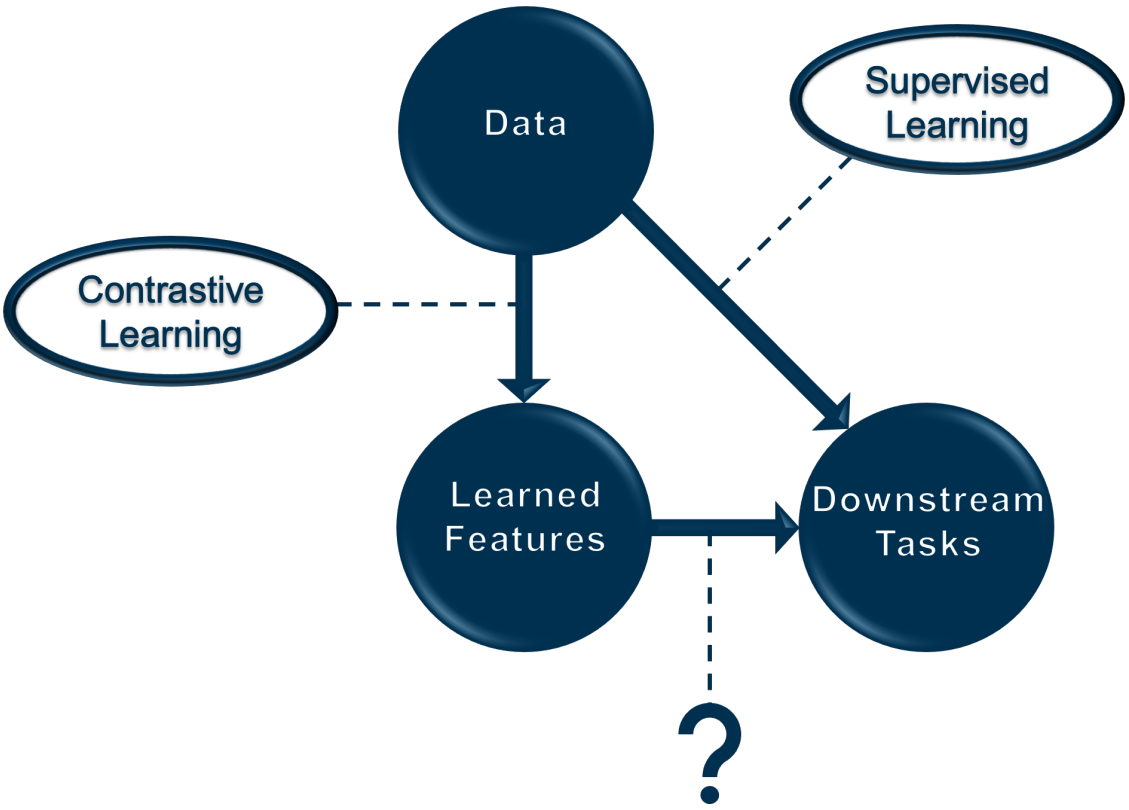


Generalizability of Representation Learning on Downstream Tasks

Master's thesis in Computer Science and Engineering

Anton Levinsson
Ziyuan Wang

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Master's thesis 2025

Generalizability for Representation Learning on Downstream Tasks

A Study of Downstream Performance in a Nonlinear ICA setting through Contrastive Learning

Anton Levinsson & Ziyuan Wang

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
Gothenburg, Sweden 2025

Generalizability for Representation Learning on Downstream Tasks
A Study of Downstream Performance in a Nonlinear
ICA setting using Contrastive Learning
Anton Levinsson & Ziyuan Wang

Supervisor: Ahmet Balcioglu, Department of Computer Science and Engineering
Examiner: Fredrik D. Johansson, Department of Computer Science and Engineering

Master's Thesis 2025
Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Wind visualization constructed in Matlab showing a surface of constant wind speed along with streamlines of the flow.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2025

Generalizability for Representation Learning on Downstream Tasks
A Study of Downstream Performance in a
Nonlinear ICA setting using Contrastive Learning
Anton Levinsson & Ziyuan Wang
Department of Computer Science and Engineering
Chalmers University of Technology

Abstract

We study the theoretical generalizability of representations learned by contrastive learning by analyzing their performance in downstream linear regression tasks. To quantify the quality of these learned features, we provide rigorous proofs for the worst-case and expected downstream performances under specified assumptions. These results offer theoretical results for analyzing the quality of features by looking at downstream tasks. To empirically verify the theory, we conduct experiments on both simulated and real-world data, following time-contrastive learning (TCL) strategies, which solve the problem of nonlinear independent component analysis (nonlinear ICA). In the real-world setting, we construct nonlinear source separation tasks using mixed audio signals from different instrument categories. Then, we propose some specific downstream tasks to verify our theories using our learned features from the TCL method and compare with the observed features. The results show that most of the defined downstream tasks are located in the confidence level of the expected performance and are bounded by the worst case, which indicates that our theory aligns with the real-world setting.

Keywords: contrastive learning, generalizability in downstream, nonliear ICA, linear regression.

Acknowledgements

We would like to express our sincere gratitude to our supervisor, Ahmet Zahid Balcıoğlu, for his guidance, support, and constructive feedback throughout the course of this thesis. His insights and encouragement have been important in shaping the direction and quality of this work.
We are also deeply grateful to our examiner, Fredrik D. Johansson, for his helpful feedback and for ensuring that the work met the necessary academic standards. His input during the process has been greatly appreciated.

Anton Levinsson & Ziyuan Wang, Gothenburg, June 2025

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Full Term
CI	Confidence Interval
ICA	Independent Component Analysis
InfoNCE	Information Noise-Contrastive Estimation
MCC	Mean Correlation Coefficient
MLP	Multilayer Perceptron
MSE	Mean Square Error
OLS	Ordinary Least Squares
R^2	Coefficient of Determination
ReLU	Rectified Linear Unit
TCL	Time-Contrastive Learning

1

Introduction

Obtaining effective features that are generalizable from neural networks has always been a challenging problem. One of the possible solutions is self-supervised learning [2, 3], which is an effective learning paradigm to learn high-dimensional representations from data without labels. In paper [2, 4, 5], the learned features from self-supervised methods can achieve competitive results and generalizable features in downstream tasks, with regard to image and text datasets.

Contrastive learning (CL)[6, 7] is a particularly successful subgroup of self-supervised techniques that use the representational distance between sample pairs. The goal of contrastive learning is to construct a representation space, where similar data points (positive pairs) are close together, and dissimilar data points (negative pairs) are far apart. Independent component analysis (ICA) [8] has been analyzed using contrastive learning as a tool to achieve blind source separation from observations with a demixing function f^{-1}. However, compared to linear ICA, nonlinear ICA tasks can help to find complex independent components if f is nonlinear. But the biggest hindrance is that the model would not be identifiable (capacity to recover the latent variables that generate the data) if there is no temporal or similar structure in the data[9]. Some solutions [10, 11] are made using deep learning to solve nonlinear ICA. Recently, a new framework has been proposed to achieve identifiability[1, 12, 13] through contrastive learning. Some auxiliary variables are introduced, making latent variables conditionally independent if the distribution of sources is provided, like the exponential family. Solving nonlinear ICA through contrastive learning allows for the recovery of features up to a linear transformation, making it a powerful tool for studying contrastive learning.

However, a systematic understanding of how different CL objectives influence generalizability across diverse downstream tasks is still lacking. We will therefore seek new theories based on previous works and conduct experiments on both synthetic and real-world data to guide us to study learned representations from contrastive learning under nonlinear ICA settings. The next step is to investigate the worst and expected performance of the learned representations in downstream tasks in a regression setting.

Our project comprises four main contributions. First, closed-form theoretical expressions are derived that characterize the worst-case and expected downstream performance of contrastive learning representations. Second, we performed exten-

sive experiments on synthetic data to verify that the results align with our theory and analyzed the potential factors that impact the quality of learned representations. Finally, we also performed experiments in a real-world setting in order to generalize our theorem, using the specific downstream tasks. The results showed that most of our downstream tasks fell within the confidence region defined by our bound.

1.1 Context

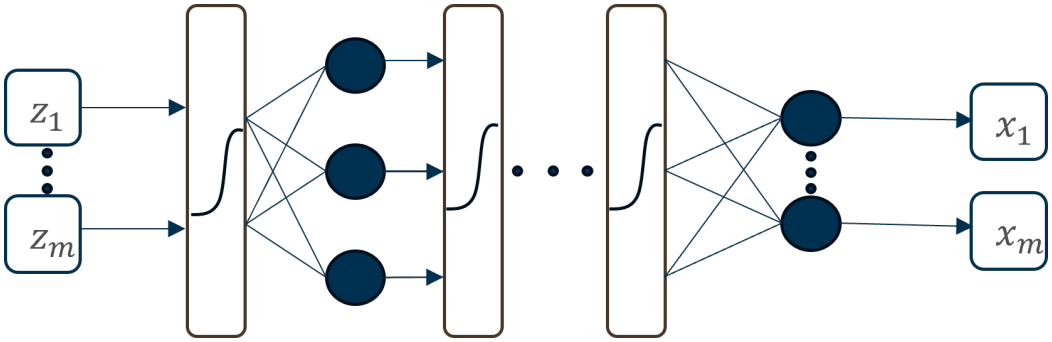
There is a sizeable body of work to understand contrastive learning and its effectiveness in downstream tasks [6, 14]. Multiple works are to prove the bound for the sample complexity in the downstream tasks. Others attempt to unravel the roles of negative samples when conducting contrastive learning or connect contrastive learning to the data generating process. The authors in [15] argue that the learned representations from CL can reduce the complexity of the labeled sample on downstream tasks with InfoNCE loss. Additionally, [16] showed that contrastive learning is closely relevant to the data-generating process. However, as suggested by Evgenia Rusak et al. in [17], there is still a gap between theory and practice when conducting CL and the extent to which CL produces features that generalize across downstream tasks remains an open question. An (ϵ, δ)-measure in paper [18] is proposed to prove the upper bound of the population loss for the downstream classification task - that is, with probability at least $1 - \delta$, the population loss does not exceed the empirical loss by more than ϵ - and the authors in paper [19] demonstrated that the generalization bound for CL does not depend on negative examples, but up to logarithmic terms, better than in paper [15].

To close this gap, this study is about whether contrastive representations retain meaningful information about latent source variables in a nonlinear ICA setting. Our focus is not only on how well these representations support downstream tasks, but also on when and why they fail to do so. By analyzing both worst-case and expected downstream performance mainly in a controlled simulation setting where the true sources are known, we aim to better understand the conditions under which contrastive learning generalize.

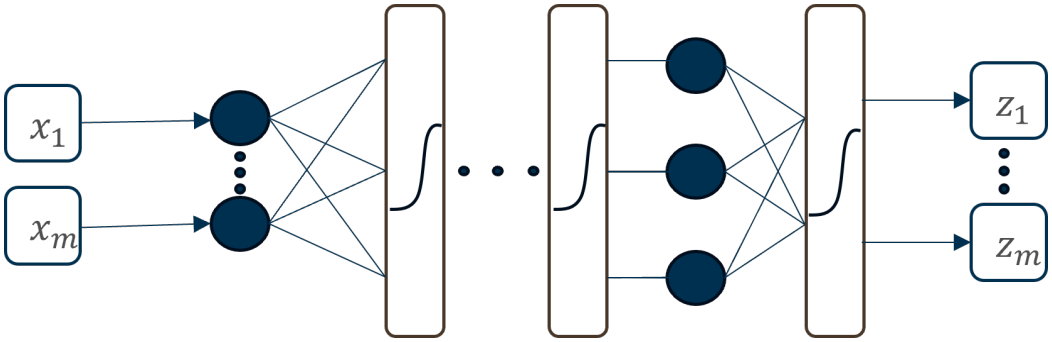
1.2 Problem

The first part of our problem is illustrated in Figure 1.1, which includes the mixing step and the learning steps. In the mixing step, the source data z is mixed by a certain nonlinear function, for example, a multilayer perceptron with activation ahead, to obtain the observed data x. This mixing step is part of the data-generating process and is not under our control. For the learning step, the use of a neural network aims to reverse the mixing step and recover the source variables s. As mentioned in Section 1 we will work in an ICA setting in which prior work [1, 12, 13], has shown that the source data z, assumed to consist of statistically independent components, can be recovered up to a linear transformation. This affine identifiability is

crucial in our setting, as it guarantees that the learned features have a meaningful structure from the original sources, making it possible to evaluate their usefulness in downstream tasks.



(a) Mixing step: The source data is transformed by a nonlinear function to produce observed signals.



(b) Learning step: A neural network attempts to recover the original sources by learning a mapping that approximates the inverse of the transformation.

Figure 1.1: Mixing step (a) versus learning step (b). The figure illustrates part of our problem setup. In the mixing step, latent source data is transformed by a nonlinear function to produce observed signals. In the learning step, a neural network attempts to recover the original sources by learning a mapping that approximates the inverse of the transformation.

Hence, it is intuitive to investigate how well our learned features capture the latent variables that are relevant for downstream tasks, where labels in the downstream are linearly determined by the latent variables. In order to quantify how good our learned representations are, we resort to find the worst downstream task and also the expected downstream performance given by a certain learned model.

More specifically, we attempt to answer the following questions in this thesis:

1. How can we identify the worst-case downstream task, that is, the linear function of the true latent variables for which our learned model performs the worst? What is the loss for the worst downstream task? What factor affects the performance of the worst downstream tasks?

2. How to quantify expected downstream performance in all possible downstream

tasks? What kind of information does expected downstream performance convey to us?

3. How far are our assumptions from real-world data? How does our theory generalize in the real-world case, given a specific downstream tasks?

1.3 Limitations

Linear Downstream Task In this work, we focus on evaluating tasks defined by a linear function of the true latent variables. This setting does not always hold in real-world applications. For example, hard classification tasks often require complex nonlinear decision boundaries, which are typically handled by adding multilayer perceptrons or other neural network architectures to model these nonlinearities. Hence, for our linear downstream task, the loss may be high even for an oracle model, which might not be evident and promising when analyzing a deficient model.

In the background of ICA Most of our research is conducted under the assumption of ICA, that is, data generated according to a group of (conditionally) independent components. However, in a real-world setting, the latent variables may not always be independent in a nonlinear setting. Although clear conclusions can potentially be drawn under our ICA setting, the performance of downstream tasks under conditions where the assumptions of ICA are violated remains unknown.

Noisy Latent Variables One notable point is that, in all related work, source noise is never mentioned. In other words, it remains unclear how the presence of noise in the sources would affect both the model and the corresponding downstream tasks. Here, we refer specifically to the noise added to the latent variables z themselves. However, if the noise is not complicated, for instance, noise that is independent of sources or white noise that is uniformly distributed in the sources, the algorithm can treat this noise as another component or can separate sources because of the independence of noise.

Simulated and Real-World Data A key limitation when analyzing the performance on downstream tasks and the relationship between downstream loss and contrastive learning loss is the differences between simulated and real-world data. In simulated settings, we have full control over the data generation process, allowing us to explicitly define the independent components and test theoretical assumptions. This enables us to quantify how well contrastive learning recovers the underlying structure and how it affects downstream performance. For real-world data, the true latent structure is unknown, and factors such as noise and dataset biases are possible additional challenges that are not present when it comes to simulated data. This gap between simulated and real world-data highlights the importance of our proposed theorem to understand how well our controlled experiments translates to practical downstream tasks.

1.4 Notation

Throughout this report, capitalized letters are used to represent matrices, and variables in lowercase represent vectors, unless otherwise stated. Below, the main symbols, operators, and notations used throughout this thesis are presented.

Table 1.1: Notation used throughout the thesis.

Functions and Operators	
$\mathbb{P}(\cdot)$	Probability of certain event.
$\mathbb{E}(\cdot)$	Expectation operator over a random variable.
$\mathrm{Var}(X)$	Variance of a random variable X.
$\mathrm{tr}(\cdot)$	Trace of a matrix.
Observed and Latent Data	
x_t	The observed time series data at time t.
$z_t = f^{-1}(x_t)$	Ground-truth independent components that generate the observed signals (simulated data).
$\hat{h}(x_t)$	The learned representation extracted from x_t by the model $\hat{h}$.
$H = [h_1, \ldots, h_N]^\top \in \mathbb{R}^{N \times n}$	Matrix for latent features for the underlying generation process; each row is one observation of the latent variables.
$\hat{H} = [\hat{h}_1, \ldots, \hat{h}_N]^\top \in \mathbb{R}^{N \times n}$	Matrix of recovered latent variables, where each row is the model's estimate of the latent variables for one observation.
$P = \hat{H}(\hat{H}^\top \hat{H})^{-1} \hat{H}^\top$	Projection matrix onto the column space of the recovered latent features $\hat{H}$.
Downstream Task Variables	
β	A downstream task vector in the source space that represents a direction along which target values are defined.
β_i	i^{th} component of the downstream task vector.
$\hat{\beta}$	The optimal linear predictor trained on the learned features $\hat{h}(x)$.
$y_i = \beta^\top f^{-1}(x_i)$	The true downstream label.
$\hat{y}_i = \hat{\beta}^\top \hat{h}(x_i)$	The predicted downstream label.
$\mathcal{L}_{ds}^{w}$	Worst downstream loss where the index w indicates the worst-case task.
$\bar{\mathcal{L}}_{ds}$	Expected downstream loss, where the bar denotes an average over downstream tasks or data.

1.5 Outline

In this chapter, we have introduced our motivation for the thesis, to study the generalisibility of contrastive learning in the setting of ICA. In Chapter 2, we will introduce the problem formulation mathematically, and a relevant background is provided to better understand our attempt to solve the problems. In Chapter 3, we present the main theoretical results from our work, including our proof for the close form of the worst downstream and expected downstream. In Chapter 4, the experimental setup is introduced for both simulated and real-world data. Afterwards, Chapter 5 presents the results of simulated data and real-world data, to support our findings from Chapter 3 and answer our research questions. In Chapter 6, we discuss our theoretical limitations and give suggestions for our future work. Finally, in Chapter 7, the conclusion of our thesis is provided.

2

Background

For the background section, we begin by recalling linear regression along with some useful corollaries, which is essential for our following theorem in Chapter 3. We will also cover the basic idea of independent component analysis and the time contrastive learning method (TCL) from paper [1].

2.1 Linear Regression

Linear regression is a fundamental machine learning method that is used to model the relationship between a set of input variables and a continuous output variable. In the context of our theoretical framework, it provides a basis for analyzing how well learned representations can capture target information. The following subsections introduce key concepts in linear regression, including ordinary least squares, projection matrix, and the coefficient of determination.

2.1.1 Ordinary Least Squares (OLS)

In our project, we focus mainly on the generalizibility of the regression downstream task of contrastive learning. In this section, we briefly introduce the basic idea of linear regression and relevant theories.

We begin with the standard linear regression model, where the observed labels $Y \in \mathbb{R}^N$ are assumed to follow a linear relationship with the feature matrix $X \in \mathbb{R}^{N \times d}$:

$$Y = X\beta + \varepsilon,$$

where $\beta \in \mathbb{R}^d$ is the true (but unknown) regression parameter, and $\varepsilon \in \mathbb{R}^N$ is the error term capturing the discrepancy between the model and the observed data.

In order for the linear model to provide valid estimates, the following classical assumptions are typically made [20]:

1. **Linearity and additive errors**: The relationship between the dependent variable Y and the input variables X can be captured linearly. The error term is additive.

2. **Independence**: The errors ε_i are independent across observations.
3. **Homoscedasticity**: The error terms have a constant variance for all, i.e., $\text{Var}(\varepsilon_i) = \sigma^2$ for all i.
4. **No multicollinearity**: The columns of X are linearly independent, i.e., $X^\top X$ is full rank and invertible.
5. **Zero-mean errors**: The error term has zero mean, i.e., $\mathbb{E}[\varepsilon] = 0$.

Under these assumptions, we derive the OLS estimator $\hat{\beta}$ by minimizing the residual sum of squares:

$$\hat{\beta} = \arg\min_{\beta} \|X\beta - Y\|_2^2.$$

Expanding the objective:

$$\|X\beta - Y\|_2^2 = (X\beta - Y)^\top (X\beta - Y),$$

then take the gradient with respect to β and set it to zero:

$$\nabla_\beta \left(\|X\beta - Y\|_2^2 \right) = 2X^\top (X\beta - Y) = 0.$$

Solving the normal equation results in:

$$X^\top X\hat{\beta} = X^\top Y, \tag{2.1}$$

from which we obtain the closed-form solution:

$$\hat{\beta} = (X^\top X)^{-1} X^\top Y. \tag{2.2}$$

The estimator from 2.2 is unbiased under the assumptions above, satisfying:

$$\mathbb{E}[\hat{\beta}] = \mathbb{E}[(X^\top X)^{-1} X^\top (X\beta + \varepsilon)] = \mathbb{E}[\beta] + (X^\top X)^{-1} X^\top \mathbb{E}[\varepsilon] = \beta.$$

However, for the fitted model, estimates of the error terms are called residuals, which are defined as:

$$\hat{\varepsilon} = Y - X\hat{\beta}. \tag{2.3}$$

A useful corollary about residuals is

$$X^\top \hat{\varepsilon} = 0,$$

which directly obtains from 2.1 by moving terms on both sides and factorizing $X^\top$. This corollary reveals that the observation vector for each feature is orthogonal to the residual vectors. If the intercept is considered in the model, i.e. X includes column of ones, then $\mathbf{1}^\top \hat{\varepsilon} = 0$, which means that

$$\sum_{i=1}^{N} \hat{\varepsilon}_i = 0,$$

.

2.1.2 Projection Matrix

Given a feature matrix $X \in \mathbb{R}^{n \times d}$ with full column rank, the *projection matrix* is defined as [21]:

$$P = X(X^\top X)^{-1} X^\top.$$

This matrix $P \in \mathbb{R}^{n \times n}$ projects any vector $Y \in \mathbb{R}^n$ onto the subspace spanned by the columns of X, i.e:

$$\hat{Y} = PY.$$

The projection matrix P satisfies the following properties:

1. **Idempotent:** $P^2 = P$

   Proof. We compute P^2:

   $$\begin{aligned} P^2 &= X(X^\top X)^{-1} X^\top X (X^\top X)^{-1} X^\top \\ &= X(X^\top X)^{-1} \left(X^\top X\right) (X^\top X)^{-1} X^\top \\ &= (X^\top X)^{-1} (X^\top X) = I. \end{aligned}$$

   Thus, P is idempotent. □

2. **Symmetric:** $P^\top = P$

   Proof. We compute the transpose of P:

   $$\begin{aligned} P^\top &= \left(X(X^\top X)^{-1} X^\top\right)^\top \\ &= \left(X^\top\right)^\top \left((X^\top X)^{-1}\right)^\top X^\top \\ &= X(X^\top X)^{-1} X^\top = P. \end{aligned}$$

   Proving that P is symmetric. □

2.1.3 Coefficient of Determination

Once the estimated coefficients $\hat{\beta}$ are obtained, the predicted response vector is given by $\hat{Y} = X\hat{\beta}$.

The coefficient of determination, denoted R^2, is a statistical metric that evaluates how well the regression model explains the variability in the dependent variable based on the independent variables. It is defined as:

$$R^2 = 1 - \frac{\|Y - \hat{Y}\|_2^2}{\|Y - \bar{Y}\mathbf{1}\|_2^2}, \tag{2.4}$$

where $\bar{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i$ and where Y is the actual observed value, $\hat{Y}$ is the predicted value from the model.

Intuitively, R^2 measures the proportion of variance in the target variable that is explained by the model. An R^2 value of 1 indicates perfect prediction, which means that all predicted values exactly match the observed values. An R^2 of 0 implies that the model performs no better than simply predicting the mean $\bar{Y}$ for all observations. Negative R^2 values suggest that the model performs worse than the mean predictor, indicating a poor fit.

2.2 Nonlinear Independent Component Analysis

Our goal is to study the downstream performance for representation learning. In order to develop theoretical guarantees, we make use of nonlinear ICA as a framework and make use of identfiability guarantees made in the literature.

Theoretical setting We study a setting similar to [13], where we assume that the data instance x_i and the auxiliary variables $u_i \in \mathbb{R}^d$ are generated by a nonlinear smooth and invertible transformation f through a latent variable z. The process can be defined as $f : \mathcal{Z} \to \mathcal{X}$, where $x \subseteq \mathbb{R}^k$ is the space of observations and $z \subseteq \mathbb{R}^n$ is the space of latent variables, $n \leq k$. The auxiliary variable can be either a vector, a categorical label, or combination of both. We assume that the latent variable z_i is conditionally exponentially distributed given u_i:

$$\log P(z_i|u_i) = \log \frac{Q_i(z_i)}{Z_i(u)} + \sum_{j=1}^{M} q_{i,j}(s_i)\lambda_{i,j}(u)$$

where $q_{i,j}$, $\lambda_{i,j}$, Q_i and Z_i are scalar-valued functions and T is the order of exponential distribution.

One special case of this is the data generating distribution for TCL [1] where the auxiliary variable is the time index of a stationary time series, such as

$$\log P(z_i|\tau) = q_{i,0}(z_i) + \sum_{j=1}^{M} q_{i,j}(s_i)\lambda_{ij}(\tau) - \log Z(\lambda_{i,1}(\tau), ..., \lambda_{i,M}(\tau)), \tag{2.5}$$

where τ indicates the segment index, $q_{i,0}$ is the stationary baseline log-pdf of latent z_i and Z is a normalization constant.

Dataset We observe the dataset $D = \{(x_i, u_i)\}_i \in [N]$ for contrastive learning, where x_i represents the observed data samples and u_i denotes the auxiliary variable.

Contrastive learning for ICA Contrastive learning is a well-known solution for nonlinear ICA. Potential solutions make use of the auxiliary variable to invert the transformation f. In the example case of TCL, a feature extractor $h(x)$ is trained with a multiclass linear classifier to predict the auxiliary variable [1]. They have shown that such a training process inverts the data generating process and identifies $\hat{f}^{-1}$ as a linear transformation of h. The TCL framework is shown in Figure 2.1 from [1].

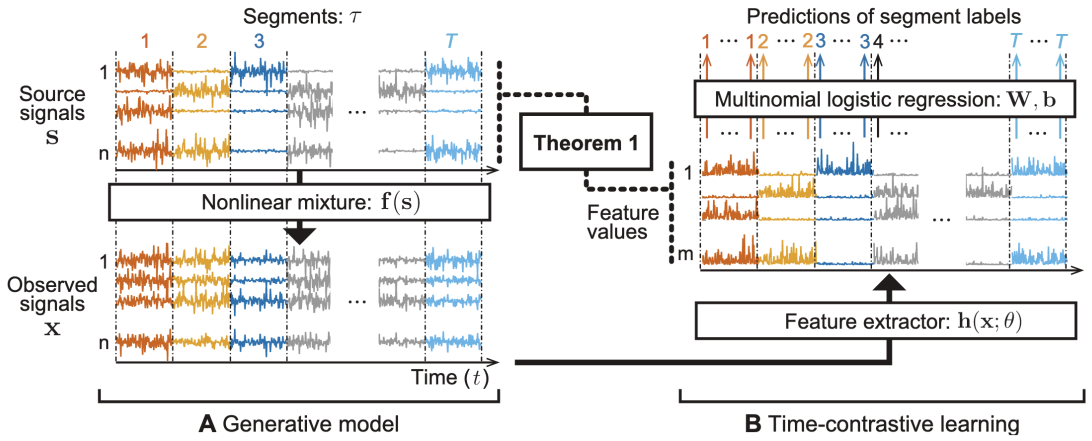


Figure 2.1: A nonlinear ICA model is learned with TCL. (A) The generative model assumes observed signals result from a nonlinear transformation of independent sources. (B) TCL trains a feature extractor to capture this nonstationarity using multinomial logistic regression, classifying data points by segment labels 1, ... , T [1].

2.2.1 ICA in Real-World

A classic example of blind source separation is the cocktail party problem, where multiple speakers talk simultaneously and we observe only mixtures of their voices through several microphones[22]. The goal is to recover the individual sources, each speaker, from the observed mixtures, each microphone, without knowing the mixing process or having access to clean source signals.

In such real-world scenarios, the observed signals are nonlinear and often lack access to clean source references or to the mixing process itself. Nevertheless, certain structural assumptions can make unsupervised separation feasible. One such assumption is segment-wise nonstationarity, where the distribution of the sources changes over time, which can be used to extract meaningful representations.

TCL is possible to use in such settings. When the data can be segmented into time intervals with statistically distinct source behavior, and when multiple views or mixtures are available per segment, TCL can be used to learn features that align with the underlying sources.

2.3 Downstream Performance

We are interested in studying the downstream task performance of representations learned from contrastive learning and developing theoretical guarantees for the learned representations. In the ICA literature, it is studied that using contrastive learning can invert the data generating process f up to certain transformations [1], our goal is to use these identifiablity guarantees to quantify the quality of the learned representations through different methods of contrastive learning, such as multinomial classification [1], regression [9], and InfoNCE loss [23].

For the model h that we learned for contrastive learning, we investigate the setting of downstream tasks $D_{ds} = \{Y_i\}_{i=1}^{N}$ with the indices i corresponding to the indices of instances in the dataset D.

2.3.1 Downstream Regression Tasks

In particular, if the downstream task is approached using linear regression, we consider the empirical loss of the optimal linear predictor (i.e. the OLS estimator) for a given target direction β.

$$\mathcal{L}(\hat{h}; \beta) = \min_{\hat{\beta}} \mathbb{E}_D[\Big((\hat{\beta}^\top \hat{h}(x) - \beta^\top z\Big)^2 |\beta] \tag{2.6}$$

This problem can be solved using an OLS estimator, and the matrix solution is added in the Appendix A.1.

Hence, it is interesting to ask what the worst case is given our learned features. We will look into the worst-case linear problem, where $y_i = \beta^\top f^{-1}(x_i)$:

$$\mathcal{L}_{ds}^{w} = \max_{\beta} \mathcal{L}(\hat{h}; \beta) \tag{2.7}$$

or the expected downstream linear problem:

$$\bar{\mathcal{L}}_{ds} = \mathbb{E}_{\beta \sim \mathbf{B}}\Big[\mathcal{L}(\hat{h}; \beta)\Big] \tag{2.8}$$

where $\mathbf{B} = \{\beta \in \mathbb{R}^N : \|\beta\| = 1\}$ represents all unit-norm linear downstream tasks. This normalization allows us to focus on the alignment between the learned representations and the ground-truth rather than the magnitude of β, which can be scaled as desired. Equation 2.7 indicates that we want to find the potential worst downstream task for any learned features and check how well we can achieve on this worst case. The potential solution for 2.7 is through the optimization method (eg: projected gradient ascent [24]). Equation 2.8 is used to measure the average performance on all downstream tasks, which can be solved by sampling β.

2.3.2 Max-Min Over Min-Max

While min-max formulations are often used in machine learning to account for difficult or uncertain tasks, our approach instead uses a max-min formulation. In a max-min setup, we fix a solution, in our case the learned representations through OLS and ask: *What is the worst task we can face and how well can we perform on it?* This is what is later referred to as *worst-case downstream performance*. In contrast, a min-max approach would instead aim to find a representation that performs well under the hardest possible task.

One reason we chose the max-min approach is that the inner minimization is a convex problem, meaning that it can be solved efficiently and exactly. The min-max approach, on the other hand, would involve non-convex optimization, making it harder to solve in practice.

2.3.3 Uniform Distribution on a Hypersphere

Our theories in Chapter 3 are based on the uniform distribution on the N-dimensional sphere. Hence, several basic properties are introduced.

Theorem 2.3.1 (Uniform distribution on a hyper-sphere [25])**.** *Let $\beta \sim \mathcal{N}(\mathbf{0}, I_n)$ be a standard multivariate normal vector in $\mathbb{R}^n$. Then the normalized vector*

$$u = \frac{\beta}{\|\beta\|_2}$$

is uniformly distributed on the unit sphere $\mathbb{S}^{n-1} = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\|_2 = 1\}$.

This result implies that a simple and efficient method to sample uniformly from the unit sphere is to draw from a multivariate standard normal distribution and normalize the resulting vector. Besides, Theorem 2.3.1 highlights the invariant rotation of normalized vectors from standard multivariate normal distribution. Moreover, it follows by symmetry that the expected value $\mathbb{E}(u)$ equals to zero.

Lemma 2.3.1. *Let $u \sim \mathrm{Unif}(\mathbb{S}^{n-1})$ and let $Q \in \mathbb{R}^{n\times n}$ be any orthogonal matrix (i.e., $Q^\top Q = I$). Then the rotated vector Qu also satisfies*

$$Qu \sim \mathrm{Unif}(\mathbb{S}^{n-1}).$$

This property is known as *rotational invariance*. It implies that the uniform distribution on the sphere is invariant under orthogonal transformations. For a rigorous proof, see [25].

For spherical symmetry, due to the spherical symmetry (or orthogonal invariance) of the uniform distribution on the unit sphere $\mathbb{S}^{n-1}$, all coordinates u_i of $u \sim \text{Unif}(\mathbb{S}^{n-1})$ have identical marginal distributions, and satisfy $\mathbb{E}[u_i] = 0$ for all $i = 1, \ldots, n$.

3

Methods

The central aim of this project is to evaluate the downstream performance of learned representations in a nonlinear ICA setting. This setting provides access to ground-truth latent variables, enabling a precise assessment of representation quality when identifiability is achieved. Building on the TCL framework[1], we analyze theoretical bounds and performance guarantees for downstream tasks under this assumption. Experiments are conducted through simulated data, and real-world will be backed up in the following sections.

The composition of this chapter is divided into three parts, brief introduction of the TCL theorem, theory for worst downstream performance, and theory for expected downstream performance.

3.1 Time-Contrastive Learning and nonlinear ICA

As introduced in Chapter 2, Hyvärinen (2016) [1] proposed an intuitive but delicate method of unsupervised learning for time series data by leveraging its nonstationary structure. He also showed that nonlinear ICA can be solved up to a linear transformation by using TCL, which is the first identifiability result for nonlinear ICA.

Theorem 3.1.1. *If the data generating process is defined as presented in Equation 2.5, with $q_{i,0}(z_i) = 0$ and $M = 1$, z and x having the same dimension, after learning the parameters of the feature extractor $\hat{h}$, the learned features $\hat{h}(x)$ are equal to z up to a linear transform.*

This theorem is based on three main assumptions. Data generation should be carried out according to Equation (2.5), i.e. exponential family. Then, this family should be in the case of $q_{i,0}(z_i) = 0$ and $M = 1$, which means that in this exponential family, each source is associated with a single modulating function $q_{i,1}$ and remains consistent across all sources.

Theorem 3.1.1 describes the success of TCL for nonlinear ICA. If it is possible to distinguish between segments, the model should learn something useful and identifiability of nonlinear ICA can be achieved.

3.2 Regression Downstream Performance Guarantee

We will first conduct studies to examine the generalizability of our learned features from contrastive learning in the regression setting.

Linear regression performance is evaluated using the estimated independent components, and compared to the performance obtained using oracle features. This was done in two different settings, expected downstream performance and worst downstream performance. To evaluate how well the learned representations generalize to typical downstream tasks, the *expected downstream performance* was assessed. This formulation serves as a baseline for downstream analysis before analyzing the worst-case scenario.

3.2.1 Assumptions For Downstream Tasks

Before the theoretical contribution is presented, the underlying assumptions must first be stated.

Assumption 3.2.1 (Linear and deterministic downstream task)**.** *The label in the downstream task is given by the linear combination of source data, i.e. given a latent vector,* $y_i = \beta^\top z_i$.

Assumption 3.2.2 (Spherical parameter domain)**.** *The parameter vector* β *for the downstream task lies on the unit sphere, i.e.,* $\beta \in S^{n-1}$.

Assumption 3.2.3 (Noise-free source data)**.** *The source data are assumed to be noise-free. Although the sources are drawn from a distribution, no additional noise is included in the generative process.*

The above assumptions are made to simplify the analysis and enable tractable theoretical results. The linearity assumption ensures that the downstream task can be interpreted directly in terms of the latent sources. Constraining the parameter β to lie on the unit sphere makes it less difficult for us to reach the theoretical result. Assuming noise-free source data eliminates confounding factors from the generative process, though this may limit the direct applicability of the results to real-world settings.

3.2.2 Worst-Case Downstream Performance

To assess the *worst-case downstream performance*, a max-min objective was formulated, as shown in Formula 2.7, where the optimal $\hat{\beta}$ conditioned on a fixed β was found using an OLS estimator. Then, the worst-case β that maximizes the loss was calculated using gradient ascend. A closed-form was also shown in the next chapter. The worst case β corresponds to the eigenvector associated with the largest eigenvalue of this matrix, which is stated formally in Theorem 3.2.1.

For the simplicity of illustration for our theorem, we stack all the latent variables

into matrix and define the matrix $H = [z_1, ..., z_N]^\top \in \mathbb{R}^{N \times n}$, as the matrix for latent features for the underlying generation process and each row is one single observation for latent variable. Similarly, $\hat{H} = [\hat{h}_1, ..., \hat{h}_n]^\top \in \mathbb{R}^{N \times n}$ is the matrix for the recovered latent variables. Besides, we define the projection matrix $P = \hat{H}(\hat{H}^\top \hat{H})^{-1}\hat{H}^\top$, which has the same properties specified in the previous section. The following mathematical derivation is based on matrix form. Especially, the matrix form for R square is added in the appendix.

Lemma 3.2.1. $\max_{x^\top x=1} x^\top Ax = \lambda_{\max}(A)$ *if* A *is a symmetric matrix, where* $\lambda_{\max}(A)$ *is the largest eigenvalue of matrix* A *[21].*

Theorem 3.2.1. *The solution to the worst downstream loss* $\mathcal{L}_{ds}^w$ *defined as 2.7, has a closed form solution:*

$$\mathcal{L}_{ds}^w = \frac{\lambda_{max}\left(H^\top (I-P)H\right)}{N}, \tag{3.1}$$

and the worst downstream task $\beta_{\texttt{worst}}$ *is the corresponding eigenvector.*

Proof. We can write the problem in matrix form

$$\max_{\beta \in \mathbf{B}} \min_{\hat{\beta}} \frac{||\hat{H}\hat{\beta} - H\beta||_2^2}{N}$$

For the inner minimization problem, it is a classic linear regression problem, with the label vector defined as $H\beta$. Hence, the OLS estimator is the solution and can be plugged into the formula.

$$\hat{\beta}^*(\beta) = (\hat{H}^\top \hat{H})^{-1}\hat{H}^\top H\beta$$

We plug the OLS estimator in the problem and use the symmetric and idempotent properties of the projection matrix, leading to the following result.

$$\mathcal{L}_{ds}^w = \max_{\beta \in \mathbf{B}} \frac{||(P-I)H\beta||_2^2}{N} = \max_{\beta \in \mathbf{B}} \frac{\beta^\top H^\top (I-P)H\beta}{N}$$

Because $\beta \in \mathbf{B}$, we can refer to Lemma 3.2.1 and the final result was concluded:

$$\mathcal{L}_{ds}^w = \max_{\beta \in \mathbf{B}} \frac{\beta^\top H^\top (I-P)H\beta}{N} = \frac{\lambda_{max}(H^\top (I-P)H)}{N}$$

□

For Lemma 3.2.1, it is similar to the well-known Rayleigh Quotient, which is computed as $\max \frac{x^\top Ax}{x^\top x}$. Our version requires the norm of x to be one.

For Theorem 3.2.1, the proof combines the first lemma with the application of the OLS estimator. The true label in our case is defined as $H\beta$. This theorem gives the closed form of loss in the worst downstream task, depending on one projection matrix P and a source matrix H. If the source data is fixed, this loss only depends

on the learned features output from the neural network. Moreover, the following Theorem 3.2.4 points out that the linear transformation of the learned features does not affect the downstream loss. This is aligned with the basic OLS estimator, which depends only on the basis of the projection matrix, and linear transformation will not affect its basis. This theory also reveals that even if we do not apply an ICA for final features, the downstream loss stays the same under our above-mentioned assumptions.

Another way to support our theorem is by utilizing projected gradient ascent to solve for β and satisfy the constraint $\beta \in \mathbf{B} = \{\beta \in \mathbb{R}^N : \|\beta\|| = 1\}$.

$$\nabla_\beta \left(\frac{\beta^\top H^\top (I - P) H \beta}{N} \right) = 2 \frac{H^\top (I - P) H \beta}{N}.$$

And the update step for β is

$$\beta'_{\text{new}} = \beta_{\text{old}} + \alpha' H^\top (I - P) H \beta_{\text{old}}.$$

$$\beta_{\text{new}} = \frac{\beta'_{\text{new}}}{\|\beta'_{\text{new}}\|}$$

where $\alpha' = \frac{2\alpha}{N}$ is the step size, because N and 2 are scalars that can be included in the original α.

In Figure 3.1, it is clear that even with different starting points, the results of the projected gradient ascent always converge to our theoretical closed-form solution.

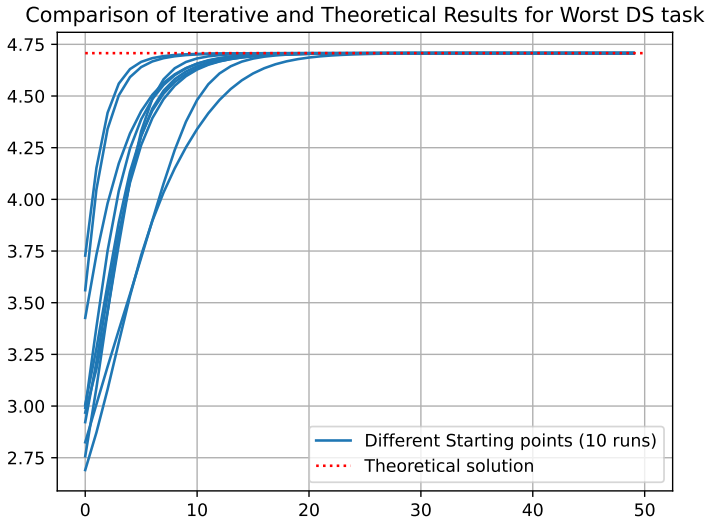


Figure 3.1: Comparison between iterative results and closed form results with ten different initialized points from standard multivariate Gaussian distribution with dimension 20 before normalization with unit norm. α is set at 0.0005 and number of iterations is set at 50 steps.

3.2.3 Expected Downstream Performance

For the analysis of expected downstream performance, it is done by sampling random linear functions β, projecting the ground-truth sources and the estimated components onto these directions, and fitting an OLS estimator to recover each projection as shown in 2.8. Averaging this over many random β directions provides an estimate of how well the learned representation aligns with typical tasks that depend linearly on the source signals.

Before going to theorems, several lemmas are needed to introduce for better understanding the following theories.

Lemma 3.2.2. *If $\beta \in \mathbb{R}^n$ is uniform distributed on n-dimensional sphere, then for the expectation over the monomial of components $\beta_{i_1}\beta_{i_2}\dots\beta_{i_k}$ is zero if k is odd, or if any index $j \in \{1,\dots,n\}$ appears an odd number of times in the multiset $\{i_1,\dots,i_k\}$.*

Proof. Because β is uniformly on the n-dimensional sphere, we can flip the sign of each component of β, yielding β', which has the same distribution as β because of symmetry of the distribution. Hence,

$$\mathbb{E}\left[\beta_{i_1}\dots\beta_{i_k}\right] = (-1)^k \cdot \mathbb{E}\left[\beta_{i_1}\dots\beta_{i_k}\right].$$

If k is odd, this flips sign, so:

$$\mathbb{E}[\beta_{i_1}\dots\beta_{i_k}] = -\mathbb{E}[\beta_{i_1}\dots\beta_{i_k}] = 0.$$

Now suppose k is even, but there is an index j that appears an odd number of times. Consider the reflection of β that flips the sign of coordinate j only. This transformation preserves the distribution (rotational invariance) but flips the sign of the monomial. Hence, the expectation remains zero. □

Lemma 3.2.3. *If $\beta \in \mathbb{R}^n$ is uniform distributed on n-dimensional sphere, then for the expectation over the product from its two arbitrary components β_i and β_j,*

$$\mathbb{E}\left[\beta_i \beta_j\right] = \begin{cases} 0 & \text{if } i \neq j \\ \frac{1}{n} & \text{if } i = j \end{cases}.$$

Proof. For every single entry $\beta_i = \frac{x_i}{\|x\|}$,

$$\mathbb{E}\left[\beta_i \beta_j\right] = \mathbb{E}_{x\sim\mathcal{N}}\left[\frac{x_i x_j}{\|x\|^2}\right],$$

where x is from the multivariate standard normal distribution $\mathcal{N}$.

Then, we need to investigate the case if i equals j.

If $i \neq j$, we resort to Lemma 3.2.2, both β_i and β_j appear one time, yielding zero.

If $i = j$, $\mathbb{E}\left[\beta_i \beta_j\right] = \mathbb{E}_{x\sim\mathcal{N}}\left[\frac{x_i^2}{\|x\|^2}\right]$. Because for each entry of x, they are i.i.d. Hence, $\mathbb{E}_{x\sim\mathcal{N}}\left[\frac{x_i^2}{\|x\|^2}\right]$ has the same value for all i.

$$\begin{aligned}\mathbb{E}\left[\beta_i \beta_j\right] = \mathbb{E}\left[\beta_i^2\right] = \mathbb{E}_{x\sim\mathcal{N}}[\frac{x_i^2}{\|x\|^2}] = \frac{1}{dim(\beta)}\mathbb{E}_{x\sim\mathcal{N}}\left[\frac{dim(\beta)x_i^2}{\|x\|^2}\right] \\ = \frac{1}{dim(\beta)}\mathbb{E}_{x\sim\mathcal{N}}\left[\sum_{i=1}^{dim(\beta)}\frac{x_i^2}{\|x\|^2}\right] = \frac{1}{dim(\beta)} = \frac{1}{n}\end{aligned}$$

□

Then, our theory for the expected loss on all possible downstream tasks is introduced. As defined in 2.8, the downstream task is drawn from the uniformly distributed n-dimensional sphere. The average loss is computed to serve as an indicator of the overall alignment between the learned features and the latent structure.

Theorem 3.2.2. *If x is drawn from a multivariate standard normal distribution $\mathcal{N}$, $\beta = \frac{x}{\|x\|}$, $\bar{\mathcal{L}}_{ds}$ defined in 2.8 can be computed by*

$$\frac{tr(A)}{N \cdot n}, \tag{3.2}$$

where matrix A is defined as $H^\top(I-P)H$, and N is the total number of samples in the matrix.

Proof. With the help of Theorem 3.2.1, we can simplify our formulated term in the matrix form, changing the maximization problem into expectation problem.

$$\begin{aligned}
\bar{\mathcal{L}}_{ds} &= \mathbb{E}_{\beta\sim\mathbf{B}}\left[\frac{\|(P-I)H\beta\|_2^2}{N}\right] = \mathbb{E}_{\beta\sim\mathbf{B}}\left[\frac{\beta^\top H^\top (I-P)H\beta}{N}\right] \\
&= \frac{1}{N}\mathbb{E}_{\beta\sim\mathbf{B}}\left[\beta^\top A\beta\right] = \frac{1}{N}\mathbb{E}_{\beta\sim\mathbf{B}}\left[\sum_{i,j}\beta_i\beta_j A_{ij}\right] \\
&= \frac{1}{N}\sum_{i,j} A_{ij}E_{\beta\sim\mathbf{B}}\left[\beta_i\beta_j\right] \\
&= \frac{1}{N}\sum_{i\neq j} A_{ij}E_{\beta\sim\mathbf{B}}\left[\beta_i\beta_j\right] + \frac{1}{N}\sum_{i=j} A_{ij}E_{\beta\sim\mathbf{B}}\left[\beta_i\beta_j\right] \\
&= \frac{\mathrm{tr}(A)}{N\cdot n} \quad \text{(from Lemma 3.2.3)}
\end{aligned}$$

□

The theorem for the expected downstream task is verified through simulation as well. We first sample a group of beta from the standard multivariate normal distribution and then normalize each β with its norm. Afterwards, using each sampled normalized β to compute the downstream loss and then average the loss over different β. The simulated results can be seen in Figure 3.2. As the number of sampled β increases, the average loss appears to converge to the closed-form value derived in Theorem 3.2.2, represented by the red dotted line.

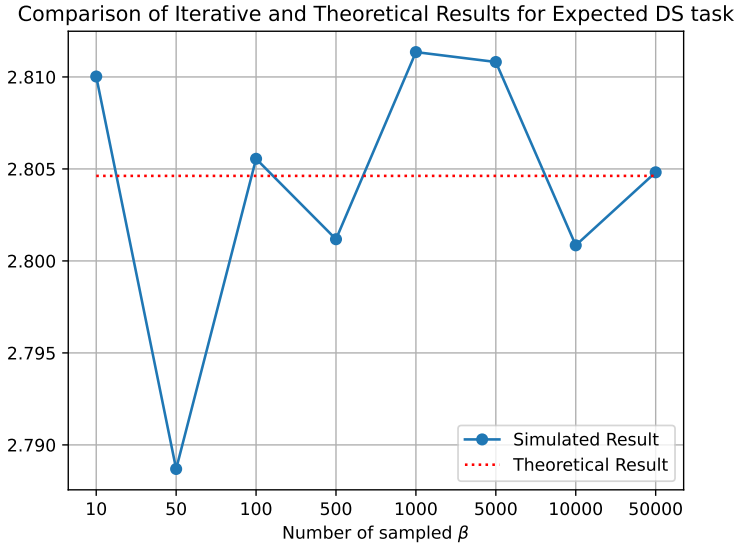


Figure 3.2: Comparison between simulated results and closed form results with different number of β.

Lemma 3.2.4. *If $\beta \in \mathbb{R}^n$ is uniformly distributed on n-dimensional sphere, then for any two different components β_i, β_j, the joint distribution of (β_i, β_j) is invariant*

under orthogonal transformations in the (β_i, β_j) plane. In particular,

$$(\beta_i, \beta_j) \overset{d}{=} \left(\frac{\beta_i + \beta_j}{\sqrt{2}}, \frac{\beta_i - \beta_j}{\sqrt{2}} \right),$$

i.e., the joint distribution is rotationally invariant in any 2D coordinate subspace.

Proof. Let $Q \in \mathbb{R}^{n \times n}$ be an orthogonal matrix that acts as the identity on all coordinates except for the i-th and j-th, where it applies the 2D rotation:

$$Q_{[i,j]} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Formally, Q can be written as a block matrix:

$$Q = \operatorname{diag}(I_{i-1}, Q_{[i,j]}, I_{n-j}),$$

where I_{i-1} and I_{n-j} are identity matrices of size $i-1$ and $n-j$.

From Lemma 2.3.1, we have

$$\beta \sim \text{Unif}(\mathbb{S}^{n-1}) \quad \Rightarrow \quad Q\beta \sim \text{Unif}(\mathbb{S}^{n-1}).$$

Let $\beta' = Q\beta$. Then β' is uniformly distributed on the sphere, and its (i, j) components are:

$$(\beta'_i, \beta'_j) = \left(\frac{\beta_i + \beta_j}{\sqrt{2}}, \frac{\beta_i - \beta_j}{\sqrt{2}} \right).$$

Therefore, by the invariance of the distribution under Q,

$$(\beta_i, \beta_j) \overset{d}{=} \left(\frac{\beta_i + \beta_j}{\sqrt{2}}, \frac{\beta_i - \beta_j}{\sqrt{2}} \right).$$

□

Lemma 3.2.5. *If $\beta \in \mathbb{R}^n$ is uniform distributed on n-dimensional sphere, then for the expectation over its four different arbitrary components β_i, β_j, β_k and β_l,*

$$\mathbb{E}\left[\beta_i \beta_j \beta_k \beta_l\right] = \frac{3}{n(n+2)}.$$

Also, for the expectation over its two arbitrary components square,

$$\mathbb{E}\left[\beta_i^2 \beta_j^2\right] = \frac{1}{n(n+2)}.$$

Proof. From Lemma 3.2.4, we can conclude

$$\mathbb{E}[\beta_1^2 \beta_2^2] = \mathbb{E}\left[\left(\frac{\beta_1 + \beta_2}{\sqrt{2}} \right)^2 \left(\frac{\beta_1 - \beta_2}{\sqrt{2}} \right)^2 \right] = \frac{1}{4} \mathbb{E}\left[(\beta_1 + \beta_2)^2 (\beta_1 - \beta_2)^2 \right].$$

Expanding the product and using symmetry gives:

$$\mathbb{E}[\beta_1^2\beta_2^2] = \frac{1}{4}\left(2\mathbb{E}[\beta_1^4] - 2\mathbb{E}[\beta_1^2\beta_2^2]\right),$$

which implies:

$$\mathbb{E}[\beta_1^4] = 3\,\mathbb{E}[\beta_1^2\beta_2^2].$$

In order to compute these two expectations, we expand the square of the norm:

$$\mathbb{E}\left[\left(\sum_{i=1}^{n}\beta_i^2\right)^2\right] = \mathbb{E}\left[\sum_{i=1}^{n}\beta_i^4 + \sum_{i\neq j}\beta_i^2\beta_j^2\right] = n\,\mathbb{E}[\beta_1^4] + n(n-1)\,\mathbb{E}[\beta_1^2\beta_2^2].$$

Since β is uniformly distributed on the unit sphere, we have:

$$\sum_{i=1}^{n}\beta_i^2 = 1 \Rightarrow \mathbb{E}\left[\left(\sum_{i=1}^{n}\beta_i^2\right)^2\right] = 1.$$

Therefore,

$$n\,\mathbb{E}[\beta_1^4] + n(n-1)\,\mathbb{E}[\beta_1^2\beta_2^2] = 1.$$

Combining this with the earlier identity $\mathbb{E}[\beta_1^4] = 3\,\mathbb{E}[\beta_1^2\beta_2^2]$, we obtain:

$$n\cdot 3\mathbb{E}[\beta_1^2\beta_2^2] + n(n-1)\,\mathbb{E}[\beta_1^2\beta_2^2] = 1,$$

which yields:

$$\mathbb{E}[\beta_1^2\beta_2^2] = \frac{1}{n(n+2)}, \quad \mathbb{E}[\beta_1^4] = \frac{3}{n(n+2)}.$$

□

Theorem 3.2.3. *If x is drawn from a multivariate standard normal distribution $\mathcal{N}$, $\beta = \frac{x}{\|x\|}$, the variance of empirical downstream loss $\mathcal{L}(\hat{h};\beta)$ defined in 2.6, $\sigma_{\hat{h}}^2$, can be computed by*

$$\sigma_{\hat{h}}^2 = var_{\beta\sim\mathbf{B}}\left[\mathcal{L}_{ds}(\hat{h};\beta)\right] = \frac{1}{N^2}\left[\frac{2tr(A^2)+tr(A)^2}{n(n+2)} - \frac{tr(A)^2}{n^2}\right], \tag{3.3}$$

where matrix A is defined as $H^\top(I-P)H$, and N is the total number of samples in the matrix.

Proof. The variance of downstream loss is defined as

$$\sigma_{\hat{h}}^2 = var_{\beta\in\mathbf{B}}\left[\mathcal{L}_{ds}(\hat{h};\beta)\right] = var_{\beta\sim\mathbf{B}}\left[\frac{1}{N}\beta^\top A\beta\right] = \mathbb{E}_{\beta\sim\mathbf{B}}\left[\left(\frac{1}{N}\beta^\top A\beta\right)^2\right] - \mathbb{E}_{\beta\sim\mathbf{B}}\left[\frac{1}{N}\beta^\top A\beta\right]^2.$$

The second has been solved in Theorem 3.2.2, which is given by $\frac{1}{N^2\cdot n^2}tr(A)^2$.

For the first term,

$$\begin{aligned}\mathbb{E}_{\beta\sim\mathbf{B}}\left[\left(\frac{1}{N}\beta^\top A\beta\right)^2\right] &= \frac{1}{N^2}\mathbb{E}_{\beta\sim\mathbf{B}}\left[\left(\beta^\top A\beta\right)^2\right] = \frac{1}{N^2}\mathbb{E}_{\beta\sim\mathbf{B}}\left[\left(\sum_{i,j}\beta_i\beta_j A_{ij}\right)\left(\sum_{k,l}\beta_k\beta_l A_{kl}\right)\right] \\ &= \frac{1}{N^2}\mathbb{E}_{\beta\sim\mathbf{B}}\left[\left(\sum_{i,j,k,l}\beta_i\beta_j\beta_k\beta_l A_{ij}A_{kl}\right)\right] \\ &= \frac{1}{N^2}\sum_{i,j,k,l}\mathbb{E}_{\beta\sim\mathbf{B}}\left[\beta_i\beta_j\beta_k\beta_l\right]A_{ij}A_{kl}\end{aligned}$$

Refer to Lemma 3.2.2, we have that $\mathbb{E}_{\beta\sim\mathbf{B}}\left[\beta_i\beta_j\beta_k\beta_l\right] = 0$ whenever all indices $i \neq j \neq k \neq l$ are distinct. The same conclusion holds when exactly two indices are equal and the other two are distinct.

Finally, this term turns out to be :

$$\begin{aligned}\mathbb{E}_{\beta\sim\mathbf{B}}\left[\left(\frac{1}{N}\beta^\top A\beta\right)^2\right] &= \frac{1}{N^2}\left[\sum_i \mathbb{E}_{\beta\sim\mathbf{B}}\left[\beta_i^4\right]A_{ii}^2 + \sum_{\substack{i,k\\ i\neq k}}\mathbb{E}_{\beta\sim\mathbf{B}}[\beta_i^2\beta_k^2]A_{ii}A_{kk}\right] \\ &= \frac{1}{N^2}\left[\sum_i \frac{3A_{ii}^2}{n\left(n+2\right)} + \sum_{\substack{i,k\\ i\neq k}}\frac{A_{ii}A_{kk}}{n\left(n+2\right)}\right] \quad \text{(from Lemma 3.2.5)} \\ &= \frac{1}{N^2}\left[\sum_i \frac{2A_{ii}^2}{n\left(n+2\right)} + \sum_{i,k}\frac{A_{ii}A_{kk}}{n\left(n+2\right)}\right] \quad \text{(rearrange terms)} \\ &= \frac{1}{N^2}\left[\frac{2}{n\left(n+2\right)}\sum_i A_{ii}^2 + \frac{1}{n\left(n+2\right)}\sum_i A_{ii}\sum_k A_{kk}\right] \\ &= \frac{1}{N^2}\left[\frac{2tr(A^2)+tr(A)^2}{n(n+2)}\right]\end{aligned}$$

Hence, $\sigma_{\hat{h}}^2 = var\beta \sim \mathbf{B}\left[\mathcal{L}_{ds}(\hat{h};\beta)\right] = \frac{1}{N^2}\left[\frac{2tr(A^2)+tr(A)^2}{n(n+2)} - \frac{tr(A)^2}{n^2}\right]$

□

3.2.4 Downstream Property in our setting

Theorem 3.2.4. *$\mathcal{L}_{ds}^w$ and $\bar{\mathcal{L}}_{ds}$ is invariant to $\hat{H}$ with an invertible linear transformation.*

Proof. For $\mathcal{L}_{ds}^w$ and $\bar{\mathcal{L}}_{ds}$, they are all decided by the projection matrix P of the learned features. Suppose that there is an invertible transformation $C^\top \in \mathbb{R}^{n\times n}$, which transforms the feature matrix $\hat{H}$ into $\hat{H}C$.

When we compute the new projection matrix for the transformed features, we need to replace $\hat{H}$ with $\hat{H}C$. Then, we have

$$P = \hat{H}C(C^\top \hat{H}^\top \hat{H}C)^{-1}C^\top \hat{H}^\top = \hat{H}CC^{-1}\hat{H}^\top \hat{H}(C^\top)^{-1}C^\top \hat{H}^\top = P$$

which shows P remains the same and then concludes the theorem. □

Theorem 3.2.4 indicates that once the learned features are obtained, the downstream performance is not affected by the application of ICA to the learned features to obtain more separated sources. This theorem also coincides with the fact that there exists a corresponding transformation of the regression coefficients such that the prediction, and hence the performance metrics such as mean squared error, remain unchanged. Therefore, whether or not ICA is applied to learned features does not have an influence on downstream performance, as long as the downstream model is flexible enough to adapt.

Theorem 3.2.5. *$\bar{\mathcal{L}}_{ds}$ specified in theorem 3.2.2 is equivalent to compute the average performance for $dim(\beta)$ orthogonal sparse downstream tasks L_{avg}, where each orthogonal sparse downstream task $\beta^{(t)}, t \in [1, dim(\beta)]$ is defined as $\beta_k^{(t)} = \begin{cases} 0 & \text{if } k \neq t \\ 1 & \text{if } k = t \end{cases}$.*

Proof. If the average loss for $\dim(\beta)$ downstream tasks are computed, it follows that:

$$\begin{aligned}
\mathcal{L}_{avg} &= \frac{1}{N \cdot \dim(\beta)} \sum_{t=1}^{\dim(\beta)} {\beta^{(t)}}^\top A \beta^{(t)} \\
&= \frac{1}{N \cdot \dim(\beta)} \sum_{t=1}^{\dim(\beta)} \sum_{ij} \beta_i^{(t)} \beta_j^{(t)} A_{ij} \\
&= \frac{1}{N \cdot \dim(\beta)} \sum_{t=1}^{\dim(\beta)} A_{tt} \\
&= \frac{1}{N \cdot \dim(\beta)} tr(A) \\
&= \bar{\mathcal{L}}_{ds}
\end{aligned}$$

□

This theorem highlights the underlying method to analyze expected downstream performance. Rather than sampling from an n-dimensional sphere, we only need to look at the sparse $\dim(\beta)$ downstream tasks. In addition to that, there is an explanation for each downstream task, because $H\beta^{(t)}$ indicates selecting the t^{th} column of source signals as label, which means that the downstream task is just to predict the t^{th} dimension of the source data.

Theorem 3.2.6. *The empirical downstream loss $\mathcal{L}(\hat{h};\beta)$ is sub-Gaussian with parameter $\sigma = \frac{\mathcal{L}_{ds}^{w}}{2}$, and satisfies the following Gaussian tail bound:*

$$\mathbb{P}\left(|\mathcal{L}(\hat{h};\beta) - \bar{\mathcal{L}}_{ds}| > t\right) \leq 2\exp\left(-\frac{t^2}{2\sigma^2}\right).$$

Proof sketch. We omit the full proof here, but since $\mathcal{L}(\hat{h};\beta)$ is supported on a bounded interval $[0, \mathcal{L}_{ds}^{w}]$, it follows directly that it is sub-Gaussian with parameter $\sigma = \frac{\mathcal{L}_{ds}^{w}}{2}$. □

This implies that the concentration of the empirical downstream loss around its mean $\bar{\mathcal{L}}_{ds}$ becomes tighter when the worst-case downstream performance $\mathcal{L}_{ds}^{w}$ is small.

Corollary 3.2.1. *Given confidence level $1-\alpha$, the empirical downstream loss $\mathcal{L}(\hat{h};\beta)$ satisfies the following confidence interval:*

$$\mathcal{L}(\hat{h};\beta) \in \left[\bar{\mathcal{L}}_{ds} - \frac{\mathcal{L}_{ds}^{w}}{2}\cdot\sqrt{2\ln\left(\frac{2}{\alpha}\right)},\ \bar{\mathcal{L}}_{ds} + \frac{\mathcal{L}_{ds}^{w}}{2}\cdot\sqrt{2\ln\left(\frac{2}{\alpha}\right)}\right]$$

with probability at least $1-\alpha$.

4

Experimental Setup

This section describes the experimental setup used to evaluate our theoretical findings. We begin by detailing how the simulated source signals were generated and mixed following prior work. We then outline our real-world audio-based experiments in both dataset construction and two different types of mixing. Finally, the downstream tasks used in a real-world setting are presented.

4.1 Simulated Data

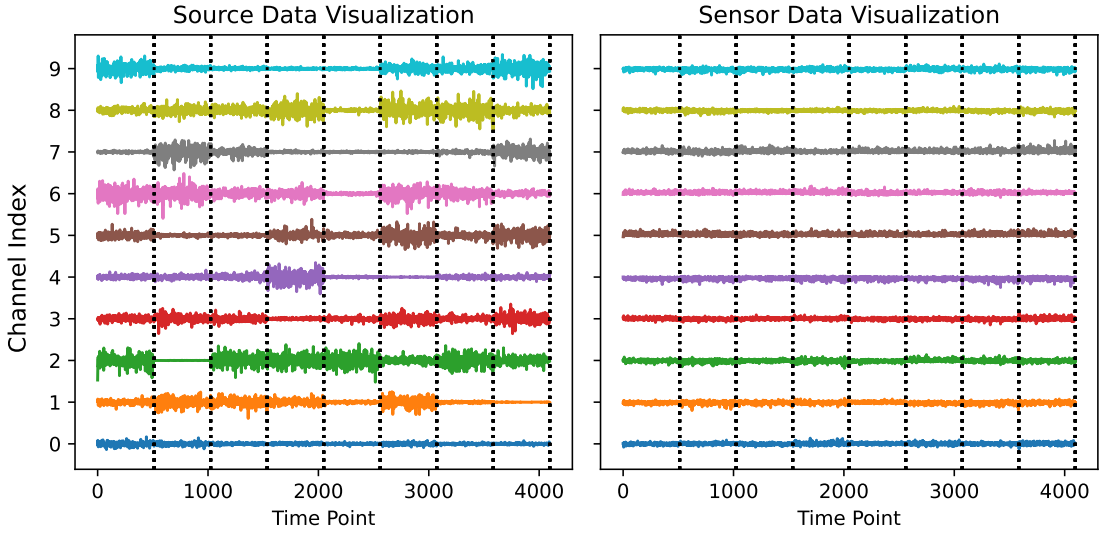


Figure 4.1: An example for simulated data generated from Laplacian distribution. For sensor data, it is mixed by a three-layer MLP. In the picture, only the first ten channels of signals are presented.

Data Generation The data generation process was carried out in the same way as in [1], generating simulated source signals with 20 channels and 512 numbers of observations in each segment from the Laplacian distribution. For each channel of the source signals, they are mutually independent. In order to modulate in a nonstationary manner, each channel within each segment of the source should be modulated by a value (i.e. $\lambda_{ij}(\tau)$ in Equation 2.5), randomly drawn from a uniform

distribution. As for nonlinear mixing function, the number of layers for mixing is decided first and the same number of orthogonal matrices are generated randomly to do mixing. Before each linear mixing, the activation function (Leaky Relu with negative slope 0.2) was used. A visualization of our simulated data in the first ten channels is shown in Figure 4.1.

Training Procedure The multilayer perceptron is trained as a feature extractor, as in the TCL paper [1], which is illustrated in Figure 1.1(b). The number of layers (L) for feature extractor is the same as for the mixing-MLP in the mixing step in Figure 1.1(a). The feature extractor uses the maxout activation function, which can be seen as a form of implicit feature selection. Specifically, effectively double the number of channels and then select the maximum over pairs before reducing it back to the original size. As for nonlinear mixing function, the same function presented in the previous paragraph was used, a mixing-MLP which is initialized by a group of orthogonal vectors with activation function(Leaky Relu with negative slope 0.2) ahead of each linear layer. To tune the hyperparameters, the entire dataset was stratified and split into training, validation, and test sets with a ratio of 70:15:15. Then the feature extractor was trained by feeding with the training set, and the Adam optimizer was used with default betas of 0.9 and 0.95, weight decay of 0.0001 and learning rate of 0.001. The batch size is 4096.

Metrics For the results, we report the most commonly used performance metrics for the evaluating of downstream tasks, including mean squared error (MSE) and the coefficient of determination (R^2). These two metrics are fundamental for linear regression tasks, measuring the average prediction error and the proportion of variance in the labels explained by the model. In order to evaluate the recovery of the source variables, we use the mean correlation coefficient (MCC), which calculates the mean of the correlations resulting from a bipartite matching of source and recovered latent variables from a cross-correlation matrix of randomized dependence coefficient (RDC) [26].

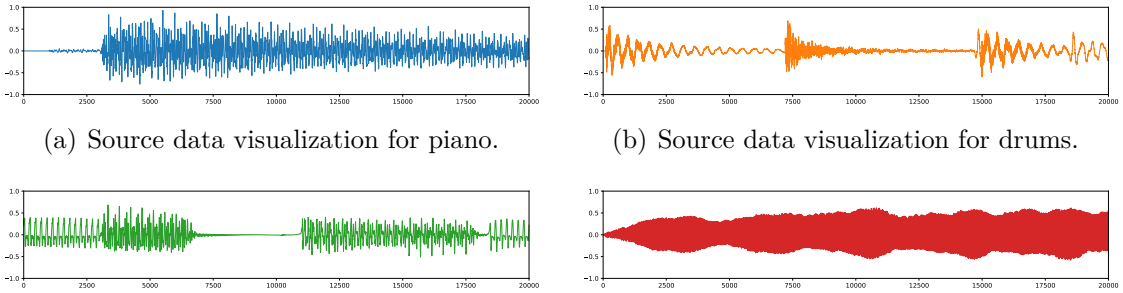
4.2 Real-World Data

In order to verify our theorems in the real-world case, practical datasets were used to carry out similar experiments and certain downstream tasks are proposed to verify our theory.

4.2.1 Data Preparation

In deciding which data to use in a real-world setting, three things were taken into account. Firstly, we wanted the data to represent a realistic case of blind source separation in order to be able to have meaningful downstream tasks. Second, in order to be able to use TCL as a framework for feature extraction, our data should have nonstationary structure. Finally, we wanted access to the original source signals to perform different mixing and also to be able to evaluate how well our learned representations perform in downstream tasks.

Dataset For our dataset, several instrument pieces were selected using the "Musical Instrument's Sound Dataset" available on Kaggle [27]. In this large dataset, four types of instruments were selected to conduct experiments: piano, violin, drums, and guitar. The sound waves for each instrument are shown in 4.2, with the amplitude on the y-axis and the x-axis representing time points. The sampling rate is fixed at 20kHz.



(a) Source data visualization for piano.

(b) Source data visualization for drums.

(c) Source data visualization for guitar.

(d) Source data visualization for violin.

Figure 4.2: One second audio samples for the four different instruments from the dataset.

Mixing Method To satisfy the temporal nonstationarity, variations over time were introduced by augmenting the source signals across segments. Individually, each source in Figure 4.2 was repeated and segmented to match the specified number of segments, and amplitude augmentations were applied independently to each of the segments that were used as true source data, which is a similar modulation in the simulated data. The sensor data was prepared in two different ways. The first way was through linear mixing as illustrated in 4.3. z_1 and z_2 represent the true source signals. The two microphones pick up one mixture each from the instruments where the more distant source is attenuated by a factor w. In doing this, two different mixtures were created and used as our sensor data. This setup was extended and three and four instruments were used to carry out our experiments. In the case with three instruments, for each microphone, the closest instrument used its original sound wave, the second closest was reduced to 0.5 and the farthest was reduced to 0.3. For four instruments, the setting was similar to that for three instruments, with the additional instrument being reduced to 0.5. Regarding the nonlinear mixing option, the same procedure explained in Section 4.1 was utilized, the source was centered with zero mean and then modulated. Figure 4.4 first presents eight segments of source data where each source has been modulated segment-wise and the right figure visualizes the sensor data after nonlinear mixing.

Training Procedure The main difference in the real-world dataset compared to the simulated data is that we now have 20.000 data points per segment, since the sample rate is 20kHz, instead of 512 data points in the simulated data. Therefore, the experiments were conducted with 4 and 8 segments, with a downsampled sampling rate of 8000, resulting in fewer data points compared to the original raw sound track. Four different instruments were used, and we used all possible combinations with

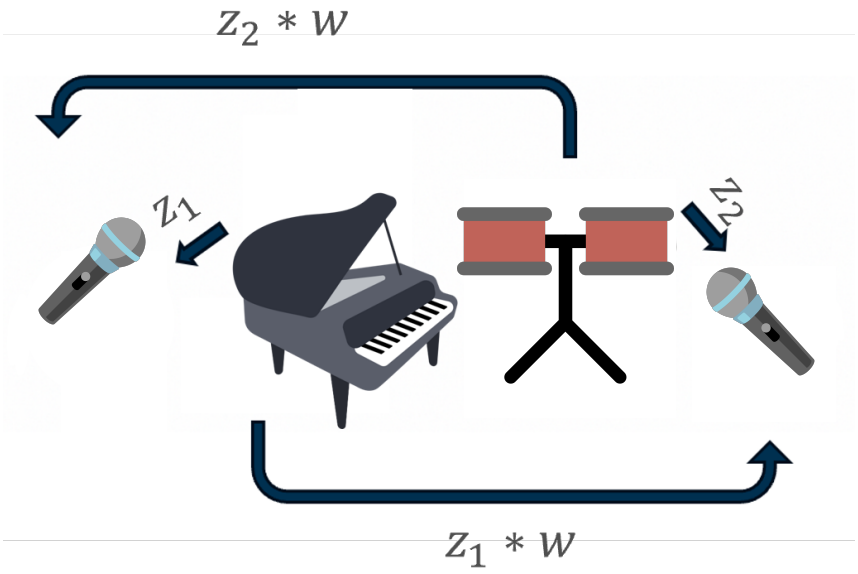


Figure 4.3: Our mixing step for real-world experiments where two instruments(piano and drums in this case) are used, z_1 and z_2 corresponding to the piano and drums. They are recorded by two microphones where w describes a decrease in amplitude for the instrument farther away. Piano icon by `Vecteezy.com`, drum and microphone icons from SVGRepo.

three instruments, respectively, and conducted experiments, both in the linear and nonlinear case. We also modified the feature extractor accordingly, as the author did in [1]. Dropout(dropout parameter set to 0.2) [28] and batch normalization [29] are used in every linear layer. For the activation function in the last linear layer, it is replaced with PRelu for better flexible representations. The training parameters remain the same as the simulated data.

Experimental Setup A realistic case of blind source separation is achieved using musical instruments. Nonstationary sources are introduced through segment-wise augmentation or modulation of the source data. Additionally, by creating a custom dataset, it becomes possible to evaluate the effectiveness of the learned representations on downstream tasks. The next step was the execution of downstream task experiments.

Defined Downstream Tasks The next step, in order to verify our theoretical results, was to perform downstream tasks. Since we worked with audio time series, downstream tasks are not possible at each data point. Instead, a sliding window was used to extract the labels. We set the window size to 500 data points with 500 hop size, which means that there is no overlapping between different windows. Then we assigned labels to each data point using the extracted label. The following downstream tasks were performed where T denotes the time points in each window i:

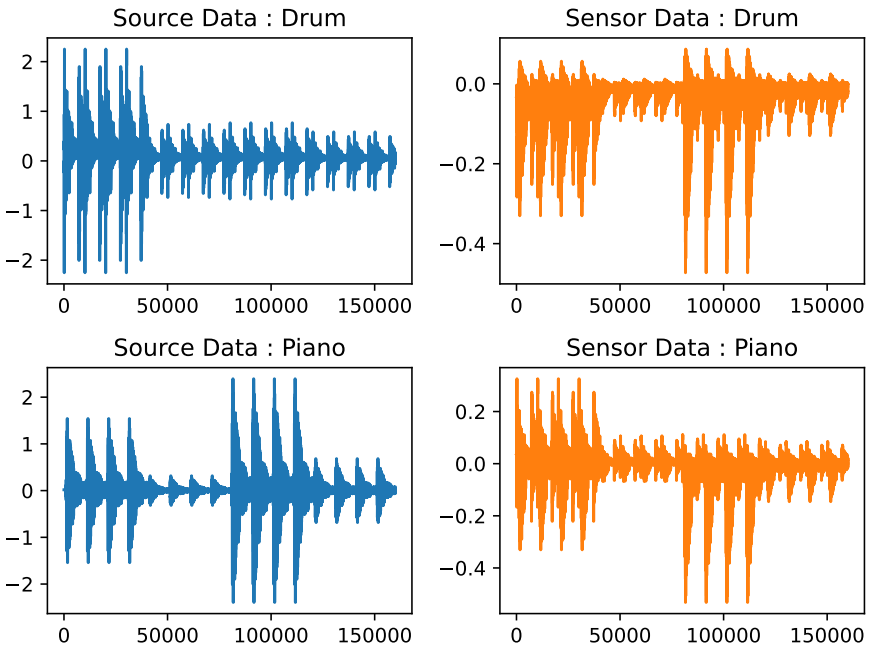


Figure 4.4: Comparison of ground truth and mixed sensor signals nonlinear mix

- **Mean Amplitude Prediction:** Predict the average amplitude of each instrument within each window:

$$\mu^{(i)} = \frac{1}{T}\sum_{t=1}^{T}|x_t^i|$$

- **RMS Amplitude Prediction:** Predict the root mean square amplitude of each instrument within each window, capturing both magnitude and energy:

$$RMS^{(i)} = \frac{1}{T}\sum_{t=1}^{T}\left(x_t^{(i)}\right)^2$$

- **Max Amplitude Prediction:** Predict the highest amplitude of each instrument in each window.

$$Max^{(i)} = \max_t |x_t^i|$$

- **Peak-to-Peak Amplitude Prediction:** Predict the difference between the maximum and minimum amplitudes of each source within each window:

$$PeakToPeak^{(i)} = \max_t |x_t^i| - \min_t |x_t^i|$$

- **Standard Deviation:** Predict the standard deviation of the amplitude values for each source in the window:

$$\sigma^{(i)} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(x_t^{(i)} - \mu^{(i)}\right)^2}$$

These tasks were designed to evaluate how well the learned representations capture relevant source characteristics in a blind source separation setting. By measuring performance on these tasks, we can assess the informativeness and generalizability of the representations with respect to the underlying generative factors, thus providing empirical support for our theoretical analysis.

5

Results

The reported results will showcase the effectiveness of the learned representations for downstream tasks. For simulated data, the expected and worst downstream performance is presented along with metrics that help to theoretically back up the results. These are presented through different experiments that help explain how various factors affect downstream performance. In addition to looking at the results obtained, we also investigated potential causes behind this.

5.1 Simulated Data

To validate our theoretical framework and evaluate the quality of the learned representations in practice, we examined a variety of experimental configurations and empirical metrics, which are the same experimental settings as presented in Section 4.1. Ranging the number of layers from 1 to 5, and also the number of segments from 8 doubling all the way to 512. Then, the MCC, MSE, and R^2 were computed. Our baseline for experiments consisted of having the number of layers in the mixing-MLP between 1 and 5 while having the feature-MLP between 1 and 5 layers, respectively. This setup allowed us to look into how downstream performance is affected when the feature extractor is of matching complexity as the mixing-MLP. The result in this section presents both the worst and expected downstream performance.

5.1.1 Mean Correlation Coefficient

The MCC in Figure 5.1, between the learned features and the true sources, peaks at intermediate depths and declines with higher segment counts, which is slightly different from the results in the TCL paper [1]. The main reason is that there is no dataset split in their original experiments and they computed the metrics based on the training set, where there is a risk of overfitting. It is true that an increasing number of segments can benefit the recovery of the source data, however, if the number of segments is really large, it will jeopardize the learning step, indicating that distinguishing between segments is a hard task.

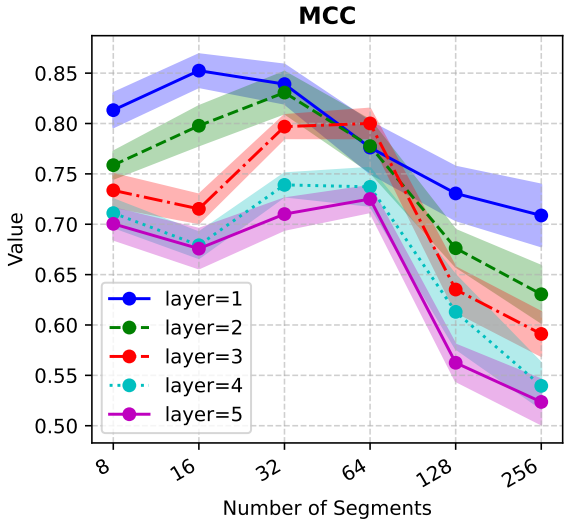


Figure 5.1: The reproduced results for mean correlation, having matching depth of feature and mixing-MLP, the same settings used in the TCL paper.

5.1.2 Worst-Case Downstream Performance

For the worst downstream performance, the main result is shown in Figure 5.2, which includes MSE before and after ICA and R^2.

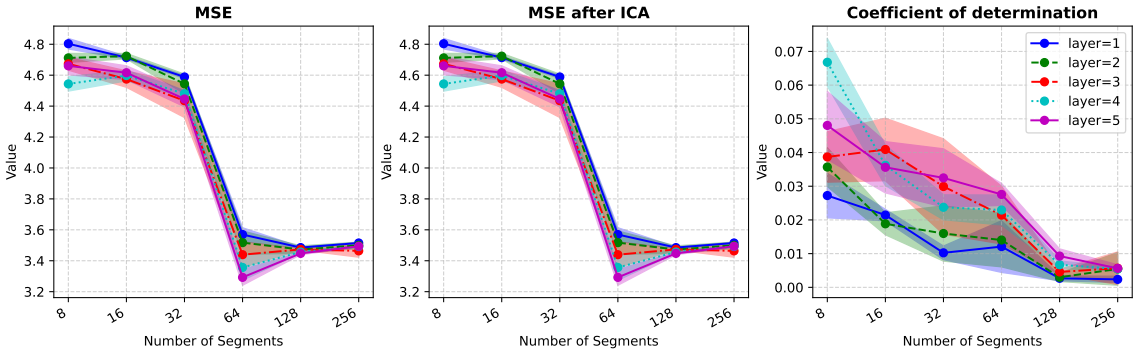
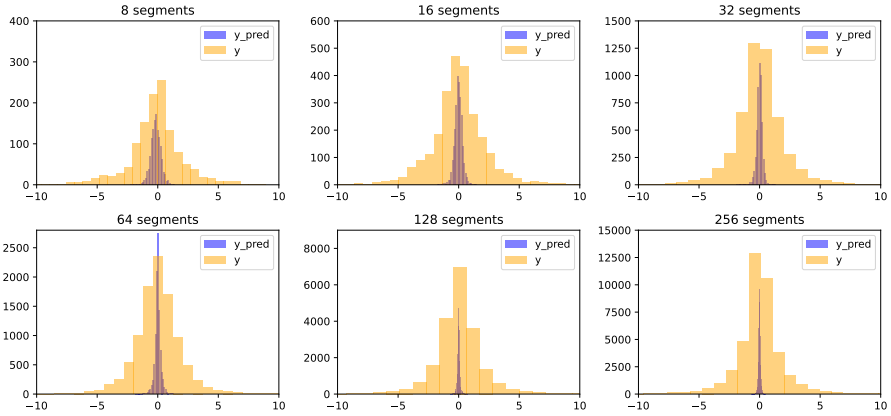


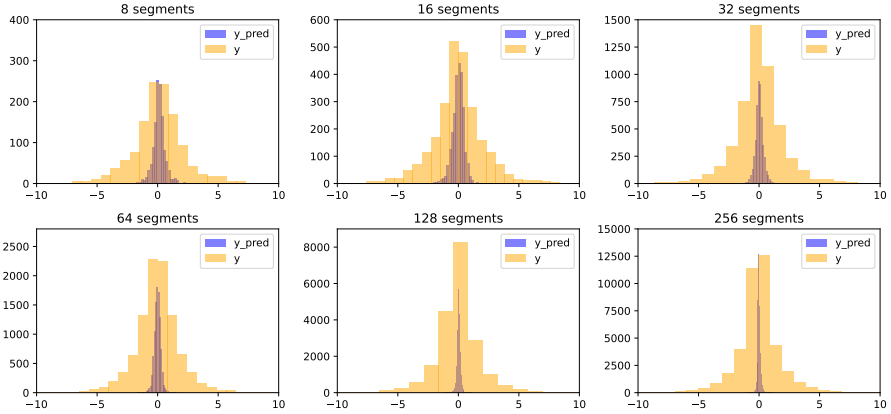
Figure 5.2: Worst downstream result followed the same setting as TCL paper, across different feature extractor depths (1-5) layers, matched to mixing-MLP depth across different number of segments.

As for MSE in the worst downstream task, it is obvious that the loss is decreasing dramatically with the number of segments increasing for all settings with different number of layers in the mixing and learning step. Especially for the number of segments from 32 to 64, the loss decreases the most. Figure 5.2 shows that the MSE after using ICA on the learned features is the same as the one without ICA, which follows our Theorem 3.2.4. However, the R^2 score in Figure 5.2 inclines to decrease to almost zero with an increasing number of segments, indicating that the worst downstream task in a larger number of segments behaves like predictions for the mean of the true label. This contradicts the intuition that a lower worst-case loss should lead to higher explained variance, yet both are decreasing.

However, a potential way to analyze how MSE changes is to decompose it into two terms, the variance of the true label and the variance of the predicted values, presented in Figure 5.3. The formula for decomposition is shown in the Appendix A.4.



(a) Distribution for true label and predicted label in the setting of layer 1.



(b) Distribution for true label and predicted label in the setting of layer 5.

Figure 5.3: An example for distribution of the true labels and predicted labels in the worst downstream tasks of simulated data generated from Laplacian distribution.

After decomposition of MSE into the two variance terms, it is intuitive to plot the distribution for the true and predicted labels. From Figure 5.3, we drew the distribution of labels in layers 1 and 5. Among the number of segments, the distribution of true labels remains stable as a bell-shaped distribution. But for the distribution of predicted labels, it begins to collapse to the mean of true label, which can also be reflected by the results of R^2 in Figure 5.2 that its value reaches almost zero in a large number of segments. And compare between layer 1 and layer 5, the former shows slightly more collapsed according to the graph, which can also be reflected from the lower R^2 in layer 1 from Figure 5.2.

5.1.3 Expected Downstream Performance

Next, we will go through our expected downstream performance, where the same metrics as those used for the worst downstream task were used, including MSE(before and after ICA) and R^2.

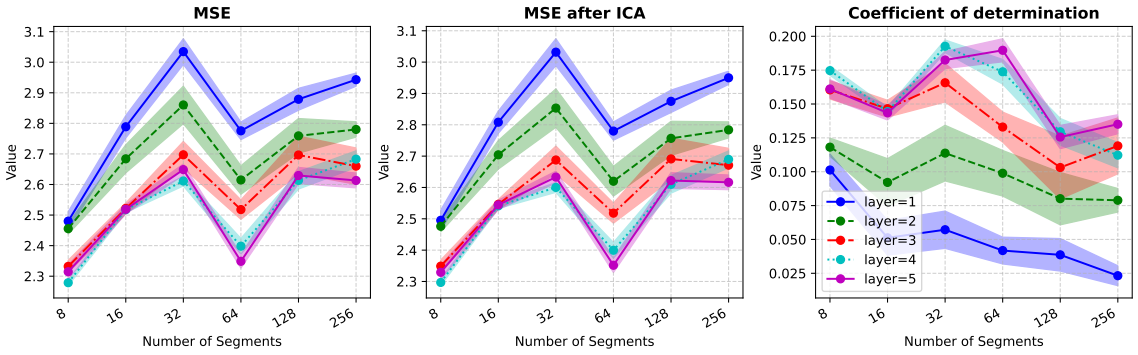
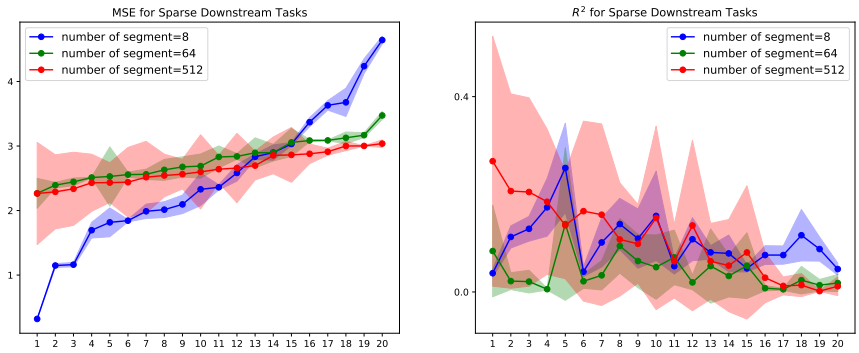


Figure 5.4: Expected downstream result followed the same setting as TCL paper, across different feature extractor depths (1-5) layers, matched to mixing-MLP depth across different number of segments.

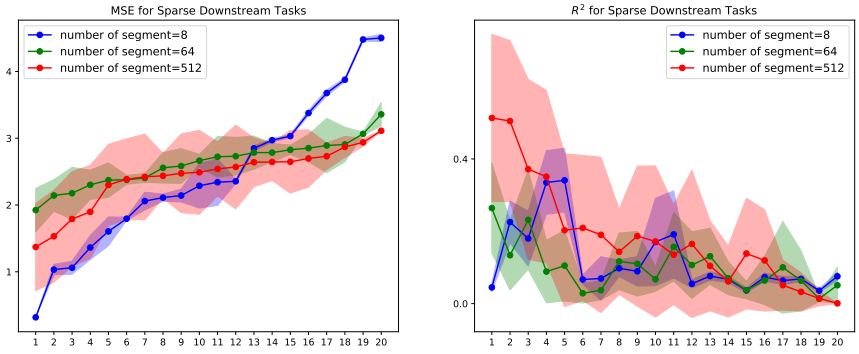
It is clear that the results for MSE before and after ICA align with our Theorem 3.2.4, which is identical to each other. However, for the overall trend of MSE, it is hard to conclude a consistent result regarding the number of segments, as well as the coefficient of determination. However, the expected downstream performance is lower and the R^2 value is higher than in the worst-case scenario, which is both reasonable and expected. Given the analysis for the expected downstream, it is difficult to analyze one downstream separately and draw the label distribution, as in Figure 5.3.

Expected Downstream However, from Theorem 3.2.5, expected downstream performance is equivalent to analyzing expected sparse downstream tasks, that is, consider each dimension of source data as downstream labels and conduct analysis. However, plotting the distribution of predictions for each downstream task makes it difficult to assess the effects of the number of segments or mixing depth, as there are too many individual tasks to analyze.

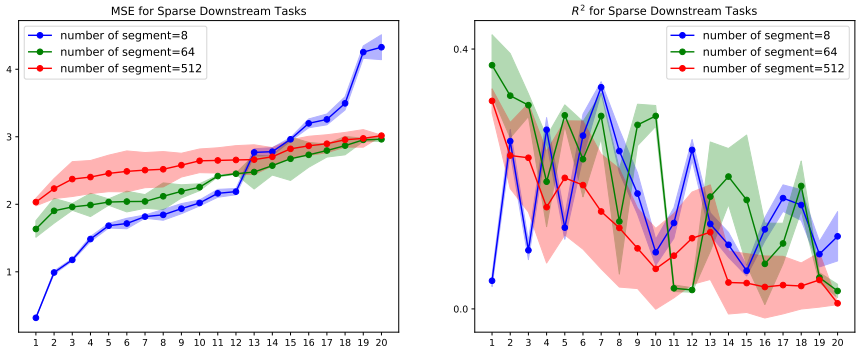
From the experimental results in Figure 5.4, it is shown that one and two layers stand out from the rest and that the results of three to five layers exhibit a similar trend. In order for better interpretation of the results mentioned above, we visualized the MSE and R^2 for each sparse downstream task in Figure 5.5 for $L = 1, 2, 5$, in ascending order of MSE. It is true that if we increase the number of segments, the performance differences between sparse downstream tasks diminish, but more variances of sparse performances are introduced, due to the instability of distinguishing the segments across all the segments. However, as we increase the number of mixing layers, the variance for the performance or R^2 decreases. And, R^2 is not close to zero even if MSE increases a lot, especially for 8 and 64 number of segments in five-layer mixing, which means that these predictions from the model deviate from the mean prediction and indicate that the model learned useful information from the recovered features.



(a) 1-layer mixing case.



(b) 2-layer mixing case.



(c) 5-layer mixing case.

Figure 5.5: MSE and R^2 for each sparse downstream tasks in the case of 1,2 and 5 layer mixing. The tasks are sorting in ascending order of MSE.

5.2 Ablation Studies

For ablation studies, we first intend to check the role of different number of layers when mixing, with a fixed feature extractor. Because the same setting as TCL paper [1] was kept, feature extractor has the same number of layer for mixture. It is significant to check what happens when the mixing is complex, how it influences the downstream tasks with a fixed feature extractor.

Then, recall our assumptions at the beginning, one of the biggest assumptions is based on noise free source, which is impossible in the real-world data. Because, it is inevitable to capture during recording. For the investigation of the affect of noise, a Gaussian noise was added with different magnitudes to the source. For the contaminated source, we will use this noisy version to train the inverted models.

5.2.1 Different mixtures with fixed extractors.

The same configuration presented in Section 5.1 was followed. However, it was observed that the effect of using different feature extractors could influence the validity of the conclusions, thus needing further experiments. The same results as in previous experiments are reported, and the impact of different feature extractors on downstream performance is investigated.

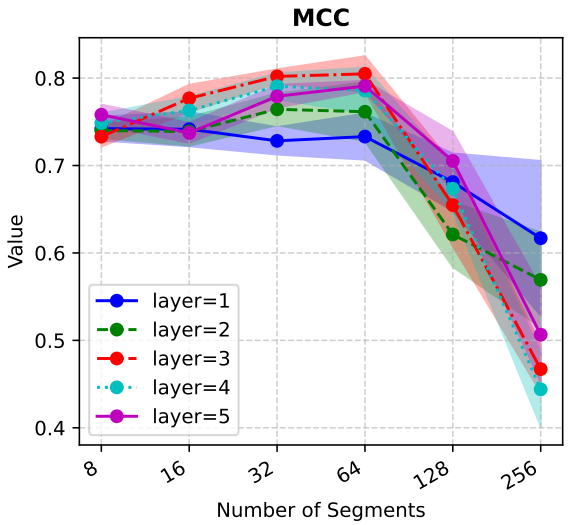
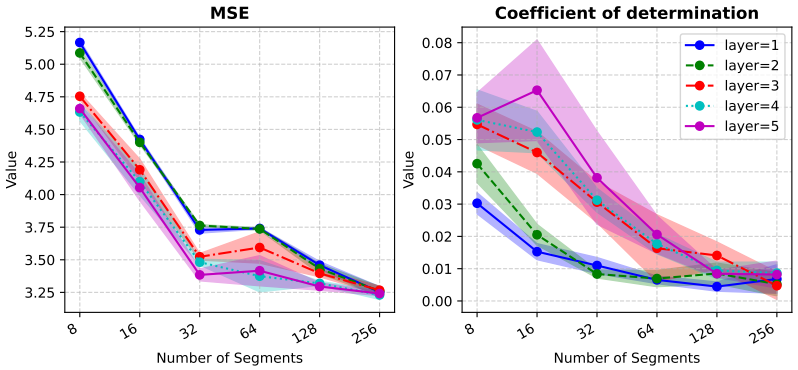


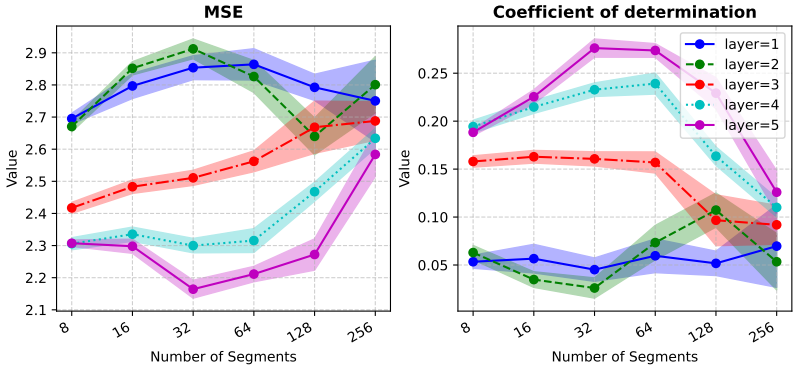
Figure 5.6: Results for mean correlation, by fixing the feature extractor.

When it comes to MCC, mixtures with three layers have the highest values at 64 number of segments, and one-layer mixing has the worst MCC among all settings. In addition, mixtures with different layers have the same turning points for the decrease in MCC. However, as shown in Figure 5.2, the MCC results suggest that using the same layer of the model to invert the data-generating process yields better MCC values with more layer of mixing.

In Figure 5.7(a), it is obvious that the worst performance is decreasing with an increasing number of segments, as well as for the coefficient of determination, which



(a) Worst Downstream Performance.



(b) Expected Downstream Performance.

Figure 5.7: Downstream performance followed the same setting as TCL paper with feature extractor depth at three layers matched to mixing-MLP depths of (1-5) layers.

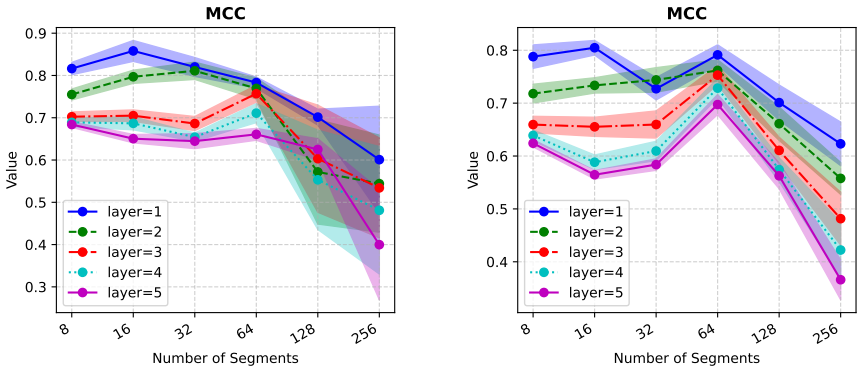
has the same results as different feature extractors, more layers of mixing leading to the prediction of the mean. Regarding expected downstream performance, high values for MSE correspond to low R^2, reflecting poor model performance or weak fit to the data. However, 5-layer mixing consistently achieves a lower MSE and higher R^2 across segments compared to models with fewer layers, as shown in Figure 5.7(b). Then the loss increases with decreasing number of layers in mixing. Possibly indicating that deeper mixing introduces more complex nonlinear transformations, which better suits the feature extractor to recover useful structure for downstream tasks. It is also notable that there is no clear trend in the expected performance as the number of segments increases. However, the curves for MSE and R^2 are mirrored-higher loss corresponds to lower R^2–which is consistent with the behavior of a reasonable model.

5.2.2 Noisy Source

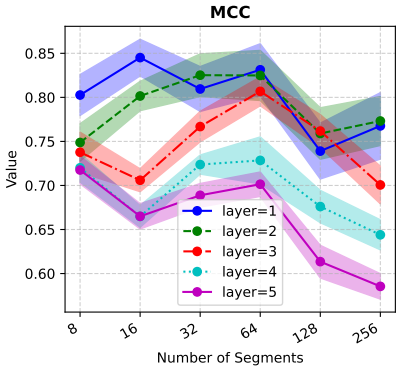
For the noisy source setting, the data are generated from a Laplacian distribution and then contaminated with Gaussian noise of varying magnitude(0.01, 0.1, 1). Then, the same training strategy is used to train the neural network and estimate

the performance of downstream tasks.

Before going through the downstream performance, results for the mean correlation are listed below in Figure 5.8. With less noise, MCC is similar to that in Figure 5.2. However, with a larger noise level, MCC for settings with less layer of mixing is not able to achieve the same level as less noise case and more number of segments are needed to improve the MCC, which indicates that classifying more segments can relieve the influence of the noise. Then, after increasing the magnitude to 1, the variance of MCC tends to increase for all the settings of segments, but the value of mean correlation is not decreasing as expected.



(a) MCC with noise magnitude 0.01 (b) MCC with noise magnitude 0.1

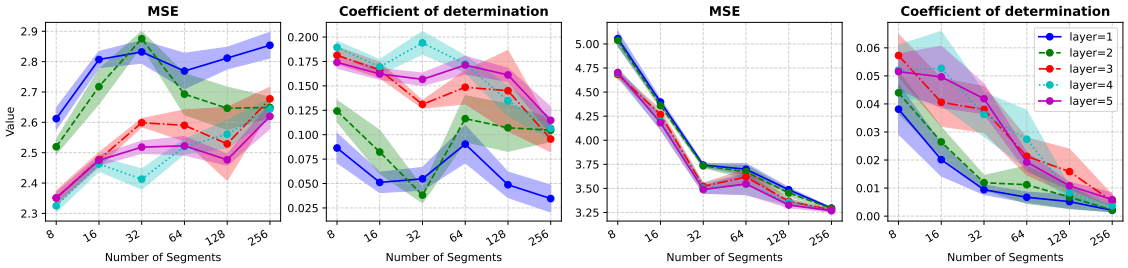


(c) MCC with noise magnitude 1

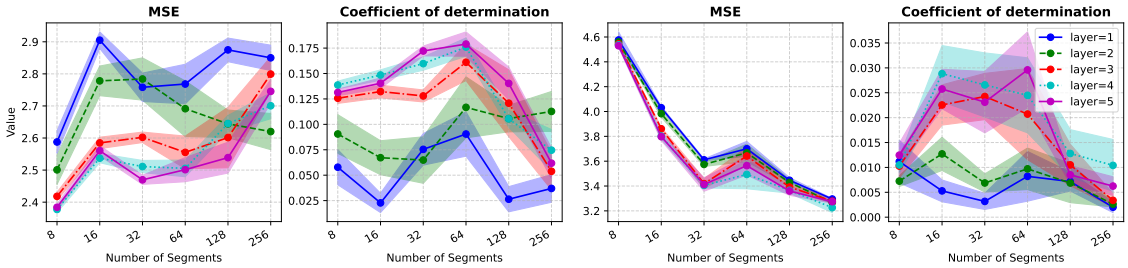
Figure 5.8: Results of MCC for simulated data generated from Laplacian distribution under different noise level.

When it comes to the worst downstream performance, the noise does not affect the overall trend of mean squared errors, except for 16 and 32 segments, with a huge jump of decreasing. However, when the noise level increases from 0.01 to 0.1, the coefficient of determination behaves differently. When the noise level is 0.1, the value for R^2 is much smaller and close to zero, indicating that noise ruins the worst downstream tasks, making its prediction a mean value as shown in Figure 5.8. However, for noise level 1, the result is even better than noise with fewer magnitudes and shows similar trend close to the free-noise situation.

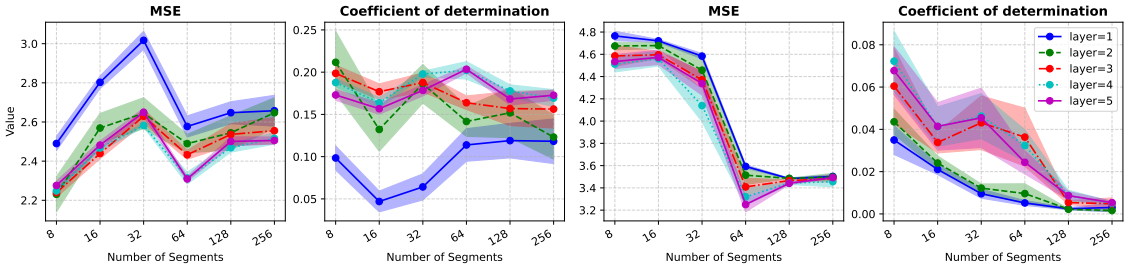
The expected downstream performance with different magnitudes of Gaussian noise



(a) Worst downstream with noise magnitude 0.01



(b) Worst downstream with noise magnitude 0.1



(c) Worst downstream with noise magnitude 1

Figure 5.9: Results of worst downstream task for simulated data generated from Laplacian distribution under different noise level.

is presented in Figure 5.9, with each row presenting the corresponding MSE and R^2 values for a given noise level. Note that the general trend, that more layers in feature and mixing MLPs result in lower loss and higher R^2, follows the pattern of Figure 5.9. Furthermore, the MSE when the noise level is 0.01 and 0.1 are quite similar but change slightly when increasing the noise magnitude to 1, where the results are more similar and slightly lower.

5.3 Real-world Data

To evaluate how our method performs beyond the simulated setting, we include experiments on real-world audio data. This section presents the results from using instrument recordings in a blind source separation setup, with the goal of understanding how well the learned features support downstream tasks. All of the results

are from settings for three and four instruments. We report the gold standard losses for our learned features in the tables. Then, we performed our defined downstream tasks and computed the losses for them. We wanted to see if the losses for our defined tasks situate in a reasonable place with the estimation of expected and worst downstream tasks.

5.3.1 Expected and Worst Downstream Performance

Table 5.1: Table for one-layer linear mixing setting and four-layer nonlinear setting with number of segments 4 and 8. V: Violin, D: Drums, P: Piano, G: Guitar. These results are computed from the test dataset and are used as gold standard bound for the following specific downstream tasks.

Mixing	Set.	Seg.	$\bar{\mathcal{L}}$	Avg R^2	$\mathcal{L}^w$	Worst R^2	MCC	Acc.
Linear	P+D+V	4	0.21	0.22	0.42	0.01	0.70	0.83
		8	0.16	0.25	0.22	0.004	0.76	0.39
	P+D+G	4	0.04	0.29	0.07	0.39	0.64	0.54
		8	0.80	0.47	2.23	0.26	0.86	0.36
	P+V+G	4	0.66	0.24	1.55	0.02	0.77	0.67
		8	0.44	0.17	1.14	0.08	0.66	0.39
	D+V+G	4	0.72	0.32	1.80	0.08	0.89	0.84
		8	0.38	0.13	0.79	0.16	0.76	0.40
	P+D+V+G	4	0.12	0.22	0.28	0.02	0.71	0.67
		8	0.32	0.20	0.72	0.37	0.72	0.57
Nonlinear	P+D+V	4	0.13	0.20	0.28	0.06	0.76	0.67
		8	0.11	0.17	0.24	0.06	0.65	0.34
	P+D+G	4	0.08	0.27	0.14	0.27	0.70	0.70
		8	0.12	0.22	0.25	0.01	0.64	0.40
	P+V+G	4	0.19	0.15	0.38	0.01	0.69	0.52
		8	0.13	0.21	0.24	0.03	0.74	0.38
	D+V+G	4	0.18	0.25	0.38	0.02	0.77	0.67
		8	0.13	0.22	0.24	0.04	0.60	0.49
	P+D+V+G	4	0.12	0.22	0.28	0.02	0.71	0.77
		8	0.11	0.14	0.24	0.05	0.72	0.42

The golden-standard results for our real-world data under different instrument settings with linear mixing are shown in Table 5.1. Nonlinear setting achieves lower loss than the linear setting overall, which coincides with our simulated results that more layers of mixing tend to have lower losses. But for the mean correlation, the linear setting is better than the nonlinear, which is intuitive because of the difficulty to recover a nonlinear transformation of the sources. It is also the same case for the accuracy; linear transformation is easier for the model to make classifications. For

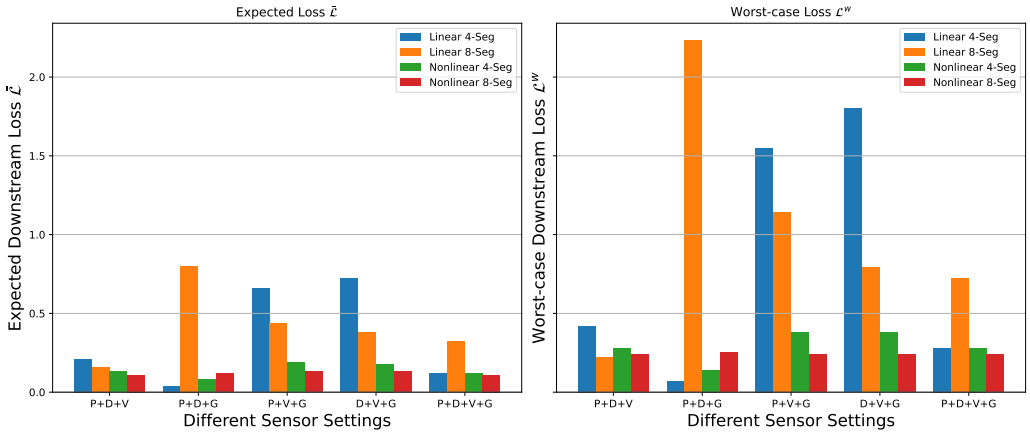


Figure 5.10: Bar chart for one-layer linear mixing setting and four-layer nonlinear setting with number of segments 4 and 8. V: Violin, D: Drums, P: Piano, G: Guitar. Presenting the expected and worst-case loss in the four different experimental settings.

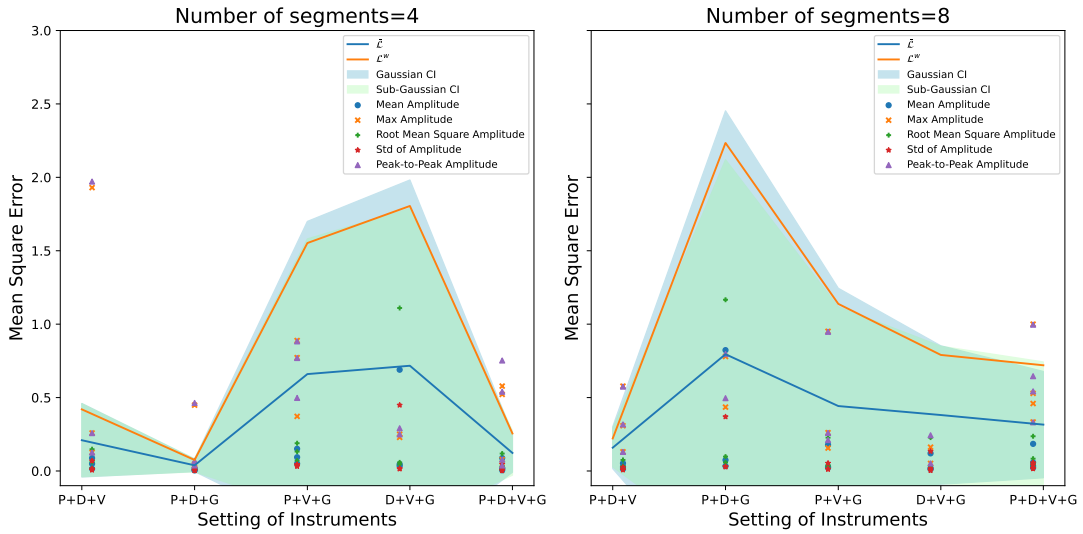
easier comparison, Figure 5.10 presents the expected downstream loss in a bar-chart.

We observe for the average downstream loss and the average R^2 score that there is no clear trend with an increasing number of segments. The worst downstream loss is, with the exception of P+D+G and P+D+V+G(linear setting), decreasing with increasing number of segments, matching our theoretical results. For the nonlinear setting, it is noticeable that with an increasing number of segments, the average loss in downstream tasks decreases, which does not align with our theoretical results at the beginning, but the decreasing average R^2 is similar to our results in Figure 5.4.

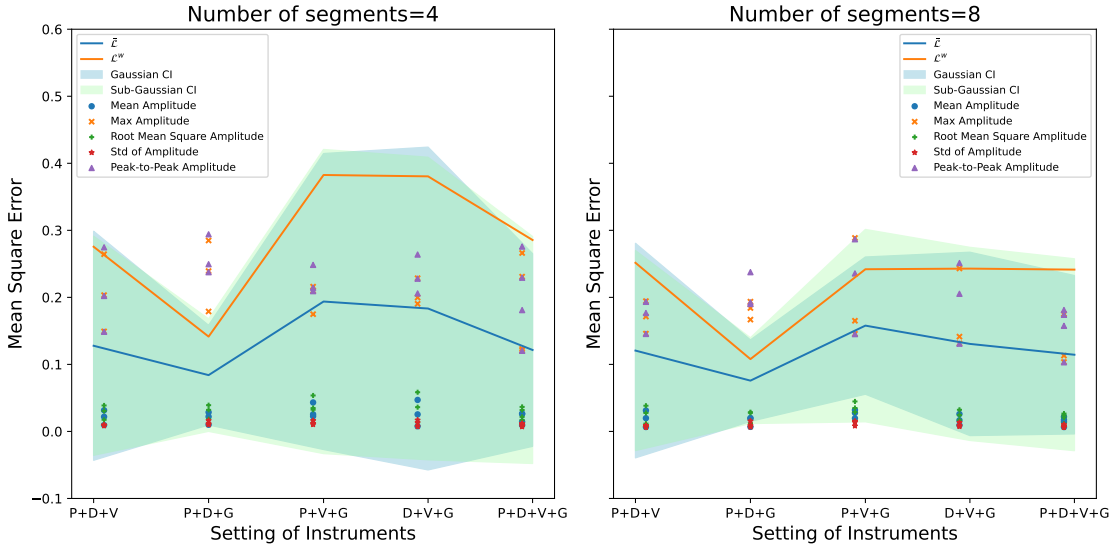
5.3.2 Defined Downstream Tasks

Figure 5.11 presents the expected downstream performance across different sensor settings, the number of segments (4 and 8), and the types of mixing (linear vs. nonlinear). When increasing the number of segments from 4 to 8, the overall downstream losses remain similar in the linear setting, but decreases in the nonlinear setting, evidenced by a lower average loss and a tighter confidence interval. This suggests that a finer segmentation enhances the representational quality of the learned features under nonlinear mixing. Figure 5.11 (a) shows the results for the linear mixing setting. Noticeably, the majority of the MSE for the downstream tasks lies within the confidence interval except for some prediction of Peak to Peak Amplitude and Max Amplitude prediction. However, in this case, the improvement from additional segments is limited, and the performance becomes more sensitive to the choice of sensor combinations.

In Figure 5.11 (b), among all the possible downstream tasks, most of them sit in the specified confidence intervals and few of them exceeded our theoretical worst cases. Especially for the P+D+G setting, numerous defined downstream tasks exceed the worst downstream loss, which is an indication of a violation of the theory that the



(a) Confidence level($\alpha = 0.01$) for the linear mixing setting.



(b) Confidence level($\alpha = 0.01$) for the nonlinear mixing setting.

Figure 5.11: Expected downstream performance(blue line) with confidence interval($\alpha = 0.01$). Yellow line indicates the worst downstream performance. Other points represent the specific downstream tasks. Light green stripe(Sub Gaussian CI) indicates the confidence level computed from sub-Gaussian tail bound from Corollary 3.2.1. Light blue stripe(Gaussian CI) indicates the confidence level under the assumption that $\mathcal{L}_{ds}(\hat{h}; \beta) \sim \mathcal{N}(\bar{\mathcal{L}}_{ds}, \sigma^2_{\hat{h}})$.

label is a linear combination of the latent variables. The outliers of our defined downstream tasks are still maximum amplitude and peak-to-peak amplitude prediction. The potential reason is that the maximum amplitude is unstable as the peak value can easily be affected by noise, outliers, or other sources of contamination. Since it relies on a single extreme point, even small disturbances or artifacts can significantly distort the measurement, making it less reliable. For increasing the number of segments, it is shown that the total variance of the downstream performance is decreasing, with the exception of P+D+G in the linear setting, which indicates that the model tries to learn some fine-grained features for better representations. For specific downstream tasks, the variance also decreases and the loss is closer to the theoretically expected one.

For the confidence intervals, we come up with two ways to compute them. One is due to the fact that the empirical loss in our case is a sub-Gaussian variable and we can easily compute the confidence interval given a confidence level α. The other way to compute the interval is to assume that the distribution of the empirical loss is from a Gaussian distribution with a mean of $\bar{\mathcal{L}}_{ds}$ and a variance of $\sigma_{\hat{h}}^2$ as stated in Theorem 3.2.2 and 3.2.3. We actually do not know how the empirical loss is distributed due to the random variable of learned features.

For these two CIs, they are close to each other. But in the linear setting, the confidence interval from sub-Gaussian tends to be bounded by that from Gaussian. That is due to the larger sub-Gaussian parameter in the linear case. In the nonlinear setting, sub-Gaussian CI has even larger ranges than that of Gaussian, yielding more defined downstream tasks.

5.3.3 Learned Features vs Sensor Data

Finally, we want to investigate how well our learned features compare to the sensor data. Despite nonlinear or linear case, most points lie above the diagonal, indicating that learned features consistently outperform raw sensor features in downstream tasks. This pattern holds across both linear and nonlinear mappings and for both segment settings (4 and 8 segments). The advantage is particularly noticeable in the nonlinear cases, where the learned representations demonstrate significantly lower MSE. These results suggest that the learned features are more informative and robust, offering improved generalization in downstream applications compared to features derived from sensor data.

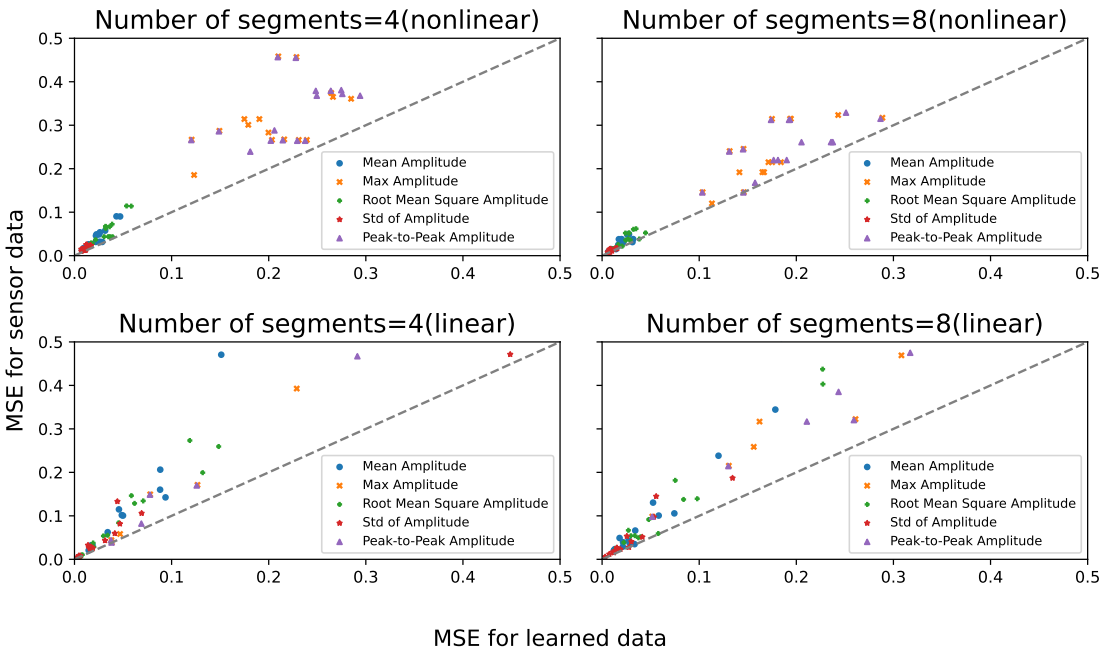


Figure 5.12: Scatter plot for the downstream tasks. X-axis represents the loss using the learned features. Y-axis represents the loss using the sensor data. Point situating in the dotted line indicates equal performance between learned features and sensor data.

6

Discussion

To analyze the results and better understand their causes, we first discuss how the theoretical assumptions and limitations affected performance. We then examine the worst-case and expected downstream results, before concluding the chapter with directions for future work.

6.1 Theoretical Assumptions and Limitations

Our theoretical framework is based on a set of simplifying assumptions that make the downstream performance analysis in the nonlinear ICA setting tractable and allow for closed-form expressions for both expected and worst downstream performance. Specifically, we assume that the downstream tasks are linear functions of the latent sources, that the task parameter vectors lie on the unit sphere, and that the source data are noise-free. However, they also introduce limitations when conducting real-world experiments.

The assumption of linear downstream tasks, combined with squared-error evaluation, allows the problem to be analyzed through ordinary least squares regression. Under this setup, both expected and worst-case performance can be expressed in terms of how well the learned features align with the true sources via linear projections. This not only makes the analysis clean, but also restricts its relevance to settings where the relationship between features and targets is approximately linear. In other domains, such as frequency prediction, downstream tasks are often nonlinear and extracted from features. As a result, the performance guarantees derived under this assumption may not be transferred directly to these more complex settings.

Limiting β to lie on the unit sphere means that we treat all directions in the latent space equally when evaluating performance. This makes analysis and gives a general baseline for how the learned representation performs across every possible task. With clear upsides, real-world applications usually have specific goals that correspond to specific directions rather than being uniform. Even though we use uniform sampling, the theory can be extended to have other distributions for downstream tasks. Thus making it possible to adapt our framework to task-specific settings. Furthermore, limiting $\|\beta\| = 1$ ensures that the loss is independent of scale and focuses purely on the alignment between the learned features and the source data, which is not

dependent on the framework and can be modified depending on the need.

Regarding the noise-free source data, which is useful for our theory, this assumption does not hold fully in real world settings. It helps to investigate the role of representations in contrastive learning, but in practice both source and sensor data are often affected by noise. In such settings, the theoretical guarantees might not strictly apply, but rather provide meaningful insights and serve as a solid baseline.

6.2 Worst Downstream Task

The worst downstream task reflects the latent direction that is the least recoverable from the learned features. This offers a valuable perspective on representation quality in highlighting which linear task performs the worst. This is useful in settings where reliability is important and can reveal blind spots that would not be discovered by expected performance.

As seen in our results, the MSE for the worst downstream task decreases as the number of segments increases. Theorem 3.2.1 helps formalize this observation, as it captures the maximum error when predicting any linear target derived from the true sources using the learned features. As the number of segments increases, the MLP gains more contrastive information, intuitively improving its ability to recover the underlying structure in the data. Although this is reflected in Figure 5.2, we also found that with a larger number of segments the model begins to collapse towards the mean, as shown in Figure 5.3. This indicates that the result stems from uniformly low errors across tasks rather than a good generalization. A possible reason is that, with larger number of classes, the model struggles to distinguish between segments due to limited samples per class, leading it to minimize the contrastive loss by learning the mean regardless of input. This effect is particularly noticeable when the number of segments greatly exceeds the dimensionality of the learned representations (i.e. 64, 128, 256 segments versus 20 dimensions), making it increasingly difficult for the model to represent all classes distinctly. This is further supported by the decrease in both R^2 and MCC with increasing number of segments.

The ablation study that was performed by fixing the number of layers for the feature extractor and adding noise to the source data,to further investigate our results. It was shown that when adding noise, a high MCC was achieved with fewer number of segments then drastically decreasing. A potential explanation is that, with fewer layers of mixing, the sensor data with noise are still able to be recovered by the model because of the less complicated data structure. However, with more layers of mixing, the data structure is more nonlinear and complicated with high-level noise added, making it difficult to recover the source data. We also pointed out that the worst downstream loss for noise level 1 is very similar to our noise-free experiment; a possible explanation for it is that the Gaussian noise dominates the data and makes the model discriminate segments from the Gaussian distribution with Laplacian noise instead.

6.3 Expected Downstream

The expected downstream performance offers a broader picture of how well the learned representation supports typical tasks. This makes it a useful indicator of general-purpose representational quality, especially when downstream tasks are not known in advance. However, this averaging can also mask important differences between individual tasks, which motivates a closer examination of the sparse downstream results.

It can be concluded from Figure 5.5 that among all the scarce downstream tasks, their performance varies. That said, increasing the number of segments can relieve the variance of performance for the sparse downstream tasks. Moreover, it is noticeable that downstream with low loss also has a low R^2 except in the larger segment setting, which indicates that those downstream tasks tend to predict the mean of the true label and learn almost nothing from the data. When the number of segments is large, for example 512, the decrease of R^2 synchronizes with the increase of loss. This phenomenon can also explain why in Figure 5.4, R^2 rocketed when the number of segments is 512.

However, counterintuitively, with more layer of mixing, sensor signals are more mingled and harder to separate, which can be reflected in the decreasing MCC in Figure 5.2. However, when it comes to downstream performance, even though the loss is similar to the case of less mixing, the R^2 score improves, indicating that the downstream task is better captured by a certain linear model.

6.4 Future Work

Several directions emerge from this work that could extend both the theoretical framework and the empirical evaluation of contrastive learning in a nonlinear ICA setting.

First, the TCL framework[1] is not the only recipe to conduct self-supervised learning to tackle nonlinear ICA problems. There are other literatures such as [12], [13], which can be used to recover features up to a linear transformation. It is intriguing to ask how these learned features perform in downstream tasks and what factors could influence the quality of these learned features.

Second, a good extension would be to compare against a fully supervised baseline trained directly on the downstream task. This approach skips the representation learning step entirely and instead trains a model from scratch. By evaluating its performance across increasing amounts of labeled data, such a comparison would help assess the sample efficiency and practical value of the learned representations.

Third, the current analysis assumes linear downstream tasks, enabling closed-form results. A natural extension would be to consider nonlinear downstream objectives, such as imposing a kernel method on the learned features to model the nonlinear relationship, which would better reflect real-world applications, where the relation-

ship between latent factors and targets is rarely linear. In addition to that, in our work, we only include downstream regression tasks. For classification tasks, due to the lack of closed-form solutions, we leave it out currently. However, classification-based downstream tasks are common in real-world applications, such as instrument classification and event detection.

Finally, in a musical setting, it is challenging to define meaningful downstream tasks at the level of individual time points, as relevant information is usually distributed over short time windows. Therefore, it becomes important to extract features segment-wise rather than point-wise — something that the TCL method does not directly support. This suggests a need for new contrastive learning approaches that can produce meaningful segment-level representations.

7

Conclusion

In our thesis, we commenced with asking the question how generalizable our learned features are from the contrastive learning recipe TCL and planned to investigate how the features are influenced with respect to the downstream tasks. To answer these questions, we provided several theoretical results which are useful for generalizability analysis. In particular, under our assumptions, we provided our theoretical close-form solution for the empirical losses, which is considered as a standard to quantify how good the learned representations are.

To summarize what we have done, we have first proved a closed-form solution for the worst and expected downstream loss, which is a function of the learned features. In addition, we derived the variance of the loss, which helps quantify how much downstream performance varies across tasks. Afterwards, we conducted the experiments on both simulated data(same settings from the TCL paper) and also using real-world data, with instruments. For simulated data, we try to investigate how different settings influence the learned features and losses. For real-world data, we found some real musical tracks and applied our own mixing procedure. Then we applied contrastive learning to obtain features to perform specific downstream tasks. We define our downstream tasks, which are to predict the statistical metrics of the music as in mean or maximum amplitude. Through our experimental results, we found that our learned features can outperform the sensor data and most of the defined downstream tasks situate in a specific confidence interval, which is a good indicator that our assumptions still align with the real-world case.

In conclusion, the quantification of downstream performances is a useful topic for understanding how well learned features can perform in downstream tasks, which sheds light on the effectiveness and generalization capabilities across different settings.

Bibliography

[1] A. Hyvarinen and H. Morioka, "Unsupervised feature extraction by time-contrastive learning and nonlinear ica," *Advances in neural information processing systems*, vol. 29, 2016.

[2] J. Devlin, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[3] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 11, pp. 4037–4058, 2020.

[4] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, 2020.

[5] J. Kim, H. Lee, J. Chang, and S. M. Park, "Generalized supervised contrastive learning," *arXiv preprint arXiv:2206.00384*, 2022.

[6] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*, pp. 1597–1607, PMLR, 2020.

[7] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pp. 776–794, Springer, 2020.

[8] B. Ans, J. Hérault, and C. Jutten, "Architectures neuromimétiques adaptatives: Détection de primitives," *Proceedings of Cognitiva*, vol. 85, pp. 593–597, 1985.

[9] A. Hyvärinen and P. Pajunen, "Nonlinear independent component analysis: Existence and uniqueness results," *Neural networks*, vol. 12, no. 3, pp. 429–439, 1999.

[10] H. Mobahi, R. Collobert, and J. Weston, "Deep learning from temporal coherence in video," in *Proceedings of the 26th annual international conference on machine learning*, pp. 737–744, 2009.

[11] R. Goroshin, J. Bruna, J. Tompson, D. Eigen, and Y. LeCun, "Unsupervised learning of spatiotemporally coherent metrics," in *Proceedings of the IEEE international conference on computer vision*, pp. 4086–4093, 2015.

[12] A. Hyvarinen and H. Morioka, "Nonlinear ica of temporally dependent stationary sources," in *Artificial Intelligence and Statistics*, pp. 460–469, PMLR, 2017.

[13] A. Hyvarinen, H. Sasaki, and R. Turner, "Nonlinear ica using auxiliary variables and generalized contrastive learning," in *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 859–868, PMLR, 2019.

[14] N. Saunshi, J. Ash, S. Goel, D. Misra, C. Zhang, S. Arora, S. Kakade, and A. Krishnamurthy, "Understanding contrastive learning requires incorporating inductive biases," in *International Conference on Machine Learning*, pp. 19250–19286, PMLR, 2022.

[15] S. Arora, H. Khandeparkar, M. Khodak, O. Plevrakis, and N. Saunshi, "A theoretical analysis of contrastive unsupervised representation learning," *arXiv preprint arXiv:1902.09229*, 2019.

[16] R. S. Zimmermann, Y. Sharma, S. Schneider, M. Bethge, and W. Brendel, "Contrastive learning inverts the data generating process," in *International Conference on Machine Learning*, pp. 12979–12990, PMLR, 2021.

[17] E. Rusak, P. Reizinger, A. Juhos, O. Bringmann, R. S. Zimmermann, and W. Brendel, "Infonce: Identifying the gap between theory and practice," *arXiv preprint arXiv:2407.00143*, 2024.

[18] W. Huang, M. Yi, X. Zhao, and Z. Jiang, "Towards the generalization of contrastive self-supervised learning," *arXiv preprint arXiv:2111.00743*, 2021.

[19] Y. Lei, T. Yang, Y. Ying, and D.-X. Zhou, "Generalization analysis for contrastive representation learning," in *International Conference on Machine Learning*, pp. 19200–19227, PMLR, 2023.

[20] G. Casella and R. Berger, *Statistical inference*. CRC press, 2024.

[21] G. Strang, *Introduction to Linear Algebra*. Wellesley, MA: Wellesley-Cambridge Press, 5 ed., 2016.

[22] *What is Independent Component Analysis?*, ch. 7, pp. 145–164. John Wiley Sons, Ltd, 2001.

[23] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2019.

[24] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[25] R. Vershynin, *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge Series in Statistical and Probabilistic Mathematics, Cambridge University Press, 2018.

[26] D. Lopez-Paz, P. Hennig, and B. Schölkopf, "The randomized dependence coefficient," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'13, (Red Hook, NY, USA), p. 1–9, Curran Associates Inc., 2013.

[27] S. Prasad, "Musical instruments sound dataset." `https://www.kaggle.com/datasets/soumendraprasad/musical-instruments-sound-dataset`, 2022. Accessed: 2025-05-09.

[28] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, p. 1929–1958, Jan. 2014.

[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37, pp. 448–456, 2015.

A

Appendix 1

A.1 Closed-form for $\mathcal{L}(\beta; \hat{h})$

Proof. We know the downstream is given by $H\beta$. The solution of $\hat{\beta}$ can be given by OLS estimator.

$$\hat{\beta} = \left(\hat{H}^\top H\right)^{-1} \hat{H}^\top H\beta.$$

Then, we can write the loss into the matrix form:

$$\begin{aligned}
\mathcal{L}(\beta; \hat{h}) &= \frac{1}{N}\left(\hat{H}\hat{\beta} - H\beta\right)^\top \left(\hat{H}\hat{\beta} - H\beta\right) \\
&= \frac{1}{N}\left(PH\beta - H\beta\right)^\top \left(PH\beta - H\beta\right) \\
&= \frac{1}{N}\left(\beta^\top H^\top H\beta - \beta^\top H^\top PH\beta\right) \\
&= \frac{1}{N}\beta^\top H^\top \left(I - P\right) H\beta,
\end{aligned}$$

where P is the projection matrix for $\hat{H}$.

Hence, $\mathcal{L}(\beta; \hat{h})$ has a closed-form solution $\frac{1}{N}\beta^\top H^\top \left(I - P\right) H\beta$.

□

A.2 Proof for Lemma 3.2.1

Proof. We use Lagrangian multiplier μ, then

$$L(x, \mu) = x^T Ax - \mu(1 - x^T x),$$

We can compute

$$\frac{\partial L}{\partial x} = (A + A^T)x - 2\mu = 2Ax - 2\mu x.$$

We set the partial derivative to be zero and we have

$$\mu x = Ax,$$

which indicates μ is the eigenvalue of matrix A. Then, the original objective function becomes

$$\max_{x^T x=1} \mu x^T x = \mu_{max} = \lambda_{max}(A).$$

□

A.3 Matrix form for R square in our setting

From definition 2.4 and with our setting for $Y = H\beta$ and $\hat{Y} = \hat{H}\hat{\beta}(\beta) = PH\beta$, then the matrix form of R square is computed as

$$Y^\top \hat{Y} = \beta^\top H^\top \hat{H}^\top (\hat{H}^\top \hat{H})^{-1} \hat{H} H\beta = \beta^\top H^\top PH\beta$$

$$\hat{Y}^\top \hat{Y} = \beta^\top H^\top P^\top PH\beta = \beta^\top H^\top PH\beta$$

$$\begin{aligned}
R^2 &= 1 - \frac{\|Y - \hat{Y}\|_2^2}{\|Y - \bar{Y}\mathbf{1}\|_2^2} \\
&= 1 - \frac{Y^\top Y - 2Y^\top \hat{Y} + \hat{Y}^\top \hat{Y}}{\|Y - \bar{Y}\mathbf{1}\|_2^2} \\
&= 1 - \frac{\beta^\top H^\top H\beta - 2Y^\top \hat{Y} + \hat{Y}^\top \hat{Y}}{\|Y - \bar{Y}\mathbf{1}\|_2^2} \\
&= 1 - \frac{\beta^\top H^\top H\beta - \beta^\top H^\top PH\beta}{\|Y - \bar{Y}\mathbf{1}\|_2^2} \\
&= 1 - \frac{\beta^\top H^\top (I - P) H\beta}{\|Y - \bar{Y}\mathbf{1}\|_2^2}
\end{aligned}$$

A.4 MSE Decomposition

$$
\begin{aligned}
MSE &= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y} + \bar{y} - \hat{y}_i)^2 \\
&= \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2 - \frac{2}{n}\sum_{i=1}^{n}(y_i - \bar{y})(\hat{y}_i - \bar{y}) + \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2 \\
&= \mathrm{var}(y) + \mathrm{var}(\hat{y}) - \frac{2}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i + \hat{y}_i - \bar{y})(\hat{y}_i - \bar{y}) \\
&= \mathrm{var}(y) - \mathrm{var}(\hat{y}) - \frac{2}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)(\hat{y}_i - \bar{y}) \\
&= \mathrm{var}(y) - \mathrm{var}(\hat{y}) - \underbrace{\frac{2}{n}\sum_{i=1}^{n}\epsilon_i(\hat{y}_i - \bar{y})}_{\epsilon \perp (y - \bar{y})} = \mathrm{var}(y) - \mathrm{var}(\hat{y})
\end{aligned}
$$

where $var(y)$ refers to the variance of true label, which is determined by the linear combination by the source data and $var(\hat{y})$ refers to the variance of predicted label, which is from our worst downstream task. ϵ_i is the residual term for sample i.

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY