A PENALIZED AUTOENCODER APPROACH FOR NONLINEAR INDEPENDENT COMPONENT ANALYSIS

Tianwen Wei

Xiaomi Inc.
Wuhan, China

Stéphane Chrétien

National Physical Laboratory
London, United Kingdom

ABSTRACT

We propose Independent Component Autoencoder (ICAE), a deep neural network-based framework for nonlinear Independent Component Analysis (ICA). The proposed method consists of a penalized autoencoder and a training objective that is to minimize a combination of the reconstruction loss and an ICA contrast. Unlike many previous ICA methods that are usually tailored to separate specific mixture, our method can recover sources from various mixtures, without prior knowledge on the nature of that mixture.

Index Terms— Independent Component Analysis, Autoencoder, Representation Learning, Nonlinear mixture

1. INTRODUCTION

Blind source separation (BSS) [1] aims at recovering unobserved latent variables or sources $\boldsymbol{s} = (s_1, \ldots, s_d)^\mathsf{T}$ from only observed signals $\boldsymbol{x} = (x_1, \ldots, x_n)^\mathsf{T}$ which are unknown functions of the sources, i.e.

$$\boldsymbol{x} = \boldsymbol{\Phi}(\boldsymbol{s}). \tag{1}$$

Independent Component Analysis (ICA) [2, 3] is the major framework for solving BSS problems which based on the assumption that the sources are statistically independent.

While linear BSS problems can be efficiently solved by ICA [2], the nonlinear ones are, on the other hand, ill-conditioned in general [4, 5]. But if $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi} \stackrel{\text{def}}{=} \boldsymbol{\Phi}^{-1}$ are properly constrained, the source separation problem can still be solved by ICA. One important example is the so called post-nonlinear (PNL) mixture [6]. The PNL mixture $\boldsymbol{\Phi} = f \circ \mathbf{A}$ consists of a linear mixture $\mathbf{A} = (a_{ij})$, followed by a channel-wise invertible nonlinear mapping f_i, i.e.

$$x_i = f_i\Big(\sum_{j=1}^{d} a_{ij} s_j\Big), \quad \forall i = 1, \ldots, n. \tag{2}$$

If one constrains the separating structure as $\boldsymbol{\Psi} = \mathbf{B} \circ g$, i.e. a composition of a channel-wise nonlinear mapping $g = (g_1, \ldots, g_n)$ and a linear mapping $\mathbf{B}$, then it is shown [6, 7] that under mild conditions the separation can be achieved via ICA. Another important class of nonlinear ICA is the linear-quadratic (LQ) mixture [8, 9]. The LQ mixture model assumes that the observed signal $\boldsymbol{x}$ is the sum of a linear function of $\boldsymbol{s}$ and a quadratic form of $\boldsymbol{s}$:

$$x_i = \sum_{j=1}^{d} a_{ij} s_j + \sum_{j=1}^{d} \sum_{k=1}^{d} b_{ijk} s_j s_k, \quad \forall i = 1, \ldots, n. \tag{3}$$

where (a_{ij}) and (b_{ijk}) are fixed scalars. Deville and Hosseini [8, 9] have shown that under some mild conditions the LQ mixture can be separated. Their solution is based on the recurrent neural network.

In this work, we propose a deep neural network-based framework for solving both linear and nonlinear ICA problems, which we shall refer to as the Independent Component AutoEncoder (ICAE). The neural network has long been a powerful tool for solving nonlinear ICA problems. For previous works we refer to [10, 11, 12, 13, 14, 15, 16, 17, 18]. The key feature of our methods is that it can successfully separate various mixtures, without prior knowledge on the nature of that mixture or the exact number of the sources. This is achieved via incorporating the reconstruction loss as part of the training objective, which is crucial for regularizing the solution. Similar approach can also be found in [19, 13, 18]. Our method is essentially an autoencoder [20] that targets a feature representation with independent components. To promote independence, we propose to penalize the representation with an ICA contrast, which can be regarded as a measure of degree of independence [2], or "nongaussianity" of the components [3]. By such penalization, the representation learned will eventually have independent components, with distributions that are non-Gaussian.

2. INDEPENDENT COMPONENT AUTOENCODER

2.1. Training objective

The pipeline of ICAE is depicted in Fig.1. For simplicity we shall assume that the number of sources d is known. The case for unknown d will be discussed in Section 3.4.2. The

This project is supported by NSFC (Grant No. 61701547). It is also funded by department of AI and cloud platform of Xiaomi Inc.

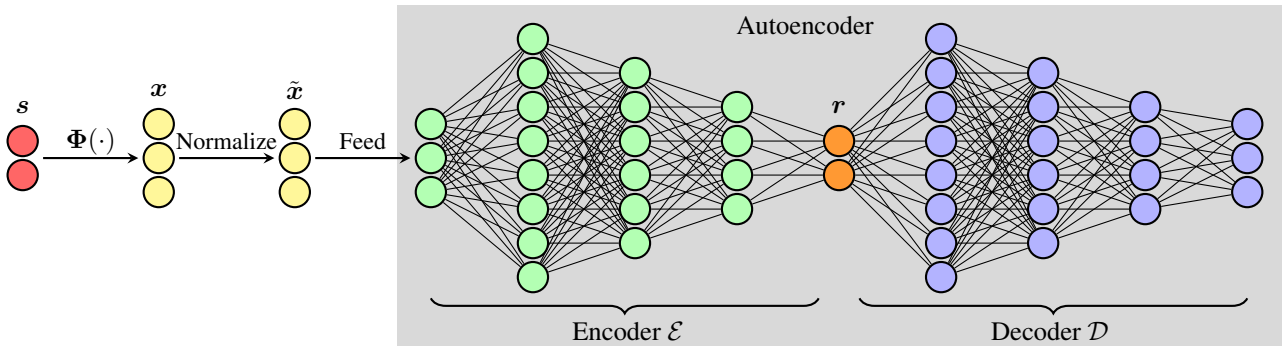

Fig. 1. From left to right: The unknown source $\boldsymbol{s}$ (red nodes) is transformed by a nonlinear function $\boldsymbol{\Phi}$. The observation $\boldsymbol{x} = \boldsymbol{\Phi}(\boldsymbol{s})$ (yellow nodes) is first normalized via $\tilde{x}_i = (x_i - \bar{x}_i)/\sigma_i$, then fed as input to an autoencoder $\mathcal{A}$ which consists of an encoder $\mathcal{E}$ (green nodes) and a decoder $\mathcal{D}$ (blue nodes). All neurons, except for those in the output layer of the encoder and decoder, are equipped with $\tanh$ activation function. The autoencoder is trained in such a way that the encoded representation $\boldsymbol{r}$ (orange nodes) has zero mean, unit variance and statistically independent components.

observed signal $\boldsymbol{x} = \boldsymbol{\Phi}(\boldsymbol{s})$ is fed after channel-wise normalization as input features to an autoencoder $\mathcal{A} = \mathcal{D} \circ \mathcal{E}$ consisting of an encoder $\mathcal{E} : \mathbb{R}^n \to \mathbb{R}^d$ and a decoder $\mathcal{D} : \mathbb{R}^d \to \mathbb{R}^n$. The autoencoder depends on trainable parameters $\boldsymbol{\theta}$, but for conciseness we will drop it from the notation. Our goal is to train the autoencoder in such a way that the encoded representation $\boldsymbol{r} = \mathcal{E}(\tilde{\boldsymbol{x}})$ has zero mean, unit variance and statistically independent components. To achieve this, we propose to minimize the following loss function:

$$L(\boldsymbol{\theta}) = D\big(\tilde{x}, \mathcal{A}(\tilde{x})\big) + \lambda G\big(\boldsymbol{r}\big), \tag{4}$$

where on the right hand side of (4):

1. The first term $D\big(\tilde{\boldsymbol{x}}, \mathcal{A}(\tilde{\boldsymbol{x}})\big)$, referred to as the *reconstruction loss*, measures the difference between input $\tilde{\boldsymbol{x}}$ and output $\mathcal{A}(\tilde{\boldsymbol{x}})$ of the autoencoder. In principle any reasonable distance metric $\mathcal{D}(\cdot, \cdot)$ should work. Here we take the simplest one, namely the l_2 loss employed in conventional autoencoder, i.e.

$$\mathcal{D}\big(\tilde{\boldsymbol{x}}, \mathcal{A}(\tilde{\boldsymbol{x}})\big) = \mathbb{E}\|\tilde{\boldsymbol{x}} - \mathcal{A}(\tilde{\boldsymbol{x}})\|^2. \tag{5}$$

   This term regularizes the learned representation, forcing it to be capable of reconstructing the input feature.

2. The second term $G(\boldsymbol{r})$, which we refer to as the *ICA contrast*, is a measure of independence of the components of $\boldsymbol{r} = (r_1, \ldots, r_d)^\mathsf{T}$. A natural candidate for G would be the mutual information. However, as evaluating the mutual information involves the estimation of the underlying pdf, which is a difficult task, we resort to surrogates. Among the many valid candidates, we choose the following ICA contrast borrowed from [21]:

$$G(\boldsymbol{r}) = -\sum_{i=1}^{d} \big|\mathbb{E}[r_i^4] - 3\big|. \tag{6}$$

   This contrast is based on the fourth-order cumulant [2, 1] and assumes a prewhitened input $\boldsymbol{r}$. To accommodate the latter requirement, we have applied mean removal and PCA based whitening to $\boldsymbol{r}$ before passing it to $G(\cdot)$.

3. Coefficient λ is a hyperparameter that balances the weights between the reconstruction loss and the ICA contrast. Clearly setting $\lambda = 0$ would lead to a conventional autoencoder, whereas pushing λ towards ∞ would resulting in a pure ICA contrast. In our experiments, we have always set $\lambda = 0.005$. The impact of this parameter on separation performance will be studied in Section 3.4.4.

2.2. Implementation details

As a common practice, the observation $\boldsymbol{x}$ is normalized before feeding it to the neural network. For the ith channel, the normalized observation is defined as $\tilde{x}_i = \frac{x_i - \bar{x}_i}{\sigma_i}$, where $\bar{x}_i$ and σ_i are respectively the sample mean and sample standard deviation of x_i. The encoder and decoder network typically consists of stacked fully-connected layers. The number of layers and number of neurons per layer determine the capacity of the autoencoder, hence they can be customized according to the application. In our experiments, both the encoder and decoder have identical structure of MLP with two hidden layers each containing 64 and 16 neurons. Neurons of all layers except for the output layers of the encoder and decoder are equipped with $\tanh$ activation function.

Code for reproducing the results presented in the current work is open sourced and can be downloaded form the internet[1].

[1] https://github.com/TianwenWei/ICAE.git

3. NUMERICAL EXPERIMENTS

3.1. Configuration

Throughout the experiments, we fix $d = 5$ and use synthetic sources all having zero mean and unit variance. The sources are plotted in the top five rows of Fig. 4. We shall focus on the following types of mixture: 1) Linear mixture; 2) PNL mixture, as defined in (2); 3) LQ mixture, as defined in (3). If not otherwise stated, we always use 10k samples as training data with feature dimension (i.e. number of mixed components) $n = 10$. For nonlinear functions such as those used in the PNL mixture, we consider the following: $f_1(x) = x^3$, $f_2(x) = \text{sigmoid}(x)$, $f_3(x) = \exp(x/3)$, $f_4(x) = \tanh(x)$ and $f_5(x) = \log\big(1+\exp(x)\big)$. These functions are all smooth and strictly increasing. We use Adam [22] to minimize the loss with learning rate 0.002 and a batch size 512. All statistics involved, such as those in (5) and (6) are computed in a per mini-batch basis. The training is considered finished after 100k iterations.

To evaluate the performance, we compute the output $\boldsymbol{r}$ of the representation layer on a test set of size 1024, then try to find a match with the ground truth sources $s_1, \ldots, s_d$. Basically, we consider r_i, the ith component of $\boldsymbol{r}$ matches s_j if either $\mathbb{E}\|s_i - r_j\|^2$ or $\mathbb{E}\|s_i + r_j\|^2$ is the smallest among all possible combinations of (i, j). Here the mathematical expectation is estimated by the sample mean over the test set. Let $\mathcal{P}$ denote the set of permutations over $\{1, \ldots, d\}$. The overall evaluation metric is defined as the average of MSEs:

$$\min_{\sigma\in\mathcal{P}} \left\{ \frac{1}{d} \sum_{i=1}^{d} \min\{\mathbb{E}\|s_i - r_{\sigma(i)}\|^2, \mathbb{E}\|s_i + r_{\sigma(i)}\|^2\} \right\},$$

which we refer to as the MSE as well.

3.2. Model selection

Sometimes the ICAE network fails to converge or converges to degenerate solutions. Hence it is crucial to train the network multiple times and perform the model selection in a principled way. Evaluating the model is a bit tricky, because a solution with lower training loss does not necessarily leads to a better separation quality. In the experiments that follow, we train the network five times with different initializations for each set of hyper-parameters, with *fixed* observation $\boldsymbol{x}$. We monitor the value of the ICA contrast during the training for each run. We shall select the network for which the ICA contrast (6) has the lowest variance computed over last 5k iterations of training. It turns out that such strategy tends to select the best model.

3.3. Baselines

We report in Table 1 the result of symmetric FastICA [23] and ANICA [18] as baselines. The FastICA algorithm is one of the most successful algorithm for linear ICA. Understandably, it does not perform well for PNL and LQ mixture. We obtained the FastICA result using Python's scikit-learn package. The ANICA algorithm is a recently proposed nonlinear ICA method which is based on an encoder-decoder network similar to ours, plus an adversarial discriminator network that promotes independence. The ANICA results in Table 1 is directly extracted from [18], where only the average correlations were given. Since those were obtained in a comparable but different setting (4k training samples, six sources) with specialized network structure that depends on the mixture, they are included here merely to give a rough baseline of a dedicated nonlinear ICA algorithm.

Table 1. A comparison of FastICA, ANICA and ICAE. Table cell "$0.001(1.000)$" means MSE $= 0.001$ and the average correlation of the estimated components and the corresponding sources is 1.000.

	linear	PNL	LQ
FastICA	0.001(1.000)	0.617(0.703)	0.227(0.886)
ANICA	- (0.999)	- (0.988)	-
ICAE	0.016(0.993)	0.025(0.986)	0.041(0.979)

3.4. Impact of hyper parameters

3.4.1. Impact of feature dimension

We have run five independent simulations for each mixture and for each $n \in \{5, 6, 7, 8, 9, 10\}$. The results as measured by average MSE are plotted in Fig. 2. We found out that the proposed method benefits greatly from the excess feature dimensions. From the plots we observe that for all types of mixture, as n grows the corresponding MSE decreases steadily and finally fall below 0.05. The low MSE obtained for small n (i.e. close to d) is due to the fact that the model often gets stuck in degenerate solutions, recovering only a subset of the sources. For larger n, this phenomenon occurs much less frequently.

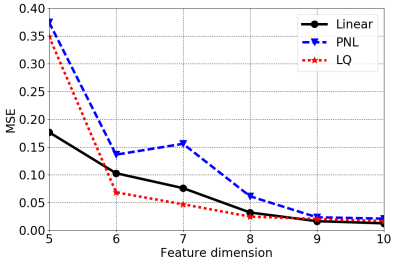

Fig. 2. Average MSE obtained for different mixtures.

3.4.2. *Impact of representation dimension*

We have also experimented with various representation dimension (number of estimated sources) $p \stackrel{\text{def}}{=} \dim(\boldsymbol{r})$, ranging from 3 to 10. The obtained reconstruction losses and MSEs are reported in Fig. 3. It can be seen that the optimal performance is obtained at $p = d$, which is more or less expected. If $p < d$, then the network cannot accurately recover the independent components due to the information bottleneck, resulting in both high reconstruction loss and high MSE. If $p > d$, then the separation performance tends to deteriorate as p increases, but if the excess dimension $p - d$ is relatively small, then the sources can still be recovered with acceptable precision. A sources vs. estimates plot for the case $p = 8$ with MSE $= 0.028$ is given in Fig 4.

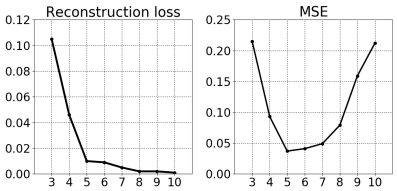

Fig. 3. Left: Reconstruction loss vs. representation dimension. Right: MSE vs. representation dimension. The number of sources $d = 5$ is fixed throughout.

3.4.3. *Impact of network complexity*

We have also experimented ICAEs with one and three hidden layers, respectively. Overall, as the number of hidden layers grows it becomes increasingly difficult to find a stable solution, while in terms of the separation performance all three models yield comparable results provided a converged network. With 1-layer ICAE, the source separation is almost always successful except for the scenario $p > d$, in which case usually only a subset of the sources can be correctly separated. When training a 3-layer ICAE, the chance of obtaining a divergent network is substantially higher than dealing with a 1-layer or 2-layer one. In our experiments approximately two thirds of the solutions given by 3-layer ICAE are degenerate or divergent. Hence in terms of the practicality the 2-layer one is preferred.

3.4.4. *Impact of hyper parameter λ*

The value of λ plays a crucial role in our model. Here we report the experiment results for a wide range of λ with the PNL mixture and $n = 10$ in Fig. 5. It can be seen that the MSE is somewhat sensitive to the value of λ. After experimenting with various sets of hyper parameters, we observe that the optimal value of λ seems to be mostly dependent only on the number of sources d and the representation dimension p. In fact, we have always fixed $\lambda = 0.005$ in the previous experiments, which seemingly have worked well.

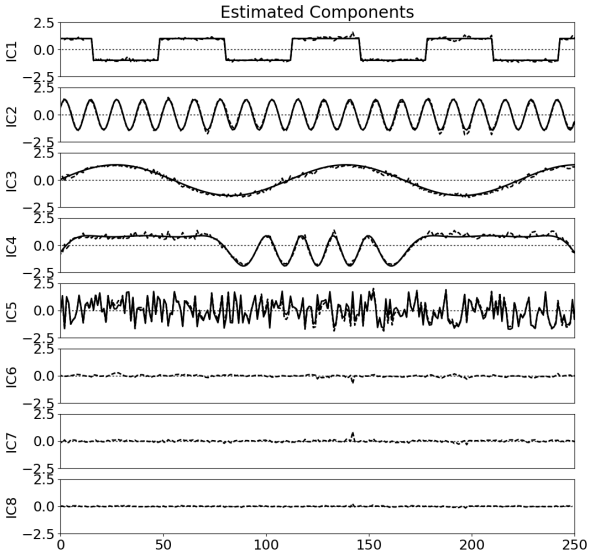

Fig. 4. Estimated components under PNL mixture with $d = 5$ and $p = 8$. The ground truth sources are plotted in solid lines for comparison. Components that correspond to the sources are labeled as IC1-IC5. Note that IC6-IC8 can be easily identified as "noise components" as they have near zero variances.

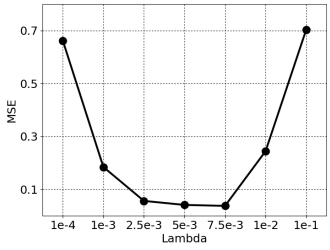

Fig. 5. Average MSE vs. λ under PNL mixture.

4. DISCUSSION

The proposed method has the advantage of being conceptually simple and easy to implement. It can successfully solve both linear and various nonlinear BSS problems, without assuming the nature of the mixture. Even in the situation where the number of the sources is not exactly known, the proposed method may still recover the sources and allows for easy identification of the noise components.

That said, we believe that there is much room for improvement within the proposed framework. For instance, there may be better choices of the regularizer for promoting independence than the fourth order cumulant used in the current work. Besides, the model selection procedure described in Section 3.2 sometimes selects a degenerate solution from the candidates. All those may be the starting points of the future work.

5. REFERENCES

[1] P. Comon and C. Jutten, *Handbook of Blind Source Separation: Independent Component Analysis and Applications*, Academic Press, 2010.

[2] P. Comon, "Independent component analysis: a new concept?," *Signal Processing*, vol. 36, no. 3, pp. 287–314, Apr. 1994.

[3] A Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis*, Wiley-Interscience, New York, 2001.

[4] A. Hyvärinen and P. Pajunen, "Nonlinear independent component analysis: Existence and uniqueness results," *Neural Networks*, pp. 429–439, 1999.

[5] A. Taleb, "A generic framework for blind source separation in structured nonlinear models," *IEEE Transactions on Signal Processing*, vol. 50, no. 8, pp. 1819–1830, Aug 2002.

[6] A. Taleb and C. Jutten, "Source separation in post-nonlinear mixtures," *IEEE Transactions on Signal Processing*, vol. 47, no. 10, pp. 2807–2820, Oct 1999.

[7] S. Achard and C. Jutten, "Identifiability of post-nonlinear mixtures," *IEEE Signal Processing Letters*, vol. 12, no. 5, pp. 423–426, May 2005.

[8] S. Hosseini and Y. Deville, "Blind separation of linear-quadratic mixtures of real sources using a recurrent structure," in *Int. Work-Conference on Artificial and Natural Neural Networks*, Berlin, Heidelberg, 2003, pp. 241–248, Springer Berlin Heidelberg.

[9] Y. Deville and S. Hosseini, "Recurrent networks for separating extractable-target nonlinear mixtures. part I: Non-blind configurations," *Signal Processing*, pp. 278–393, 2009.

[10] G. Burel, "Blind separation of sources: A nonlinear neural algorithm," *Neural Networks*, pp. 937–942, 1992.

[11] H. H. Yang, S-I. Amari, and A. Cichocki, "Information-theoretic approach to blind separation of sources in non-linear mixture," *Signal Processing*, vol. 64, no. 3, pp. 291 – 300, 1998.

[12] L. B. Almeida, "Linear and nonlinear ICA based on mutual informationthe misep method," *Signal Processing*, vol. 84, no. 2, pp. 231 – 245, 2004, Special Section on Independent Component Analysis and Beyond.

[13] Q. V. Le, A. Karpenko, J. Ngiam, and A. Y. Ng, "ICA with reconstruction cost for efficient overcomplete feature learning," in *Advances in Neural Information Processing Systems (NIPS)*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. 2011, pp. 1017–1025, Curran Associates, Inc.

[14] L. Dinh, D. Krueger, and Y. Bengio, "NICE: Non-linear independent components estimation," Workshop contribution in ICLR, 2015.

[15] Aapo Hyvärinen and Hiroshi Morioka, "Unsupervised feature extraction by time-contrastive learning and nonlinear ICA," in *NIPS 2016*, 2016.

[16] J-I. Hirayama, A. Hyvärinen, and M. Kawanabe, "SPLICE: Fully tractable hierarchical extension of ICA with pooling," in *Proceedings of the International Conference on Machine Learning (ICML)*, International Convention Centre, Sydney, Australia, Aug 2017, vol. 70, pp. 1491–1500.

[17] Y. Deville and L. T. Duarte, "An overview of blind source separation methods for linear-quadratic and post-nonlinear mixtures," in *Proceedings of the 12th International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA 2015)*, Liberec, Czech Republic, aug 2015, pp. 155–167, Springer International Publishing Switzerland.

[18] P. Brakel and Y. Bengio, "Adversarial non-linear independent component analysis," ArXiv, 2017.

[19] J. Schmidhuber, "Learning factorial codes by predictability minimization," *Neural Computation*, vol. 4, no. 6, pp. 863–879, 1992.

[20] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, aug 2013.

[21] T. Wei, "A convergence and asymptotic analysis of the generalized symmetric FastICA algorithm," *IEEE Transactions on Signal Processing*, vol. 63, pp. 6445–6458, Dec 2015.

[22] D. P. Kingma and J. L. Ba, "Adam: a method for stochastic optimization," in *Proceedings of The International Conference on Learning Representations (ICLR)*, 2015.

[23] A. Hyvärinen and E. Oja, "Independent component analysis: Algorithms and applications," *Neural Networks*, vol. 13, no. 4-5, pp. 411–430, 2000.