JMLR: Workshop and Conference Proceedings 44 (2015) 242-259 NIPS 2015

The 1st International Workshop "Feature Extraction: Modern Questions and Challenges"

Modular Autoencoders for Ensemble Feature Extraction

Henry W J Reeve HENRYWJREEVE@GMAIL.COM
School of Computer Science
The University of Manchester
Manchester, UK

Gavin Brown GAVIN.BROWN@MANCHESTER.AC.UK
School of Computer Science
The University of Manchester
Manchester, UK

Editor: Afshin Rostamizadeh

Abstract

We introduce the concept of a Modular Autoencoder (MAE), capable of learning a set of diverse but complementary representations from unlabelled data, that can later be used for supervised tasks. The learning of the representations is controlled by a trade off parameter, and we show on six benchmark datasets the optimum lies between two extremes: a set of smaller, independent autoencoders each with low capacity, versus a single monolithic encoding, outperforming an appropriate baseline. In the present paper we explore the special case of linear MAE, and derive an SVD-based algorithm which converges several orders of magnitude faster than gradient descent.

Keywords: Modularity, Autoencoders, Diversity, Unsupervised, Ensembles

1. Introduction

In a wide variety of Machine Learning problems we wish to extract information from high dimensional data sets such as images or documents. Dealing with high dimensional data creates both computational and statistical challenges. One approach to overcoming these challenges is to extract a small set of highly informative features. These features may then be fed into a task dependent learning algorithm. In representation learning these features are learnt directly from the data (Bengio et al., 2013).

We consider a *modular* approach to representation learning. Rather than extracting a single set of features, we extract multiple sets of features. Each of these sets of features is then fed into a separate learning module. These modules may then be trained independently, which addresses both computational challenges, by being easily distributable, and statistical challenges, since each module is tuned to just a small set of features. The outputs of the different classifiers are then combined, giving rise to a classifier ensemble.

Ensemble methods combine the outputs of a multiplicity of models in order to obtain an enriched hypothesis space whilst controlling variance (Friedman et al., 2001). In this work we shall apply ensemble methods to representation learning in order to extract several subsets of features for an effective classifier ensemble. Successful ensemble learning results from a fruitful trade-off between accuracy and diversity within the ensemble. Diversity

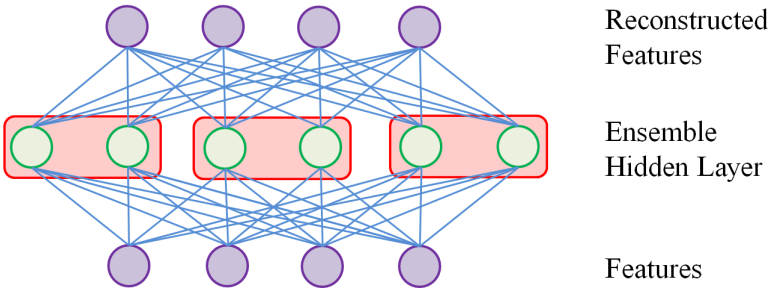

Figure 1: A Modular Autoencoder (MAE).

is typically encouraged, either through some form of randomisation, or by encouraging diversity through supervised training (Brown et al., 2005).

We investigate an unsupervised approach to learning a set of diverse but complementary representations from unlabelled data. As such, we move away from the recent trend towards coupled dimensionality reduction in which the tasks of feature extraction and supervised learning are performed in unison Gönen (2014); Mohri et al. (2015). Whilst coupled dimensionality reduction has been shown to improve accuracy for certain classification tasks Gönen (2014), the unsupervised approach allows us to use unlabelled data to learn a transferable representation which may be used on multiple tasks without the need for retraining Bengio et al. (2013).

We show that one can improve the performance of a classifier ensemble by first learning a diverse collection of modular feature extractors in a purely unsupervised way (see Section 4) and then training a set of classifiers independently. Features are extracted using a Modular Autoencoder trained to simultaneously minimise reconstruction error and maximise diversity amongst reconstructions (see Section 2). Though the MAE framework is entirely general to any activation function, in the present paper we focus on the linear case and provide an efficient learning algorithm that converges several orders of magnitude faster than gradient descent (see Section 3). The training scheme involves a hyper-parameter λ. We provide an upper bound on λ, enabling a meaningful trade off between reconstruction error and diversity (see Section 2.2).

2. Modular Autoencoders

A *Modular Autoencoder* consists of an ensemble $\mathcal{W} = \{(\boldsymbol{A}_i, \boldsymbol{B}_i)\}_{i=1}^{M}$ consisting of M autoencoder modules $(\boldsymbol{A}_i, \boldsymbol{B}_i)$, where each module consists of an encoder map $\boldsymbol{B}_i : \mathbb{R}^D \to \mathbb{R}^H$ from a D-dimensional feature space $\mathbb{R}^D$ to an H-dimensional representation space $\mathbb{R}^H$, and a decoder map $\boldsymbol{A}_i : \mathbb{R}^D \to \mathbb{R}^H$. For reasons of brevity we focus on the linear case, where $\boldsymbol{A}_i \in \mathbb{M}_{D\times H}(\mathbb{R})$ and $\boldsymbol{B}_i \in \mathbb{M}_{H\times D}(\mathbb{R})$ are matrices. See Figure 1.

In order to train our Modular Autoencoders $\mathcal{W}$ we introduce the following loss function

$$L_\lambda(\mathcal{W}, \boldsymbol{x}) := \frac{1}{M}\sum_{i=1}^{M} \overbrace{\left|\left|\boldsymbol{A}_i\boldsymbol{B}_i\boldsymbol{x} - \boldsymbol{x}\right|\right|^2}^{\text{reconstruction error}} - \lambda \cdot \frac{1}{M}\sum_{i=1}^{M} \overbrace{\left|\left|\boldsymbol{A}_i\boldsymbol{B}_i\boldsymbol{x} - \frac{1}{M}\sum_{j=1}^{M}\boldsymbol{A}_j\boldsymbol{B}_j\boldsymbol{x}\right|\right|^2}^{\text{diversity}}, \quad (1)$$

for feature vectors $\boldsymbol{x} \in \mathbb{R}^D$. The loss function $L_\lambda(\mathcal{W}, \boldsymbol{x})$ is inspired by (but not identical to) the Negative Correlation Learning approach of by Liu and Yao for training supervised ensembles of neural networks (Liu and Yao, 1999)[1]. The first term corresponds to the squared reconstruction error typically minimised by Autoencoders (Bengio et al., 2013). The second term encourages the reconstructions to be diverse, with a view to capturing different factors of variation within the training data. The hyper-parameter λ, known as the diversity parameter, controls the degree of emphasis placed upon these two terms. We discuss its properties in Sections 2.1 and 2.2. Given a data set $\mathcal{D} \subset \mathbb{R}^D$ we train a Modular Autoencoder to minimise the error $E_\lambda(\mathcal{W}, \mathcal{D})$, the loss function $L_\lambda(\mathcal{W}, \boldsymbol{x})$ averaged across the data $\boldsymbol{x} \in \mathcal{D}$.

2.1 Between two extremes

To understand the role of the diversity parameter λ we first look at the two extremes of $\lambda = 0$ and $\lambda = 1$. If $\lambda = 0$ then no emphasis is placed upon diversity. Consequently $L_0(\mathcal{W}, \boldsymbol{x})$ is precisely the average squared error of the individual modules $(\boldsymbol{A}_i, \boldsymbol{B}_i)$. Since there is no interaction term, minimising $L_0(\mathcal{W}, \boldsymbol{x})$ over the training data is equivalent to training each of the auto-encoder modules independently, to minimise squared error. Hence, in the linear case $E_0(\mathcal{W}, \mathcal{D})$ is minimised by taking each $\boldsymbol{B}_i$ to be the projection onto the first H principal components of the data covariance (Baldi and Hornik, 1989).

If $\lambda = 1$ then, by the Ambiguity Decomposition (Krogh et al., 1995),

$$L_1(\mathcal{W}, \boldsymbol{x}) = \left|\left| \frac{1}{M} \sum_{i=1}^{M} \boldsymbol{A}_i \boldsymbol{B}_i \boldsymbol{x} - \boldsymbol{x} \right|\right|^2 .$$

Hence, minimising $L_1(\mathcal{W})$ is equivalent to minimising squared error for a single large Autoencoder $(\boldsymbol{A}, \boldsymbol{B})$ with an $M \cdot H$-dimensional hidden layer, where $\boldsymbol{B} = [\boldsymbol{B}_1^T, \cdots, \boldsymbol{B}_M^T]^T$ and $\boldsymbol{A} = M^{-1}[\boldsymbol{A}_1, \cdots, \boldsymbol{A}_M]$.

Consequently, moving λ between 0 and 1 corresponds to moving from training each of our autoencoder modules independently through to training the entire network as a single monolithic autoencoder.

2.2 Bounds on the diversity parameter

The diversity parameter λ may be set by optimising the performance of a task-specific system using the extracted sets of features on a validation set. Theorem 1 shows that the search region may be restricted to the closed unit interval $[0, 1]$.

Theorem 1 *Suppose we have a data set $\mathcal{D}$. The following dichotomy holds:*

- *If $\lambda \leq 1$ then* $\inf E_\lambda(\mathcal{W}, \mathcal{D}) \geq 0$.
- *If $\lambda > 1$ then* $\inf E_\lambda(\mathcal{W}, \mathcal{D}) = -\infty$.

In both cases the infimums range over possible parametrisations for the ensemble $\mathcal{W}$.

Moreover, if the diversity parameter $\lambda > 1$ there exist ensembles $\mathcal{W}$ with arbitrarily low error $E_\lambda(\mathcal{W}, \mathcal{D})$ and arbitrarily high average reconstruction error.

1. See the Appendix for details.

Theorem 1 is a special case of Theorem 3, which is proved the appendix.

3. An Efficient Algorithm for Training Linear Modular Autoencoders

One method to minimise the error $E_\lambda(\mathcal{W}, \mathcal{D})$ would be to apply some form of gradient descent. However, Linear Modular Autoencoders we can make use of the Singular Value Decomposition to obtain a fast iterative algorithm for minimising the error $E_\lambda(\mathcal{W}, \mathcal{D})$ (see Algorithm 1).

Algorithm 1 Backfitting for Linear Modular Autoencoders

Inputs: $D \times N$ data matrix $\boldsymbol{X}$, diversity parameter λ, number of hidden nodes per module H, number of modules M, maximal number of epochs T,
Randomly generate $\{(\boldsymbol{A}_i, \boldsymbol{B}_i)\}_{i=1}^{M}$ and set $\Sigma \leftarrow \boldsymbol{X}\boldsymbol{X}^T$,
for t $= 1$ **to** T **do**
  for i $= 1$ **to** M **do**
    $\boldsymbol{Z}_i \leftarrow M^{-1} \sum_{j \neq i} \boldsymbol{A}_j \boldsymbol{B}_j$
    $\boldsymbol{\Phi} \leftarrow (\boldsymbol{I}_D - \lambda \cdot \boldsymbol{Z}_i)\, \Sigma\, (\boldsymbol{I}_D - \lambda \cdot \boldsymbol{Z}_i)^T$, where $\boldsymbol{I}_D$ denotes the $D \times D$ identity matrix.
    $\boldsymbol{A}_i \leftarrow [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_H]$, where $\{\boldsymbol{u}_1, \cdots, \boldsymbol{u}_H\}$ are the top eigenvectors of $\boldsymbol{\Phi}$.
    $\boldsymbol{B}_i \leftarrow (1 - \lambda \cdot (M-1)/M)^{-1} \cdot \boldsymbol{A}_i{}^T (\boldsymbol{I}_D - \lambda \cdot \boldsymbol{Z}_i)$
  end for
end for
return Decoder-Encoder pairs $(\boldsymbol{A}_i, \boldsymbol{B}_i)_{i=1}^{M}$

Algorithm 1 is a simple greedy procedure reminiscent of the back-fitting algorithm for additive models (Friedman et al., 2001). Each module is optimised in turn, leaving the parameters for the other modules fixed. The error $E_\lambda(\mathcal{W}, \mathcal{D})$ decreases every epoch until a critical point is reached.

Theorem 2 *Suppose that $\Sigma = \boldsymbol{X}\boldsymbol{X}^T$ is of full rank. Let $(\mathcal{W}_t)_{t=1}^{T}$ be a sequence of parameters obtained by Algorithm 1. For every epoch $t = \{1, \cdots, T\}$, we have $E_\lambda(\mathcal{W}_{t+1}, \mathcal{D}) < E_\lambda(\mathcal{W}_t, \mathcal{D})$, unless $\mathcal{W}_t$ is a critical point for $E_\lambda(\cdot, \mathcal{D})$, in which case $E_\lambda(\mathcal{W}_{t+1}, \mathcal{D}) \leq E_\lambda(\mathcal{W}_t, \mathcal{D})$.*

Theorem 2 justifies the procedure in Algorithm 1. The proof is given in Appendix B. We compared Algorithm 1 with (batch) gradient descent on an artificial data set consisting of 1000 data points randomly generated from a Gaussian mixture data set consisting of equally weighted spherical Gaussians with standard deviation 0.25 and a mean drawn from a standard multivariate normal distribution. We measured the time for the cost to stop falling by at least $\epsilon = 10^{-5}$ per epoch for both Algorithm 1 and (batch) gradient descent. The procedure was repeated ten times. The two algorithms performed similarly in terms of minimum cost attained, with Algorithm 1 attaining slightly lower costs on average. However, as we can see from Table 1, Algorithm 1 converged several orders of magnitude faster than gradient descent.

	Algorithm 1	Gradient Descent	Speed up
Minimum	0.1134 s	455.2 s	102.9×
Mean	1.4706 s	672.9 s	1062.6×
Maximum	4.9842 s	1871.5 s	6685.4×

Table 1: Convergence times for Algorithm 1 and batch gradient descent.

4. Empirical results

In this section we demonstrate the efficacy of Modular Autoencoders for extracting useful sets features for classification tasks. In particular, we demonstrate empirically that we can improve the performance of a classifier ensemble by first learning a diverse collection of modular feature extractors in an unsupervised way.

Our methodology is as follows. We take a training data set $\mathcal{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$ consisting of pairs of feature vectors $\boldsymbol{x}_n$ and class labels y_n. The data set $\mathcal{D}$ is pre-processed so each of the features have zero mean. We first train a Modular Autoencoder $\mathcal{W} = (\boldsymbol{A}_i, \boldsymbol{B}_i)$. For each module i we take C_i to be the 1-nearest neighbour classifier with the data set $\mathcal{D}_i = \{(\boldsymbol{B}_i\boldsymbol{x}_n, y_n)\}_{n=1}^N$. The combined prediction of the ensemble on a test point $\boldsymbol{x}$ is defined by taking a modal average of the class predictions $\{C_i(\boldsymbol{B}_i\boldsymbol{x})\}_{i=1}^M$.

We use a collection of six image data sets from Larochelle et al. (2007), *Basic, Rotations, Background Images* and *Background Noise* variants of *MNIST* as well as *Rectangles* and *Convex*. In each case we use a Modular Autoencoder consisting of ten modules ($M = 10$), each consisting of ten hidden nodes ($H = 10$). The five-fold cross-validated test error is shown as a function of the diversity parameter λ. We contrast with a natural baseline approach *Bagging Autoencoders* (BAE) in which we proceed as described, but the modules $(\boldsymbol{A}_i, \boldsymbol{B}_i)$ are trained independently on bootstrapped samples from the data. In all cases, as the diversity parameter increases from zero the test error for features extracted using Modular Autoencoders falls well below the level attained by Bagging Autoencoders. As $\lambda \to 1$ the ensemble error begins to rise, sometimes sharply.

5. Understanding Modular Autoencoders

In this section we analyse the role of encouraging diversity in an unsupervised way with Modular Autoencoders and the impact this has upon supervised classification.

5.1 A more complex decision boundary

We begin by considering a simple two-dimensional example consisting of a Gaussian mixture with three clusters. In this setting we use a Linear Modular Autoencoder consisting of two modules, each with a single hidden node, so each of the feature extractors is simply a projection onto a line. We use a linear Softmax classifier on each of the extracted features. The probabilistic outputs of the individual classifiers are then combined by taking the mean average. The predicted label is defined to be the one with the highest probability. Once again we observe the same trend as we saw in Section 4 - encouraging diversity leads to a substantial drop in the test error of our ensemble, with a test error of $21.3 \pm 1.3\%$ for $\lambda = 0$ and $12.8 \pm 1.0\%$ for $\lambda = 0.5$.

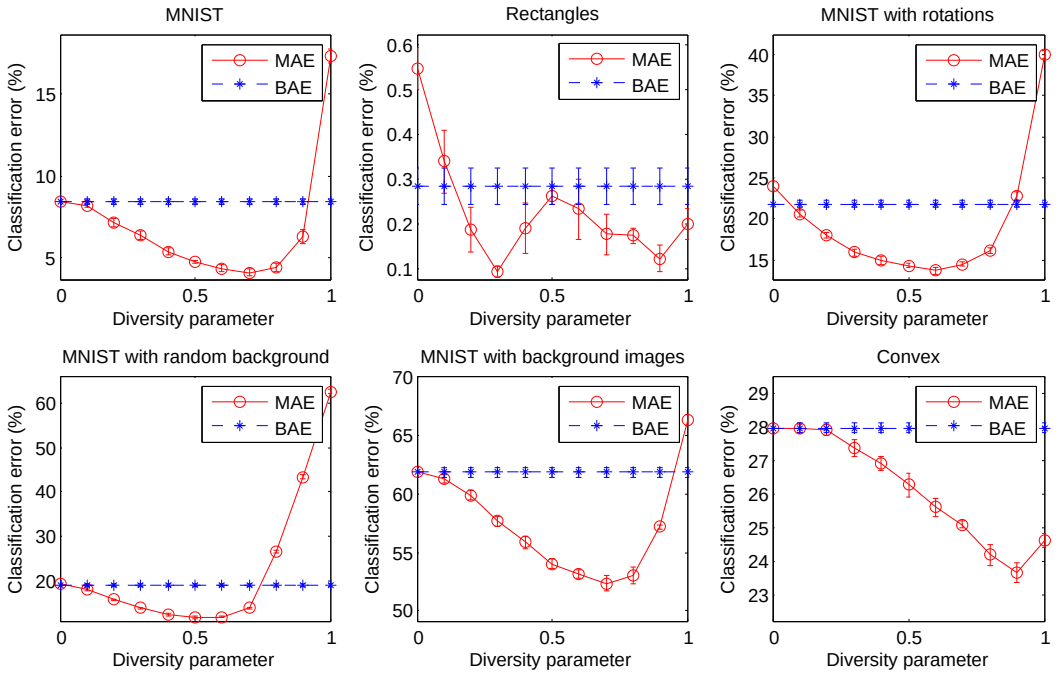

Figure 2: Test error for Modular Autoencoders (MAE) and Bagging Autoencoders (BAE).

To see why this is the case we contrast the features extracted when $\lambda = 0$ with those extracted when $\lambda = 0.5$. Figure 3 shows the projection of the class densities onto the two extracted features when $\lambda = 0$. No emphasis is placed upon diversity the two modules are trained independently to maximise reconstruction. Hence, the features extract identical information and there is no ensemble gain. Figure 5 shows the resultant decision boundary; a simple linear decision boundary based upon a single one-dimensional classification.

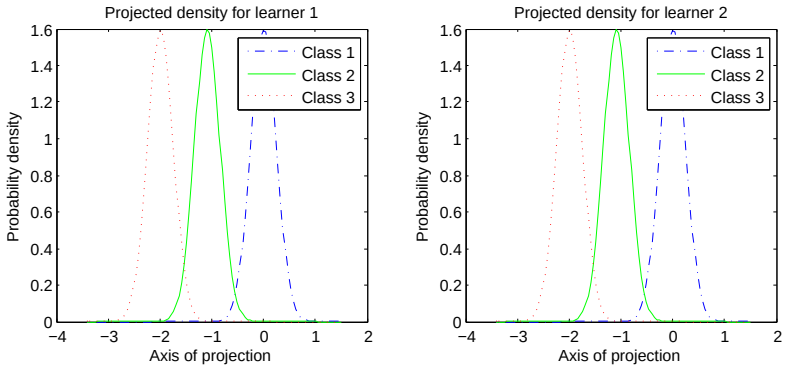

Figure 3: Projected class densities with $\lambda = 0$.

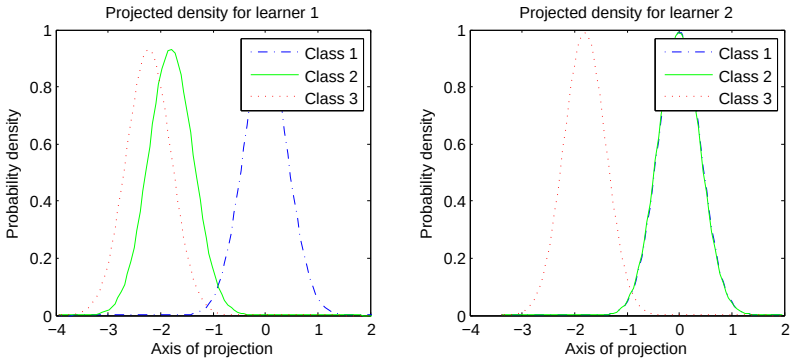

Figure 4: Projected class densities with $\lambda = 0.5$.

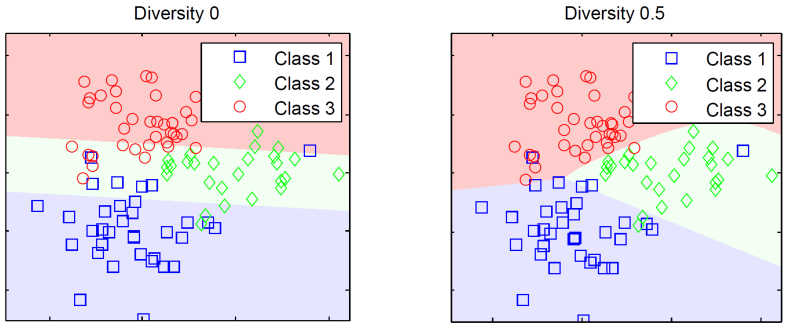

Figure 5: The decision boundary for $\lambda = 0$ (left) and $\lambda = 0.5$ (right).

In contrast, when $\lambda = 0.5$ the two features yield diverse and complementary information. As we can see from Figure 4, one feature separates class 1 from classes 2 and 3, and the other separates class 3 from classes 1 and 2. As we can see from the right of Figure 5, the resulting decision boundary accurately reflects the true class boundaries, despite being based upon two independently trained one-dimensional classifiers. This leads to the reduction in test error for $\lambda = 0.5$.

In general, Modular Autoencoders trained with the loss function defined in (1) extract diverse and complementary sets of features, whilst reflecting the main factors of variation within the data. Simple classifiers may be trained independently based upon these sets of features, so that the combined ensemble system gives rise to a complex decision boundary.

5.2 Diversity of feature extractors

In this Section we give further insight into the effect of diversity upon Modular Autoencoders. We return to the empirical framework of Section 4. Figure 6 plots two values test error for features extracted with Linear Modular Autoencoders. We plot both the average individual error of the classifiers (without ensembling the outputs) and the test error of the ensemble. In every case the average individual error rises as the diversity parameter moves

away from zero. Nonetheless, the ensemble error falls as the diversity parameter increases (at least initially).

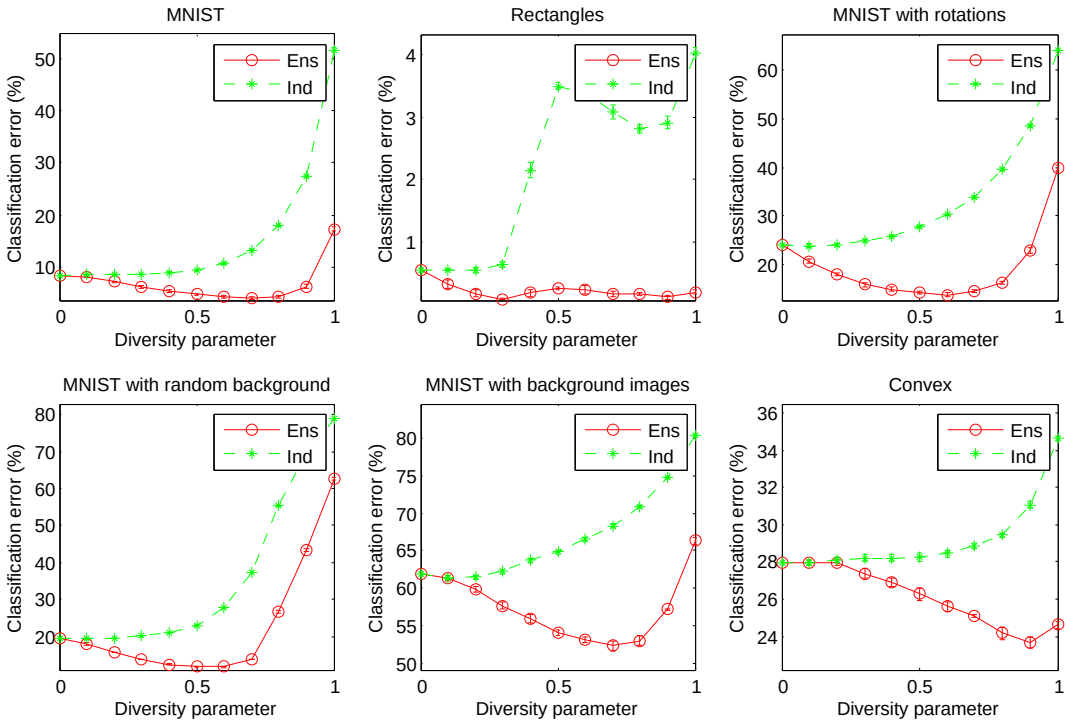

Figure 6: Test error for the ensemble system (Ens) and the average individual error (Ind) . Note that as the diversity parameter λ increases, the individual modules sacrifice their own performance for the good of the overall set of modules - the average error rises, while the ensemble error falls.

To see why the ensemble error falls whilst the average individual error rises we consider the metric structure of the different sets of extracted features. To compare the metric structure captured by different feature extractors, both with one another, and with the original feature space, we use the concept of *distance correlation* introduced by (Székely et al., 2007).

Given a feature extractor map F (such as $\boldsymbol{x} \mapsto \boldsymbol{B}_i\boldsymbol{x}$) we compute $D\left(F, \mathcal{D}\right)$, the distance correlation based upon the pairs $\{(F(x), x) : x \in \mathcal{D}\}$. The quantity $D\left(F, \mathcal{D}\right)$ tells us how faithfully the extracted feature space for a feature map F captures the metric structure of the original feature space. For each of our data sets we compute the average value of $D\left(F, \mathcal{D}\right)$ across the different feature extractors. To reduce computational cost we restrict ourselves to a thousand examples of both train and test data, $\mathcal{D}^{\text{red}}$. Figure 7 shows how the average value of $M^{-1}\sum_{i=1}^{M} D\left(\boldsymbol{B}_i, \mathcal{D}^{\text{red}}\right)$ varies as a function of the diversity parameter. As we increase the diversity parameter λ we also reduce the emphasis on reconstruction accuracy. Hence, increasing λ reduces the degree to which the extracted features accurately

reflect the metric structure of the original feature space. This explains the fall in individual classification accuracy we observed in Figure 6.

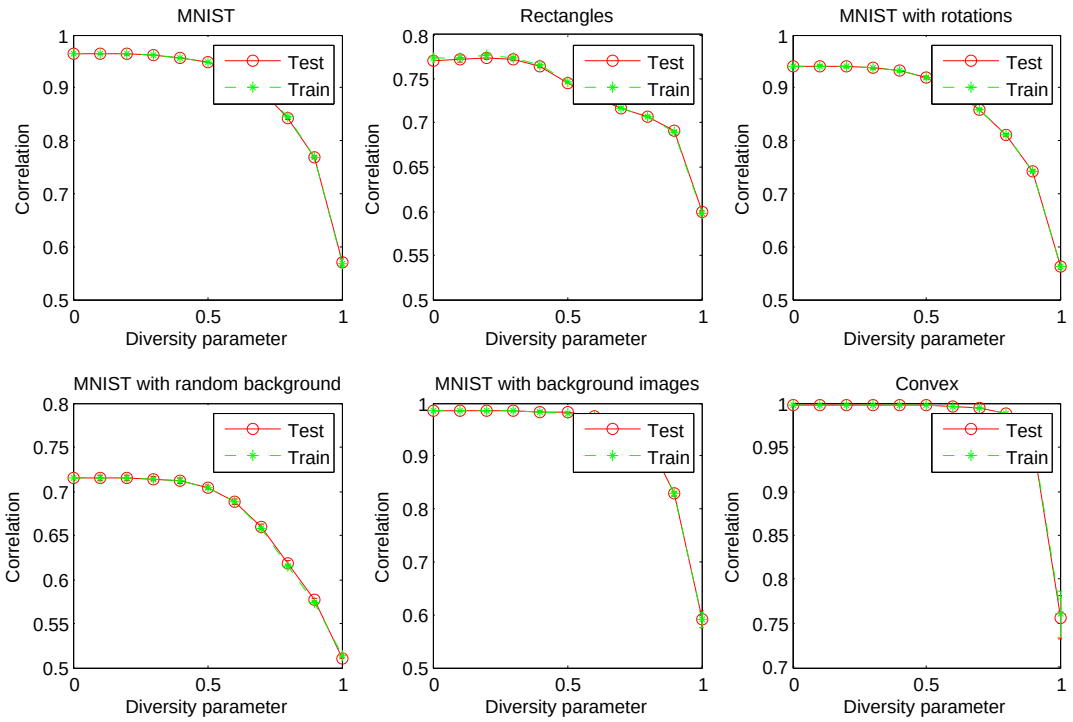

Figure 7: Average distance correlation between extracted features.

Given feature extractor maps F and G (such as $\boldsymbol{x} \mapsto \boldsymbol{B}_i\boldsymbol{x}$ and $\boldsymbol{x} \mapsto \boldsymbol{B}_j\boldsymbol{x}$), on a data set $\mathcal{D}$ we compute $C\left(F, G, \mathcal{D}\right)$, the distance correlation based upon the pairs $\{(F(\boldsymbol{x}), G(\boldsymbol{x})) : \boldsymbol{x} \in \mathcal{D}\}$. The quantity $C\left(F, G, \mathcal{D}\right)$ gives us a measure of the correlation between the metric structures induced by F and G. Again, to reduce computational cost we restrict ourselves to a thousand examples of both train and test data, $\mathcal{D}^{\text{red}}$. To measure the degree of diversity between our different sets of extracted features we compute the average pairwise correlation $C\left(\boldsymbol{B}_i, \boldsymbol{B}_j, \mathcal{D}^{\text{red}}\right)$, averaged across all pairs of distinct feature maps $\boldsymbol{B}_i, \boldsymbol{B}_j$ with $i \neq j$. Again we restrict ourselves to a thousand out-of-sample examples. Figure 8 shows how the degree of metric correlation between the different sets of extracted features falls as we increase the diversity parameter λ. Increasing λ places an increasing level of emphasis on a diversity of reconstructions. This diversity results in the different classifiers making different errors from one another enabling the improved ensemble performance we observed in Section 4.

6. Discussion

We have introduced a modular approach to representation learning where an ensemble of auto-encoder modules is learnt so as to achieve a diversity of reconstructions, as well

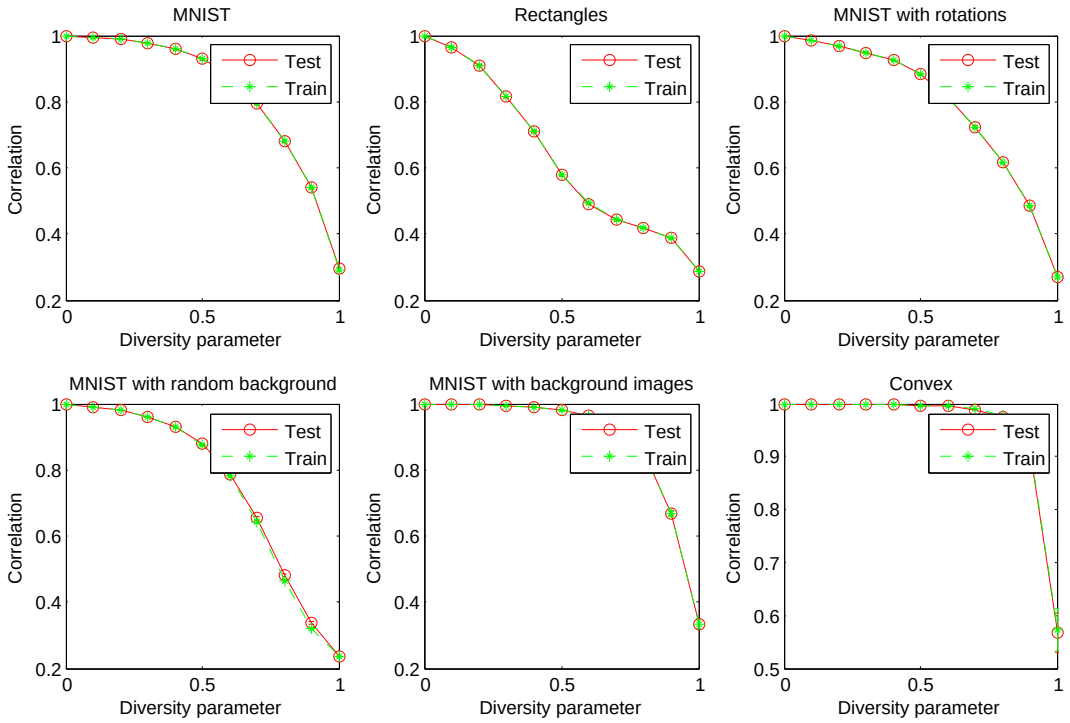

Figure 8: Average pairwise distance correlation between different feature extractors.

as maintaining low reconstruction error for each individual module. We demonstrated empirically, using six benchmark data sets, that we can improve the performance of a classifier ensemble by first learning a diverse collection of modular feature extractors in an unsupervised way. We explored Linear Modular Autoencoders and derived an SVD-based algorithm which converges three orders of magnitude faster than gradient descent. In forthcoming work we extend this concept beyond the realm of auto-encoders and into a broader framework of modular manifold learning.

Acknowledgments

The research leading to these results has received funding from EPSRC Centre for Doctoral Training grant EP/I028099/1, the EPSRC Anyscale project EP/L000725/1 and from the AXLE project funded by the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement no 318633. We would also like to thank Kostas Sechidis, Nikolaos Nikolaou, Sarah Nogueira and Charlie Reynolds for their useful comments, and the anonymous referee for suggesting several useful references.

References

Pierre Baldi and Kurt Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, 2(1):53–58, 1989.

Yoshua Bengio, Aaron Courville, and Pierre Vincent. Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1798–1828, 2013.

Gavin Brown, Jeremy Wyatt, Rachel Harris, and Xin Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20, 2005.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

Mehmet Gönen. Coupled dimensionality reduction and classification for supervised and semi-supervised multilabel learning. *Pattern recognition letters*, 38:132–141, 2014.

Anders Krogh, Jesper Vedelsby, et al. Neural network ensembles, cross validation, and active learning. 1995.

Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the 24th international conference on Machine learning*, pages 473–480. ACM, 2007.

Yong Liu and Xin Yao. Ensemble learning via negative correlation. *Neural Networks*, 12 (10):1399–1404, 1999.

Mehryar Mohri, Afshin Rostamizadeh, and Dmitry Storcheus. Foundations of coupled nonlinear dimensionality reduction. *arXiv preprint arXiv:1509.08880v2*, 2015.

Gábor J Székely, Maria L Rizzo, Nail K Bakirov, et al. Measuring and testing dependence by correlation of distances. *The Annals of Statistics*, 35(6):2769–2794, 2007.

Appendix A. Modular Regression Networks

We shall consider the more general framework of *Modular Regression Networks* (MRN) which encompasses Modular Autoencoders (MAE).

A Modular Regression Network $\mathcal{F} = \{F_i\}_{i=1}^M$ is an ensemble system consisting of M mappings $F_i : \mathbb{R}^D \to \mathbb{R}^Q$. The MRN $\mathcal{F}$ is trained using the following loss,

$$L_\lambda(\mathcal{F}, \boldsymbol{x}, \boldsymbol{y}) := \frac{1}{M}\sum_{i=1}^{M} \overbrace{||F_i(\boldsymbol{x}) - \boldsymbol{y}||^2}^{\text{error}} - \lambda \cdot \frac{1}{M}\sum_{i=1}^{M} \overbrace{\left|\left|F_i(\boldsymbol{x}) - \overline{F}(\boldsymbol{x})\right|\right|^2}^{\text{diversity}}, \tag{2}$$

where $\boldsymbol{x}$ is a feature vector, $\boldsymbol{y}$ a corresponding output, and $\overline{F}$ denotes the arithmetic average $\overline{F} := \frac{1}{M}\sum_{i=1}^{M} F_i$. Given a data set $\mathcal{D} = \{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^N$ we let $E_\lambda(\mathcal{F}, \mathcal{D})$ denote the loss $L_\lambda(\mathcal{F}, \boldsymbol{x}, \boldsymbol{y})$ averaged over $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}$. The MRN $\mathcal{F}$ is trained to minimise $E_\lambda(\mathcal{F}, \mathcal{D})$.

A.1 Investigating the loss function

Proposition 1 *Given $\lambda \in [0, \infty)$ and an MRN $\mathcal{F}$, for each example $(\boldsymbol{x}, \boldsymbol{y})$ we have*

$$L_\lambda(\mathcal{F}, \boldsymbol{x}, \boldsymbol{y}) = (1-\lambda) \cdot \frac{1}{M}\sum_i ||F_i(\boldsymbol{x}) - \boldsymbol{y}||^2 + \lambda \cdot \left|\left|\overline{F}(\boldsymbol{x}) - \boldsymbol{y}\right|\right|^2.$$

Proof The result may be deduced from the Ambiguity Decomposition (Krogh et al., 1995). ■

The following proposition relates MRNs to Negative Correlation Learning (Liu and Yao, 1999).

Proposition 2 *Given an MRN $\mathcal{F}$ and $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}$ we have*

$$\frac{\partial L_\lambda(\mathcal{F}, \boldsymbol{x}, \boldsymbol{y})}{\partial F_i} = \frac{2}{M} \cdot \left((F_i(\boldsymbol{x}) - \boldsymbol{y}) - \lambda \cdot \left(F_i(\boldsymbol{x}) - \overline{F}(\boldsymbol{x})\right)\right).$$

Proof This follows from the definitions of $L_\lambda(\mathcal{F}, \boldsymbol{x}, \boldsymbol{y})$ and $\overline{F}(\boldsymbol{x})$. ■

In Negative Correlation Learning each network F_i is updated in parallel with rule

$$\theta^i \leftarrow \theta^i - \alpha \cdot \frac{\partial F_i}{\partial \theta^i}\left((F_i(\boldsymbol{x}) - \boldsymbol{y}) - \lambda \cdot \left(F_i(\boldsymbol{x}) - \overline{F}(\boldsymbol{x})\right)\right),$$

for each example $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}$ in turn, where θ^i denotes the parameters of F_i and α denotes the learning rate (Liu and Yao, 1999, Equation 4). By Proposition 2 this is equivalent to training a MRN $\mathcal{F}$ to minimise $E_\lambda(\mathcal{F}, \mathcal{D})$ with stochastic gradient descent, using the learning rate $M/2 \cdot \alpha$.

A.2 An upper bound on the diversity parameter

We now focus on a particular class of networks. Suppose that there exists a vector-valued function $\varphi(\boldsymbol{x}; \rho)$, parametrised by ρ. We assume that φ is sufficiently expressive that for

each possible feature vector $\boldsymbol{x} \in \mathbb{R}^D$ there exists a choice of parameters ρ with $\varphi(\boldsymbol{x};\rho) \neq \mathbf{0}$. Suppose that for each i there exists a weight matrix $\boldsymbol{W}^i$ and parameter vector ρ^i such that $F_i(\boldsymbol{x}) = \boldsymbol{W}^i \varphi\left(\boldsymbol{x};\rho^i\right)$. We refer to such networks as *Modular Linear Top Layer Networks* (MLT). This is the natural choice in the context of regression and includes Modular Autoencoders with linear outputs.

Theorem 3 *Suppose we have a MLT $\mathcal{F}$ and a dataset $\mathcal{D}$. The following dichotomy holds:*

- *If $\lambda \leq 1$ then* $\inf E_\lambda(\mathcal{F},\mathcal{D}) \geq 0$.
- *If $\lambda > 1$ then* $\inf E_\lambda(\mathcal{F},\mathcal{D}) = -\infty$.

In both cases the infimums range over possible parametrisations for the MRN $\mathcal{F}$.

Moreover, if $\lambda > 1$ there exists parametrisations of $\mathcal{F}$ with arbitrarily low error $E_\lambda(\mathcal{F},\mathcal{D})$ and arbitrarily high squared loss for the ensemble output $\overline{F}$ and average squared loss for the individual regression networks F.

Proof It follows from Proposition 1 that whenever $\lambda \leq 1$, $L_\lambda(\mathcal{F},\boldsymbol{x},\boldsymbol{y}) \geq 0$ for all choices of $\mathcal{F}$ and all $(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{D}$. This implies the consequent in the case where $\lambda \leq 1$.

We now address the implications of $\lambda > 1$.

Take $(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{y}}) \in \mathcal{D}$. By the (MLT) assumption we may find parameters ρ so that $\varphi(\tilde{\boldsymbol{x}};\rho) \neq \mathbf{0}$. Without loss of generality we may assume that $0 \neq c = \varphi_1(\tilde{\boldsymbol{x}};\rho)$, where $\varphi_1(\boldsymbol{x};\rho)$ denotes the first coordinate of $\varphi(\boldsymbol{x};\rho)$. We shall leave ρ fixed and obtain a sequence $(\mathcal{F}_q)_{q\in\mathbb{N}}$, where for each q we have $F_i^q(\boldsymbol{x}) = \boldsymbol{W}^{(i,q)}\varphi(\boldsymbol{x};\rho)$ by choosing $\boldsymbol{W}^{(i,q)}$.

First take $\boldsymbol{W}^{(i,q)} = 0$ for all $i = 3,\cdots,M$, so

$$\overline{F}(\boldsymbol{x}) = \frac{1}{M}\left(F_1(\boldsymbol{x}) + F_2(\boldsymbol{x})\right).$$

In addition we choose $W_{kl}^{(1,q)} = W_{kl}^{(2,q)} = 0$ for all $k > 1$ or $l > 1$. Finally we take $W_{11}^{(1,q)} = c\cdot\left(q^2+q\right)$ and $W_{11}^{(2,q)} = -c\cdot q^2$. It follows that for each $(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{D}$ we have,

$$\begin{aligned}
F_1(\boldsymbol{x}) &= \boldsymbol{W}^{(1,q)}\varphi(\boldsymbol{x},\rho) = \left(c(q^2+q)\varphi_1(\boldsymbol{x};\rho),0,\cdots,0\right)\\
F_1(\boldsymbol{x}) &= \boldsymbol{W}^{(2,q)}\varphi(\boldsymbol{x},\rho) = \left(-cq^2\varphi_1(\boldsymbol{x};\rho),0,\cdots,0\right),\\
\overline{F}(\boldsymbol{x}) &= \left(M^{-1}cq\varphi_1(\boldsymbol{x};\rho),0,\cdots,0\right).
\end{aligned}$$

Noting that $\varphi_1(\boldsymbol{x};\rho) = c \neq 0$ and $(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{y}}) \in \mathcal{D}$ we see that,

$$\frac{1}{N}\sum_{n=1}^{N}\left(\overline{F}(\boldsymbol{x}_n) - \boldsymbol{y}_n\right)^2 = \Omega(q^2). \tag{3}$$

On the other hand we have,

$$\frac{1}{N}\sum_{n=1}^{N}\left(F_1(\boldsymbol{x}_n) - \boldsymbol{y}_n\right)^2 = \Omega(q^4),$$

and clearly for all i,

$$\frac{1}{N}\sum_{n=1}^{N}\left(F_1(\boldsymbol{x}_n)-\boldsymbol{y}_n\right)^2>0.$$

Hence,

$$\frac{1}{N}\sum_{n=1}^{N}\frac{1}{M}\sum_{i=1}^{M}\left(F_1(\boldsymbol{x}_n)-\boldsymbol{y}_n\right)^2=\Omega(q^4).$$

Combining with Equation (3) this gives

$$E_\lambda\left(\mathcal{F}^q,\mathcal{D}\right)=(1-\lambda)\cdot\Omega(q^4)+\lambda\cdot\Omega(q^2).$$

Since $\lambda>1$ this implies

$$E_\lambda\left(\mathcal{F}^q,\mathcal{D}\right)=-\Omega(q^4). \tag{4}$$

By Equations 3 and 4 we see that for any $Q_1,Q_2>1$, by choosing q sufficiently large we have

$$E_\lambda\left(\mathcal{F}^q,\mathcal{D}\right)<-Q_1,$$

and

$$\frac{1}{N}\sum_{n=1}^{N}\frac{1}{M}\sum_{i=1}^{M}\left(F_i(\boldsymbol{x}_n)-\boldsymbol{y}_n\right)^2\geq\frac{1}{N}\sum_{n=1}^{N}\left(\overline{F}(\boldsymbol{x}_n)-\boldsymbol{y}_n\right)^2>Q_2,$$

This proves the second item. ■

The impact of Theorem 3 is that whenever $\lambda>1$, minimising E_λ will result in one or more parameters diverging. Moreover, the resultant solutions may be arbitrarily bad in terms of training error, leading to very poor choices of parameters.

Theorem 4 *Suppose we have a MLT $\mathcal{F}$ on a data set $\mathcal{D}$. Suppose we choose $i\in\{1,\cdots,M\}$ and fix F_j for all $j\neq i$. The following dichotomy holds:*

- *If $\lambda<\frac{M}{M-1}$ then* $\inf E_\lambda\left(\mathcal{F},\mathcal{D}\right)>-\infty$.
- *If $\lambda>\frac{M}{M-1}$ then* $\inf E_\lambda\left(\mathcal{F},\mathcal{D}\right)=-\infty$.

In both cases the infimums range over possible parameterisations for the function f_i, with f_j fixed for $j\neq i$.

Proof We fix F_j for $j\neq i$. For each pair $(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}$ we consider $L_\lambda\left(\mathcal{F}\right)$ for large $||F_i(\boldsymbol{x})||$. By Proposition 1 we have

$$\begin{aligned}L_\lambda\left(\mathcal{F},\boldsymbol{x},\boldsymbol{y}\right)&=(1-\lambda)\cdot\frac{1}{M}\Omega\left(||F_i(\boldsymbol{x})||^2\right)+\lambda\cdot\Omega\left(||\frac{1}{M}F_i(\boldsymbol{x})||^2\right)\\&=\left(1-\lambda\cdot\frac{M-1}{M}\right)\cdot\Omega\left(||F_i(\boldsymbol{x})||^2\right).\end{aligned}$$

Hence, if $\lambda < \frac{M}{M-1}$ we see that $L_\lambda(\mathcal{F}, \boldsymbol{x}, \boldsymbol{y})$ is bounded from below for each example $(\boldsymbol{x}, \boldsymbol{y})$, for all choices of F_i. This implies the first case.

In addition, the fact that $\varphi(\boldsymbol{x}_1, \rho) \neq 0$ for some choice of parameters ρ means that we may choose a sequence of parameters such that $||F_i(\boldsymbol{x})|| \to \infty$ for one or more examples $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}$. Hence, if $\lambda > \frac{M}{M-1}$, we may choose weights so that $L_\lambda(\mathcal{F}, \boldsymbol{x}, \boldsymbol{y}) \to -\infty$ for some examples $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}$. The above asymptotic formula also implies that $L_\lambda(\mathcal{F}, \boldsymbol{x}, \boldsymbol{y})$ is uniformly bounded from above when $\lambda > \frac{M}{M-1}$. Thus, we have $\inf E_\lambda(\mathcal{F}, \mathcal{D}) = -\infty$. ■

Appendix B. Derivation of the Linear Modular Autoencoder Training Algorithm

In what follows we fix D, N, and $H < D$ and define

$$\mathcal{C}_{D,H} := \left\{(\boldsymbol{A}, \boldsymbol{B}) : \boldsymbol{A} \in \mathbb{R}^{D\times H}, \boldsymbol{B} \in \mathbb{R}^{H\times D}\right\}.$$

We data set $\mathcal{D} \subset \mathbb{R}^D$, with D features and N examples, and let $\boldsymbol{X}$ denote the $D \times N$ matrix given by $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N]$.

Given any $\lambda \in [0, \infty)$ we define our error function by

$$\begin{aligned} E_\lambda(\mathcal{W}, \mathcal{D}) =& \frac{1}{N}\sum_{n=1}^{N}\left(\frac{1}{M}\sum_{j=1}^{M}\left(||\boldsymbol{x}_n - \boldsymbol{A}_j\boldsymbol{B}_j\boldsymbol{x}_n||^2 - \lambda\cdot\left|\left|\boldsymbol{A}_j\boldsymbol{B}_j\boldsymbol{x}_n - \frac{1}{M}\sum_{k=1}^{M}A_kB_kx_n\right|\right|^2\right)\right) \\ =& \frac{1}{N\cdot M}\sum_{i=1}^{M}\left(||\boldsymbol{X} - \boldsymbol{A}_j\boldsymbol{B}_j\boldsymbol{X}||^2 - \lambda\cdot\left|\left|\boldsymbol{A}_j\boldsymbol{B}_j\boldsymbol{X} - \frac{1}{M}\sum_{k=1}^{M}A_kB_k\boldsymbol{X}\right|\right|^2\right), \end{aligned}$$

where $\mathcal{W} = ((\boldsymbol{A}_i, \boldsymbol{B}_i))_{i=1}^M \in (\mathcal{C}_{D,P})^M$, and $||\cdot||$ denotes the Frobenius matrix norm.

Proposition 3 *Suppose we take $\boldsymbol{X}$ so that $\Sigma = \boldsymbol{X}\boldsymbol{X}^T$ has full rank D and choose $\lambda < M/(M-1)$. We pick some $i \in \{1, \cdots, M\}$, and fix $\boldsymbol{A}_j, \boldsymbol{B}_j$ for each $j \neq i$. Then we find $(\boldsymbol{A}_i, \boldsymbol{B}_i)$ which minimises $E_\lambda(\mathcal{W}, \mathcal{D})$ by*

1. *Taking $\boldsymbol{A}_i$ to be the matrix whose columns consist of the D unit eigenvectors with largest eigenvalues for the matrix*

$$\left(\boldsymbol{I}_D - \frac{\lambda}{M}\sum_{j\neq i}\boldsymbol{A}_j\boldsymbol{B}_j\right)\Sigma\left(\boldsymbol{I}_D - \frac{\lambda}{M}\sum_{j\neq i}\boldsymbol{A}_j\boldsymbol{B}_j\right)^T,$$

2. *Choosing $\boldsymbol{B}_i$ so that*

$$\boldsymbol{B}_i = \left(1 - \lambda\cdot\frac{M-1}{M}\right)^{-1}\cdot\boldsymbol{A}_i^T\left(\boldsymbol{I}_D - \frac{\lambda}{M}\sum_{j\neq i}\boldsymbol{A}_j\boldsymbol{B}_j\right).$$

Moreover, for any other decoder-encoder pair $\left(\tilde{\boldsymbol{A}}_i, \tilde{\boldsymbol{B}}_i\right)$ *which also minimises* $E_\lambda(\mathcal{W}, \mathcal{D})$ *(with the remaining pairs* $\boldsymbol{A}_j, \boldsymbol{B}_j$ *fixed) we have* $\tilde{\boldsymbol{A}}_i\tilde{\boldsymbol{B}}_i = \boldsymbol{A}_i\boldsymbol{B}_i$.

Proposition 3 implies the following proposition from Section 3.

Theorem 2 *Suppose that* Σ *is of full rank. Let* $(\mathcal{W}_t)_{t=1}^T$ *be a sequence of parameters obtained by Algorithm 1. For every epoch* $t = \{1, \cdots, T\}$*, we have* $E_\lambda(\mathcal{W}_{t+1}, \mathcal{D}) < E_\lambda(\mathcal{W}_t, \mathcal{D})$*, unless* $\mathcal{W}_t$ *is a critical point for* $E_\lambda(\cdot, \mathcal{D})$*, in which case* $E_\lambda(\mathcal{W}_{t+1}, \mathcal{D}) \leq E_\lambda(\mathcal{W}_t, \mathcal{D})$.

Proof By Proposition 3, each update in Algorithm 1 modifies a decoder-encoder pair $(\boldsymbol{A}_i, \boldsymbol{B}_i)$ so as to minimise $E_\lambda(\mathcal{W}, \mathcal{D})$, subject to the condition that $(\boldsymbol{A}_j, \boldsymbol{B}_j)$ remain fixed for $j \neq i$. Hence, $E_\lambda(\mathcal{W}_{t+1}, \mathcal{D}) \leq E_\lambda(\mathcal{W}_t, \mathcal{D})$.

Now suppose $E_\lambda(\mathcal{W}_{t+1}, \mathcal{D}) = E_\lambda(\mathcal{W}_t, \mathcal{D})$ for some t. Note that $E_\lambda(\mathcal{W}_{t+1}, \mathcal{D})$ is a function of $\mathcal{C} = \{\boldsymbol{C}_i\}_{i=1}^M$ where $\boldsymbol{C}_i = \boldsymbol{A}_i\boldsymbol{B}_i$ for $i = 1, \cdots, M$. We shall show that $\mathcal{C}_t$ is a critical point in terms for E_λ. Since $E_\lambda(\mathcal{W}_{t+1}, \mathcal{D}) = E_\lambda(\mathcal{W}_t, \mathcal{D})$ we must have $\boldsymbol{C}_i^{t+1} = \boldsymbol{C}_i^t$ for $i = 1, \cdots, M$. Indeed, Proposition 3 implies that Algorithm 1 only modifies $\boldsymbol{C}_i$ when $E_\lambda(\mathcal{W}, \mathcal{D})$ is reduced (although the individual matrices $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$ may be modified). Since $\boldsymbol{C}_i^{t+1} = \boldsymbol{C}_i^t$ we may infer that $\boldsymbol{C}_i^t$ attains the minimum value of $E_\lambda(\mathcal{W}, \mathcal{D})$ over the set of parameters such that $\boldsymbol{C}_j = \boldsymbol{C}_j^t$ for all $j \neq i$. Hence, at the point $\mathcal{C}_t$ we have $\partial E_\lambda / \partial \boldsymbol{C}_i = 0$ for each $i = 1, \cdots, M$. Thus, $\partial E_\lambda / \partial \boldsymbol{A}_i = 0$ and $\partial E_\lambda / \partial \boldsymbol{B}_i = 0$, for each i, by the chain rule. ■

To prove Proposition 3 we require two intermediary lemmas. The first is a theorem concerning Rank Restricted Linear Regression.

Theorem 5 *Suppose we have* $D \times N$ *data matrices* $\boldsymbol{X}, \boldsymbol{Y}$*. We define a function* $E : \mathcal{C}_{D,H} \to \mathbb{R}$ *by*

$$E(\boldsymbol{A}, \boldsymbol{B}) = ||\boldsymbol{Y} - \boldsymbol{A}\boldsymbol{B}\boldsymbol{X}||^2.$$

Suppose that the matrix $\boldsymbol{X}\boldsymbol{X}^T$ *is invertible and define* $\Sigma := (\boldsymbol{Y}\boldsymbol{X}^T)(\boldsymbol{X}\boldsymbol{X}^T)^{-1}(\boldsymbol{X}\boldsymbol{Y}^T)$*. Let* $\boldsymbol{U}$ *denote the* $N \times D$ *matrix who's columns are the* D *unit eigenvectors of* Σ *with largest eigen-values. Then the minimum for* E *is attained by taking,*

$$\begin{aligned} \boldsymbol{A} &= \boldsymbol{U} \\ \boldsymbol{B} &= \boldsymbol{U}^T(\boldsymbol{Y}\boldsymbol{X}^T)(\boldsymbol{X}\boldsymbol{X}^T)^{-1}. \end{aligned}$$

Proof See Baldi and Hornik (1989, Fact 4). ■

Note that the minimal solution is not unique. Indeed if $\boldsymbol{A}, \boldsymbol{B}$ attain the minimum, then so does $\boldsymbol{A}\boldsymbol{C}$, $\boldsymbol{C}^{-1}\boldsymbol{B}$ for any invertible $H \times H$ matrix $\boldsymbol{C}$.

Lemma 6 *Suppose we have $D \times N$ matrices $\boldsymbol{X}$ and $\boldsymbol{Y}_1, \cdots, \boldsymbol{Y}_Q$, and scalars $\alpha_1, \cdots, \alpha_Q$ such that $\sum_{q=1}^{Q} \alpha_q > 0$. Then we have*

$$\begin{aligned}\arg \min_{(\boldsymbol{A},\boldsymbol{B}) \in \mathcal{C}_{D,H}} \left\{ \sum_{q=1}^{Q} \alpha_q ||\boldsymbol{Y}_q - \boldsymbol{ABX}||^2 \right\} \\ = \arg \min_{(\boldsymbol{A},\boldsymbol{B}) \in \mathcal{C}_{D,H}} \left\{ || \left(\sum_{q=1}^{Q} \tilde{\alpha}_q \boldsymbol{Y} \right) - \boldsymbol{ABX}||^2 \right\},\end{aligned}$$

where $\tilde{\alpha}_q = \alpha_q / \left(\sum_{q'=1}^{Q} \alpha_{q'} \right)$.

Proof We use the fact that under the Frobenius matrix norm, $||\boldsymbol{M}||^2 = \mathrm{tr}(\boldsymbol{M}\boldsymbol{M}^T)$ for matrices $\boldsymbol{M}$, where tr denotes the trace operator. Note also that the trace operator is linear and invariant under matrix transpositions. Hence, we have

$$\sum_{q=1}^{Q} \alpha_q \, ||\boldsymbol{Y}_q - \boldsymbol{ABX}||^2$$

$$\begin{aligned}&= \sum_{q=1}^{Q} \alpha_q \cdot \mathrm{tr} \left((\boldsymbol{Y}_q - \boldsymbol{ABX})(\boldsymbol{Y}_q - \boldsymbol{ABX})^T \right) \\ &= \sum_{q=1}^{Q} \alpha_q \cdot \mathrm{tr} \left(\boldsymbol{Y}_q \boldsymbol{Y}_q^T - 2(\boldsymbol{AB})\boldsymbol{X}\boldsymbol{Y}_q^T + (\boldsymbol{AB})\boldsymbol{X}\boldsymbol{X}(\boldsymbol{AB})^T \right) \\ &= \sum_{q=1}^{Q} \alpha_q \, ||\boldsymbol{Y}_q||^2 - \mathrm{tr} \left(2(\boldsymbol{AB})\boldsymbol{X} \left(\sum_{q=1}^{Q} \alpha_q \boldsymbol{Y}_q \right)^T \right) + \mathrm{tr} \left(\left(\sum_{q=1}^{Q} \alpha_q \right) (\boldsymbol{AB})\boldsymbol{X}\boldsymbol{X}^T(\boldsymbol{AB})^T \right).\end{aligned}$$

Note that we may add constant terms (ie. terms not depending on $\boldsymbol{A}$ or $\boldsymbol{B}$) and multiply by positive scalars without changing the minimising argument. Hence, dividing by $\sum_{q=1}^{Q} \alpha_q > 0$ and adding a constant we see that the minimiser of the above expression is equal to the minimiser of

$$\mathrm{tr} \left((\boldsymbol{AB})\boldsymbol{X}\boldsymbol{X}^T(\boldsymbol{AB})^T \right) + \mathrm{tr} \left(2(AB)X \left(\sum_{q=1}^{Q} \tilde{\alpha}_q \boldsymbol{Y}_q \right)^T \right) + \mathrm{tr} \left(\left(\sum_{q=1}^{Q} \tilde{\alpha}_q \boldsymbol{Y}_q \right) \left(\sum_{q=1}^{Q} \tilde{\alpha}_q \boldsymbol{Y}_q \right)^T \right).$$

Moreover, by the linearity of the trace operator this expression is equal to

$$\left|\left| \left(\sum_{q=1}^{Q} \tilde{\alpha}_q \boldsymbol{Y}_q \right) - \boldsymbol{ABX} \right|\right|^2.$$

This proves the lemma. ■

Proof [Proposition 3] We begin observing that if we fix $\boldsymbol{A}_j, \boldsymbol{B}_j$ for $j \neq i$, then minimising $E_\lambda(\mathcal{W}, \mathcal{D})$ is equivalent to minimising

$$||\boldsymbol{X} - \boldsymbol{A}_i\boldsymbol{B}_i\boldsymbol{X}||^2 - \lambda\left(1 - \frac{1}{M}\right)^2 \left|\left|\frac{1}{M-1} \cdot \boldsymbol{S}_{-i} - \boldsymbol{A}_i\boldsymbol{B}_i\boldsymbol{X}\right|\right|^2 - \frac{\lambda}{M^2}\sum_{j\neq i} ||(M\boldsymbol{A}_j\boldsymbol{B}_j\boldsymbol{X} - \boldsymbol{S}_{-i}) - \boldsymbol{A}_i\boldsymbol{B}_i\boldsymbol{X}||^2,$$

where $\boldsymbol{S}_{-i} = \sum_{j\neq i} \boldsymbol{A}_j\boldsymbol{B}_j\boldsymbol{X}$. This holds as the above expression differs from $E_\lambda(\mathcal{W}, \mathcal{D})$ only by a multiplicative factor of NM and some constant terms which do not depend upon $\boldsymbol{A}_i, \boldsymbol{B}_i$.

By Lemma 6, minimising the above expression in terms of $\boldsymbol{A}_i, \boldsymbol{B}_i$ is equivalent to minimising

$$||\boldsymbol{Y} - \boldsymbol{A}_i\boldsymbol{B}_i\boldsymbol{X}||, \tag{5}$$

with

$$\boldsymbol{Y} = \left(1 - \lambda\left(\left(1 - \frac{1}{M}\right)^2 + \frac{M-1}{M^2}\right)\right)^{-1} \cdot \left(\boldsymbol{X} - \lambda \cdot \left(\left(1 - \frac{1}{M}\right)^2 \frac{1}{M-1} \cdot \boldsymbol{S}_{-i} + \frac{1}{M^2}\sum_{j\neq i}(M\boldsymbol{A}_j\boldsymbol{B}_j\boldsymbol{X} - \boldsymbol{S}_{-i})\right)\right).$$

Here we use the fact that $\lambda < M/(M-1)$, so

$$1 - \lambda\left(\left(1 - \frac{1}{M}\right)^2 + \frac{M-1}{M^2}\right) = 1 - \lambda \cdot \frac{M-1}{M} > 0.$$

We may simplify our expression for $\boldsymbol{Y}$ as follows,

$$\boldsymbol{Y} = \left(1 - \lambda \cdot \frac{M-1}{M}\right)^{-1} \cdot \left(\boldsymbol{I}_D - \frac{\lambda}{M}\sum_{j\neq i}\boldsymbol{A}_j\boldsymbol{B}_j\right)\boldsymbol{X}.$$

By Theorem 5, we may minimise the expression in 5 by taking $\boldsymbol{A}_i$ to be the matrix whose columns consist of the D unit eigenvectors with largest eigenvalues for the matrix

$$\left(\boldsymbol{I}_D - \frac{\lambda}{M}\sum_{j\neq i}\boldsymbol{A}_j\boldsymbol{B}_j\right)(\boldsymbol{X}\boldsymbol{X}^T)\left(\boldsymbol{I}_D - \frac{\lambda}{M}\sum_{j\neq i}\boldsymbol{A}_j\boldsymbol{B}_j\right)^T,$$

and setting

$$\boldsymbol{B}_i = \left(1 - \lambda \cdot \frac{M-1}{M}\right)^{-1} \cdot \boldsymbol{A}_i^T\left(\boldsymbol{I}_D - \frac{\lambda}{M}\sum_{j\neq i}\boldsymbol{A}_j\boldsymbol{B}_j\right).$$

This completes the proof of the proposition. ■