

---

# Input margins can predict generalization too:

## Supplementary material

---

Anonymous Author(s)

Affiliation

Address

email

### A Constrained margin ablation

This section demonstrates the effect of several hyperparameters on the performance of constrained margins. We analyse the selection of the number of principal components, the number of samples, as well as the effect of clipping.

#### A.1 Number of principal components

In order to better understand the interaction between the selection of the number of principal components and predictive power, we calculate the mean constrained margin using 1 to 50 principal components for all the development set tasks (tasks 1 to 5). We once again make use of 5 000 samples. However, in this case, the first order Taylor approximation is used to reduce the computational burden. The result of this analysis is shown in Figure 1. We indicate the number of principal components selected by the Kneedle algorithm (applied to the principal components in descending order of explained variance) for each task with a star.

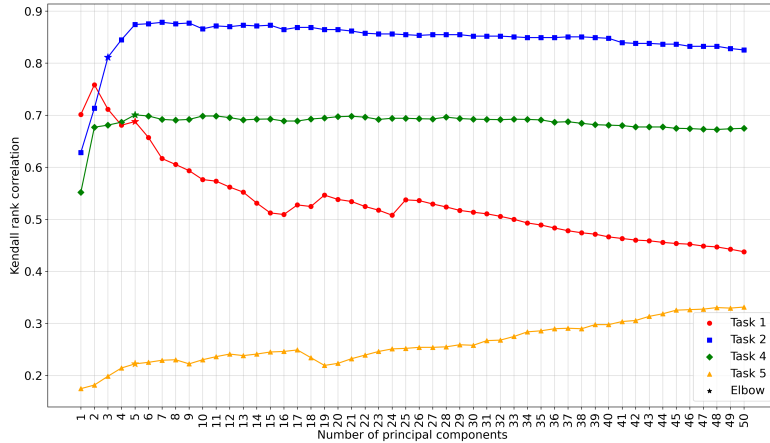


Figure 1: Predictive performance (Kendall’s rank correlation) as a function of the number of principal components for Task 1 (red circles), 2 (blue squares), 4 (green diamonds), and 5 (yellow triangles). The number of principal components reported on per task in the main paper is indicated with a star.

One observes that the elbow method selects the number of components in a near-optimal fashion for Task 1, 2, and 4. Furthermore, the optimal number is generally very low, whereafter the correlation decreases. Task 5 (which is the task for which constrained margins produce the lowest performance)

behaves in a contrary manner, as the ranking correlation increases as the number of components becomes larger. We find that it only reaches a maximum rank correlation of 0.4 at 270 components (not shown here).

## A.2 Number of samples

We have used 5 000 samples to calculate the mean constrained margin for each task (and the same number for all other margin measurements). It is worth determining what effect the number of samples has on the final performance. In Figure 2 we show the Kendall’s rank correlation between mean constrained margin and test accuracy for the development set using 500 to 5 000 samples (using the modified DeepFool algorithm).

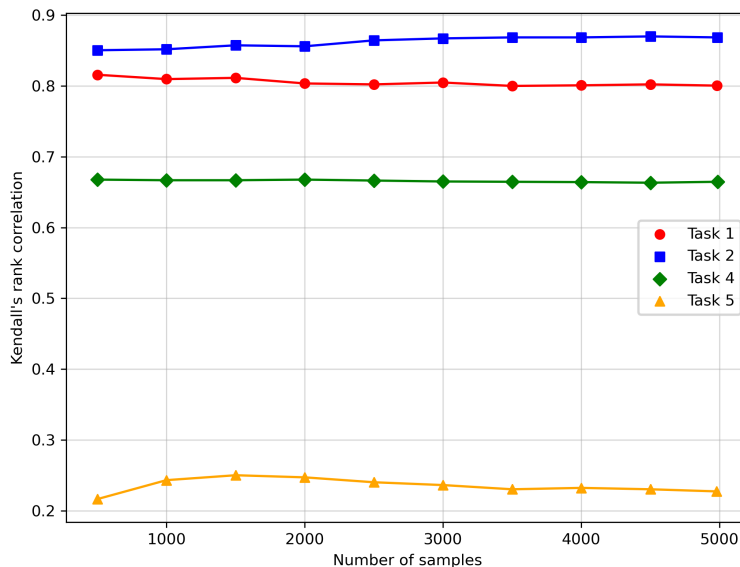


Figure 2: Predictive performance of constrained margins (Kendall’s rank correlation) as a function of the number of principal components for Task 1 (red circles), 2 (blue squares), 4 (green diamonds), and 5 (yellow triangles).

One observes that the rank correlation plateaus rather quickly for most tasks, and one can likely get away with only using 500 to 1 000 samples per model. However, to mitigate any effect that the stochastic selection of training samples can have on the reproducibility of the results, we have chosen to use 5 000 throughout. To this end, we show the number of principal components selected, as well as the number of samples used for each task in Table 1, note that Task 6 and 7 use the maximum number of samples available.

Table 1: Number of principal components and samples used for each task to calculate constrained margins. Tasks 6 and 7 use the maximum number of samples available for the dataset.

Task	Dataset	Components	Samples
1	CIFAR10	5	5 000
2	SVHN	3	5 000
4	CINIC10	5	5 000
5	CINIC10	5	5 000
6	OxFlowers	8	2 040
7	OxPets	3	3 680
8	FMNIST	4	5 000
9	CIFAR10 (augmented)	5	5 000

### 31 A.3 Enforcing bound constraints

32 Our modified DeepFool algorithm (Algorithm 1 in the main paper) enforces bound constraints on  
 33 the sample by clipping  $\hat{x}$  to stay within the minimum and maximum feature values of the dataset  
 34 after each step (see line 10 of the algorithm). Since the original images have pixel values between  
 35 0 and 1, the z-normalised data has a strict lower and upper bound. Allowing  $\hat{x}$  to deviate outside  
 36 these values will produce boundaries that cannot exist in practice. Given that the original DeepFool  
 37 algorithm does not include any form of bound constraints, we analyse the effect clipping has on the  
 38 performance of constrained and standard input margins. Table 2 shows the Kendall’s rank correlation  
 39 per task with and without clipping.

Table 2: Kendall’s rank correlation between mean margin and test accuracy for constrained and standard input margins with and without clipping.

Task	Constrained		Input	
	Clipped	Unclipped	Clipped	Unclipped
1	0.8040	<b>0.8088</b>	<b>-0.1235</b>	-0.1239
2	<b>0.8672</b>	0.8463	<b>0.6730</b>	0.6716
4	<b>0.6651</b>	0.6576	<b>0.2224</b>	0.2163
5	<b>0.2292</b>	0.1984	<b>-0.0367</b>	-0.0655
6	<b>0.8008</b>	0.7990	<b>-0.2190</b>	-0.2194
7	0.5027	<b>0.5133</b>	0.3144	<b>0.3162</b>
8	<b>0.6004</b>	0.4672	-0.1849	<b>-0.1521</b>
9	<b>0.8145</b>	0.8024	<b>0.1089</b>	0.1048
Average	<b>0.6605</b>	0.6366	<b>0.0943</b>	0.0935

40 It is evident that clipping has little effect on standard input margins – this makes sense, given that  
 41 samples on the decision boundary are generally very close to the training sample. However, in  
 42 the case of constrained margins, we observe that clipping improves the results in most cases, and  
 43 especially so for Task 8. This demonstrates that enforcing the bound constraints is a useful inclusion.

## 44 B Extended margin comparison

45 This section contains additional results relevant to Section 4.2 in the main paper. We compare using  
 46 the first-order Taylor approximation to DeepFool, and also the selection of hidden layers.

### 47 B.1 Comparison of Taylor and DeepFool

48 For constrained and standard input margins, we have experimented with using both the first-order  
 49 Taylor approximation as well as the DeepFool method to calculate the distance to the decision  
 50 boundary. Here we do a full comparison between the different methods. Tables 3 and 4 show the  
 51 predictive performance of all the variations using Kendall’s rank correlation and CMI, respectively.

Table 3: Kendall’s rank correlation between mean margin and test accuracy for constrained, standard input, and hidden margins for the PGDL dataset. DF indicates margins calculated using the DeepFool algorithm, while Taylor indicates the first-order Taylor approximation.

Task	Constrained (DF)	Constrained (Taylor)	Input (DF)	Input (Taylor)	Hidden 1st (Taylor)	Hidden all (Taylor)
1	<b>0.8040</b>	0.6991	-0.1235	0.0265	0.5794	0.7825
2	<b>0.8672</b>	0.8281	0.6730	0.6841	0.7037	0.8281
4	0.6651	0.6966	0.2224	0.6251	<b>0.7958</b>	0.2707
5	0.2292	0.2381	-0.0367	0.3571	<b>0.5427</b>	0.1329
6	<b>0.8008</b>	0.6753	-0.2190	-0.1351	0.4427	0.2839
7	<b>0.5027</b>	0.4192	0.3144	0.3215	0.3623	0.3481
8	<b>0.6004</b>	0.3419	-0.1849	-0.1233	-0.0656	0.1859
9	<b>0.8145</b>	0.7258	0.1089	0.1573	0.7097	0.4556
Average	<b>0.6605</b>	0.5780	0.0943	0.2392	0.5088	0.4110

Table 4: Conditional Mutual Information between mean margin and generalization gap for constrained, standard input, and hidden margins for the PGDL dataset. DF indicates margins calculated using the DeepFool algorithm, while Taylor indicates the first-order Taylor approximation.

Task	Constrained (DF)	Constrained (Taylor)	Input (DF)	Input (Taylor)	Hidden 1st (Taylor)	Hidden all (Taylor)
1	<b>39.37</b>	23.77	01.36	00.07	09.40	29.78
2	<b>51.12</b>	43.37	05.01	06.12	37.74	32.23
4	21.48	22.18	03.49	14.95	<b>34.73</b>	00.79
5	05.12	05.42	00.73	08.46	<b>19.11</b>	01.55
6	<b>30.52</b>	10.65	01.77	00.57	04.24	01.36
7	12.60	<b>12.91</b>	02.16	01.47	05.04	05.81
8	<b>13.54</b>	03.70	00.68	00.70	00.36	00.91
9	<b>51.46</b>	18.61	00.80	00.29	23.74	04.75
Average	<b>28.15</b>	17.58	02.00	04.08	16.80	09.65

One observes that constrained margins are significantly improved if the more accurate DeepFool method is applied, while standard input margins actually perform worse when using DeepFool. Due to the high dimensionality of hidden layers, it is computationally infeasible to apply DeepFool to hidden margins. However, note that constrained margins calculated using the Taylor approximation still outperform hidden margins calculated using the Taylor approximation.

## B.2 Comparison of hidden-layer selection

As mentioned in Section 4.1, there are three different methods that have previously been used to select relevant hidden layers when calculating hidden margins. In Table 5 we compare all variations: Using only the first ('First') or last ('Last') layer [6], the average margin over three equally spaced layers ('Equally spaced') [11], and average over all layers ('All') [17].

Table 5: Kendall's rank correlation between mean hidden margin and test accuracy using different hidden layer selections.

Task	First	Last	Equally spaced	All
1	0.5794	<b>0.8294</b>	0.4688	0.7825
2	0.7037	0.7135	<b>0.8686</b>	0.8281
4	<b>0.7958</b>	0.1066	0.6778	0.2707
5	<b>0.5427</b>	0.0089	0.2798	0.1329
6	<b>0.4427</b>	0.2365	0.2211	0.2839
7	<b>0.3623</b>	0.3179	0.3055	0.3481
8	-0.0656	<b>0.2068</b>	-0.0944	0.1859
9	<b>0.7097</b>	0.3831	0.4677	0.4556
Average	<b>0.5088</b>	0.3503	0.3994	0.4110

It is clear that the selection of hidden layers plays a significant role in the overall performance of hidden margins, and we observe a large variation per task between the different methods. While we have used the two best-performing methods as a benchmark to compare with in the main paper ('First' and 'All'), this biases the comparison in favour of hidden margins, as there is no method at present to determine a priori which hidden layer selection will perform best for a given task.

## 67 C Derivation of constrained margins (Equation (5))

68 This section uses the same notation as defined in Section 3.2 of the main paper. We first describe  
 69 the standard linear approximation of the margin following Huang et al. [32], before deriving the  
 70 constrained margin of Equation (5) as numbered in the main paper.

71 Any function  $f$  can be approximated with its differential at point  $\mathbf{x}$  using

$$\hat{f}(\mathbf{x} + \mathbf{d}) = f(\mathbf{x}) + H\mathbf{d} \quad (1)$$

$$\text{where } H = \nabla_{\mathbf{x}} f(\mathbf{x}) \quad (2)$$

72 that is, the Jacobian of the output with regard to the input features at point  $\mathbf{x}$ . We aim to find the  
 73 smallest  $\|\mathbf{d}\|$  for some norm  $\|\cdot\|$  such that  $f(\mathbf{x}) \neq f(\mathbf{x} + \mathbf{d})$ , or

$$f_j(\mathbf{x} + \mathbf{d}) \geq f_i(\mathbf{x} + \mathbf{d}) \quad (3)$$

74 If we approximate  $f(\cdot)$  with  $\hat{f}(\cdot)$ , this implies:

$$\begin{aligned} f_j(\mathbf{x}) + \nabla_{\mathbf{x}} f_j(\mathbf{x}) \cdot \mathbf{d} &\geq f_i(\mathbf{x}) + \nabla_{\mathbf{x}} f_i(\mathbf{x}) \cdot \mathbf{d} \\ \implies (\nabla_{\mathbf{x}} f_j(\mathbf{x}) - \nabla_{\mathbf{x}} f_i(\mathbf{x})) \cdot \mathbf{d} &\geq f_i(\mathbf{x}) - f_j(\mathbf{x}) \end{aligned} \quad (4)$$

75 where  $\nabla_{\mathbf{x}} f_k(\mathbf{x})$  is the gradient vector of the  $k^{th}$  output value of  $f$  with regard to input  $\mathbf{x}$ . Then, as  
 76 shown in [32], the maximum  $\|\mathbf{d}\|$  will be at:

$$\|\mathbf{d}\| = \frac{f_i(\mathbf{x}) - f_j(\mathbf{x})}{\|\nabla_{\mathbf{x}} f_j(\mathbf{x}) - \nabla_{\mathbf{x}} f_i(\mathbf{x})\|_*} \quad (5)$$

77 where  $\|\cdot\|$  and  $\|\cdot\|_*$  are dual norms. Specifically, if  $\|\cdot\|$  is the  $L2$  norm, then:

$$\mathbf{d} = \frac{f_i(\mathbf{x}) - f_j(\mathbf{x})}{\|\nabla_{\mathbf{x}} f_j(\mathbf{x}) - \nabla_{\mathbf{x}} f_i(\mathbf{x})\|_2^2} (\nabla_{\mathbf{x}} f_j(\mathbf{x}) - \nabla_{\mathbf{x}} f_i(\mathbf{x})) \quad (6)$$

78

$$\text{and } \|\mathbf{d}\|_2 = \frac{f_i(\mathbf{x}) - f_j(\mathbf{x})}{\|\nabla_{\mathbf{x}} f_j(\mathbf{x}) - \nabla_{\mathbf{x}} f_i(\mathbf{x})\|_2} \quad (7)$$

79 Equations (6) and (7) provide the standard linear approximation of the margin as used by various  
 80 authors [11, 16].

81 The derivation process for constrained margins is identical – it is only the calculation of the Jacobian  
 82 that differs, as the gradient is calculated with regard to the transformed features rather than the  
 83 original features. Note that the size and direction of the update are calculated with regard to the  
 84 transformed features but the actual step is given in the original feature space.

85 Let  $P_m$  be the matrix constructed from the first  $m$  principal components as column vectors:

$$P_m = [\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_m]^T \quad (8)$$

86 The new parameterisation  $\mathbf{x}'$  of any point  $\mathbf{x}$  is then approximated by:

$$\mathbf{x}' = P_m \mathbf{x} \quad (9)$$

87 where  $\mathbf{x}$  is a column vector. Let  $B_m$  be the pseudoinverse of  $P_m$ . Since the full  $P_N$ , when all  
 88 components are selected, is orthogonal,  $(P_N)^{-1} = (P_N)^T$  and  $B_m$  then equals the first  $m$  rows of  
 89  $(P_N)^T$ . Then we can express each individual term  $x_k$  in terms of the elements of  $\mathbf{x}'$ :

$$x_k \approx \sum_s b_{k,s} x'_s = \sum_s p_{s,k} x'_s \quad (10)$$

90 with  $b_{r,c}$  the element in  $B_m$  at row  $r$  and column  $c$ , and  $p_{r,c}$  at the same position in  $P_m$ .

91 Let the Jacobian as used in Equations (1) to (7) be given by

$$H_{r,c} = \left. \frac{\delta f_r}{\delta x_c} \right|_{\mathbf{x}} \quad (11)$$

92 Then, assuming  $X$  input features we can use the existing  $N \times X$  Jacobian, to calculate the new  
 93  $N \times m$  Jacobian in terms of  $\mathbf{x}'$  rather than  $\mathbf{x}$ , using the chain rule:

$$\begin{aligned} H'_{r,c} &= \frac{\delta f_r(\mathbf{x})}{\delta x'_c} \\ &= \frac{\delta f_r(\mathbf{x})}{\delta x_1} \cdot \frac{dx_1}{dx'_c} + \dots + \frac{\delta f_r(\mathbf{x})}{\delta x_n} \cdot \frac{dx_n}{dx'_c} \\ &= H_{r,1}p_{c,1} + \dots + H_{r,n}p_{c,n} \\ &= \nabla_{\mathbf{x}} f_r(\mathbf{x}) \cdot \mathbf{p}_c \end{aligned} \quad (12)$$

94 where  $\mathbf{p}_c$  is the  $c^{th}$  row of  $P$ , transposed. Then each row  $\mathbf{h}'_r$  of the new Jacobian in terms of  $\mathbf{x}'$  is  
 95 given by

$$\mathbf{h}'_r = \nabla_{\mathbf{x}} f_r(\mathbf{x}) P^T \quad (13)$$

96 Equation (13) which can be used directly in the adjusted version of Equations (6) and (7), such that

$$\begin{aligned} \mathbf{d} &= \frac{f_i(\mathbf{x}) - f_j(\mathbf{x})}{\|\mathbf{h}'_j - \mathbf{h}'_i\|_2^2} (\mathbf{h}'_j - \mathbf{h}'_i) \\ &= \frac{f_i(\mathbf{x}) - f_j(\mathbf{x})}{\|(\nabla_{\mathbf{x}} f_j(\mathbf{x}) - \nabla_{\mathbf{x}} f_i(\mathbf{x})) P^T\|_2^2} (\nabla_{\mathbf{x}} f_j(\mathbf{x}) - \nabla_{\mathbf{x}} f_i(\mathbf{x})) P^T \end{aligned} \quad (14)$$

$$\text{and } \|\mathbf{d}\|_2 = \frac{f_i(\mathbf{x}) - f_j(\mathbf{x})}{\|[\nabla_{\mathbf{x}} f_j(\mathbf{x}) - \nabla_{\mathbf{x}} f_i(\mathbf{x})] P^T\|_2} \quad (15)$$

97 In effect, we start at point  $\mathbf{x}$  (the only point we have a model output for), and then use the gradient in  
 98 the lower dimensional space to find the minimal distance  $\|\mathbf{d}\|_2$ .