheme.

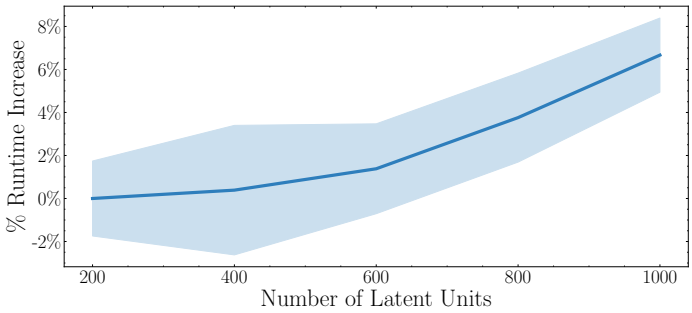


Figure 9: **Scaling to large models.** With a subsampling rate fixed at $M = 50$, TANGOS incurs only a small percentage increase in runtime as the number of neurons in the penultimate hidden layer increases dramatically.

F Dataset and Regularizer Details

We perform our experiments on 20 real-world publicly available datasets obtained from (Dua et al., 2017). They are summarized in Table 4. Further information and the source files used for each of the respective datasets can be found at: `https://archive.ics.uci.edu/ml/machine-learning-databases/<UCISource>/` where `<UCI Source>` denotes the datasets unique identifier as listed in Table 4. Standard preprocessing was applied including standardization of features, one hot encoding of categorical variables, median imputation of missing values and log transformations of highly skewed feature distributions. Furthermore, for computational feasibility, datasets with over 1000 samples were reduced in size. In these cases the first 1000 samples from the original UCI Source file were used. In Table 5 we summarize the regularizers considered in this work.

Table 4: **Dataset descriptions.** Summary of the datasets considered in this work.

Dataset	UCI Source	Type	Feature size	Sample Size	Reference
Facebook (FB)	"00368"	Regression	21	495	Moro et al. (2016)
Boston (BH)	[1]	Regression	13	506	Harrison Jr & Rubinfeld (1978)
Weather (WE)	"00514"	Regression	45	1000	Cho et al. (2020)
Wine Quality (WQ)	"wine-quality"	Regression	11	1000	Cortez et al. (2009)
Bioconcentration (BC)	"00510"	Regression	45	779	Grisoni et al. (2015)
Skillcraft (SC)	"00272"	Regression	18	1000	Thompson et al. (2013)
Forest Fire (FF)	"forest-fires"	Regression	39	517	Cortez & Morais (2007)
Protein (PR)	"00265"	Regression	9	1000	Dua et al. (2017)
Student (ST)	"00320"	Regression	56	649	Cortez & Silva (2008)
Abalone (AB)	"abalone"	Regression	9	1000	Waugh (1995)
Heart (HE)	"statlog"	Classification	20	270	Dua et al. (2017)
Breast (BR)	"breast-cancer-wisconsin"	Classification	9	699	Street et al. (1993)
Cervical (CE)	"00383"	Classification	136	858	Fernandes et al. (2017)
Credit (CR)	"credit-screening"	Classification	40	677	Dua et al. (2017)
HCV (HC)	"00571"	Classification	12	615	Hoffmann et al. (2018)
Australian (AU)	"statlog"	Classification	55	690	Quinlan (1987)
Tumor (TU)	"primary-tumor"	Classification	25	339	Michalski et al. (1986)
Entrance (EN)	"00582"	Classification	38	666	Hussain et al. (2018)
Thoracic (TH)	"00277"	Classification	24	470	Zikeba et al. (2013)
Soybean (SO)	"soybean"	Classification	484	683	Fisher & Schlimmer (1988)

[1] See: `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_boston.html`

Table 5: **Overview of regularizers.** Description of benchmarks considered in this work and their implementations.

Regularizer	Reference	Implementation
Baseline	NA	Paszke et al. (2019)
L1	Tibshirani (1996)	Paszke et al. (2019)
L2	Hoerl & Kennard (1970)	Paszke et al. (2019)
Dropout	Hinton et al. (2012)	Paszke et al. (2019)
Batch Norm	Ioffe & Szegedy (2015)	Paszke et al. (2019)
Input Noise	Krizhevsky et al. (2012)	Paszke et al. (2019)
Mixup	Zhang et al. (2018)	Zhang et al. (2018)
TANGOS	This work	Paszke et al. (2019)

G Ablation Study

`TANGOS` is designed with joint application of both regularization on specialization and orthogonalization in mind. Having empirically demonstrated strong overall results, an immediate question is the dynamics of the two regularizers, and how they interact to affect performance. Specifically, we consider the performance gain due to joint regularization effects over applying each regularizer separately.

This includes three separate settings: 1) when the specialization regularizer is applied independently (`SpecOnly`), here we set $\lambda_2 = 0$ and search over $\lambda_1 \in \{1, 10, 100\}$; 2) when the orthogonalization is applied separately (`OrthOnly`), we set $\lambda_1 = 0$ and search over $\lambda_2 \in \{0.1, 1\}$; and lastly 3) when both are applied jointly (`TANGOS`), i.e. searching over $\lambda_1 \in \{1, 10, 100\}$ and $\lambda_2 \in \{0.1, 1\}$. We report the result of the ablation study in Table 6. We empirically observe that the joint effects of both regularizers (i.e. `TANGOS`) is crucial to achieve consistently good performance.

Combining these results with what we observed in Figure 3, we hypothesize that applying just specialization regularization, with no regard for diversity, can inadvertently force the neurons to *attend to overlapping regions* in the input space. Correspondingly, simply enforcing orthogonalization, with no regard for sparsity, will likely result in neurons attending to non-overlapping yet *spurious* regions in the input. Thus, we conclude that the two regularizers have distinct, but complementary, effects that work together to achieve the desired regularization effect.

Table 6: **Ablation study.** Generalization performance on different ablation settings.

	Classification (Mean NLL)			Regression (MSE)		
Dataset	**BR**	**CR**	**HC**	**BC**	**BH**	**WQ**
`NoReg`	0.0726	0.4633	0.3321	0.3343	0.1977	0.6732
`SpecOnly`	0.0742	0.4466	0.3837	0.3099	0.1842	0.6714
`OrthOnly`	0.0716	0.3696	**0.2073**	0.2692	0.1916	0.6529
`TANGOS`	**0.070**	**0.3633**	0.2191	**0.2472**	**0.1826**	**0.6379**

H TANGOS BEHAVIOR ANALYSIS

In this section we apply TANGOS to a simple image classification task using a convolutional neural network (CNN) and provide a qualitative analysis of the behavior of the learned network. This analysis is conducted on the MNIST dataset (LeCun et al., 1998) using the recommended split resulting in 60,000 training and 10,000 validation examples.

In this experiment we train a standard CNN architecture (as described in Table 7) with a penultimate hidden layer of 10 neurons for 10 epochs with Adam optimizer and a learning rate of 0.001. We also apply L2 regularization with weight 0.001. After each epoch, the model is evaluated on the validation set where the epoch achieving the best validation performance stored for further analysis. Two models are trained under this protocol. One model which applied TANGOS to the penultimate hidden layer with $\lambda_1 = 100$, $\lambda_2 = 0.1$ and $M = 25$ and a baseline model which does not apply TANGOS.

Table 7: MNIST Convolutional Neural Network Architecture.

Layer Type	Hyperparameters	Activation Function
Conv2d	Input Channels:1 ; Output Channels:16 ; Kernel Size:5 ; Stride:2 ; Padding:1	ReLU
Conv2d	Input Channels:16 ; Output Channels:32 ; Kernel Size:5 ; Stride:2 ; Padding:1	ReLU
Flatten	Start Dimension:1	
Linear	Input Dimension: 512 ; Output Dimension: 256	ReLU
Linear	Input Dimension: 256 ; Output Dimension: 10	
Linear	Input Dimension: 10 ; Output Dimension: 10	

In this section we examine the gradients of each of the 10 neurons in the penultimate hidden layer with respect to each of the input dimensions of a given image. TANGOS is designed to reward orthogonalization and specialization of these gradient attributions which can be evaluated qualitatively by inspection. In all plots that follow we apply a min-max scaling across all hidden units for fair comparison. Both strong positive and strong negative values for attributions may be interpreted as a latent unit attending to a given input dimension. In Figure 10 we provide results for the baseline model applied to a test image where, in line with similar analyses in previous works such as Crabbé & van der Schaar (2022), we note that the way in which hidden units attend to the input is highly entangled. In contrast to this, in Figure 11, we include the same plot for the TANGOS trained model on the same image. In this case, each hidden unit does indeed produce relatively sparse and orthogonal attributions as desired. These results were consistent across the test images.

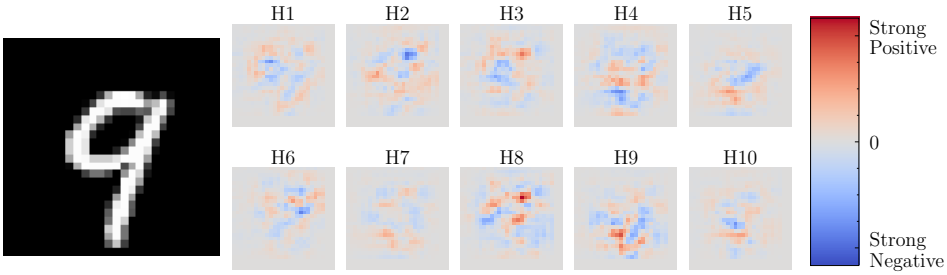


Figure 10: **Without TANGOS Training.** Gradient attributions with respect to each of the 10 hidden neurons. These results suggest significant overlap among the gradient attributions.

We can glean further insight into the TANGOS trained model by examining the role of individual neurons across multiple test images. In Figure 12, we provide the gradient attributions for hidden neuron 5 (H5) from our previous discussion across twelve test images. This neuron appears to discriminate between an open or a closed loop at the lower left of the digit. Indeed this is a key aspect of distinction between the set of digits $\{2, 6, 8, 0\}$ (first row) and $\{9, 5, 3\}$ (second row). We also include digits where this visual feature is less useful as they contain no lower-left loop either open or closed (third row). This hypothesis can be further examined by analyzing the values of these

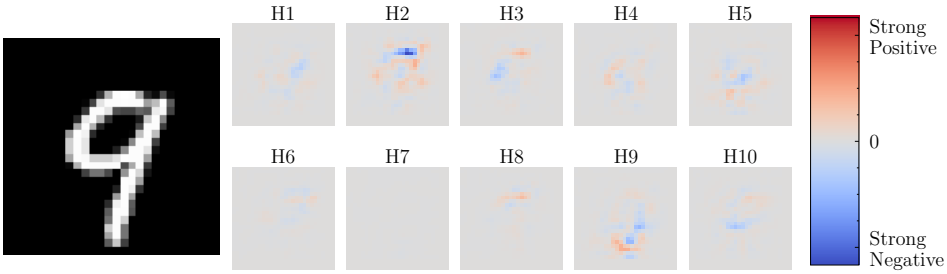


Figure 11: **With TANGOS Training.** TANGOS encourages gradient attributions to be sparse with minimal overlap.

activations. We note that the first two rows typically have higher magnitude with opposite signs while the third row has lower magnitude activations. In Table 8 we summarize the effect of these activation scores on class probabilities by accounting for the weights connecting to each of the ten classes. As one might expect, the weight connections between the hidden neuron and classes on the first row and the second row have opposite signs indicating that neuron 5 does indeed indeed discriminate between these classes.

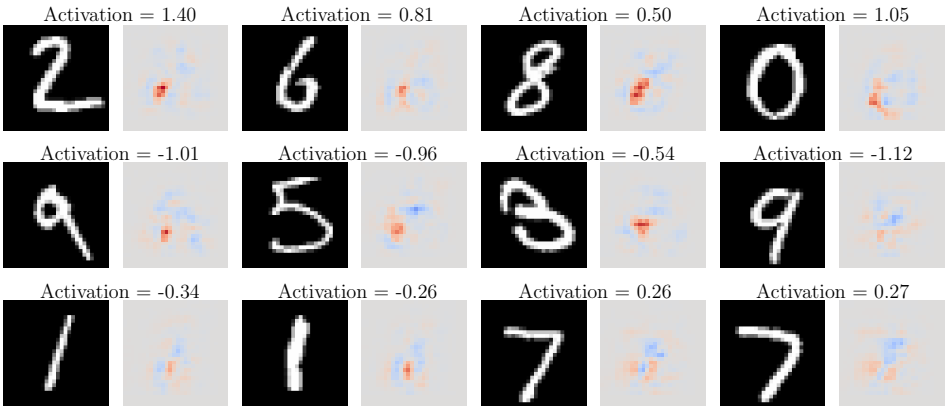


Figure 12: **Hidden Neuron 5.** This neuron attempts to discriminate whether inputs contain a closed loop on the lower left of their digit. Inputs with a closed lower loop incur highly positive activations (first row). Inputs with open lower loops incur highly negative activations (second row). While ambiguous inputs with no lower loop at all tend to produce low magnitude activations (third row).

Table 8: **Neuron 5 Class Weights.** Weights connecting neuron 5 to each of the ten classes and a summary of their combined effect with the neuron activation. The classification influence column provides a categorical indication of the magnitude of the contribution to each class output for a fixed activation magnitude. This is determined by the magnitude of the connecting weight where: Low $\in [0, 0.59]$, Medium $\in [0.59, 1.17]$, and High $\in [1.17, 1.76]$.

Class label	Weight connection	Increases class probability if activation is	Classification influence
0	1.0883	Positive	Medium
1	-0.6826	Negative	Medium
2	1.5298	Positive	High
3	-1.5862	Negative	High
4	-0.2516	Negative	Low
5	-0.6008	Negative	Medium
6	0.9065	Positive	Medium
7	0.3524	Positive	Low
8	0.7362	Positive	Medium
9	-1.7608	Negative	High

I TABULAR ARCHITECTURES AND BOOSTING

While non-neural methods such as XGBoost (Chen & Guestrin, 2016) and CatBoost (Prokhorenkova et al., 2018) are still considered state of the art for tabular data (Grinsztajn et al., 2022), much progress has been made in recent years to close the gap. In this section we provide a proof of concept experiment applying TANGOS to a state of the art transformer architecture for tabular data proposed in Gorishniy et al. (2021). Specifically, this architecture combines a Feature Tokenizer which transforms features into embeddings with a multi-layer Transformer (Vaswani et al., 2017). In our experiments, we use a 3 layer Transformer with a 32 dimensional feature embedding size and 4 attention heads. Following the original paper we use Reglu activations, a hidden layer size of 43 corresponding to a ratio of $\frac{4}{3}$ with the embedding size, Kaiming initialization (He et al., 2015), and AdamW optimizer (Loshchilov & Hutter, 2017). Finally, we apply a learning rate of 0.001. We compare this architecture with and without TANGOS regularization applied which we refer to as "Baseline" and "+ TANGOS" respectively. We set $\lambda_1 = 1$ and $\lambda_2 = 0.01$ which were found to be reasonable default values for specialization and orthogonalization in our experiments in Section 5.

We ran our experiments on the Jannis (Guyon et al., 2019) and Higgs (Baldi et al., 2014) datasets. These are both classification datasets consisting of 83733 and 98050 examples respectively. These datasets were selected as they represent a significant number input examples along with a middling number of input features relative to the other tabular datasets explored in this work (54 and 28 respectively). We follow the experimental protocol of the boosting comparison in Grinsztajn et al. (2022) using the same training, validation and test splits and reporting mean test accuracy over three runs. Therefore we obtain the same results for boosting as reported in that work.

The results of this experiment are reported in Table 9 where we find that TANGOS does indeed have a positive effect on the FT-Transformer performance. While we do not claim that TANGOS regularization results in neural networks that outperform Boosting methods, these results indicate that TANGOS regularization can contribute to closing the gap and may play a key role when combined with other methods as highlighted in Kadra et al. (2021). We believe this to be an important area for future research and, in particular, expect that architecture specific developments of the ideas presented in this work may provide further improvements on the results obtained in this section.

Table 9: **FT-Transformer Architecture and Boosting.** Adding TANGOS regularization can contribute to closing the gap between state of the art tabular architectures and leading boosting methods. We report mean accuracy $\pm$ standard deviation.

Dataset	FT-Transformer		Boosting	
	Baseline	+ TANGOS	XGBoost	CatBoost
Jannis	0.714 ± 0.002	$\mathbf{0.720} \pm 0.000$	0.711 ± 0.000	$\mathbf{0.724} \pm 0.001$
Higgs	0.721 ± 0.002	$\mathbf{0.723} \pm 0.000$	0.717 ± 0.000	$\mathbf{0.728} \pm 0.001$