Data Decomposition for Constrained Visual Learning

Calvin Murdock

April 2020
CMU-ML-20-106

Machine Learning Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Simon Lucey (Chair)
Katerina Fragkiadaki
Deva Ramanan
James Hays (Georgia Institute of Technology)

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

This research was funded by the Department of Transportation award number DTRT13GUTC26, National Science Foundation award numbers IIS1116583, IIS1418523, IIS1526033, IIS1617953, IIS1925281, and a contract from ARGO AI LLC.

Keywords: computer vision, representation learning, component analysis, constrained optimization, deep neural networks, sparse approximation

For my parents

Abstract

With the increasing prevalence of large datasets of images, machine learning has all but overtaken the field of computer vision. In place of specialized domain knowledge, many problems are now dominated by deep neural networks that are trained end-to-end on collections of labeled examples. But can we trust their predictions in real-world applications? Purely data-driven approaches can be thwarted by high dimensionality, insufficient training data variability, intrinsic problem ambiguity, or adversarial vulnerability. In this thesis, we address two strategies for encouraging more effective generalization: 1) integrating prior knowledge through inference constraints 2) theoretically motivated model selection. While inherently challenging for feed-forward deep networks, they are prevalent in traditional techniques for data decomposition such as component analysis and sparse coding. Building upon recent connections between deep learning and sparse approximation theory, we develop new methods to bridge this gap between deep and shallow learning.

We first introduce a formulation for data decomposition posed as approximate constraint satisfaction, which can accommodate richer instance-level prior knowledge. We apply this framework in Semantic Component Analysis, a method for weakly-supervised semantic segmentation with constraints that encourage interpretability even in the absence of supervision. From its close relationship to standard component analysis, we also derive Additive Component Analysis for learning nonlinear manifold representations with roughness-penalized additive models.

Then, we propose Deep Component Analysis, an expressive model of constrained data decomposition that enforces hierarchical structure through multiple layers of constrained latent variables. While it can again be approximated by feed-forward deep networks, exact inference requires an iterative algorithm for minimizing approximation error subject to constraints. This is implemented using Alternating Direction Neural Networks, recurrent neural networks that can be trained discriminatively with backpropagation. Generalization capacity is improved by replacing nonlinear activation functions with constraints that are enforced by feedback connections. This is demonstrated experimentally through applications to single-image depth prediction with sparse output constraints.

Finally, we propose a technique for deep model selection motivated by sparse approximation theory. Specifically, we interpret the activations of feed-forward deep networks with rectified linear units as algorithms for approximate inference in structured nonnegative sparse coding models. These models are then compared by their capacities for achieving low mutual coherence, which is theoretically tied to the uniqueness and robustness of sparse representations. This provides a framework for jointly quantifying the contributions of architectural hyperparameters such as depth, width, and skip connections without requiring expensive validation on a specific dataset. Experimentally, we show correlation between a lower bound on mutual coherence and validation error across a variety of common network architectures including DenseNets and ResNets. More broadly, this suggests promising new opportunities for understanding and designing deep learning architectures based on connections to structured data decomposition.

Contents

List of Figures

List of Tables

1 Introduction

In the case of all things which have several parts and in which the totality is not, as it were, a mere heap, but the whole is something beside the parts, there is a cause; for even in bodies contact is the cause of unity in some cases, and in others viscosity or some other such quality.

Aristotle, 350 B.C.E.

Understanding the composition of images is an essential task in many problems within the field of computer vision. Examples include segmenting images into semantically related regions like in Figure 1.1, interpreting contextual cues for classifying the subjects of images, or recovering the three-dimensional structure of scenes by separating shape from shading. This essentially amounts to extracting specific patterns of information from collections of thousands, millions, or even billions of visual features. While seemingly intractable, real-world visual data are often rich with structure that limits their complexity and enables effective learning. Even though the typical image resolution of a digital photograph is large, the number of realistic natural images is extremely small relative to that of all combinations of possible pixel values. For example, physical laws precisely limit how light can travel throughout a scene, semantic meanings dictate how object appearances correlate, and human biases affect how images–and questions about images–are framed. One of the major goals in computer vision is to achieve levels of performance comparable to human vision.

Since David Marr's pioneering work on the computational foundations of vision [94], effectively leveraging prior knowledge has been an important challenge in emulating vision. While the underlying task of extracting three-dimensional information from two-dimensional projections of a scene is fundamentally ill-posed,

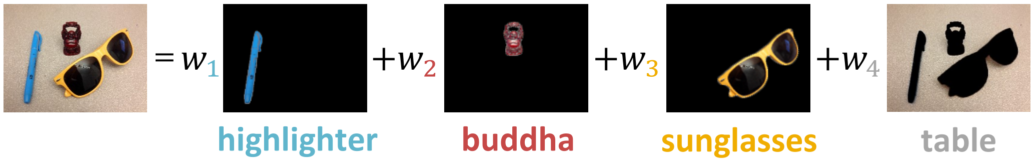


Figure 1.1: Many tasks within computer vision can be posed as decomposing an image into its constituent parts. In the task of semantic segmentation, pixels are grouped into non-overlapping parts that correspond to different objects within a scene. Even without extensive supervision, prior knowledge such as object color consistency can encourage more accurate segmentations.

humans leverage a multitude of subconscious cues to resolve ambiguities and construct an accurate mental model of the world. For example, binocular disparity allows for the triangulation of light projected onto our retinas, shading is used to infer surface normals and integrate the geometric structure of objects, while perspective and high-level semantic knowledge enable correlating image size with relative distance [59].

Building upon the successes of human vision, there has been a long history of introspective work within the computer vision community towards incorporating prior cues for resolving visual ambiguities [122]. These approaches are typically either derived from geometric constraints or learned from correlation patterns in large datasets of labeled images. Despite their foundations in the physical properties of light within a scene, computational implementations for enforcing image constraints often rely on simplifying approximations that can limit their effectiveness. For example, many algorithms rely on unrealistic assumptions such as rigidity [18], Lambertian reflectance [45], or surface smoothness [142]. In contrast, learning-based techniques instead approach visual inference as a data-driven prediction problem. Given enough representative examples, correlation patterns corresponding to visual structure can be learned instead of enforced explicitly. Figure 1.2 provides an intuitive example of how pixel-level image segmentations could be found only through image-level supervision, prior constraints such as object color consistency, and the co-occurrence of image features. In fact, deep neural networks trained for the task of image classification have been found to automatically learn object localization without any strong pixel-level annotations [113].

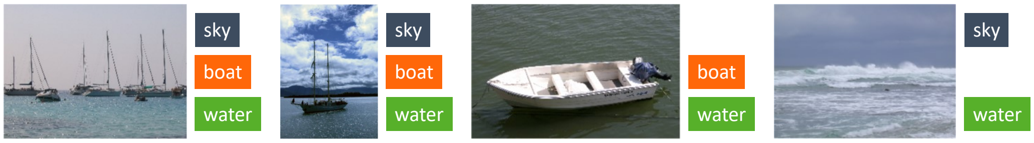


Figure 1.2: Large collections of images can provide context that enables effective visual inference using prior knowledge about the structure of images. In the task of weakly supervised image segmentation, color consistency constraints alongside co-occurrences of image features like the texture of water enable pixel-level object labels to be inferred from image-level annotations of training data.

Due to limited model complexity and computational inefficiency, early learning-based methods were restricted to simplified problems such as the co-segmentation of subjects shared within small collections images [125]. However, recent advances in deep neural networks have spawned a bevy of new methods for solving much more complex problems by mining extensive datasets of images for predictive patterns of correlation. Through multiple layers of parameterized nonlinear transformations, these models have a high capacity for learning functions that accurately map input images to output predictions. When trained on sufficiently large sets of example pairs of input and output data, the learned functions can then be applied to predict unknown outputs from novel images not seen during training. Even though they were originally designed for higher-level unstructured tasks such as image classification [78], deep neural networks have since achieved unparalleled performance in a wide variety of tasks. New datasets, loss functions, and network configurations have quickly expanded their scope to include a much wider range of structured applications that once required hard-coded assumptions or explicit geometric reasoning. Examples include predicting depth maps [42], surface normals [150], and optical flow [41].

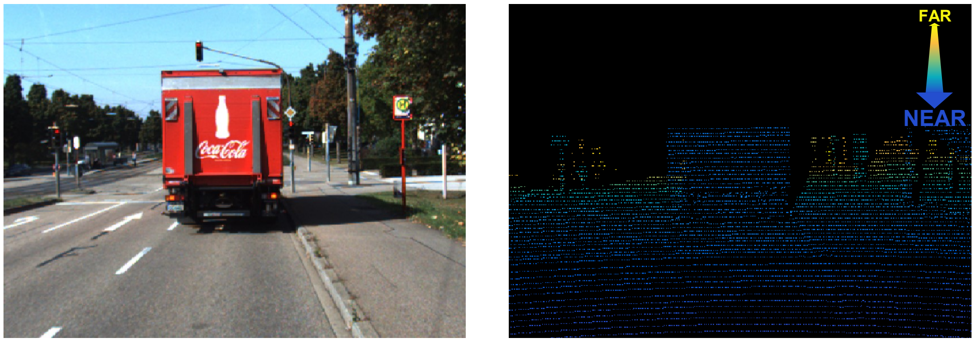
1.1 Motivation

Surprisingly, many state-of-the-art methods now use task-agnostic, “black-box” models that do not consider any of the rich prior knowledge and structure associated with these problems. As a result, they can fall victim to unpredictable failure modes that prevent effective generalization. This behavior can occur when image content is insufficient to resolve inherent ambiguities or when the distribution of the training data differs substantially from that of the testing data. But even with sufficient training data, poor generalization has also been observed in adversarial scenarios where small, imperceptible changes to the input can completely alter the output predictions [137]. Ideally, architectures should be selected to reduce their sensitivity to these input perturbations for improving generalization performance, but this is not yet possible due to a fundamental lack of theoretical understanding [157]. Instead, deep network architectures are still largely designed through human ingenuity and ad-hoc experimentation.

Within computer vision, there has been significant progress in developing new architectures that can learn effective image representations across a wide range of different applications. This can be seen through the community's quick adoption of the newest state-of-the-art deep networks from AlexNet [73] to VGGNet [132], ResNets [54], DenseNets [62], and so on. But this begs the question: why do some deep network architectures work better than others? Despite years of groundbreaking empirical results, an answer to this question still remains elusive.

The difficulty in comparing network architectures arises from the lack of a theoretical foundation for characterizing their generalization capacities. Shallow machine learning techniques like support vector machines [31] were aided by theoretical tools like the VC-dimension [146] for determining when their predictions could be trusted to avoid overfitting. Deep neural networks, on the other hand, have eschewed similar analyses due to their complexity. Theoretical explorations of deep network generalization [108] are often disconnected from practical applications and rarely provide actionable insight into how architectural hyperparameters contribute to performance.

For real-world applications like self-driving cars, interpretability and robustness are key requirements for encouraging user trust in the output of computer vision algorithms. Unfortunately, current deep neural network architectures lack both the the theoretical tools to guarantee good generalization performance and the ability to enforce agreement with prior knowledge. As such, classical techniques for enforcing constraints and fusing multiple data sources like LiDAR still play a key role in perception pipelines, as shown in Figure 1.3. This can be partially attributed to the difficulty in enforcing prediction constraints that encode



(a) Dense Image Data

(b) Sparse LiDAR Constraints

Figure 1.3: Real-world applications of computer vision like self-driving cars require robust and interpretable machine learning algorithms. They also tend to be rich with structure, prior domain knowledge, and multiple sources of complementary data such as (a) high-resolution but ambiguous image data and (b) sparse but accurate LiDAR measurements. We propose techniques inspired by data decomposition to effectively model this rich structure through constraints.

prior structure with feed-forward computations wherein multiple outputs are constructed independently from one another. This differs from classical techniques for data decomposition which can naturally enforce constraints during optimization and are often also associated with strong theoretical performance guarantees.

The computational framework of data decomposition has seen numerous applications in the fields of computer and human vision alike. Founded on the assumption that useful representations should be able to accurately reconstruct input data, classical computational techniques like component analysis and matrix factorization attempt to approximately decompose a set of data points $\boldsymbol{x}_i$ for $i = 1, \ldots, n$ into linear combinations of shared representative components $\boldsymbol{b}_j$ with individualized weights w_{ij} so that:

$$\forall i = 1, \ldots, n : \quad \boldsymbol{x}_i \approx \sum_{j=1}^{k} w_{ij} \boldsymbol{b}_j, \quad \{w_{ij}, \boldsymbol{b}_j\} \in \mathcal{C} \tag{1.1}$$

The representations $\boldsymbol{w}_i$ and the model parameters $\boldsymbol{b}_j$ are restricted by constraints in the set $\mathcal{C}$ to enforce prior knowledge. For example, nonnegativity constraints have demonstrated the ability to decompose images into more natural components corresponding to localized parts [80]. Similarly, sparsity has been shown to give rise to feature locality and frequency selectivity. The resulting learned features are very similar those observed experimentally in the mammalian primary visual cortex [112].

Learning an image decomposition amounts to finding parameters and representations that minimize the average reconstruction error subject to these constraints, as formalized in Equation 1.2.

$$\underset{w_{ij},\boldsymbol{b}_j}{\arg\min} \sum_{i=1}^{n} \left\| \boldsymbol{x}_i - \sum_{j=1}^{k} w_{ij}\boldsymbol{b}_j \right\|_2^2 \quad \text{s.t.} \quad \{w_{ij}, \boldsymbol{b}_j\} \in \mathcal{C} \tag{1.2}$$

One important benefit of this framework is its generality; depending on the application of interest, different features of the data can be emphasized simply by modifying the constraints in $\mathcal{C}$. Furthermore, certain constraints such as sparsity give representations with theoretically advantageous properties such as uniqueness, robustness, and generalization guarantees. However, despite these advantages, classical techniques for shallow representation learning have greatly reduced modeling capacity in comparison to deep alternatives.

1.2 Contributions

In this thesis, we propose new extensions that bridge the gap between classical data decomposition techniques and modern deep learning.

First, we observe that because of image variability and differing spatial layouts, some constraints would be inconsistent and impossible to enforce with traditional approaches based on matrix factorization. To broaden its scope, we derive a novel formulation of data decomposition posed as approximate constraint satisfaction that are able to handle richer prior knowledge. Instead of learning components that minimize average reconstruction error, we minimize their proximity to components that exactly reconstruct each training example. These auxiliary decompositions correspond to affine equality constraints, which can be enriched by other instance-level prior knowledge. We apply this new method to Semantic Component Analysis (SCA) where semantic segmentations is directly posed as image decomposition [101]. Through informative constraints that encourage spatially localized, non-overlapping regions, we achieve interpretable results on small datasets even in the absence of pixel-level annotations.

Despite their ability to effectively incorporate a wide range of prior knowledge, the assumption of linearity in classical data decomposition techniques has also limited its applicability to more complex visual learning applications. While it has been shown to be remarkably accurate for some data like aligned images of Lambertian objects such as faces [7], even small perturbations can introduce nonlinearities that bias the results. Kernel PCA handles nonlinear interactions by performing data decomposition in an implicit higher-dimensional reproducing kernel Hilbert space [129], but it is not optimized to effectively reconstruct the input data. Alternatively, manifold learning assumes that meaningful representations should preserve the local geometry of input data [140, 9]. However, these methods are often computationally expensive, difficult to interpret, and sensitive to noise. To address these issues, we propose Additive Component Analysis (ACA), a novel method for nonlinear component analysis that fits an unsupervised additive model [20] to data [102]. We extend our constraint satisfaction framework for data decomposition to explicitly optimize reconstruction error subject to intuitive constraints that penalize the roughness of the learned manifold. This framework can also be generalized to accommodate data restricted to known manifolds, as demonstrated in Approximate Grassmannian Projections, a method for subspace-valued data decomposition [103].

Despite advances in modeling nonlinearities, shallow data decomposition techniques are still limited by their attainable model capacity. Recently, deep neural networks have emerged as the preferred alternative to component analysis for representation learning of visual data. Their ability to jointly learn multiple layers of abstraction has been shown to allow for encoding increasingly complex features such as textures and object parts [81]. Unfortunately, “black-box” deep learning models are not yet well understood and do not share the same interpretability enabled by the intuitive constraints of data decomposition. In order to bridge the gap between these two techniques, we introduce the framework of Deep Component Analysis (DeepCA), a multilayer sparse coding model that shares the same practical advantages of deep learning [100]. Building upon theoretical connections to multilayer convolutional sparse coding [117, 135], DeepCA allows deep neural networks to encode prior knowledge through recurrent feedback connections that explicitly enforce constraints.

In addition to broadening the applicability of data decomposition, this relationship also provides a novel perspective for conceptualizing deep learning techniques. Despite their unrivaled practical advantages, deep neural networks still not well understood intuitively or theoretically [157]. By considering feed-forward deep networks as approximate solutions to sparse coding problems [117], we indirectly analyze complicated deep network architectures through the sparse dictionary structures that they induce. Sparse approximation theory specifies conditions that guarantee sparse representations to be uniquely identifiable and robust to input perturbations [39, 40]. Though not necessary, these properties are closely related to their ability to effectively memorize individual training examples and generalize to unseen validation data. We quantify deep networks by their maximum capacity to achieve these properties using the minimum deep frame potential, a bound on the induced dictionary structure. We propose to use this architecture-dependent measure as a cue for dataless model selection that does not require computationally expensive training and validation. Experimentally, we show correlation with validation error across different state-of-the-art families of architectures that are commonly used in computer vision applications.

1.3 Thesis Organization

In Chapter 2, we provide necessary background material. This includes an overview of techniques for both shallow and deep representation learning, different constraints and regularizers, methods for enforcing this prior knowledge with proximal optimization algorithms, theoretical challenges involved with characterizing generalization capacity in deep neural networks, and the foundations of sparse approximation theory.

In Chapter 3, we provide an interpretation of data decomposition as approximate constraint satisfaction. In Section 3.1 we present a novel formulation for data decomposition that supports a wide range of rich data-dependent constraints and propose a general optimization strategy for learning. In Section 3.2, we apply this framework to the problem of weakly-supervised image segmentation. Through instance-level constraints that explicitly encode the compositional structure of individual images through non-overlapping consistency constraints, we achieve semantically interpretable segmentations with few training examples and incomplete annotations. In Section 3.3, we use the same framework to relax the linearity assumptions of standard data decomposition through Additive Component Analysis. Instead of learning linear subspaces that span fixed component vectors, we learn smooth curvilinear manifolds constructed as linear combinations of nonlinear smoothing splines. To increase representational power, we compose multiple layers in a manner similar to deep learning. This allows for modeling more complex nonlinear interactions like image translation that cannot be effectively represented with curvilinear manifolds.

In Chapter 4, we describe theoretical and conceptual connections between deep learning and sparse approximation. In Section 4, we propose Deep Component Analysis, a novel framework for multi-layer data decomposition. Inference in these models can be performed using Alternating Direction Neural Networks, recurrent deep networks that implement an optimization algorithm for constrained optimization. We apply these networks to the task of single-image depth prediction with sparse output constraints and show that recurrent feedback connections robustly enforce prior knowledge for improved generalization performance. Finally, in Section 4.2, we show that these connections allow different deep networks to be quantified and compared indirectly using the minimum deep frame potential, a data-independent measure of architecture-induced dictionary structure. Correlations with validation error across a variety of practical densely connected and residual networks demonstrate the promising potential for better understanding deep learning through the lens of data decomposition.

This thesis is composed of material from the following publications:

[101] Calvin Murdock and Fernando De la Torre. Semantic component analysis. In *International Conference on Computer Vision (ICCV)*, 2015.

[102] Calvin Murdock and Fernando De la Torre. Additive component analysis. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[103] Calvin Murdock and Fernando De la Torre. Approximate grassmannian intersections: Subspace-valued subspace learning. In *International Conference on Computer Vision (ICCV)*, 2017.

[100] Calvin Murdock, Ming-Fang Chang, and Simon Lucey. Deep component analysis via alternating direction neural networks. In *European Conference on Computer Vision (ECCV)*, 2018.

[104] Calvin Murdock and Simon Lucey. Dataless model selection with the deep frame potential. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

2 Background

Identifying the underlying structure of data is one of the most important tasks in machine learning. As technological advancements facilitate the construction of datasets with increasing size and dimensionality, data analysis is becoming more challenging due to computational constraints and the curse of dimensionality. In the field of computer vision especially, data often consist of thousands if not millions of features, resulting in drastically increased training data requirements. Thus, alternative representations are necessary for efficiently and robustly encoding data for use in high-level applications such as recognition. In this chapter, we provide an overview of different techniques for learning image representations along with some theoretical tools for evaluating and predicting their effectiveness.

2.1 Visual Representation Learning

Despite their high dimensionality, real-world data often concentrate near manifolds with lower intrinsic dimensionality [105]. For example, while the typical image resolution of digital photographs is large, the space of natural images occupies an extremely small volume in comparison to that of all possible pixel instantiations. Because the geometric nature of data is typically unknown, a variety of properties have been proposed for encouraging the extraction of meaningful low-dimensional representations. Techniques for data decomposition are founded on the implicit assumption that useful representations are those that can accurately reconstruct input data. However, to enable effective generalization and interpretability, modeling assumptions or regularization often must be employed. Alternatively, manifold learning has acheived much success under the assumption that meaningful representations should preserve the local geometry of input data. However, these methods are often computationally expensive, difficult to interpret, and sensitive to noise.

2.1.1 Data Decomposition and Component Analysis

Component analysis methods play a key role in many computer vision applications due to its ability for linear and non-linear dimensionality reduction, denoising, feature extraction and exploratory data analysis [34]. Principal Component Analysis (PCA) fits a low dimensional subspace to data by finding directions of maximal variance. Though successful in restricted settings, early applications such as Eigenfaces [144] were unable to produce interpretable components. This was partially resolved through NMF, which demonstrated the ability to decompose images into more natural components corresponding to localized parts [80]. Numerous extensions have since been proposed to improve interpretability through localization constraints [84] or sparsity-inducing regularization [60]. Other approaches have explicitly modeled the physical process of occlusion by introducing additional latent variables that encode the ordering of objects in the scene [57]. However, all of these methods still require that all objects in different images be aligned, which is impractical for real images.

Data decomposition techniques that rely on matrix factorization approximate a matrix $\mathbf{X}$ (with data instances $\boldsymbol{x}_i$ as its columns) as the product of two matrices $\mathbf{W}$ and $\mathbf{B}$, i.e. as $\mathbf{X} \approx \mathbf{B}\mathbf{W}^\intercal$. This factorization approximately decomposes data $\boldsymbol{x} \in \mathbb{R}^d$ into linear combinations of learned components in $\mathbf{B} \in \mathbb{R}^{d \times k}$. In other words, data points are represented as linear combinations of a shared set of basis components, i.e. $\boldsymbol{x}_i \approx \mathbf{B}\boldsymbol{w}_i = \sum_j w_{ij}\boldsymbol{b}_j$ where $\boldsymbol{b}_j$ are the columns of $\mathbf{B}$ and $\boldsymbol{w}_i$ are the columns of $\mathbf{W}$. This is typically accomplished by minimizing reconstruction error subject to constraints $\mathcal{C}$ on the coefficients that serve to resolve ambiguity or incorporate prior knowledge such as low-rank structure or sparsity. Despite this, matrix factorization approaches are limited in their ability to incorporate more complicated priors. It is also unclear how they could be effectively applied to structured tasks like image segmentation in which semantic regions are known to be spatially localized in distinct, non-overlapping regions. Later in Section 3.2, we develop a technique that addresses these issues by allowing for rich, instance-level constraints that can depend on image content. This allows for the interpretable decomposition of image data into semantic segmentations.

While the problem of learning both the components and coefficients is typically non-convex, its structure naturally suggests simple alternating minimization strategies that are often guaranteed to converge [155]. However, these techniques typically require careful initialization in order to avoid poor local minima. This differs from backpropagation with stochastic gradient descent wherein random initializations are often sufficient. Alternatively, we consider a nested optimization problem that separates learning from inference:

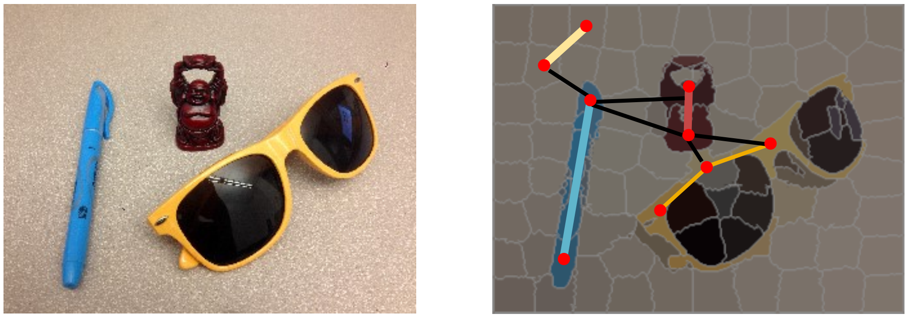
$$\arg\min_{\mathbf{B}} \sum_{i=1}^{n} \|\boldsymbol{x}^{(i)} - \mathbf{B}\boldsymbol{f}(\boldsymbol{x}^{(i)})\|_2^2 \quad \text{s.t.} \;\; \boldsymbol{f}(\boldsymbol{x}) = \arg\min_{\boldsymbol{w}\in\mathcal{C}} \; \|\boldsymbol{x} - \mathbf{B}\boldsymbol{w}\|_2^2 \tag{2.1}$$

Here, the inference function $\boldsymbol{f} : \mathbb{R}^d \to \mathbb{R}^k$ is a potentially nonlinear transformation that maps data to their corresponding representations by solving an optimization problem with fixed parameters. For unconstrained PCA with orthogonal components, this inference problem has a simple closed-form solution given by the linear transformation $\boldsymbol{f}^{\text{PCA}}(\boldsymbol{x}) = \mathbf{B}^{\mathsf{T}}\boldsymbol{x}$. Substituting this into Equation 2.1 results in a linear autoencoder with one hidden layer and tied weights, which has the same unique global minimum but can be trained by backpropagation [5].

With general constraints, inference typically cannot be accomplished in closed form but must instead rely on an iterative optimization algorithm. However, if this algorithm is composed as a finite sequence of differentiable transformations, then the model parameters can still be learned in the same way by backpropagating gradients through the steps of the inference algorithm. Later in Section 4, we extend this idea by representing an algorithm for inference as a recurrent neural network unrolled to a fixed number of iterations. This allows for very efficient learning for a general class of models with a wide variety of constraints and regularizers.

2.1.2 Prior Knowledge, Constraints, and Regularizers

A variety of techniques have been proposed for integrating constraints for enforcing prior knowledge about image structure. An example is shown in Figure 2.1,



(a) Original Image (b) Similarity Graph

Figure 2.1: The structure of images can be effectively represented though similarity graphs of image regions. Inference over these graphs can be used to enforce spatial consistency constraints for more accurate image segmentations with limited training data.

where similarities between the appearance of image regions are represented using superpixel graphs. In Section 3.2, we show how inference over these graphs can enforce spatial consistency constraints for image components resulting in interpretable image decompositions into semantic regions.

Some other methods have attempted to explicitly address the need for representations that are invariant to uninformative image variations. This is usually accomplished by simultaneously aligning and decomposing the images in an alternating manner. For example, [46] introduced discrete latent variables that select from predefined linear image transformations. Similarly, [67] learned translation-invariant appearance and occlusion models for videos. To be able to scale to higher parametric models, [35] proposed parameterized component anlaysis. However, these types of methods are typically restricted to small parametric classes of image transformations (e.g. translation or rotation) and cannot account for multiple objects or strong changes in pose.

Identifying and localizing the semantic classes within an image is an example of a task for which invariances cannot be easily parametrized. In addition to accounting for non-rigid transformations, large intra-class appearance variations must also be considered. Thus, none of the techniques described above would be able to give a semantically-meaningful separation into classes.

Without pixel-wise labeling of training images, simple discriminative models are no longer viable. Some weakly-supervised approaches attempt to simultaneously learn discriminative classifiers alongside object locations through alternating methods like multiple-instance learning [58, 30] or matrix completion [21]. Others use graphical models that enforce consistency both within and across images to ensure class similarity [147, 154, 159]. However, exact inference in these models is typically intractable, so approximate methods must be used instead. Furthermore, all of these methods require large, non-convex optimization problems that are sensitive to initialization and do not scale well to large data sets. Leveraging the recent work in the optimization of deep networks, approaches based on CNNs have resulted in high-quality segmentations even without full supervision [121, 114, 128, 116, 120]. However, none of these approaches can be used for the unsupervised clustering of images into semantically-meaningful regions.

Instead, most approaches to this problem incorporate prior knowledge about class appearance and image composition to guide image segmentations or bounding box localizations. If fully-supervised training data is available, the most effective method is to train discriminative models that can be used to directly classify individual image regions. These local predictions are typically guided towards global consistency using prior knowledge such as local similarity [23, 47, 68], contextual geometric constraints [141], or agreement between multiple independent segmentations [2, 65]. Unlike component analysis, most methods for visual recognition are fully-supervised and make use of bounding boxes or pixel-wise segmentations to locate objects of interest. However, this type of manual labeling is time consuming, error-prone, and potentially suboptimal [109]. On the other hand, the increasing prevalence of large image collections emphasizes the need for fully- or partially-automated techniques for analyzing and archiving their content.

2.1.3 Nonlinear Dimensionality Reduction

Numerous attempts have been made to model more complex data by incorporating nonlinearities within a component analysis framework. The most prominent example is kernel PCA [129], which handles nonlinear interactions implicitly by performing PCA in a higher-dimensional reproducing kernel Hilbert space using the kernel trick. However, it is not optimized to effectively reconstruct the input data which limits its applicability. While out-of-sample inference is enabled via a representer theorem, there is no clear back-projection from the latent space to the original input space due to the pre-image problem [76]. Furthermore, the computational requirements of kernel PCA prevent its use on large datasets. Similarly, Gaussian Process Latent Variable Models [77] provide a general probabilistic interpretation of nonlinear PCA, but still suffer from many of the same issues due to the kernelized covariance function. Recently, approximate kernel methods have been proposed to improve computational efficiency. In [123], data are explicitly mapped to a randomized feature space in which inner products approximate kernel function evaluations. Using this idea, random nonlinear features have led to scalable algorithms for nonlinear PCA [90]. However, these approaches all first transform the input data, preventing their effective application to data reconstruction and denoising.

Methods for manifold learning find low-dimensional data representations by minimizing local geometric distortions, e.g. [140, 9]. Most often formalized as eigendecompositions, these algorithms do not learn explicit mappings to the latent space and thus cannot support back-projection or out-of-sample extensions directly [13]. Furthermore, these techniques tend to be topologically unstable, relying on unintuitive hyper-parameters (e.g. neighborhood size) that require careful tuning in order to avoid degenerate behavior like short circuiting [4].

Unlike parametric methods that have a fixed complexity, nonparametric methods can adapt to the data, allowing for the representation of a wide range of nonlinearities. However, they are ineffective in high-dimensional settings due to the large amount of training data required to effectively characterize full data distributions [151]. To address this issue, additive models consider a smaller class of nonparametric functions that decompose into sums of univariate functions considering each input dimension independently [20] via smoothing splines, piecewise polynomial functions with roughness penalties that encourage functions with small second derivatives [151]. Other nonparametric methods have also generalized the notion of principal components as geometric objects passing through the center of data [52, 115], but they cannot generally be used for dimensionality reduction. The method that is most similar to ACA is [24], which also learns explicit nonparametric functions to minimize a least-squares objective, but requires good initializations and is intractable for large datasets.

2.1.4 Deep Neural Networks

Deep neural networks have emerged as the preferred technique for representation learning of visual data. Their ability to jointly learn multiple layers of abstraction has been shown to allow for encoding increasingly complex features such as textures and object parts [81]. Unlike with component analysis, inference is given in closed-form by design. Autoencoders are unsupervised deep networks attempt to reconstruct data by learning explicit nonlinear mappings to and from latent representations. While shown to be equivalent to PCA in the linear case [5], nonlinear activation functions and stacking can enable a rich class of nonlinear representations [149]. In fact, some deep learning models can be interpreted as learning data manifolds [11] or lower-dimensional distributions [70].

Deep networks have had the most success in fully-supervised scenarios due to the ability to train image representations jointly with objectives such as classifcation or regression. Specifically, a representation is constructed by passing an image $\boldsymbol{x}$ through the composition of alternating linear transformations with parameters $\mathbf{B}_j$ and $\boldsymbol{b}_j$ and fixed nonlinear activation functions ϕ_j for layers $j = 1, \ldots, l$ as follows:

$$\boldsymbol{f}^{\mathrm{DNN}}(\boldsymbol{x}) = \phi_l\big(\mathbf{B}_l^{\mathsf{T}} \cdots \phi_2(\mathbf{B}_2^{\mathsf{T}}(\phi_1(\mathbf{B}_1^{\mathsf{T}}\boldsymbol{x} - \boldsymbol{b}_1) - \boldsymbol{b}_2) \cdots - \boldsymbol{b}_l\big) \tag{2.2}$$

Then, learning is accomplished using first-order optimization techniques that minimize a loss function ℓ that measures discrepancy with training annotations. Updates are made jointly with respect to the parameters in all layers via backpropagation, an application of the chain rule for computing gradients of function compositions. The general optimization problem with supervision $\boldsymbol{y}$ is:

$$\operatorname*{arg\,min}_{\{\mathbf{B}_j, \boldsymbol{b}_j\}} \sum_{i=1}^{n} \ell(\boldsymbol{f}^{\mathrm{DNN}}(\boldsymbol{x}^{(i)}),\, \boldsymbol{y}^{(i)}) \tag{2.3}$$

Instead of considering the forward pass of a neural network as an arbitrary nonlinear function, we interpret it as a method for approximate inference in an unsupervised generative model. This follows from previous work which has shown it to be equivalent to bottom-up inference in a probabilistic graphical model [119] or approximate inference in a multi-layer convolutional sparse coding model [117, 135]. However, these approaches have limited practical applicability due to their reliance on careful hyperparameter selection and specialized optimization algorithms. While ADMM has been proposed as a gradient-free alternative to backpropagation for parameter learning [139], we use it only for inference which allows for simpler learning using backpropagation with arbitrary loss functions.

Recurrent feedback has been proposed to improve performance by iteratively refining predictions, especially for applications such as human pose estimation or image segmentation where outputs have complex correlation patterns [22, 8, 83]. While some methods also implement feedback by directly unrolling iterative algorithms, they are often geared towards specific applications such as graphical model inference [29, 61], solving under-determined inverse problems [50, 37, 136], or image alignment [86]. Similar to [156], we provide in Section 4 a more general mechanism for low-level feedback in arbitrary neural networks that is motivated by the more interpretable goal of minimizing reconstruction error subject to constraints on network activations.

While these methods employ explicit nonlinear mappings for reconstructing the original data, it is not yet clear how different regularization techniques and model architectures affect the space of learnable nonlinear functions [157], so they tend to require significant engineering effort and still often result in overfitting and poor interpretability. Due to the vast space of possible deep network architectures and the computational difficulty in training them, deep model selection is still largely been guided by ad-hoc engineering and human ingenuity. While progress slowed in the years following early breakthroughs [78], recent interest in deep learning architectures began anew due to empirical successes largely attributed to computational advances like efficient training using GPUs and rectified linear unit (ReLU) activation functions [73]. Since then, numerous architectural changes have been proposed. For example, much deeper networks with residual connections were shown to achieve consistently better performance with fewer parameters [54]. Building upon this, densely connected convolutional networks with skip connections between more layers yielded even better performance [62].

While theoretical explanations for these improvements were lacking, consistent experimentation on standardized benchmark datasets continued to drive empirical success.

However, due to slowing progress and the need for increased accessibility of deep learning techniques to a wider range of practitioners, more principled approaches to architecture search have recently gained traction. Motivated by observations of extreme redundancy in the parameters of trained networks [36], techniques have been proposed to systematically reduce the number of parameters without adversely affecting performance. Examples include sparsity-inducing regularizers during training [1] or through post-processing to prune the parameters of trained networks [56]. Constructive approaches to model selection like neural architecture search [44] instead attempt to compose architectures from basic building blocks through tools like reinforcement learning. Efficient model scaling has also been proposed to enable more effective grid search for selecting architectures subject to resource constraints [138]. While automated techniques can match or even surpass manually engineered alternatives, they require a validation dataset and and rarely provide insights transferable to other settings.

2.2 Theoretical Foundations

We are motivated by theoretical connections between deep neural networks and sparse approximation. Consider the feed-forward deep neural network from Equation 2.3, which is constructed as the composition of linear transformations with parameters $\mathbf{B}_j$ and nonlinear activation functions ϕ_j. Equivalently, it can be represented as $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{w}_l$ where $\boldsymbol{w}_j = \mathbf{B}_j^\top \boldsymbol{w}_{j-1} - \boldsymbol{b}_j$ for $j = 1, \dots, l$ and $\boldsymbol{w}_0 = \boldsymbol{x}$. In many modern state-of-the-art networks, the ReLU activation function has been adopted due to its effectiveness and computational efficiency. It can also be interpreted as the nonnegative soft-thresholding proximal operator associated with the function Φ in Equation 4.3, a nonnegativity constraint and a sparsity-inducing ℓ_1 penalty with a weight determined by the scalar bias parameter λ.

$$\begin{gathered}\Phi(\boldsymbol{w}) = \mathbb{I}(\boldsymbol{w} \geq \mathbf{0}) + \lambda \left\|\boldsymbol{w}\right\|_1 \\ \phi(\boldsymbol{x}) = \operatorname{ReLU}(\boldsymbol{x} - \lambda\mathbf{1}) = \arg\min_{\boldsymbol{w}} \tfrac{1}{2} \left\|\boldsymbol{w} - \boldsymbol{x}\right\|_2^2 + \Phi(\boldsymbol{w})\end{gathered} \tag{2.4}$$

Thus, the forward pass of a deep network is equivalent to a layered thresholding pursuit algorithm for approximating the solution of a multi-layer sparse coding model [117]. Results from shallow sparse approximation theory can then be adapted to bound the accuracy of this approximation, which improves as mutual coherence decreases, and indirectly analyze other theoretical properties of deep networks like uniqueness and robustness.

2.2.1 Deep Network Generalization

To better understand the implicit benefits of different network architectures, there have been adjacent theoretical explorations of deep network generalization. These works are often motivated by the surprising observation that good performance can still be achieved using highly over-parametrized models with degrees of freedom that surpass the number of training data. This contradicts many commonly accepted ideas about generalization, spurning new experimental explorations that have demonstrated properties unique to deep learning. Examples include the ability of deep networks to express random data labels [158] with a tendency towards learning simple patterns first [3]. While exact theoretical explanations are lacking, empirical measurements of network sensitivity such as the Jacobian norm have been shown to correlate with generalization [111]. Similarly, Parseval regularization [98] encourages robustness by constraining the Lipschitz constants of individual layers.

Due to the difficulty in analyzing deep networks directly, other approaches have instead drawn connections to the rich field of sparse approximation theory. The relationship between feed-forward neural networks and principal component analysis has long been known for the case of linear activations [5]. More recently, nonlinear deep networks with ReLU activations have been linked to multilayer sparse coding to prove theoretical properties of deep representations [117]. This connection has been used to motivate new recurrent architecture designs that resist adversarial noise attacks [124], improve classification performance [134], or enforce prior knowledge through output constraints [99].

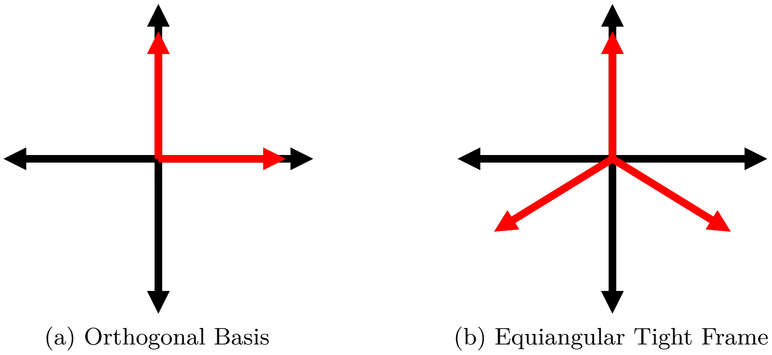


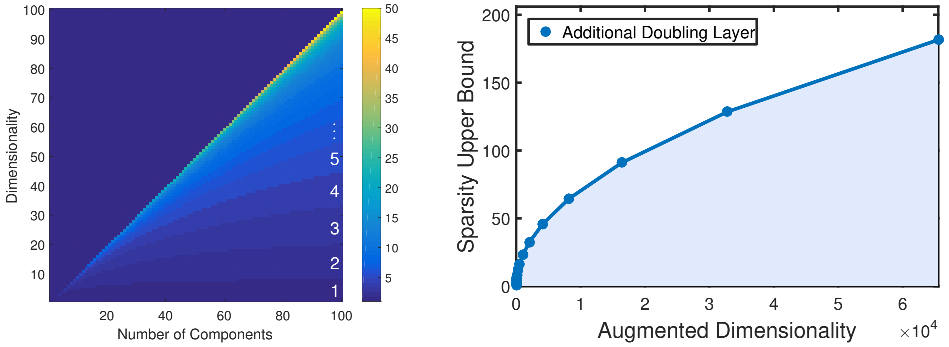
Figure 2.2: A comparison between (a) an orthogonal basis and (b) an overcomplete equiangular tight frame. Overcomplete representations allow for redundantly representing data in higher dimensions. Low mutual coherence and sparsity constraints ensure representation efficiency, uniqueness, and robustness.

2.2.2 Sparse Approximation Theory

Sparse approximation theory considers representations of data vectors $\boldsymbol{x} \in \mathbb{R}^d$ as sparse linear combinations $\boldsymbol{x} \approx \sum_j w_j \boldsymbol{b}_j = \mathbf{B}\boldsymbol{w}$ of atoms from an over-complete dictionary $\mathbf{B} \in \mathbb{R}^{d \times k}$. The number of atoms k is greater than the dimensionality d and the number of nonzero coefficients $\|\boldsymbol{w}\|_0$ in the representation $\boldsymbol{w} \in \mathbb{R}^k$ is constrained to be small.

Through applications like compressed sensing [38], sparsity has been found to exhibit theoretical properties that enable data representation with efficiency far greater than what was previously thought possible. Central to these results is the requirement that the dictionary be "well-behaved," essentially ensuring that its columns are not too similar. For undercomplete matrices with $k \leq d$, this is satisfied by enforcing orthogonality, but overcomplete dictionaries require other conditions. Specifically, we focus our attention on mutual coherence μ of the dictionary $\mathbf{B}$, the maximum magnitude normalized inner product of all pairs of dictionary atoms. Equivalently, it is the maximum magnitude off-diagonal element in the Gram matrix $\mathbf{G} = \tilde{\mathbf{B}}^\top \tilde{\mathbf{B}}$ where the columns of $\tilde{\mathbf{B}}$ have unit norm:

$$\mu = \max_{i \neq j} \frac{|\boldsymbol{b}_i^\top \boldsymbol{b}_j|}{\|\boldsymbol{b}_i\| \, \|\boldsymbol{b}_j\|} = \max_{i,j} |(\mathbf{G} - \mathbf{I})_{ij}| \tag{2.5}$$



(a) Sparsity Level Bounds

(b) Capacity with Increasing Depth

Figure 2.3: Visualizations of upper bounds on the number of nonzero elements required for solutions of overdetermined linear systems to be unique (a). In a DeepCA model, as additional layers are added to double the activation dimensionality, the uniqueness capacity increases sublinearly with respect to the dimensionality of the augmented shallow system (b).

Figure 2.2 shows an example comparing a complete orthogonal basis with an overcomplete dictionary that has minimal mutual coherence.

We are primarily motivated by the observation that a model's capacity for low mutual coherence increases along with its capacity for both the *memorization* of training data–through unique representations–and the *generalization* to validation data–through robustness to input perturbations.

With an overcomplete dictionary, there is an infinite space of coefficients $\boldsymbol{w}$ that can exactly reconstruct any data point as $\boldsymbol{x} = \mathbf{B}\boldsymbol{w}$, which would not support discriminative representation learning. However, if representations from a mutually incoherent dictionary are sufficiently sparse, then they are necessarily optimal and unique [39]. Specifically, if $\|\boldsymbol{w}\|_0 < \frac{1}{2}(1+\mu^{-1})$, then $\boldsymbol{w}$ is the unique, sparsest representation for $\boldsymbol{x}$. Furthermore, if $\|\boldsymbol{w}\|_0 < (\sqrt{2}-0.5)\mu^{-1}$, then it can be found efficiently by convex optimization through ℓ_1 regularization. Thus, minimizing the mutual coherence of a dictionary increases its capacity for uniquely representing data points for improved *memorization.* Figure 2.3 demonstrates the effect of dictionary size on the minimum achievable mutual coherence.

Sparse representations are also robust to input perturbations [40]. Specifically, given a noisy datapoint $\boldsymbol{x} = \boldsymbol{x}_0 + \boldsymbol{z}$ where $\boldsymbol{x}_0$ can be represented exactly as $\boldsymbol{x}_0 = \mathbf{B}\boldsymbol{w}_0$ with $\|\boldsymbol{w}_0\|_0 \leq \frac{1}{4}\left(1 + \mu^{-1}\right)$ and the noise $\boldsymbol{z}$ has bounded magnitude $\|\boldsymbol{z}\|_2 \leq \epsilon$, then $\boldsymbol{w}_0$ can be approximated by solving the ℓ_1-penalized LASSO problem:

$$\arg\min_{\boldsymbol{w}} \|\boldsymbol{x} - \mathbf{B}\boldsymbol{w}\|_2^2 + \lambda \|\boldsymbol{w}\|_1 \tag{2.6}$$

It's solution is stable and the approximation error is bounded from above in Equation 2.7, where $\delta(\boldsymbol{x}, \lambda)$ is a constant.

$$\|\boldsymbol{w} - \boldsymbol{w}_0\|_2^2 \leq \frac{(\epsilon + \delta(\boldsymbol{x}, \lambda))^2}{1 - \mu(\mathbf{B})(4\|\boldsymbol{w}\|_0 - 1)} \tag{2.7}$$

Thus, minimizing the mutual coherence of a dictionary decreases the sensitivity of its sparse representations for improved *generalization*. This is similar to evaluating input sensitivity using the Jacobian norm [111]. However, instead of estimating the average perturbation error over validation data, it bounds the worst-cast error over all possible data.

3 Data Decomposition as Approximate Constraint Satisfaction

Despite their support for constrained inference, traditional matrix factorization is often unable to incorporate more complicated priors. Specifically, since the learned components $\boldsymbol{b}_j$ are shared amongst all training examples, these techniques have limited applicability to structured tasks that are naturally represented by data-dependent constraints. For example, the task of image segmentation can be described as decomposing each image into a unique set of semantic regions that are spatially localized to distinct regions. This prior knowledge cannot be effectively represented as a single constraint set $\mathcal{C}$.

3.1 Instance-Level Data Decomposition

Instead of approximating data as combinations of shared parameters, we propose an exact data decomposition of each image feature vector into its own distinct set of instance components $\boldsymbol{h}_{ij}$, as shown in Equation 3.1.

$$\boldsymbol{x}_i \approx \sum_{j=1}^{k} w_{ij} \boldsymbol{b}_j \quad \Longrightarrow \quad \boldsymbol{x}_i = \sum_{j=1}^{k} w_{ij} \boldsymbol{h}_{ij} \tag{3.1}$$

Exposing these latent components allows for easily incorporating instance-level semantic constraints $\mathcal{C}_i$ related to *a priori* knowledge about individual data points $\boldsymbol{x}_i$, such as the layout and composition of objects within images.

While learning $m \times n$ components from only n training examples may seem intractable, we restrict the learning process by explicit enforcing similarity between instance components with same index j, which could, for example, correspond to different semantic classes. We formalize this intuition with the optimization problem in Equation 3.2, which constrains the global image feature vector $\boldsymbol{x}_i$ to equal a linear combination of its constrained instance components $\boldsymbol{h}_{ij}$ while minimizing the sum of weighted distances to exemplar components $\boldsymbol{b}_j$.

$$\operatorname*{arg\,min}_{w_{ij},\boldsymbol{h}_{ij},\boldsymbol{b}_j} \sum_{i=1}^{n} \sum_{j=1}^{k} w_{ij}^2 \left\|\boldsymbol{h}_{ij} - \boldsymbol{b}_j\right\|_2^2 \quad \text{s.t.} \quad \sum_{j=1}^{k} w_{ij}\boldsymbol{h}_{ij} = \boldsymbol{x}_i,\ \{w_{ij}, \boldsymbol{h}_{ij}\} \in \mathcal{C}_i \tag{3.2}$$

This formulation attempts to regularize the solution for $\boldsymbol{h}_{ij}$ by shrinking them towards related instance components while adhering to the instance-level constraints in $\mathcal{C}_i$ that can vary across data points $\boldsymbol{x}_i$. Learning the shared model parameters $\boldsymbol{b}_j$ then amounts to finding a set of components that is closest to satisfying all of the constraints, i.e. the approximate intersection of sets $\boldsymbol{C}_i$ for each instance i in the training set. Unlike matrix factorization approaches, explicitly decomposing a shared basis into separate instance components allows for richer constraints that would otherwise not be possible.

Because it appears to be accomplishing a very different goal, it is natural to ask how our approximate constraint satisfaction objective differs from the standard reconstruction error minimization objective in Equation 1.2. Equation 3.3 below shows an equivalent objective after substituting the auxiliary decomposition of $\boldsymbol{x}_i$ from Equation 3.1.

$$\operatorname*{arg\,min}_{w_{ij},\boldsymbol{h}_{ij},\boldsymbol{b}_j} \sum_{i=1}^{n} \left\| \sum_{j=1}^{k} w_{ij}\left(\boldsymbol{h}_{ij} - \boldsymbol{b}_j\right) \right\|_2^2 \quad \text{s.t.} \quad \sum_{j=1}^{k} w_{ij}\boldsymbol{h}_{ij} = \boldsymbol{x}_i,\ \{w_{ij}, \boldsymbol{h}_{ij}\} \in \mathcal{C}_i \tag{3.3}$$

When expanded, the squared norm introduces additional cross terms in the form of $(\boldsymbol{h}_{ij} - w_{ij}\boldsymbol{b}_j)^\intercal(\boldsymbol{h}_{ik} - w_{ik}\boldsymbol{f}_k)$ for $k \neq j$ that do not appear in our objective. Despite this, unconstrained versions of both formulations achieve exactly the same solutions, justifying our interpretation of data decomposition as approximate constraint satisfaction.

To see why these two problems are equivalent, consider solving Equation 3.2 only for the auxiliary components $\boldsymbol{h}_{ij}$ with the shared parameters $\boldsymbol{b}_j$ and latent representations $\boldsymbol{w}_i$ fixed. This decomposes into independent subproblems for each data instance $i = 1, \ldots, n$. If we concatenate the variables so that $\mathbf{G}_i = [w_{i1}\boldsymbol{h}_{i1}, \ldots, w_{ik}\boldsymbol{h}_{ik}]$ and $\mathbf{F}_i = [w_{i1}\boldsymbol{b}_{i1}, \ldots, w_{ik}\boldsymbol{b}_{ik}]$ The scaled instance components in $\mathbf{G}_i$ can be found by solving the optimization problem in Equation 3.4, where $\mathbf{A} = \mathbf{1}_m^\intercal \otimes \mathbf{I}_d$ and $\otimes$ denotes the Kronecker product.

$$\underset{\mathbf{G}_i}{\arg\min} \, \|\mathbf{G}_i - \mathbf{F}_i\|_F^2 \quad \text{s.t.} \;\; \mathbf{A}\text{vec}(\mathbf{G}_i) = \boldsymbol{x}_i \tag{3.4}$$

Note that this is simply the projection of the scaled shared components in $\mathbf{F}_i$ onto the affine subspace defined by the equality constraint of the exact instance decomposition. Thus, its solution is given in closed form in Equation 3.5, where $\mathbf{A}^+\boldsymbol{x}_i$ is a point on the affine subspace and the columns of $\mathbf{N}$ form an orthonormal basis for the nullspace of $\mathbf{A}$.

$$\text{vec}(\mathbf{G}_i) = \mathbf{A}^+\boldsymbol{x}_i + \mathbf{N}\mathbf{N}^\intercal \left(\text{vec}(\mathbf{F}_i) - \mathbf{A}^+\boldsymbol{x}_i\right) \tag{3.5}$$

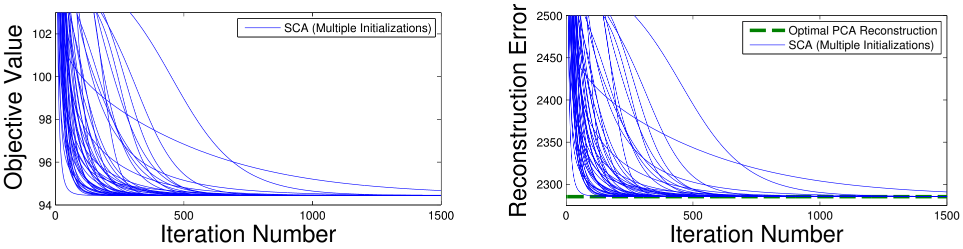
Here, $\mathbf{A}^+\boldsymbol{x}_i = \frac{1}{k}\boldsymbol{x}_i \otimes \mathbf{1}$ and $\mathbf{N}\mathbf{N}^\intercal = \left(\mathbf{I} - \frac{1}{k}\mathbf{1}\mathbf{1}^\intercal\right) \otimes \mathbf{I}$. After some simplification, the instance components $\boldsymbol{h}_{ij}$ can be found from Equation 3.6:

$$w_{ij}\boldsymbol{h}_{ij} = w_{ij}\boldsymbol{b}_{ij} + \tfrac{1}{k}\left(\boldsymbol{x}_i - \mathbf{B}\boldsymbol{w}_i\right) \tag{3.6}$$

Intuitively, this can be interpreted as distributing the current approximation error equally among the instance components $\boldsymbol{h}_{ij}$ so that they sum to $\boldsymbol{x}_i$. Plugging this back into our problem in Equation 3.2 gives the original reconstruction error minimization objective function from Equation 1.2 rescaled by k^{-1}. Thus, both problems have exactly the same solutions.

3.1.1 Alternating Optimization

Parameter learning for this problem naturally lends itself to an efficient alternating minimization algorithm inspired by the problem of finding the approximate intersection of convex sets. After initialization, we fix the shared components $\boldsymbol{b}_j$



(a) Objective Value

(b) Reconstruction Error

Figure 3.1: Our algorithm's convergence without constraints on synthetic data and 50 random initializations. Despite its alternating nature, our approach is robust to initialization and typically converges very quickly in both objective value (a) and reconstruction error from projection onto the exemplar components (b).

and jointly solve for the latent representations $\boldsymbol{w}_i$ and instance components $\boldsymbol{h}_{ij}$.

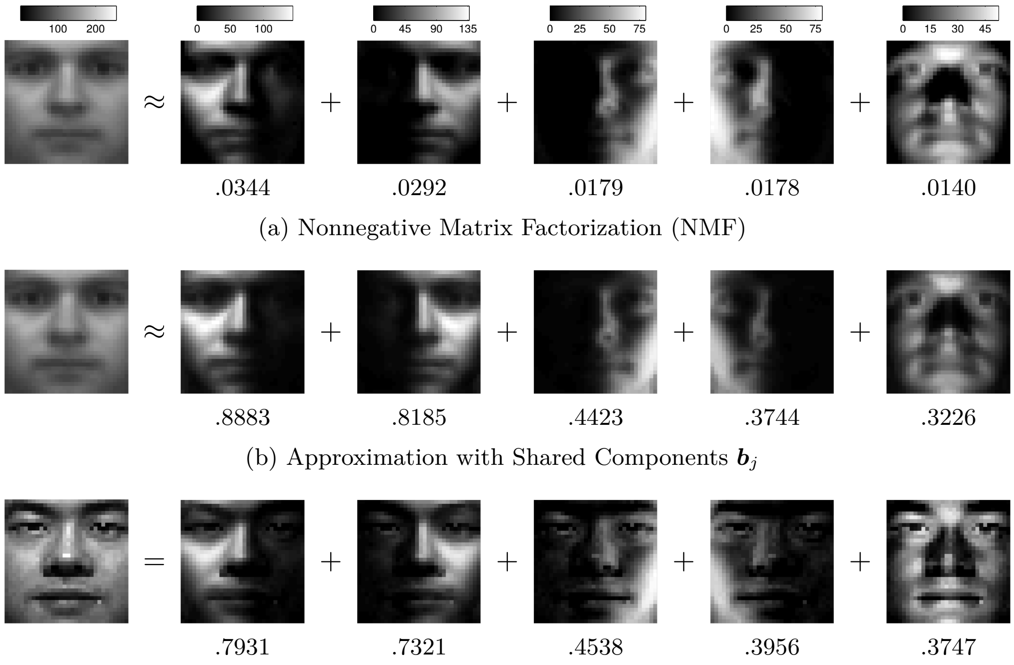
$$\begin{gathered} \underset{w_{ij},\tilde{\boldsymbol{h}}_{ij}}{\arg\min} \sum_{j=1}^{k} \tilde{\boldsymbol{h}}_{ij}^{\mathsf{T}} \begin{bmatrix} \mathbf{I} & -\boldsymbol{b}_j \\ -\boldsymbol{b}_j^{\mathsf{T}} & \boldsymbol{b}_j^{\mathsf{T}}\boldsymbol{b}_j \end{bmatrix} \tilde{\boldsymbol{h}}_{ij} \quad \text{s.t. } \tilde{\boldsymbol{h}}_{ij} = \begin{bmatrix} w_{ij}\boldsymbol{h}_{ij} \\ w_{ij} \end{bmatrix}, \\ \sum_{j=1}^{k} \begin{bmatrix} \mathbf{I}_d & \mathbf{0} \end{bmatrix} \tilde{\boldsymbol{h}}_{ij} = \boldsymbol{x}_i,\ \{w_{ij}, \boldsymbol{h}_{ij}\} \in \mathcal{C}_i \end{gathered} \tag{3.7}$$

Then, with these variables fixed, we solve for the shared components $\boldsymbol{b}_j$.

$$\underset{\boldsymbol{b}_j}{\arg\min} \sum_{i=1}^{n} w_{ij}^2 \left\| \boldsymbol{h}_{ij} - \boldsymbol{b}_j \right\|_2^2 = \frac{\sum_{i=1}^{n} w_{ij}^2 \boldsymbol{h}_{ij}}{\sum_{i=1}^{n} w_{ij}^2} \tag{3.8}$$

This process is repeated until convergence. Despite the nonconvexity of our problem, each of these subproblems is convex and the alternating optimization procedure has been shown to converge consistently to good solutions for a variety of applications. In Figure 3.1, we demonstrate empirically that our algorithm converges quickly and is robust to initialization. Later in Section 3.2, we show that this alternating strategy is also effective for the problem of weakly-supervised semantic segmentation.

While a separate set of instance components are learned for each image, the exemplar components $\boldsymbol{b}_j$ can be represented simply as the weighted average of

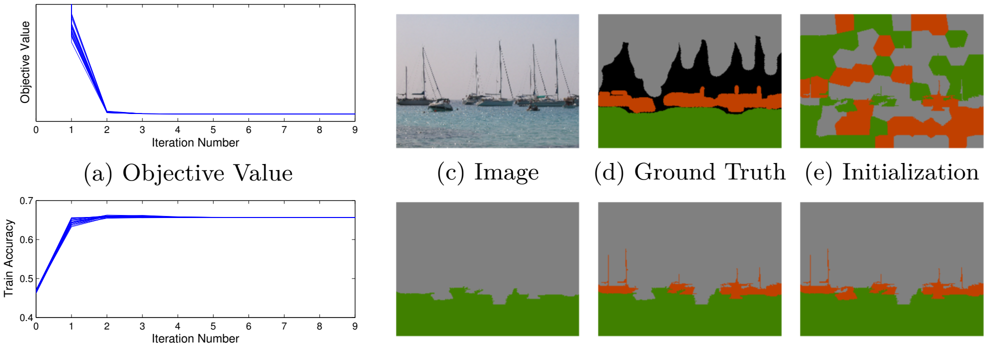


(a) Nonnegative Matrix Factorization (NMF)

(b) Approximation with Shared Components $\boldsymbol{b}_j$

(c) Decomposition with Instance Components $\boldsymbol{h}_{ij}$

Figure 3.2: A comparison of the components found through instance decomposition with traditional nonnegative matrix factorization (NMF). The first column in each row shows a reconstructed image while the next five columns show the components used and the corresponding coefficients that minimize its reconstruction error. Row (i) uses the basis found through matrix factorization, (ii) the shared exemplar components $\boldsymbol{b}_j$ of SCA, and (iii) the instance components $\boldsymbol{h}_{ij}$ of SCA that exactly reconstruct the image. The qualitative similarity between these components and the comparable reconstruction performance suggests a close relationship between SCA and traditional matrix factorization, despite their different objective functions.



(a) Objective Value

(c) Image

(d) Ground Truth

(e) Initialization

(b) Training Pixel Accuracy

(f) Iteration 1

(g) Iteration 2

(h) Iteration 3

Figure 3.3: Left: The convergence of our algorithm on the MSRC2 data set with weak labels. Showing 20 random initializations, both the objective value (a) and the training accuracy (b) consistently converge to the same values after only around 3 iterations. Right: Example segmentations at different points in the training process. After iteration 1, the large water and sky regions are successfully found, while iterations 2 and 3 segment the smaller boats.

all instance components $\boldsymbol{h}_{ij}$ sharing the same index j. Importantly, unlike other methods employing high-dimensional and over-complete bases, the many instance components of SCA are *not* estimated independently; they are related through the smaller set of exemplar components, which can be interpreted as a shared basis representative of the training data.

Despite their seemingly unfamiliar construction, we empirically found that the exemplar components of SCA share close connections between the bases learned through traditional matrix factorization techniques. For comparison purposes, we use the shared exemplar components as a basis that can approximately reconstruct data in the same manner as PCA or NMF. Without any additional semantic constraints $\mathcal{C}_i$, this basis consistently achieves reconstruction performance comparable to that of PCA despite the different objective function. In addition, by introducing nonnegativity constraints on both w_{ij} and $\boldsymbol{h}_{ij}$, the resulting exemplar components are qualitatively similar to the basis vectors found through NMF. This is shown in Figure 3.2, which gives a visual comparison between our method and NMF.

3.1.2 Robustness via Trimmed Averaging

The Grassmann Average [53] (GA) is a recent method for scalable dimensionality reduction that represents data points as one-dimensional subspaces and constructs a leading component as their spherical average. This is very similar to our method which also represents components as weighted averages.

Specifically, GA can be considered a special case of our problem for a single component ($m = 1$) with the additional constraints $w_i = \pm 1$ and $\|\boldsymbol{b}\|_2 = 1$. After incorporating these constraints, Equation 3.2 can be written as:

$$\arg\max_{w_i, \boldsymbol{b}} \sum_{i=1}^{n} w_i \boldsymbol{x}_i^\intercal \boldsymbol{b} \quad \text{s.t. } w_i = \pm 1, \ \|\boldsymbol{b}\|_2 = 1 \tag{3.9}$$

Note that $w_i = 1$ if and only if $\boldsymbol{x}_i^\intercal \boldsymbol{b}$ is positive. Thus, the objective can be equivalently represented by replacing the multiplication of $\boldsymbol{x}_i^\intercal \boldsymbol{b}$ by w_i with an absolute value, resulting in exactly the same problem solved by GA.

One of the main benefits of GA is that robustness can be easily incorporated simply by using the robust feature-wise trimmed average (in which the smallest and largest $P\%$ of values are ignored) in place of the ordinary average, which is highly sensitive to outliers. We apply this same idea to introduce robustness to our algorithm as well, which was found to be particularly effective in cases when supervision is minimal or altogether unavailable. However, while GA must rely on greedy methods for acquiring more than just the leading component (which could affect what is considered to be an outlier), our algorithm is able to estimate multiple components simultaneously.

3.2 Semantic Component Analysis (SCA)

Real-world images are often composed of a number of distinct (but semantically-related) regions. A natural aim of visual learning is to find these meaningful regions in an unsupervised or weakly-supervised manner. For instance, consider Figure 3.4(a): it is clear that there are four component objects that can explain the given images. The question is how to recover these semantic components with minimal supervision. Algorithms that approach this problem face many challenges, primarily in dealing with large intra-class variability in appearance, illumination, and pose.

A generative model for image formation can be considered as mixing a number of semantic components: one for each class present within an image. While the same local image features (e.g. quantized sift descriptors) may appear in instances from different classes, the *distributions* of features within semantic regions are often distinct across classes. If these global image features could be unmixed into their semantic components–each representing consistent segmentations belonging only to a single class–then recognition tasks could be simplified dramatically. This problem motivates a component analysis (CA) approach to image understanding in which an image is decomposed into *semantic* components.

Image decomposition is often accomplished through matrix factorization techniques, such as Principal Component Analysis (PCA) [144], Non-negative Matrix Factorization (NMF) [80], or Probabilistic Latent Semantic Analysis (pLSA) [133]. These methods approximate data as linear combinations of latent factors by minimizing total reconstruction error. While some variations of these approaches can result in localized, semantically-meaningful, or parts-based image decompositions, they are generally unable to adhere to a key property of image formation: objects are *occlusive*, i.e. image formation is nonlinear in pixel space because an object occludes everything behind it. Thus, images tend to consist of contiguous groups of pixels that belong only to a single object class. On the other hand, matrix decompositions represent each pixel as a superposition of multiple components. Since they rely on a *shared* basis that only *approximates* the original data, modifying these methods to enforce semantically-meaningful components by incorporating such nonlinear pixel-level constraints with real-word, unaligned images is nontrivial.

In the last decade, image classification has become an incredibly active research topic with widespread applications. Most methods for visual recognition are fully-supervised and make use of bounding boxes or pixel-wise segmentations to locate objects of interest. However, this type of manual labeling is time consuming, error-prone, and potentially suboptimal [109]. On the other hand, the increasing prevalence of large image collections emphasizes the need for fully- or partially-automated techniques for analyzing and archiving their content.

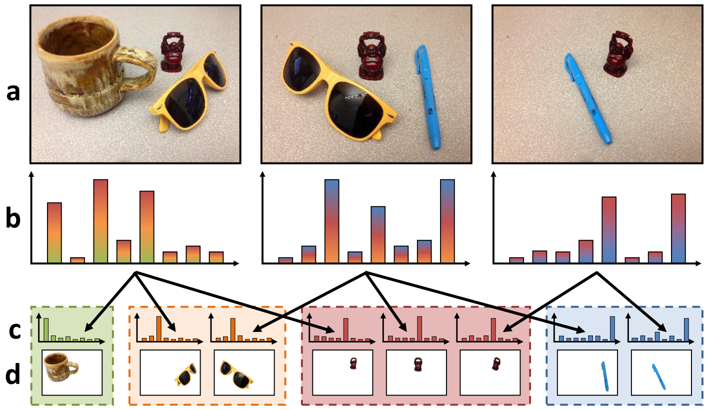


Figure 3.4: An overview of Semantic Component Analysis (SCA) applied to the task of unsupervised object discovery. (a) From a set of images containing multiple classes, (b) Bag-of-Words features are extracted pooling information from the entire image. (c) SCA decomposes these global representations into component histograms associated with meaningful component objects. The segments corresponding to these object histograms are shown in (d).

To demonstrate its ability to incorporate richer prior knowledge, we apply our novel formulation of instance-level data decomposition to the task of image segmentation. We introduce Semantic Component Analysis (SCA), a novel method for visual data decomposition that finds semantic factorizations of visual data. Figure 3.4 illustrates SCA applied to Bag-of-Words (BoW) histograms extracted from input images. Our algorithm decomposes these global image features into class-specific histograms (Figure 3.4c) constructed from partitions of semantically-related image segments (Figure 3.4d). While existing factorization methods use a global basis common to all images, the key idea of SCA is the introduction of instance-specific sets of components allowing for more complex image constraints and priors. Specifically, we enforce that object partitions be spatially-consistent. This type of coherence would not be not possible with a global basis because instances of the same class vary in appearance and location across images.

Images are often composed of a number of distinct (but semantically-related) regions. A natural aim of visual learning is to find these meaningful regions with minimal supervision. While supervised approaches to image segmentation rely on labeled training examples, acquiring this data can be time-consuming and error-prone. Instead, we consider the process of image formation as mixing a number of semantic components: one for each class present within an image. While the same local image features may appear in instances from different classes, the *distributions* of features within semantic regions are often distinct across classes. If these global image features could be unmixed into their semantic components–each representing consistent segmentations belonging only to a single class–then recognition tasks could be simplified dramatically.

3.2.1 Semantic Constraints for Segmentation

Ideally, we seek a semantically-interpretable technique for CA that represents each class as a single component. In order to encourage that this be the case in the absence of pixel-level annotations, we must rely on priors and constraints that summarize assumptions about how classes are represented in images. Specifically, we note that images tend to be separated into spatially-consistent partitions of object classes. However, because of intra-class variability and differing spatial layouts across images, these constraints would be inconsistent and impossible to enforce in traditional matrix factorization approaches.

Instead, we propose an exact data decomposition of each image feature $\boldsymbol{x}_i$ into it's own distinct set of instance components $\mathbf{H}_i$ (with columns $\boldsymbol{h}_{ij}$) in lieu of a shared basis:

$$\boldsymbol{x}_i = \mathbf{H}_i \boldsymbol{w}_i = \sum_{j=1}^{m} w_{ij} \boldsymbol{h}_{ij} \quad \forall i = 1, \ldots, n. \tag{3.10}$$

Here, n represents the size of the dataset and m represents the total number of semantically-related groups of components (i.e. object classes) that we consider. Observe that having a separate set of components for each image–where the basis $\mathbf{H}_i$ depends on the image index i–differs from traditional CA methods which use a global basis common to every image.

In order to encourage semantic in the absence of pixel-level annotations, we must rely on priors and constraints that summarize assumptions about how classes are represented in images. Specifically, we note that images tend to be separated into spatially-consistent partitions of object classes. However, because of intra-class variability and differing spatial layouts across images, these constraints would be inconsistent and impossible to enforce in traditional matrix factorization approaches.

This formulation assumes *additive* image representations, meaning that an image's global feature vector $\boldsymbol{x}_i$ can be expressed as the sum of its segment feature vectors. Note that many shallow representations share this property, including all average-pooled local features. We represent images using simple ℓ_1-normalized Bag-of-Words histograms over dense SIFT descriptors [91] quantized to $d = 1024$ dictionary elements. we begin with an over-segmentation of each image into p_i locally-consistent superpixel feature vectors of dimensionality d. Let $\mathbf{S}_i \in \mathbb{R}^{d \times p_i}$ be a matrix with the i^{th} image's normalized superpixel features $\boldsymbol{s}_{ik}$ as its columns. Let q_{ik} represent the proportion of the image taken up by the k^{th} superpixel and denote by $\boldsymbol{q}_i$ the vector with these values as its elements. Thus, due to its additivity, $\boldsymbol{x}_i = \mathbf{S}_i \boldsymbol{q}_i$. That is, the image histogram $\boldsymbol{x}_i$ is a convex combination of its superpixel histograms $\boldsymbol{s}_{ik}$.

To account for object class occlusion in the image, we enforce that the instance components $\boldsymbol{h}_{ij}$ come from non-overlapping partitions of superpixels by defining indicator variables $z_{ijk} \in \{0, 1\}$ that are 1 if the k^{th} superpixel belongs only to the j^{th} class and 0 otherwise. Let $\boldsymbol{z}_{ij}$ be the column vector formed by stacking the z_{ijk} for all k. Then, the weighted component histograms can be written as $w_{ij}\boldsymbol{h}_{ij} = \mathbf{S}_i \text{diag}(\boldsymbol{q}_i)\boldsymbol{z}_{ij}$, where w_{ij} represents the proportion of the i^{th} image belonging to the j^{th} class. This also constrains the component by $w_{ij} = \boldsymbol{q}_i^\intercal \boldsymbol{z}_{ij}$ so that $0 \le w_{ij} \le 1$ and $\sum_{j=1}^{m} w_{ij} = 1$.

While the over-segmentation of images into superpixels provides some local spatial consistency, many superpixels could still make up a single object. Thus, we incorporate an additional regularization term borrowed from the spectral clustering and co-segmentation literature [68] that promotes smoothness between superpixels. Specifically, we define a similarity matrix $\mathbf{W}_i$ that assigns each pair of superpixels in an image a weight determined by their spatial proximity and color similarity. Denote by $\mathbf{L}_i$ the normalized graph Laplacian constructed from $\mathbf{W}_i$. Enforcing that the quantity $\boldsymbol{z}_{ij}^\intercal \mathbf{L}_i \boldsymbol{z}_{ij}$ be small (less than a threshold parameter ρ) encourages nearby superpixels with similar color to take on the same label. Figure 3.5 shows an example of this.

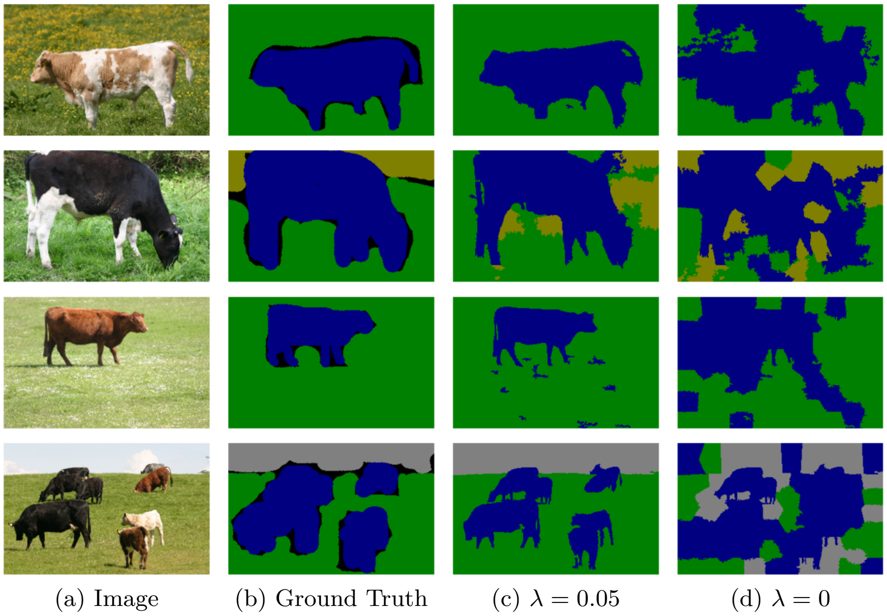


Figure 3.5: A comparison of segmentation results both with (c) and without (d) spatial consistency regularization, which encourages segmentations that better adhere to object boundaries. By taking into account local similarities within images, spurious errors can be avoided resulting in segmentations that better match the ground truth (b).

Note that this set of constraints is non-convex since we enforce z_{ijk} to be binary, which would make optimization difficult. Thus, we first relax this constraint by allowing z_{ijk} to take on values within the continuous interval $[0, 1]$. Since $\sum_{j=1}^{m} z_{ijk} = 1$, z_{ijk} can be interpreted as the degree to which the k^{th} superpixel in the i^{th} image belongs to the j^{th} class. The solution can then be rounded by selecting the class with the highest value in order to produce a discrete segmentation.

These semantic instance constraints are summarized as follows in Equation 3.11:

$$\begin{aligned}\mathcal{C}_i = \Big\{ w_{ij}, \boldsymbol{h}_{ij} \,:\, w_{ij}\boldsymbol{h}_{ij} = \mathbf{S}_i \text{diag}(\boldsymbol{q}_i)\boldsymbol{z}_{ij},\ \boldsymbol{z}_{ij}^{\mathsf{T}}\mathbf{L}_i\boldsymbol{z}_{ij} \leq \rho, \\ w_{ij} = \boldsymbol{q}_i^{\mathsf{T}}\boldsymbol{z}_{ij},\ \sum_{j=1}^{k}\boldsymbol{z}_{ij} = \mathbf{1},\ \mathbf{0} \leq \boldsymbol{z}_{ij} \leq \mathbf{1} \Big\}\end{aligned} \tag{3.11}$$

This constraint set is very general and can be easily adapted to include additional

Table 3.1: SCA segmentation results on synthetic data.

	aeroplane	cow	building	car	sheep	tree	grass	marble	stone	bark	total
	1	2	3	4	5	6	7	8	9	10	
Cluster (GT)	02	66	41	52	00	96	98	96	49	96	77
Cluster (Super)	00	29	96	32	03	39	52	29	31	28	36
SCA (None)	76	73	84	65	84	01	86	88	81	75	77
SCA (Weak)	80	81	90	74	86	53	81	83	82	88	82
SCA (Full)	77	78	85	74	78	55	86	88	88	92	85

image priors or modified to be applicable to tasks even beyond image segmentation. Even so, these simple, intuitive constraints surprisingly still result in semantically-meaningful decompositions. Furthermore, because this set is convex, it allows for convenient optimization

The original objective function from Equation 3.2 can then be updated to incorporate these constraints as shown in Equation 3.12.

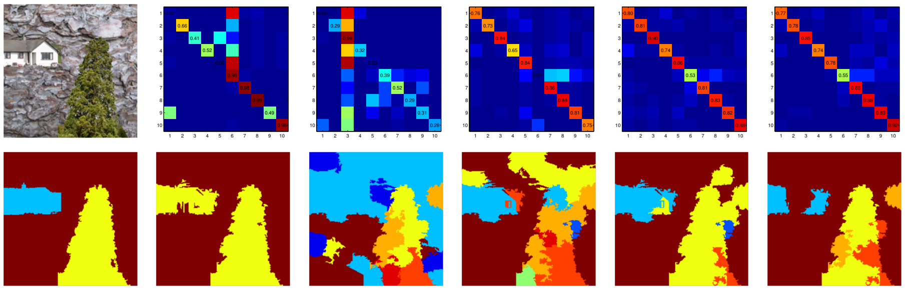
$$\begin{aligned} \underset{w_{ij},\boldsymbol{z}_{ij},\boldsymbol{b}_j}{\arg\min} \sum_{i=1}^{n}\sum_{j=1}^{k} \|\mathbf{S}_i \mathrm{diag}(\boldsymbol{q}_i)\boldsymbol{z}_{ij} - w_{ij}\boldsymbol{b}_j\|_2^2 + \lambda \boldsymbol{z}_{ij}^{\mathsf{T}} \mathbf{L}_i \boldsymbol{z}_{ij} \\ \text{s.t. } w_{ij} = \boldsymbol{q}_i^{\mathsf{T}} \boldsymbol{z}_{ij},\ \sum_{j=1}^{k} \boldsymbol{z}_{ij} = \mathbf{1},\ \mathbf{0} \leq \boldsymbol{z}_{ij} \leq \mathbf{1} \end{aligned} \tag{3.12}$$

Note that the formulation described thus far does not require any training labels, various levels of supervision can be easily incorporated by simply fixing certain known elements during training. In particular, weak supervision can be included by forcing the coefficients w_{ij} for all absent classes to be zero, which effectively requires summing only over those classes present in an image.

3.2.2 Experimental Results

To demonstrate the effectiveness of our method, we evaluate it against a number of datasets with varying levels of superivsion.

First, we consider synthetic data with minimal controlled intra-class variation. Specifically, we use 500 training images and 200 testing images generated by first



Ground Truth Cluster (GT) Cluster SCA (None) SCA (Weak) SCA (Full)

Figure 3.6: Top: Confusion matrices for the accuracies in Table 3.1. Bottom: Example segmentations for the different methods. Increasing levels of supervision improve segmentation consistency.

selecting one of three backgrounds from the Salzburg Texture Image Database [75] and then randomly placing up to 7 rescaled objects segmented from the MSRC2 dataset [130], for a total of 10 classes. There is a maximum of 50% overlap with other objects and the image edges (simulating occlusion), and there are 2.9 classes per image on average.

Table 3.1 shows the segmentation performance of our algorithm with varying levels of supervision using a BoW dictionary size of 1024 with smoothness regularization parameter $\lambda = 0.05$ and using a robust trimmed average with $P = 20\%$. In the unsupervised setting, clusters were permuted and assigned to class labels in order to maximize average training accuracy. As unsupervised baselines, we also compare k-medians clustering of both ground-truth segments and independent superpixels. Even though our method is based on superpixels, its performance is very close to the clustering of ground-truth segments, even performing better on smaller classes. This is likely due to the joint assignment of all classes within an image according to the image formation constraints in $\mathcal{C}_i$. Simply clustering superpixels results in very poor performance because small regions do not contain enough class-specific features.

While increasing the level of supervision improved accuracy somewhat (especially for "tree", which is visually similar to the background classes such as "grass"), our algorithm was generally able to cluster the image regions into the

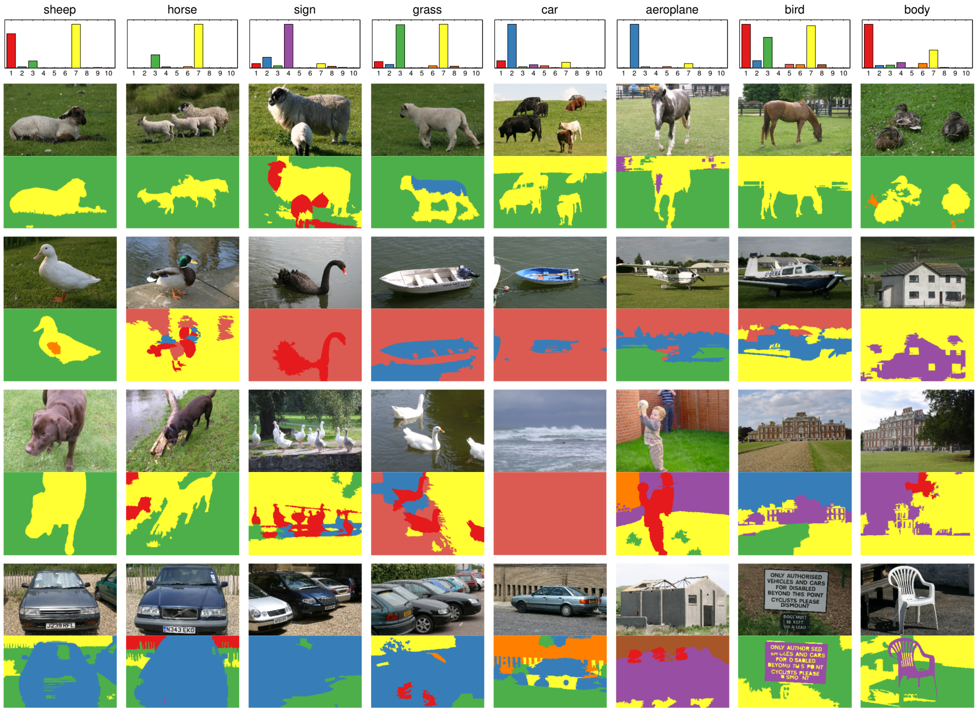


Figure 3.7: Example unsupervised segmentation results on the MSRC2 dataset. The bar plots on top show the proportion of pixels associated with a given ground truth class that were assigned to each of the 10 unsupervised clusters. Below are example images and the resulting segmentations achieved by our algorithm, showing clear separation into semantically-meaningful groups.

correct semantic classes even with minimal training. Class confusion matrices and example segmentations are shown in Figure 3.6.

We also evaluated our algorithm on the MSRC2 dataset [130], which contains 591 images segmented into 21 ground truth classes. We first applied our method in the unsupervised setting with $m = 10$ latent classes. Smoothness regularization was used with $\lambda = 2$ along and exemplar components were computed using the median, i.e. with $P = 50\%$. Example qualitative results are shown in Figure 3.7. Note that the resulting groups are semantically related and generally give a good separation between classes. For example, nearly all "aeroplane" pixels were assigned to cluster 2, which also included pixels associated with other man-made objects such as "car" and "boat".

Table 3.2: SCA segmentation results on the MSRC2 dataset.

	[147]	[19]	[89]	**SCA (None)**	**SCA (Weak)**	**SCA (Full)**
Total Acc.	67	69	71	60	70	77

Table 3.3: SCA segmentation results on the Sift Flow dataset.

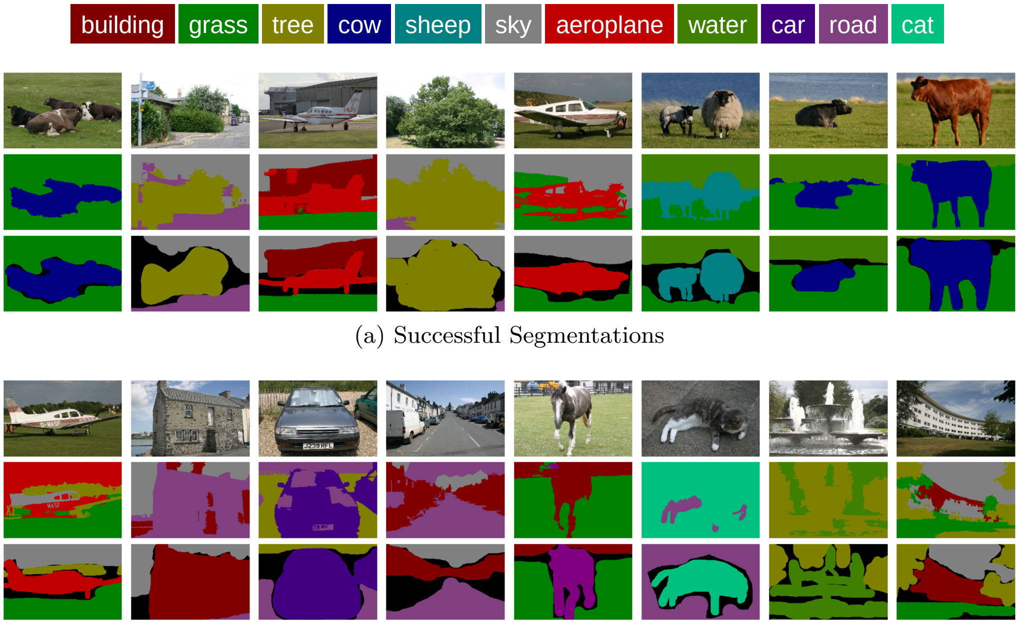
	[147]	[148]	[154]	**SCA (None)**	**SCA (Weak)**	**SCA (Full)**
Avg. Acc.	14	21	28	14	19	25

We tested our algorithm with weak labels provided at both training and testing time. To provide context, we also show results of our method when training without any labels and with full pixel-level annotations. We used the standard method for separating the training and testing data [130]. Table 3.2 summarizes our results in terms of total pixel accuracy in comparison to other methods. Despite the simplicity of our algorithm, we achieve comparable performance to many state-of-the-art systems specifically engineered for the task. Figure 3.8 shows some example successful and unsuccessful segmentations.

Finally, we evaluated performance on the challenging Sift Flow dataset [87], which contains 2688 total images (200 of which are used only for testing) and 33 classes, with an average of 4.43 classes per image. Following [154], we predict weak labels of testing images using linear SVMs trained on 4096-dimensional features extracted from the last fully-convolutional layer (fc6) in the pre-trained Caffe CNN [66]. Table 3.3 shows average class accuracy in comparison to other methods. Results from unsupervised and fully-supervised training are also shown for comparison. We again achieve comparable performance to other methods that are designed specifically for weakly-supervised semantic segmentation and use much richer feature sets (color, GIST, and superpixel locations) and priors (e.g. objectness, ILP, and discriminative appearance models.)

3.2.3 Conclusion

We outlined a general framework for explicitly introducing interpretability to component analysis. This was accomplished through an alternative objective function (rather than the traditional least squares reconstruction error from ma-



(a) Successful Segmentations

(b) Failure Cases

Figure 3.8: Example weakly-supervised segmentations from the MSRC2 dataset showing both (a) successful and (b) unsuccessful cases. Typical failure cases occur because of confusion between visually similar classes that commonly co-occur (e.g. sky and road) or when different classes have very similar color (e.g. the gray cat on the road.)

trix factorization) that exposes instance components which can be constrained using prior information. Specifically, we formalized an intuitive observation: images tend to be partitioned into spatially-consistent, non-overlapping regions that belong only to a single class. Despite their simplicity, these constraints allow for the semantically-meaningful clustering of image regions. Requiring only BoW features and superpixel color similarities, our algorithm is easily-implementable, efficient, and robust to initialization. Furthermore, varying levels of supervision can be incorporated trivially.

Even without manual engineering, fine-tuning, or over-fitting to a particular dataset, we achieve competitive performance on standard weakly-supervised semantic segmentation tasks. Our approach is general, allowing for the simple inclusion of additional constraints and priors with the potential to improve these results even further. SCA could also be easily adapted to numerous other applications beyond semantic segmentation, including time series analysis, background modeling in videos, etc.

3.3 Additive Component Analysis (ACA)

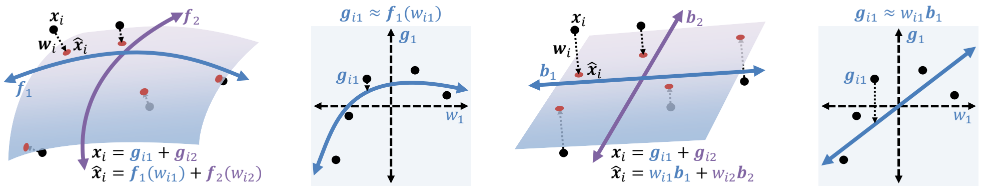
Despite the complexity of factors involved in determining the organization of pixels in an image, one prominent hypothesis suggests that images concentrate near manifolds with low intrinsic dimensionality determined by their underlying degrees of freedom [105]. One of its earliest successes was found in the area of face recognition through the Eigenfaces algorithm [144], a holistic data-driven approach for representing data as linear combinations of learned components.

In addition to enabling richer constraints that better encode prior knowledge in applications like image segmentation, instance-level data decomposition can also be adapted to address the basic modeling restriction common to many component analysis techniques: linearity. We propose Additive Component Analysis (ACA), a novel method for nonlinear component analysis. The motivating hypothesis underlying our approach is that reconstructed input data should vary smoothly with respect to lower-dimensional representations, relaxing the strict linearity assumption of PCA. Our approach can be interpreted as an unsupervised additive model [20] constructed to predict training data from latent input variables, effectively fitting a smooth manifold to data with complexity controlled by an intuitive roughness penalty. An overview is shown in Figure 3.9, along with comparisons to PCA.

We generalize Equation 1.2 by instead approximating data as the sum of learned nonlinear basis functions $\boldsymbol{f}_j$ evaluated at some latent variables w_{ij}, resulting in approximations given by the additive model $\boldsymbol{x}_i \approx \boldsymbol{f}(\boldsymbol{w}_i) = \sum_j \boldsymbol{f}_j(w_{ij})$. The resulting optimization problem is shown in Equation 3.13.

$$\underset{w_{ij}, \boldsymbol{f}_j}{\arg\min} \sum_{i=1}^{n} \left\| \boldsymbol{x}_i - \sum_{j=1}^{k} \boldsymbol{f}_j(w_{ij}) \right\|_2^2 + \lambda \left\| \boldsymbol{w}_i \right\|_2^2 \quad \text{s.t. } \boldsymbol{f}_j \in \mathcal{F} \tag{3.13}$$

Here, we aim to learn both the basis functions $\boldsymbol{f}_j$ and latent variables $\boldsymbol{w}_i$. We constrain the basis functions to belong to the set $\mathcal{F}$ In addition, in order to



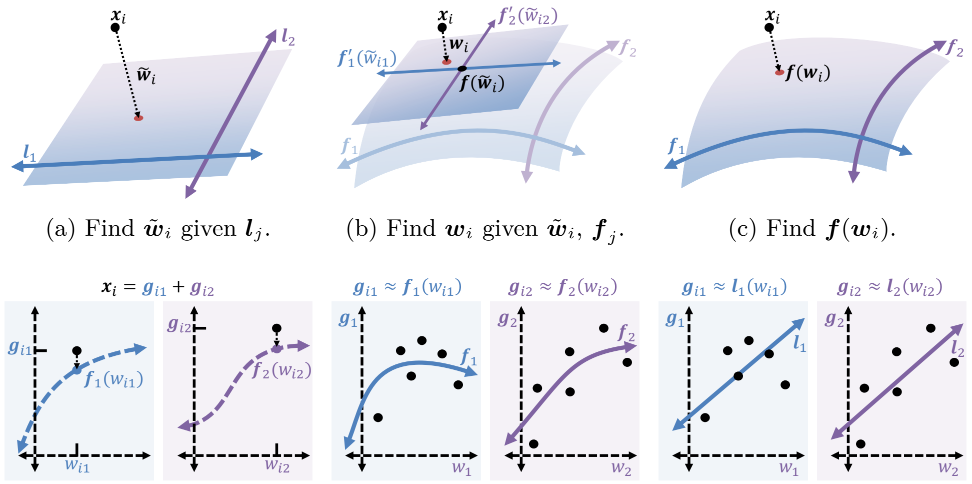
(a) Additive Component Analysis (ACA) (b) Principal Component Analysis (PCA)

Figure 3.9: An overview of ACA (a) in comparison to PCA (b) on the task of fitting a two-dimensional surface to three-dimensional data. Both methods minimize the sum of squared distances between the data and their orthogonal projections. However, ACA learns a nonlinear manifold resulting in reduced reconstruction error. The key to our approach is the decomposition of each data point $\boldsymbol{x}_i$ into a sum of target components $\boldsymbol{g}_{ij}$, which allows the basis functions to be learned through simple, univariate regression.

compress the latent space and constrain the domains of the basis functions, we enforce that the latent representations belong to a closed set $\mathcal{W}$, implemented using a small amount of ℓ_2 regularization with a fixed hyperparameter of $\lambda = 0.01$.

This objective essentially minimizes the error in approximating data by projecting them onto an m-dimensional curvilinear manifold. Example learned basis functions can be found in Figure 3.11. In addition, Figure 3.12 visualizes higher-dimensional basis functions for image data by evaluating them at different intervals of the corresponding training latent variables.

Optimization for this problem presents an interesting challenge. A common approach for similar constrained component analysis problems (e.g. non-negative matrix factorization [14], dictionary learning [72], etc.) is alternating minimization. With one set of variables fixed, the resulting problem is usually much simpler. In our case, however, this is not so. With the latent representations $\boldsymbol{w}_i$ fixed, the optimization problem reduces to that of a supervised additive model, which must be solved using an iterative backfitting algorithm, often requiring many iterations to converge [20]. To enable simpler optimization, we again introduce additional auxiliary variables by decomposing $\boldsymbol{x}_i$ into a sum of target components $\boldsymbol{g}_{ij}$, which we enforce with an affine equality constraint. Our optimization



(a) Find $\tilde{\boldsymbol{w}}_i$ given $\boldsymbol{l}_j$. (b) Find $\boldsymbol{w}_i$ given $\tilde{\boldsymbol{w}}_i$, $\boldsymbol{f}_j$. (c) Find $\boldsymbol{f}(\boldsymbol{w}_i)$.

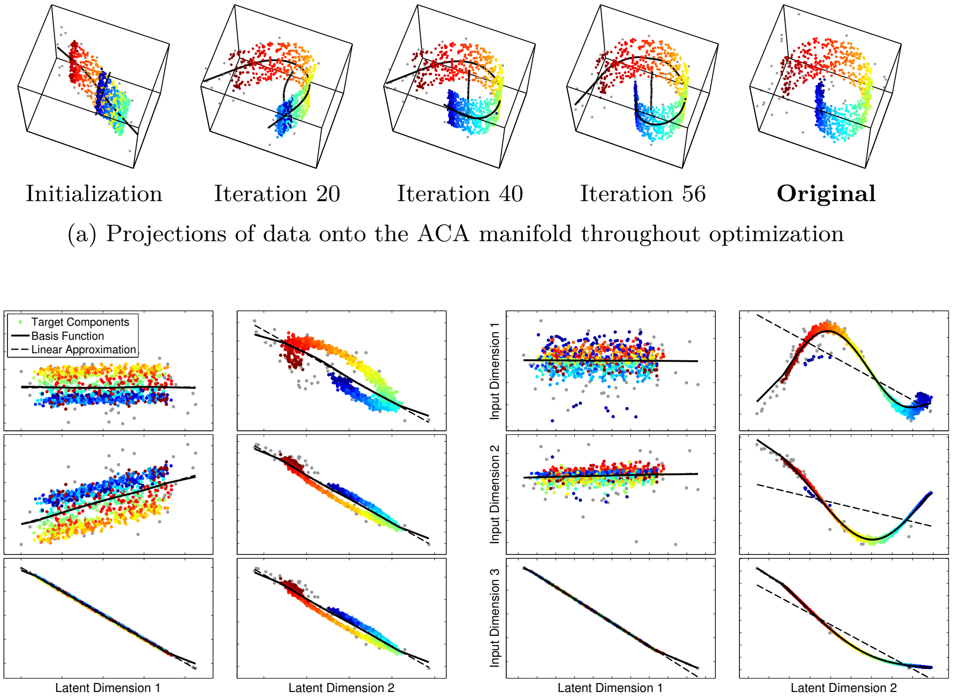
(d) Find $\boldsymbol{g}_{ij}$ given $\boldsymbol{f}(\boldsymbol{w}_i)$. (e) Find $\boldsymbol{f}_j$ given w_{ij}, $\boldsymbol{g}_{ij}$. (f) Find $\boldsymbol{l}_j$ given w_{ij}, $\boldsymbol{g}_{ij}$.

Figure 3.10: A visualization of one iteration of our alternating optimization procedure. (a) First, approximate latent variables $\tilde{\boldsymbol{w}}_i$ are found by projecting each data point $\boldsymbol{x}_i$ onto the affine subspace defined by the linear basis function approximations $\boldsymbol{l}_j$, initialized using PCA. (b) Then, $\boldsymbol{w}_i$ is updated by projecting $\boldsymbol{x}_i$ onto the tangent space at the point $\boldsymbol{f}(\boldsymbol{w}_i)$. This step is repeated multiple times with smaller step sizes for increased accuracy. (c) The result is an approximate orthogonal projection of $\boldsymbol{x}_i$ onto the manifold. (d) The target components $\boldsymbol{g}_{ij}$ are then found by equally redistributing the reconstruction error between them. (e,f) Finally, the basis functions $\boldsymbol{f}_j$ and their linear approximations $\boldsymbol{l}_j$ are found using simple univariate regression.

problem can then be equivalently written as follows:

$$\underset{w_{ij},\boldsymbol{g}_{ij},\boldsymbol{f}_j}{\arg\min} \sum_{i=1}^{n} \sum_{j=1}^{k} \left\| \boldsymbol{g}_{ij} - \boldsymbol{f}_j(w_{ij}) \right\|_2^2 \quad \text{s.t.} \quad \sum_{j=1}^{k} \boldsymbol{g}_{ij} = \boldsymbol{x}_i,\ \boldsymbol{f}_j \in \mathcal{F} \tag{3.14}$$

Unlike the original problem formulation in Equation 3.13, our formulation based on instance-level data decompositions again admits an efficient alternating minimization algorithm similar to the one described previously in Section 3.1. With $\boldsymbol{f}_j$ fixed, we first solve Equation 3.15, which essentially amounts to project-



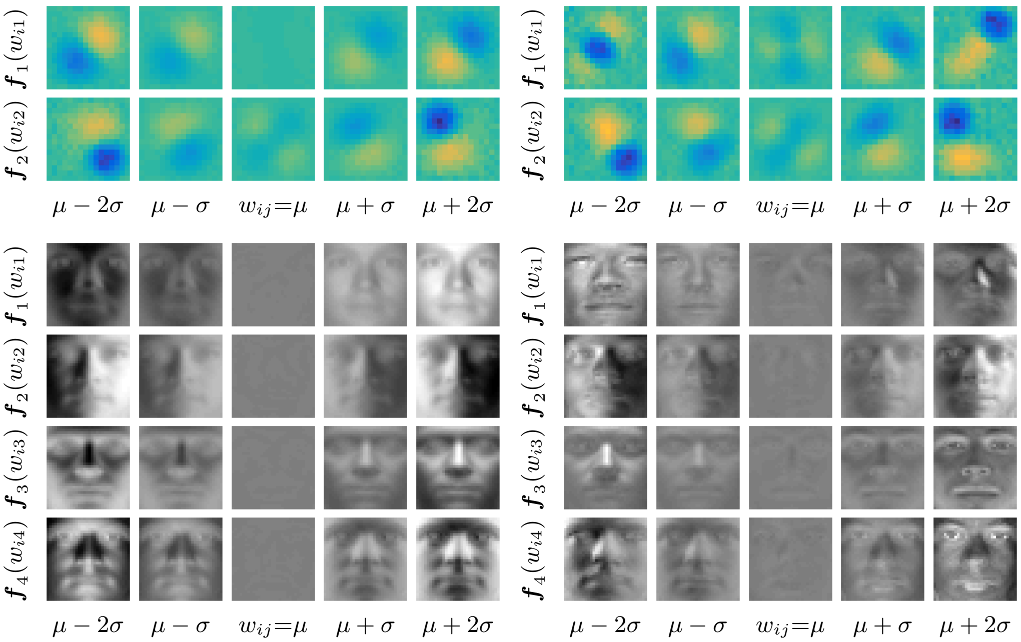
Initialization Iteration 20 Iteration 40 Iteration 56 **Original**

(a) Projections of data onto the ACA manifold throughout optimization

(b) PCA Initialization

(c) ACA Solution

Figure 3.11: An example of our optimization procedure applied to a synthetic dataset of a two dimensional surface embedded in three dimensions. Gaussian noise and uniformly random outliers were also added. (a) The original data points are shown on the right. On the left are their denoised projections using the learned basis functions (shown in black) throughout optimization. Starting from a linear subspace at initialization, the basis functions adapt to the nonlinear structure of the data, resulting in a near perfect reconstruction of the true underlying manifold. Also shown is a comparison between the basis vectors learned by (b) PCA and the nonlinear basis functions learned by (c) ACA on the synthetic dataset. The two latent dimensions vary along the horizontal axis while the three input dimensions vary along the vertical axis. The values of the target component dimensions are shown as colored points while their linear and smoothing spline approximations are shown as dotted and solid black lines respectively. Observe that ACA is able to find alternate decompositions of the data points in which the resulting target components can be well approximated by smooth functions.



(a) $\rho = 0.0001$

(b) $\rho = 0.001$

Figure 3.12: A visualization of the basis functions learned by ACA on both synthetic and real image data for a variety of roughness parameters (a-b). The value of the latent variable w_{ij} varies along the horizontal axis while the basis function index $j = 1, \ldots, m$ varies along the vertical axis.

ing $\boldsymbol{x}_i$ onto the learned manifold.

$$\underset{\boldsymbol{w}_i \in \mathcal{W}, \boldsymbol{g}_{ij}}{\arg\min} \sum_{j=1}^{k} \left\| \boldsymbol{g}_{ij} - \boldsymbol{u}_{ij} - w_{ij} \boldsymbol{f}'_j(\tilde{w}_{ij}) \right\|_2^2 + \lambda \left\| \boldsymbol{w}_i \right\|_2^2 \quad \text{s.t.} \quad \sum_{j=1}^{k} \boldsymbol{g}_{ij} = \boldsymbol{x}_i \tag{3.15}$$

Solving this directly is difficult due to the nonlinear basis functions $\boldsymbol{f}_j$. However, since we enforce that they be smooth with small second derivatives, they can be effectively approximated by first-order Taylor expansions centered around some approximate solutions $\tilde{w}_{ij}$ as $\boldsymbol{f}_j(w_{ij}) \approx \boldsymbol{u}_{ij} + w_{ij} \boldsymbol{f}'_j(\tilde{w}_{ij})$ where $\boldsymbol{u}_{ij} = \boldsymbol{f}_j(\tilde{w}_{ij}) - \tilde{w}_{ij} \boldsymbol{f}'_j(\tilde{w}_{ij})$. Here, the set of partial derivatives $\boldsymbol{f}'_j(\tilde{w}_{ij})$ span the tangent space of the manifold, reducing to the same linear least squares problem from Section 3.1. The initial $\tilde{w}_{ij}$ can be found by projecting $\boldsymbol{x}_i$ onto the affine subspace defined by linear approximations $\boldsymbol{l}_j$ to the spline functions $\boldsymbol{f}_j$. This approximation can be improved by repeatedly updating $\boldsymbol{w}_i$ with decreasing step sizes.

Afterward, following a derivation similar to that of Equation 3.6, the target components $\boldsymbol{g}_{ij}$ are given by the closed form solution in Equation 3.16.

$$\boldsymbol{g}_{ij} = \boldsymbol{f}_j(w_{ij}) + \tfrac{1}{k}\Big(\boldsymbol{x}_i - \sum_{j=1}^{k} \boldsymbol{f}_j(w_{ij})\Big) \tag{3.16}$$

With $\boldsymbol{w}_i$ and $\boldsymbol{g}_{ij}$ fixed, we update the basis functions $\boldsymbol{f}_j$ by solving Equation 3.17.

$$\underset{\boldsymbol{f}_j}{\arg\min} \sum_{i=1}^{n} \left\|\boldsymbol{g}_{ij} - \boldsymbol{f}_j(w_{ij})\right\|_2^2 \quad \text{s.t. } \boldsymbol{f}_j \in \mathcal{F} \tag{3.17}$$

This is a standard univariate regression problem mapping the latent variables w_{ij} to the target components $\boldsymbol{g}_{ij}$.

3.3.1 Curvilinear Smoothness Constraints

We restrict the basis functions $\boldsymbol{f}_j$ to be roughness-penalized smoothing splines [32] due to their generality and efficient computation. Thus, they must belong to the set $\mathcal{F}$, which is defined as:

$$\mathcal{F} = \left\{\boldsymbol{f} : \mathbb{R} \to \mathbb{R}^d \ : \ \int (f_p''(x))^2 dx \leq \gamma,\ \forall p = 1, \ldots, d\right\} \tag{3.18}$$

These constraints are implemented with a roughness penalty that balances approximation accuracy and complexity with a roughness hyperparameter ρ. The solution to the problem in Equation 3.17 is a cubic spline with knots τ_i corresponding to each of the training points $\boldsymbol{x}_i$, which can be expressed as a linear combination of spline basis functions $\boldsymbol{f}_j(x) = \sum_{t=1}^{n_b} \boldsymbol{c}_{tj} b_t(x)$ with coefficient vectors $\boldsymbol{c}_{tj} \in \mathbb{R}^d$ [32]. In our implementation, we use B-spline basis functions because they have bounded support resulting in sparse, banded matrices and linear-time inverse computations [33]. Furthermore, their evaluation and derivatives can be efficiently computed using a simple recursive formula [151].

The constraint set in Equation 3.18 can be equivalently expressed as:

$$\mathcal{F} = \left\{\boldsymbol{f}(x) = \sum_{t=1}^{n_b} \boldsymbol{c}_t b_t(x) \ : \ \int \sum_{s=1}^{n_b} \boldsymbol{c}_s^\mathsf{T} \sum_{t=1}^{n_b} \boldsymbol{c}_t b_s''(x) b_t''(x) dx \leq \gamma\right\} \tag{3.19}$$

where $\mathbf{B}_j(i,t) = b_{tj}(w_{ij})$ and $\mathbf{\Omega}_j(s,t) = \int b''_{sj}(x)b''_{tj}(x)dx$.

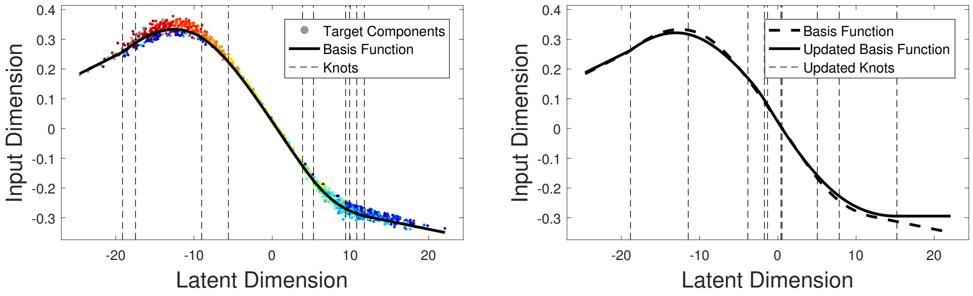
Thus, the infinite-dimensional problem in Equation 3.17 can be reduced to the simple, regularized linear least-squares problem in Equation 3.20 from which the optimal spline coefficients $\mathbf{C}_j$ may be found in closed-form, where $\mathbf{G}_j = \left[\boldsymbol{g}_{1j}, \ldots, \boldsymbol{g}_{nj}\right]^{\mathsf{T}}$ and $\mathbf{C}_j = [\boldsymbol{c}_{1j}, \ldots, \boldsymbol{c}_{n_b j}]$.

$$\underset{\mathbf{C}_j}{\arg\min} \, \|\mathbf{G}_j - \mathbf{B}_j\mathbf{C}_j\|_2^2 + \rho\mathbf{C}_j^{\mathsf{T}}\mathbf{\Omega}_j\mathbf{C}_j \tag{3.20}$$

3.3.2 Approximate Stochastic Optimization

While the optimization procedure described in the previous section is memory efficient due to the separability of the cost function into smaller subproblems, solving for all latent variables at each iteration can be computationally expensive (and possibly redundant) for extremely large datasets. Ideally, we would instead prefer to take a stochastic approach that considers only a random subset of the data at each iteration. The basis functions could then be updated with a certain step size by taking a weighted average with the parameters from the previous iteration. However, since the knot locations of the spline functions change at each iteration, their corresponding parameters are not comparable.

To overcome this issue, we propose an approach that approximates the spline functions from the previous iteration using the knots from the current iteration so that their parameters can be averaged. Specifically, we use Schoenberg's variation diminishing spline approximation [95, 92], a simple and efficient method for function approximation that does not require solving a linear system of equations as with the roughness-penalized spline approximation. To understand this method, first recall that spline functions can be interpreted geometrically as smoothed versions of their control polygons, which are piecewise-linear functions with vertices located at specific control points. For a cubic spline $f(w) = \sum_t c_t b_t(w)$ with a knot vector $\boldsymbol{\tau}$, these control points have coordinates (τ_t^*, c_t) where $\tau_t^* = \frac{1}{3}(\tau_{t+1} + \tau_{t+2} + \tau_{t+3})$ are the knot averages of $\boldsymbol{\tau}$. Similarly, for any function f, it's variation diminishing cubic spline approximation is given by $(Vf)(w) = \sum_t f(\tau_t^*)b_t(w)$ where the coefficients are given directly as function evaluations at the knot averages. Thus, before updating the basis function



(a) Original Basis Function

(b) Updated Approximation

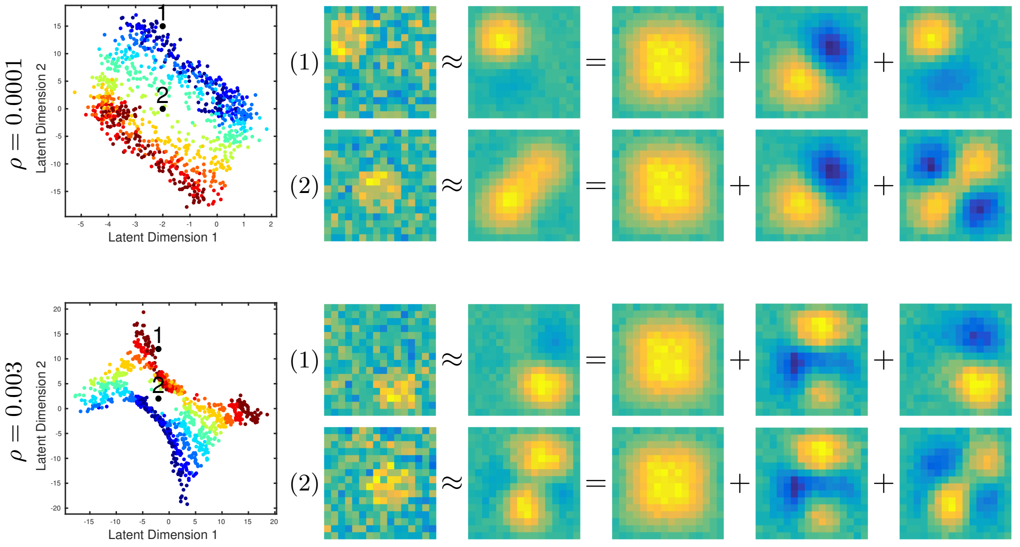
Figure 3.13: An example of the variation diminishing spline approximation used in our stochastic optimization technique for parameter averaging of basis functions with different knot locations. The original basis function (a) was fitted to target components with knot locations denoted by dotted vertical lines while the updated basis function (b) was approximated with different knot locations without requiring expensive least-squares fitting.

parameters from the previous iteration, we take a variation diminishing spline approximation of their control polygons evaluated at the new knot averages from the current iteration, essentially resulting in a linear interpolation of the control points. While this is only a rough approximation as shown in Figure 3.13, it leads to effective learning with significantly reduced training time.

3.3.3 Deep Composition of Additive Models

Despite their generality, additive models can only represent a relatively small set of possible multivariate functions due to the curvilinear assumption. Thus, the space of manifolds that can be learned with ACA is also limited. Consider, for example, a dataset of noisy images containing translated circles. Its intrinsic dimensionality equals two because there are only two independent dimensions of variation: horizontal and vertical location. However, the underlying nonlinearities cannot be effectively modeled with ACA, resulting in poor latent separability and reconstruction performance. This is demonstrated in Figure 3.14.

This fundamental limitation of component analysis is a result of the restricted additive interactions allowed between latent variables. To address this, we pro-

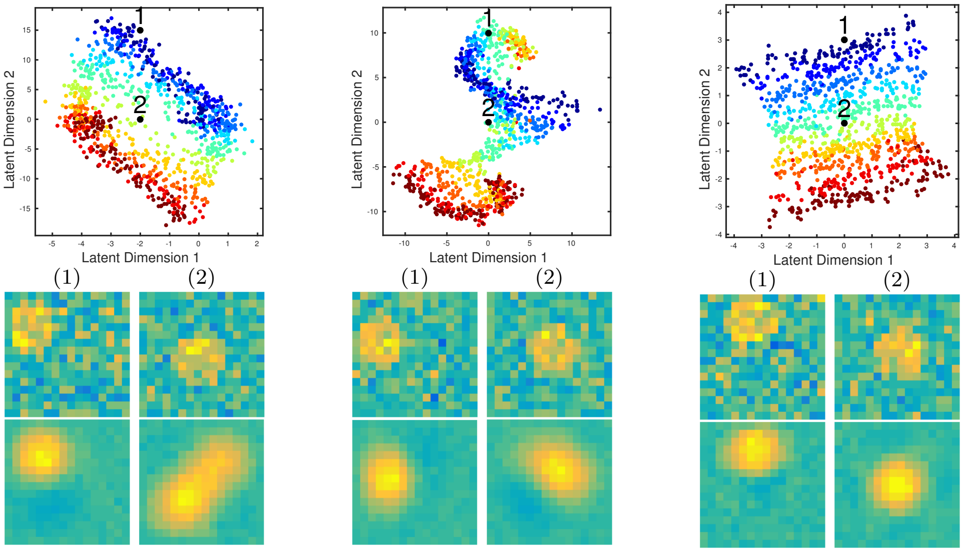


(a) $\boldsymbol{w}_i$ (b) $\boldsymbol{x}_i$ (c) $\hat{\boldsymbol{x}}_i$ (d) $\boldsymbol{m}$ (e) $\boldsymbol{f}_1$ (f) $\boldsymbol{f}_2$

Figure 3.14: A synthetic example demonstrating the difficulty in modeling nonlinearities due to translation with additive components. For two different values of the roughness parameter ρ (top and bottom), the latent space of two-dimensional representations $\boldsymbol{w}_i$ is shown (a) indicating two example images (b). ACA learns an approximate reconstruction $\hat{\boldsymbol{x}}_i$ (c) equal to the sum of a mean image m (d) and two component images (e,f) given as the evaluated basis functions $\boldsymbol{f}_j(w_{ij})$. Because the components are additive, reconstruction performance suffers even when the basis function complexity is increased with a higher roughness parameter ρ.

pose a deep extension of our approach that stacks multiple ACA layers together, increasing representational power by composing ℓ additive models $\boldsymbol{f}^k$ constructed as the sum of basis functions $\boldsymbol{f}_j^k$ for $j = 1, \ldots, m_k$ where $m_{k-1} < m_k < d$ so that $\boldsymbol{f} = \boldsymbol{f}^\ell \circ \boldsymbol{f}^{\ell-1} \circ \cdots \circ \boldsymbol{f}^1$. Similar approaches have seen much success within the area of deep learning, partially due to the observation that increasing depth can allow for comparable expressivity with exponentially fewer parameters [12]. Indeed, in Figure 3.15, we show that deep ACA successfully models translation, resulting in reduced reconstruction error and an interpretable two-dimensional representation in which latent variables correspond to different spatial dimensions.

While function composition makes optimization more difficult, we use the an



(a) One Layer ($m = 2$)

(b) Two Layers (Greedy) ($\boldsymbol{m} = [2, 4]$)

(c) Two Layers (Joint) ($\boldsymbol{m} = [2, 4]$)

Figure 3.15: A demonstration of the increased representational power provided by the composition of additive models, comparing ACA with one layer (a), with two layers trained greedily layer by layer (b), and with two layers trained jointly (c). The learned low-dimensional latent space (top) indicates two numbered example points. The corresponding original images (middle) are compared alongside the denoised reconstructions (bottom). Deep ACA results in better performance and more interpretable representations by jointly learning richer interactions between latent variables.

approach similar to the Method of Auxiliary Coordinates (MAC) [25] to learn the parameters of all layers jointly using essentially the same procedure described in Figure 3.10. This leads to a more interpretable latent space in comparison to a greedy approach that learns the parameters of each layer independently, as shown in Figure 3.15.

We now aim to infer a set of latent variables $\boldsymbol{w}_i^k \in \mathbb{R}^{m_k}$ for $k = 1, \ldots, \ell$, where $\boldsymbol{w}_i^1$ is our low-dimensional representation and the others are constrained to be intermediate layer outputs, i.e. $\boldsymbol{w}_i^{k+1} = \boldsymbol{f}^k(\boldsymbol{w}_i^k)$ for $k = 1, \ldots, \ell - 1$. For simpler notation, we fix $\boldsymbol{w}_i^{\ell+1} = \boldsymbol{x}_i$ and denote $\boldsymbol{f}^{k\uparrow} = \boldsymbol{f}^\ell \circ \boldsymbol{f}^{\ell-1} \circ \cdots \circ \boldsymbol{f}^k$, giving the optimization problem in Equation 3.21. Our optimization procedure can then proceed as usual.

$$\mathop{\arg\min}_{\substack{\boldsymbol{f}_j^k \in \mathcal{F},\\ \boldsymbol{w}_i \in \mathcal{W}}} \sum_{i=1}^{n} \sum_{k=1}^{\ell} \left\| \boldsymbol{x}_i - \boldsymbol{f}^{k\uparrow}(\boldsymbol{w}_i^k) \right\|_2^2 \text{ s.t. } \boldsymbol{w}_i^{k+1} = \boldsymbol{f}^k(\boldsymbol{w}_i^k) \tag{3.21}$$

To enable effective learning of the intermediate layers, we ignore the equality constraint when solving for the latent variables so that each $\boldsymbol{f}^{k\uparrow}(\boldsymbol{w}_i^k)$ optimally reconstructs $\boldsymbol{x}_i$. (Note that this bears some similarity to deeply-supervised deep neural networks, in which intermediate loss functions encourage the discriminability of hidden layers [79].) In other words, deep ACA can be interpreted as learning a sequence of manifolds with decreasing dimensionality so that $\boldsymbol{w}_i^k$ can be found by orthogonally projecting $\boldsymbol{x}_i$ onto the m_k-dimensional manifold defined by $\boldsymbol{f}^{k\uparrow}$.

As before, we first approximate the latent variables by fixing the basis functions and iteratively projecting $\boldsymbol{x}_i$ onto the resulting manfiold's tangent space, which is constructed as the first-order Taylor expansion of $\boldsymbol{f}^{k\uparrow}(\boldsymbol{w}_i^k)$ around $\tilde{\boldsymbol{w}}_i^k$:

$$\begin{gathered} \mathbf{D}_i^k = [\boldsymbol{f}_1^{k\prime}(\tilde{w}_{ij}^k), \ldots, \boldsymbol{f}_{m_k}^{k\prime}(\tilde{w}_{ij}^k)],\, \mathbf{D}_i^{k\uparrow} = \mathbf{D}_i^\ell \mathbf{D}_i^{\ell-1} \cdots \mathbf{D}_i^k \\ \boldsymbol{f}^{k\uparrow}(\boldsymbol{w}_i^k) \approx \mathbf{D}_i^{k\uparrow} \boldsymbol{w}_i^k + \boldsymbol{u}_i^k,\, \boldsymbol{u}_i^k = \boldsymbol{f}^{k\uparrow}(\tilde{\boldsymbol{w}}_i^k) - \mathbf{D}_i^{k\uparrow} \tilde{\boldsymbol{w}}_i^k \end{gathered} \tag{3.22}$$

Analogous to Equation 3.15, the result can be solved in closed-form.

We again decompose each set of latent variables into target components so that $\boldsymbol{w}_{ij}^{k+1} = \sum_{j=1}^{m_k} \boldsymbol{g}_{ij}^k$. After reintroducing the equality constraint, they can then be given as:

$$\boldsymbol{g}_{ij}^k = \boldsymbol{f}_j(w_{ij}^k) + \frac{1}{m_k}\Big(\boldsymbol{w}_i^{k+1} - \sum_{j=1}^{m_k} \boldsymbol{f}_j(w_{ij}^k)\Big) \tag{3.23}$$

Finally, we can again fit the basis functions $\boldsymbol{f}^k(w_{ij}^k)$ to the target components $\boldsymbol{g}_{ij}^{k+1}$ using standard regression.

3.3.4 Experimental Results

In this section, we evaluate the effectiveness of our method through qualitative and quantitative analyses on a variety of synthetic and real datasets. This is intended to demonstrate the wide applicability of ACA and to encourage its use as a simple alternative to PCA. Specifically, we demonstrate robustness to noise,

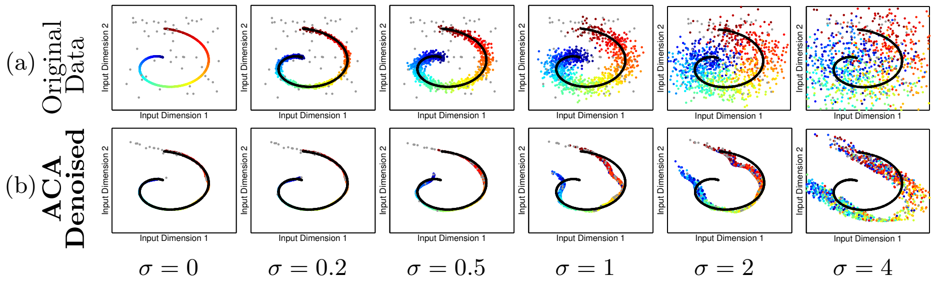


Figure 3.16: A visualization of ACA applied to synthetic data with extreme amounts of noise. In each column, different amounts of Gaussian noise are added to points along the manifold from Figure 3.11 (shown in color) along with uniformly random outliers (shown in grey). Top views of the original noisy data (a) are compared to the denoised reconstructions achieved by ACA (b). Even with large amounts of noise, ACA recovers the underlying structure of the data very well resulting in consistent low-dimensional representations.

improved denoising and reconstruction performance, and more interpretable representations with better separation of semantic categories, including large-scale experiments on the MNIST dataset.

Because ACA explicitly optimizes reconstruction accuracy, it is naturally very robust to noise unlike most approaches to manifold learning that require estimation of a neighborhood graph using pairwise distances. To demonstrate this, we constructed a synthetic dataset consisting of 1000 points sampled along a curved two-dimensional manifold embedded in three-dimensions. We then added various amounts of Gaussian noise along with 50 uniformly random outliers. A visualization of this data can be seen in Figure 3.16 and qualitative comparisons are shown in Figure 3.17 with a variety of nonlinear dimensionality reduction techniques. ACA consistently results in superior low-dimensional representations even in the presence of extreme noise. Importantly, unlike the other compared nonlinear methods, ACA trivially supports reconstruction of the underlying manifold for visualization of denoised data.

In image data, “noise” can take a variety of forms, including sensor noise, cast shadows, misalignment, occlusions, etc. Due to its ability to model complex nonlinear structure, ACA results in perceptually more accurate image reconstructions

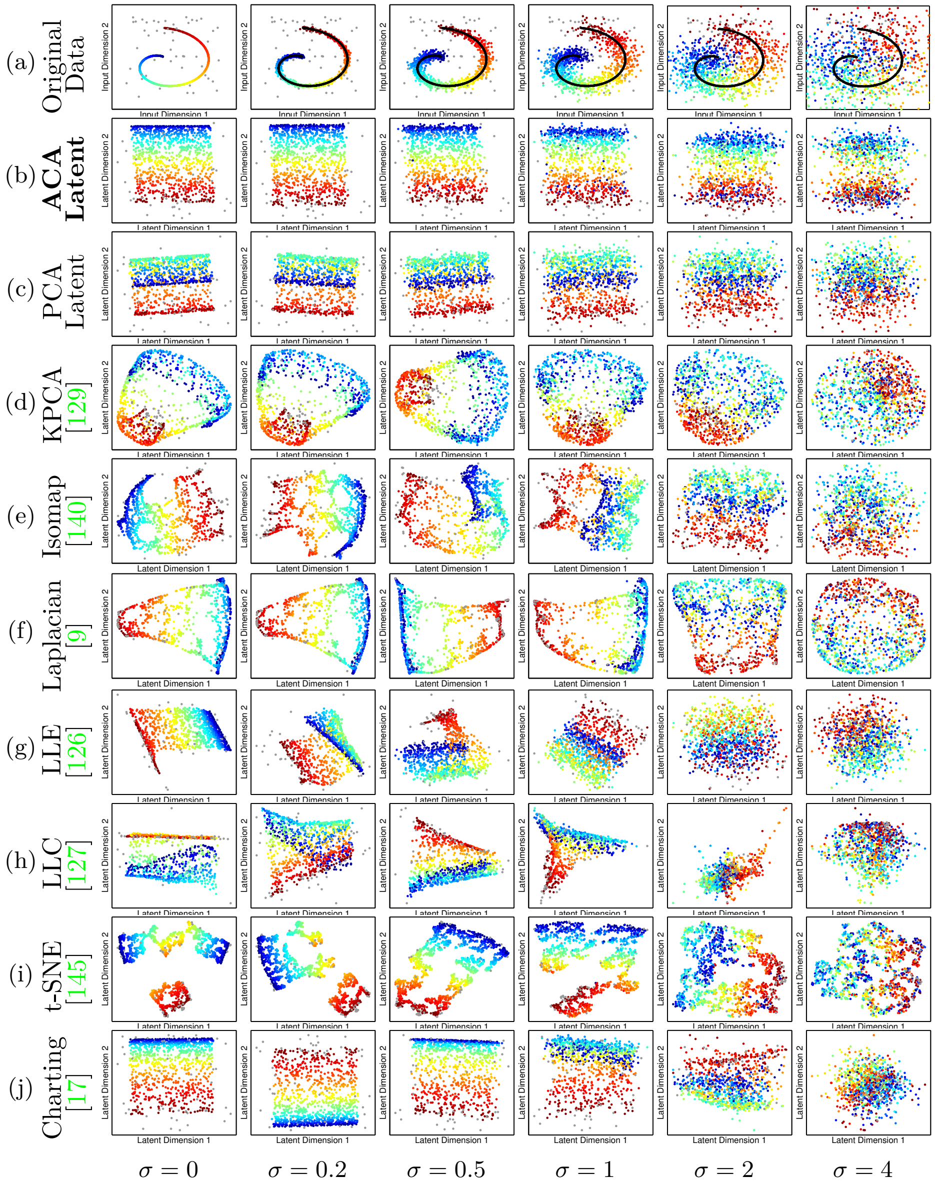


Figure 3.17: Qualitative comparisons showing our method's superior robustness to noise. (a) Different amounts of noise are added to points corresponding to the columns of Figure 3.16. The corresponding low-dimensional latent spaces of (b) ACA are compared to those of (c) PCA and a variety of other nonlinear dimensionality reduction techniques (c-j).

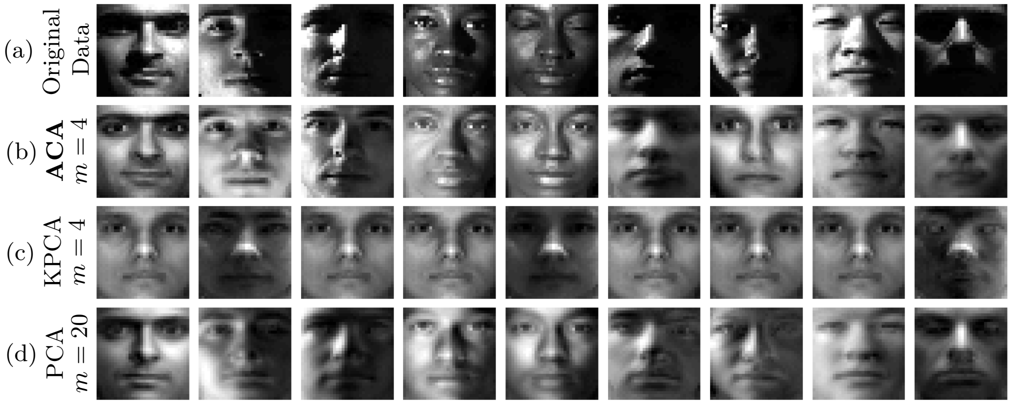


Figure 3.18: A demonstration of the invariance and complex denoising capabilities of ACA. Given images of faces under a variety of different lighting conditions (a), dimensionality reduction was performed using ACA (b) and KPCA (c) with 4 components and PCA (d) with 20 components. Because it is able to learn rich nonlinearities, ACA achieves more perceptually plausible denoised images that retain more detail with fewer components.

that are invariant to many of these sources. This is demonstrated in Figure 3.18 on the Extended Yale Face Database B [82], which contains partially aligned images of faces under different lighting conditions. Dimensionality reduction was performed on ZCA whitened images using ACA and PCA with 4 and 20 components respectively, giving similar average mean-squared reconstruction error. Example components are visualized in Figure 3.12. Also compared was Kernel PCA with 4 components and a Gaussian kernel with parameter $\sigma^2 = 2$. Since KPCA does not directly enable back-projection, approximate pre-images were found using fixed-point iterations [97]. The resulting de-whitened image reconstructions for ACA are perceptually more plausible, resulting in better shadow removal while preserving more details for improved identity preservation.

In addition to enabling accurate data reconstruction, ACA can also encode complex invariances due to the underlying smoothness constraints. This results in low-dimensional representations that are useful for high-level tasks such as exploratory data analysis and clustering. This is demonstrated in Figure 3.19, which shows how the parameter ρ affects class separability and data reconstruction

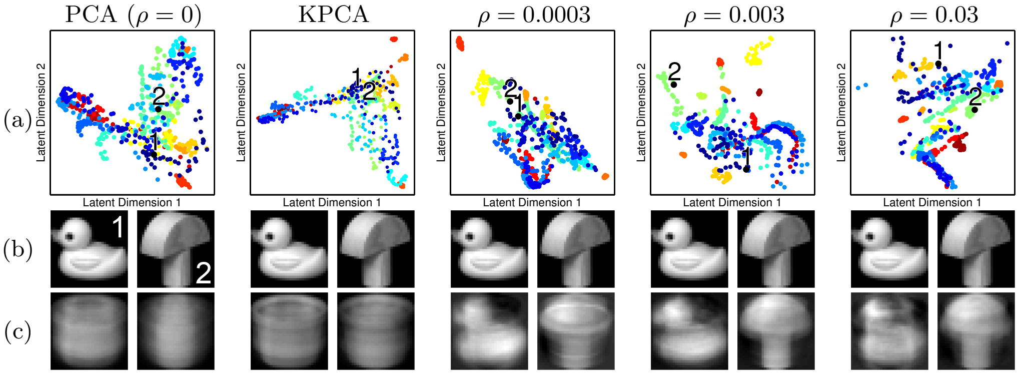


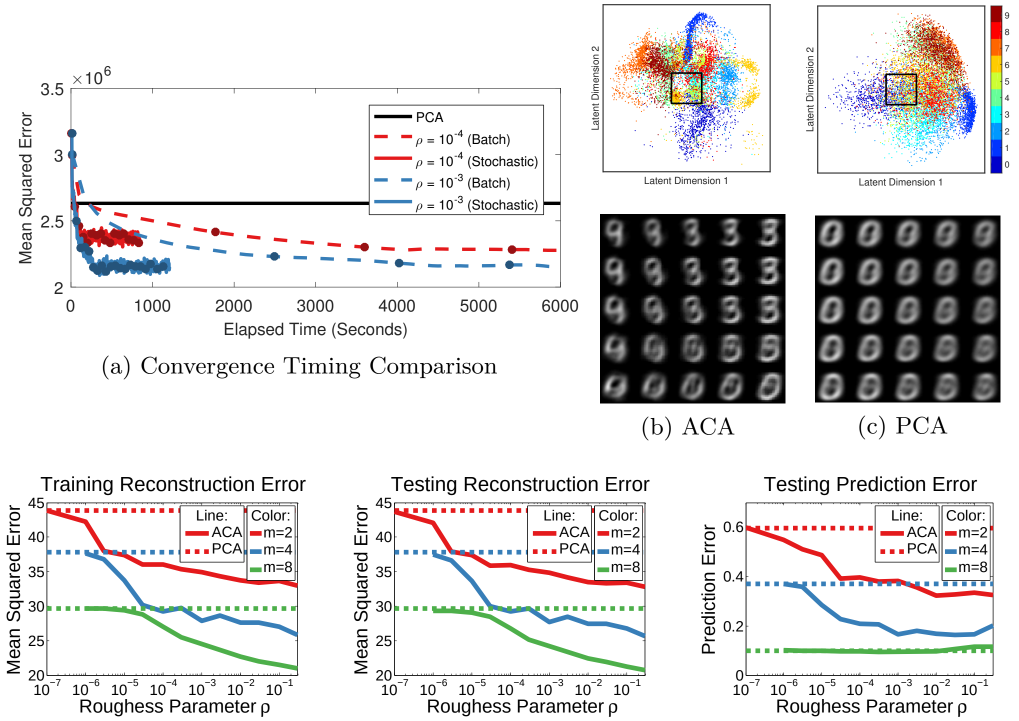
Figure 3.19: A visualization of how the roughness parameter ρ affects the performance of ACA on novel testing data. Two-dimensional latent embeddings (a) of images from the COIL-20 dataset are shown in columns for PCA, KPCA, and ACA with different values of ρ. Two example images (b) are also shown, along with their reconstructions (c). Increasing ρ can improve image reconstruction performance and the separability of object classes (shown as different colors in the latent space), but can also reduce performance due to overfitting.

performance on unseen testing data. In this experiment, approximately half of the 1440 processed images from the COIL-20 dataset [107] were used as training data for a two layer deep ACA model with $\boldsymbol{m} = [2, 4]$ and a varying roughness parameter. The learned models were then applied to the remaining testing images and the corresponding two-dimensional representations plotted alongside example image reconstructions. Increasing ρ allows for rougher basis functions with higher complexity, giving better separability of categories (shown as different colors) in the latent space in comparison to PCA and KPCA with a Gaussian kernel ($\sigma^2 = 7$). However, it can also lead to overfitting and poor reconstruction accuracy of test images.

Finally, we demonstrate large-scale results on the MNIST dataset [78] containing 60k training images and 10k testing images, which is prohibitive for many nonlinear dimensionality reduction implementations. In Figure 3.20, training error is shown against elapsed time using both batch and stochastic optimization with a batch size of 1000. Stochastic optimization leads to much faster convergence in less than 10 minutes with an unoptimized Matlab implementation. Also shown are the resulting two-dimensional latent representations and example reconstructions of the testing images. While batch optimization leads to slightly lower training error, the over-separated latent space indicates overfitting in comparison to stochastic optimization. Note that while some techniques designed specifically for low-dimensional visualization (e.g. t-SNE [145]) may result in better class separation, they cannot reconstruct the input or be applied to new data, leading to limited applicability. In Figure 3.20, quantitative results are also shown demonstrating reconstruction performance and nearest-neighbor classification performance.

3.3.5 Conclusion

Additive Component Analysis combines the simplicity and broad applicability of linear component analysis with the nonlinear representational power of manifold learning. It produces robust and interpretable latent representations given by memory-efficient models optimized for data reconstruction. This results in significantly improved performance, especially in the presence of noise, enabling the detailed analysis and visualization of large, real-world datasets. Furthermore, the composition of multiple ACA layers can overcome the modeling limitations of additive components, with parameters that can be learned jointly with the same memory-efficient optimization procedure. We believe that this demonstrates the encouraging potential for nonparametric deep learning using compositions of additive models as an alternative to standard linear transformations with fixed nonlinear activation functions. This could potentially lead to adaptive representational power with far fewer parameters, reduced overfitting due to the underlying smoothness assumption, and superior robustness.



(a) Convergence Timing Comparison

(b) ACA

(c) PCA

(d) Training Reconstruction

(e) Testing Reconstruction

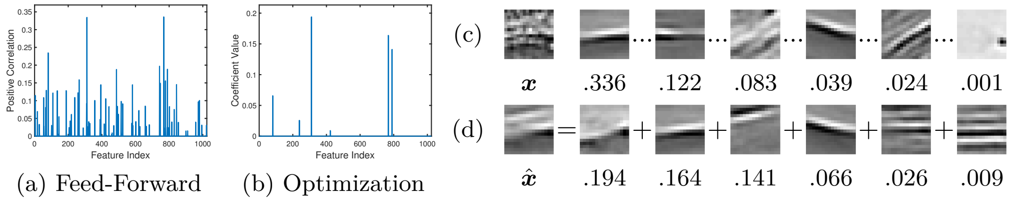
(f) Testing Prediction

Figure 3.20: The effect of our approximate stochastic optimization scheme applied to the MNIST dataset. Reconstruction error (a) is plotted throughout training with stochastic optimization with solid dots shown every 20 iterations. The resulting two-dimensional test data embeddings for $\rho = 10^{-3}$ (b) are compared against to those of PCA (c) alongside grids of reconstructed images from the regions indicated by black squares. Also shown are results on the MNIST datset, showing (a) reconstruction error of training images, (b) reconstruction error of testing images, and (c) testing nearest-neighbor classification error. Performance is compared between PCA and ACA for a variety of roughness parameters ρ and numbers of components m.

4 Deep Network Inference as Data Decomposition

Deep convolutional neural networks have achieved remarkable success in the field of computer vision. While far from new [78], the increasing availability of extremely large, labeled datasets along with modern advances in computation with specialized hardware have resulted in state-of-the-art performance in many problems, including essentially all visual learning tasks. Examples include image classification [62], object detection [63], and semantic segmentation [28]. Despite a rich history of practical and theoretical insights about these problems, modern deep learning techniques typically rely on task-agnostic models and poorly-understood heuristics. However, recent work [86, 143, 16] has shown that specialized architectures incorporating classical domain knowledge can increase parameter efficiency, relax training data requirements, and improve performance.

Prior to the advent of modern deep learning, optimization-based methods like component analysis and sparse coding dominated the field of representation learning. These techniques use structured matrix factorization to decompose data into linear combinations of shared components. Latent representations are inferred by minimizing reconstruction error subject to constraints that enforce properties like uniqueness and interpretability. Importantly, unlike feed-forward alternatives that construct representations in closed-form via independent feature detectors, this iterative optimization-based approach naturally introduces conditional dependence between features in order to best explain data, a useful phenomenon commonly referred to as “explaining away” within the context of graphical models [11]. An example of this effect is shown in Figure 4.1, which compares sparse

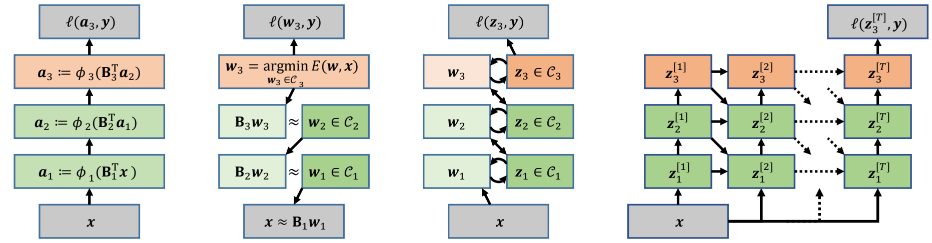


(a) Feed-Forward (b) Optimization

Figure 4.1: An example of the "explaining away" conditional dependence provided by optimization-based inference. Sparse representations constructed by feed-forward nonnegative soft thresholding (a) have many more non-zero elements due to redundancy and spurious activations (c). On the other hand, sparse representations found by ℓ_1-penalized, nonnegative least-squares optimization (b) yield a more parsimonious set of components (d) that optimally reconstruct approximations of the data.

representations constructed using feed-forward soft thresholding with those given by optimization-based inference with an ℓ_1 penalty. While many components in an overcomplete set of features may have high-correlation with an image, constrained optimization introduces competition between components resulting in more parsimonious representations.

Component analysis methods are also often guided by intuitive goals of incorporating prior knowledge into learned representations. For example, statistical independence allows for the separation of signals into distinct generative sources [69], non-negativity leads to parts-based decompositions of objects [80], and sparsity gives rise to locality and frequency selectivity [112]. Due to the difficulty of enforcing intuitive constraints like these with feed-forward computations, deep learning architectures are instead often motivated by distantly-related biological systems [131] or poorly-understand internal mechanisms such as covariate shift [64] and gradient flow [55]. Furthermore, while a theoretical understanding of deep learning is fundamentally lacking [158], even non-convex formulations of matrix factorization are often associated with guarantees of convergence [6], generalization [88], uniqueness [48], and even global optimality [51].


$\ell(\mathbf{a}_3, \mathbf{y})$
$\mathbf{a}_3 \coloneqq \phi_3(\mathbf{B}_3^T \mathbf{a}_2)$
$\mathbf{a}_2 \coloneqq \phi_2(\mathbf{B}_2^T \mathbf{a}_1)$
$\mathbf{a}_1 \coloneqq \phi_1(\mathbf{B}_1^T \mathbf{x})$
$\mathbf{x}$
$\ell(\mathbf{w}_3, \mathbf{y})$
$\mathbf{w}_3 = \operatorname{argmin}_{\mathbf{w}_3 \in \mathcal{C}_3} E(\mathbf{w}, \mathbf{x})$
$\mathbf{B}_3\mathbf{w}_3$
$\approx$
$\mathbf{w}_2 \in \mathcal{C}_2$
$\mathbf{B}_2\mathbf{w}_2$
$\approx$
$\mathbf{w}_1 \in \mathcal{C}_1$
$\mathbf{x} \approx \mathbf{B}_1\mathbf{w}_1$
$\ell(\mathbf{z}_3, \mathbf{y})$
$\mathbf{w}_3$
$\mathbf{z}_3 \in \mathcal{C}_3$
$\mathbf{w}_2$
$\mathbf{z}_2 \in \mathcal{C}_2$
$\mathbf{w}_1$
$\mathbf{z}_1 \in \mathcal{C}_1$
$\mathbf{x}$
$\ell(\mathbf{z}_3^{[T]}, \mathbf{y})$
$\mathbf{z}_3^{[1]}$
$\mathbf{z}_3^{[2]}$
$\mathbf{z}_3^{[T]}$
$\mathbf{z}_2^{[1]}$
$\mathbf{z}_2^{[2]}$
$\mathbf{z}_2^{[T]}$
$\mathbf{z}_1^{[1]}$
$\mathbf{z}_1^{[2]}$
$\mathbf{z}_1^{[T]}$
$\mathbf{x}$

(a) Feed-Forward (b) DeepCA (c) Optimization (d) Unrolled Optimization

Figure 4.2: A comparison between feed-forward neural networks and the proposed deep component analysis (DeepCA) model. While standard deep networks construct learned representations as feed-forward compositions of nonlinear functions (a), DeepCA instead treats them as unknown latent variables to be inferred by constrained optimization (b). To accomplish this, we propose a differentiable inference algorithm that can be expressed as a recurrent generalization of feed-forward networks (c) that can be unrolled to a fixed number of iterations for learning via backpropagation (d).

4.1 Deep Component Analysis

In order to unify the intuitive and theoretical insights of component analysis with the practical advances made possible through deep learning, we introduce the framework of Deep Component Analysis (DeepCA). This novel model formulation can be interpreted as a multilayer extension of traditional component analysis in which multiple layers are learned jointly with intuitive constraints intended to encode structure and prior knowledge. DeepCA can also be motivated from the perspective of deep neural networks by relaxing the implicit assumption that the input to a layer is constrained to be the output of the previous layer, as shown in Equation 4.1 below. In a feed-forward network (left), the output of layer j, denoted $\boldsymbol{a}_j$, is given in closed-form as a nonlinear function of $\boldsymbol{a}_{j-1}$. DeepCA (right) instead takes a generative approach in which the latent variables $\boldsymbol{w}_j$ associated with layer j are *inferred* to optimally reconstruct $\boldsymbol{w}_{j-1}$ as a linear combination of learned components subject to some constraints $\mathcal{C}_j$.

$$\text{Feed-Forward: } \boldsymbol{a}_j = \phi(\mathbf{B}_j^{\mathsf{T}} \boldsymbol{a}_{j-1}) \implies \text{DeepCA: } \mathbf{B}_j \boldsymbol{w}_j \approx \boldsymbol{w}_{j-1} \text{ s.t. } \boldsymbol{w}_j \in \mathcal{C}_j \quad (4.1)$$

From this perspective, intermediate network "activations" cannot be found in closed-form but instead require explicitly solving an optimization problem. While a variety of different techniques could be used for performing this inference, we propose the Alternating Direction Method of Multipliers (ADMM) [15]. Importantly, we demonstrate that after proper initialization, a single iteration of this algorithm is equivalent to a pass through an associated feed-forward neural network with nonlinear activation functions interpreted as proximal operators corresponding to penalties or constraints on the coefficients. The full inference procedure can thus be implemented using Alternating Direction Neural Networks (ADNN), recurrent generalizations of feed-forward networks that allow for parameter learning using backpropagation. A comparison between standard neural networks and DeepCA is shown in Figure 4.2. Experimentally, we demonstrate that recurrent passes through convolutional neural networks enable better sparsity control resulting in consistent performance improvements in both supervised and unsupervised tasks without introducing any additional parameters.

In addition to practical advantages, our model also provides a novel perspective for conceptualizing deep learning techniques. Specifically, rectified linear unit (ReLU) activation functions [49], which are ubiquitous among many state-of-the-art models in a variety of applications, are equivalent to ℓ_1-penalized, sparse projections onto non-negativity constraints. Alongside the interpretation of feed-forward networks as single-iteration approximations of reconstruction objective functions, this suggests new insights towards better understanding the effectiveness of deep neural networks from the perspective of sparse approximation theory.

Deep Component Analysis generalizes inference in the original component analysis model of Equation 3.2 by introducing additional layers $j = 1, \dots, l$ with parameters $\mathbf{B}_j \in \mathbb{R}^{p_{j-1} \times p_j}$. Optimal DeepCA inference can then be accomplished via Equation 4.2, where we use penalty function notation $\Phi_j : \mathbb{R}^{p_j} \to \mathbb{R}$ in place of constraint sets.

$$\boldsymbol{f}^*(\boldsymbol{x}) = \underset{\{\boldsymbol{w}_j\}}{\arg\min} \sum_{j=1}^{l} \tfrac{1}{2} \left\| \boldsymbol{w}_{j-1} - \mathbf{B}_j \boldsymbol{w}_j \right\|_2^2 + \Phi_j(\boldsymbol{w}_j) \quad \text{s.t. } \boldsymbol{w}_0 = \boldsymbol{x} \tag{4.2}$$

Note that hard constraints can still be represented by indicator functions $I(\boldsymbol{w}_j \in \mathcal{C}_j)$ that equal zero if $\boldsymbol{w}_j \in \mathcal{C}_j$ and infinity otherwise. While we use pre-multiplication with a weight matrix $\mathbf{B}_j$ to simplify notation, our method also supports any linear transformation by replacing transposed weight matrix multiplication with its corresponding adjoint operator. For example, the adjoint of convolution is transposed convolution, a popular approach to upsampling in convolutional networks [110].

If the penalty functions are convex, this problem is also convex and can be solved using standard optimization methods. Instead of alternating optimization for parameter learning, however, we use backpropagation like in standard feed-forward networks by replacing the inference function in Equation 2.3 with an optimization algorithm for solving Equation 4.2 unrolled to a fixed number of iterations. This allows for arbitrary loss functions while still enforcing constraints on the inferred outputs.

4.1.1 From Activation Functions to Constraints

Before introducing our inference algorithm, we first discuss the connection between penalties and their nonlinear proximal operators, which forms the basis of the close relationship between DeepCA and traditional neural networks. Ubiquitous within the field of convex optimization, proximal algorithms [118] are methods for solving nonsmooth optimization problems. Essentially, these techniques work by breaking a problem down into a sequence of smaller problems that can often be solved in closed-form by proximal operators $\phi : \mathbb{R}^d \to \mathbb{R}^d$ associated with penalty functions $\Phi : \mathbb{R}^d \to \mathbb{R}$ given by the solution to the following optimization problem, which generalizes projection onto a constraint set:

$$\phi(\boldsymbol{w}) = \arg\min_{\boldsymbol{w}'} \tfrac{1}{2} \left\|\boldsymbol{w} - \boldsymbol{w}'\right\|_2^2 + \Phi(\boldsymbol{w}') \tag{4.3}$$

Within the framework of DeepCA, we interpret nonlinear activation functions in deep networks as proximal operators associated with convex penalties on latent coefficients in each layer. hile this connection cannot be used to generalize all nonlinearities, many can naturally be interpreted as proximal operators. For example, the sparsemax activation function is a projection onto the probability simplex [96]. Similarly, the ReLU activation function is a projection onto the nonnegative orthant. When used with a negative bias $\boldsymbol{b}$, it is equivalent to nonnegative soft-thresholding $\mathcal{S}_{\boldsymbol{b}}^{+}$, the proximal operator associated with nonnegative

ℓ_1 regularization:

$$\Phi^{\ell_1^+}(\boldsymbol{w}) = I(\boldsymbol{w} \geq 0) + \textstyle\sum_p b_p \left| w_p \right| \quad \Longrightarrow \quad \phi^{\ell_1^+}(\boldsymbol{w}) = \mathcal{S}_{\boldsymbol{b}}^+(\boldsymbol{w}) = \mathrm{ReLU}(\boldsymbol{w} - \boldsymbol{b}) \quad (4.4)$$

While this equivalence has been noted previously as a means to theoretically analyze convolutional neural networks [117], DeepCA supports optimizing the bias $\boldsymbol{b}$ as an ℓ_1 penalty hyperparameter via backpropagation for adaptive regularization, which results in better control of representation sparsity.

In addition to standard activation functions, DeepCA also allows for enforcing additional constraints that encode prior knowledge if their corresponding proximal operators can be computed efficiently. For our example of single-image depth prediction with a sparse set of known outputs $\boldsymbol{y}$ provided as prior knowledge, the penalty function on the final output $\boldsymbol{w}_l$ is $\boldsymbol{\Phi}_l(\boldsymbol{w}_l) = I(\mathbf{S}\boldsymbol{w}_l = \boldsymbol{y})$ where the selector matrix $\mathbf{S}$ extracts the indices corresponding to the known outputs in $\boldsymbol{y}$. The associated proximal operator $\boldsymbol{\phi}_l$ projects onto this constraint set by simply correcting the outputs that disagree with the known constraints. Note that this would not be an effective output nonlinearity in a feed-forward network because, while the constraints would be technically satisfied, there is nothing to enforce that they be consistent with neighboring predictions leading to unrealistic discontinuities. In contrast, DeepCA inference minimizes the reconstruction error at each layer subject to these constraints by taking multiple iterations through the network.

4.1.2 Alternating Direction Neural Networks

With the model parameters fixed, we solve our DeepCA inference problem using the Alternating Direction Method of Multipliers (ADMM), a general optimization technique that has been successfully used in a wide variety of applications [15]. To derive the algorithm applied to our problem, we first modify our objective function by introducing auxiliary variables $\boldsymbol{z}_j$ that we constrain to be equal to the unknown coefficients $\boldsymbol{w}_j$, as shown in Equation 4.5 below.

$$\operatorname*{arg\,min}_{\{\boldsymbol{w}_j, \boldsymbol{z}_j\}} \sum_{j=1}^{l} \tfrac{1}{2} \left\| \boldsymbol{z}_{j-1} - \mathbf{B}_j \boldsymbol{w}_j \right\|_2^2 + \Phi_j(\boldsymbol{z}_j) \quad \text{s.t. } \boldsymbol{w}_0 = \boldsymbol{x}, \forall j : \boldsymbol{w}_j = \boldsymbol{z}_j \qquad (4.5)$$

From this, we construct the augmented Lagrangian $\mathcal{L}_\rho$ with dual variables $\boldsymbol{\lambda}$ and a quadratic penalty hyperparameter $\rho = 1$:

$$\mathcal{L}_\rho = \sum_{j=1}^{l} \tfrac{1}{2} \left\| \boldsymbol{z}_{j-1} - \mathbf{B}_j \boldsymbol{w}_j \right\|_2^2 + \Phi_j(\boldsymbol{z}_j) + \boldsymbol{\lambda}_j^\mathsf{T} (\boldsymbol{w}_j - \boldsymbol{z}_j) + \tfrac{\rho}{2} \left\| \boldsymbol{w}_j - \boldsymbol{z}_j \right\|_2^2 \tag{4.6}$$

The ADMM algorithm then proceeds by iteratively minimizing $\mathcal{L}_\rho$ with respect to each set of variables with the others fixed, breaking our full inference problem into smaller pieces that can each be solved in closed form. Due to the decoupling of layers in our DeepCA model, the latent activations can be updated incrementally by stepping through each layer in succession, resulting in faster convergence and computations that mirror the computational structure of deep neural networks. With only one layer, our objective function is separable and so this algorithm reduces to the classical two-block ADMM, which has extensive convergence guarantees [15]. For multiple layers, however, our problem becomes non-separable and so this algorithm can be seen as an instance of cyclical multi-block ADMM with quadratic coupling terms. While our experiments have shown this approach to be effective in our applications, theoretical analysis of its convergence properties is still an active area of research [27].

A single iteration of our algorithm proceeds by taking the following steps for all layers $j = 1, \ldots, l$ in succession:

1.) First, $\boldsymbol{w}_j$ is updated by minimizing the Lagrangian after fixing the associated auxiliary variable $\boldsymbol{z}_j$ from the previous iteration along with that of the previous layer $\boldsymbol{z}_{j-1}$ from the current iteration:

$$\begin{aligned} \boldsymbol{w}_j^{[t+1]} &:= \arg\min_{\boldsymbol{w}_j} \mathcal{L}_\rho(\boldsymbol{w}_j, \boldsymbol{z}_{j-1}^{[t+1]}, \boldsymbol{z}_j^{[t]}, \boldsymbol{\lambda}_j^{[t]}) \\ &= \left(\mathbf{B}_j^\mathsf{T} \mathbf{B}_j + \rho \mathbf{I} \right)^{-1} (\mathbf{B}_j^\mathsf{T} \boldsymbol{z}_{j-1}^{[t+1]} + \rho \boldsymbol{z}_j^{[t]} - \boldsymbol{\lambda}_j^{[t]}) \end{aligned} \tag{4.7}$$

This is an unconstrained linear least squares problem, so it's solution is given by solving a linear system of equations.

2.) Next, $\boldsymbol{z}_j$ is updated by fixing the newly updated $\boldsymbol{w}_j$ along with the next layer's coefficients $\boldsymbol{w}_{j+1}$ from the previous iteration:

$$\begin{aligned}\boldsymbol{z}_j^{[t+1]} &:= \arg\min_{\boldsymbol{z}_j} \mathcal{L}_\rho(\boldsymbol{w}_j^{[t+1]}, \boldsymbol{w}_{j+1}^{[t]}, \boldsymbol{z}_j, \boldsymbol{\lambda}_j^{[t]}) \\ &= \phi_j\big(\tfrac{1}{\rho+1}\mathbf{B}_{j+1}\boldsymbol{w}_{j+1}^{[t]} + \tfrac{\rho}{\rho+1}(\boldsymbol{w}_j^{[t+1]} + \tfrac{1}{\rho}\boldsymbol{\lambda}_j^{[t]})\big) \\ \boldsymbol{z}_l^{[t+1]} &:= \phi_j\big(\boldsymbol{w}_j^{[t+1]} + \tfrac{1}{\rho}\boldsymbol{\lambda}_j^{[t]}\big)\end{aligned} \tag{4.8}$$

This is the proximal minimization problem from Equation 4.3, so its solution is given in closed form via the proximal operator $\boldsymbol{\phi}_j$ associated with the penalty function $\boldsymbol{\Phi}_j$. Note that for $j \neq l$, its argument is a convex combination of the current coefficients $\boldsymbol{w}_j$ and feedback that enforces consistency with the next layer.

3.) Finally, the dual variables $\boldsymbol{\lambda}_j$ are updated with the constraint violations scaled by the penalty parameter ρ.

$$\boldsymbol{\lambda}_j^{[t+1]} := \boldsymbol{\lambda}_j^{[t]} + \rho(\boldsymbol{w}_j^{[t+1]} - \boldsymbol{z}_j^{[t+1]}) \tag{4.9}$$

This process is then repeated until convergence. Though not available as a closed-form expression, in the next section we demonstrate how this algorithm can be posed as a recurrent generalization of a feed-forward neural network.

Our inference algorithm essentially follows the same pattern as a deep neural network: for each layer, a learned linear transformation is applied to the previous output followed by a fixed nonlinear function. Building upon this observation, we implement it using a recurrent network with standard layers, thus allowing the model parameters to be learned using backpropagation.

Recall that the $\boldsymbol{w}_j$ update in Equation 4.7 requires solving a linear system of equations. While differentiable, this introduces additional computational complexity not present in standard neural networks. To overcome this, we implicitly assume that the parameters in over-complete layers are Parseval tight frames, i.e. so that $\mathbf{B}_j\mathbf{B}_j^\mathsf{T} = \mathbf{I}$. This property is theoretically advantageous in the field of sparse approximation [26] and has been used as a constraint to encourage robustness in deep neural networks [98]. However, in our experiments we found that it was unnecessary to explicitly enforce this assumption during training; with appropriate learning rates, backpropagating through our inference algorithm was enough to ensure that repeated iterations did not result in diverging sequences of variable updates. Thus, under this assumption, we can simplify the update in

Algorithm 1: Feed-Forward

```
Input: $\boldsymbol{x}$, $\{\mathbf{B}_j, \boldsymbol{b}_j\}$
Output: $\{\boldsymbol{w}_j\}$, $\{\boldsymbol{z}_j\}$
Initialize: $\boldsymbol{z}_0 = \boldsymbol{x}$
for $j = 1, \ldots, l$ do
    Pre-activation:
      $\boldsymbol{w}_j := \mathbf{B}_j^\mathsf{T} \boldsymbol{z}_{j-1}$
    Activation:
      $\boldsymbol{z}_j := \phi_j(\boldsymbol{w}_j - \boldsymbol{b}_j)$
end
```

Algorithm 2: Alternating Direction Neural Network

```
Input: $\boldsymbol{x}$, $\{\mathbf{B}_j, \boldsymbol{b}_j\}$
Output: $\{\boldsymbol{w}_j^{[T]}\}$, $\{\boldsymbol{z}_j^{[T]}\}$
Initialize: $\{\boldsymbol{\lambda}_j^{[0]}\} = \mathbf{0}$, $\{\boldsymbol{w}_j^{[1]}, \boldsymbol{z}_j^{[1]}\}$ from Algorithm 1
for $t = 1, \ldots, T-1$ do
    for $j = 1, \ldots, l$ do
        Dual: Update $\boldsymbol{\lambda}_j^{[t]}$ (Eq. 4.9)
        Pre-activation: Update $\boldsymbol{w}_j^{[t+1]}$ (Eq. 4.10)
        Activation: Update $\boldsymbol{z}_j^{[t+1]}$ (Equ. 4.8)
    end
end
```

Equation 4.7 using the Woodbury matrix identity as follows:

$$\boldsymbol{w}_j^{[t+1]} := \tilde{\boldsymbol{z}}_j^{[t]} + \tfrac{1}{\rho+1} \mathbf{B}_j^\mathsf{T} \big(\boldsymbol{z}_{j-1}^{[t+1]} - \mathbf{B}_j \tilde{\boldsymbol{z}}_j^{[t]} \big), \quad \tilde{\boldsymbol{z}}_j^{[t]} := \boldsymbol{z}_j^{[t]} - \tfrac{1}{\rho} \boldsymbol{\lambda}_j^{[t]} \tag{4.10}$$

As this only involves simple linear transformations, our ADMM algorithm for solving the optimization problem in our inference function $\boldsymbol{f}^*$ can be expressed as a recurrent neural network that repeatedly iterates until convergence. In practice, however, we unroll the network to a fixed number of iterations T for an approximation of optimal inference so that $\boldsymbol{f}^{[T]}(\boldsymbol{x}) \approx \boldsymbol{f}^*(\boldsymbol{x})$. Our full algorithm is summarized in Algorithms 1 and 2.

4.1.3 Generalization of Feed-Forward Networks

Given proper initialization of the variables, a single iteration of this algorithm is identical to a single pass through a feed-forward network. Specifically, if we let $\boldsymbol{\lambda}_j^{[0]} = \mathbf{0}$ and $\boldsymbol{z}_j^{[0]} = \mathbf{B}_j^\mathsf{T} \boldsymbol{z}_{j-1}^{[1]}$, where we again denote $\boldsymbol{z}_0^{[1]} = \boldsymbol{x}$, then $\boldsymbol{w}_j^{[1]}$ is equivalent to the pre-activation of a neural network layer:

$$\boldsymbol{w}_j^{[1]} := \mathbf{B}_j^\mathsf{T} \boldsymbol{z}_{j-1}^{[1]} + \tfrac{1}{\rho+1} \mathbf{B}_j^\mathsf{T} \big(\boldsymbol{z}_{j-1}^{[1]} - \mathbf{B}_j (\mathbf{B}_j^\mathsf{T} \boldsymbol{z}_{j-1}^{[1]}) \big) = \mathbf{B}_j^\mathsf{T} \boldsymbol{z}_{j-1}^{[1]} \tag{4.11}$$

Similarly, if we initialize $\boldsymbol{w}_{j+1}^{[0]} = \mathbf{B}_{j+1}^\mathsf{T} \boldsymbol{w}_j^{[1]}$, then $\boldsymbol{z}_j^{[1]}$ is equivalent to the corresponding nonlinear activation using the proximal operator ϕ_j:

$$\boldsymbol{z}_j^{[1]} := \phi_j \big(\tfrac{1}{\rho+1} \mathbf{B}_{j+1} (\mathbf{B}_{j+1}^\mathsf{T} \boldsymbol{w}_j^{[1]}) + \tfrac{\rho}{\rho+1} \boldsymbol{w}_j^{[1]} \big) = \phi_j \big(\boldsymbol{w}_j^{[1]} \big) \tag{4.12}$$

Thus, one iteration of our inference algorithm is equivalent to the standard feed-forward neural network given in Equation 2.2, i.e. $\boldsymbol{f}^{[1]}(\boldsymbol{x}) = \boldsymbol{f}^{\mathrm{DNN}}(\boldsymbol{x})$, where nonlinear activation functions are interpreted as proximal operators corresponding to the penalties of our DeepCA model. Additional iterations through the network lead to more accurate inference approximations while explicitly satisfying constraints on the latent variables.

4.1.4 Learning by Backpropagation

With DeepCA inference approximated by differentiable ADNNs, the model parameters can be learned in the same way as standard feed-forward networks. Extending the nested component analysis optimization problem from Equation 2.1, the inference function $\boldsymbol{f}^{[T]}$ can be used as a generalization of feed-forward network inference $\boldsymbol{f}^{[1]}$ for backpropagation with arbitrary loss functions L that encourage the output to be consistent with provided supervision $\boldsymbol{y}^{(i)}$, as shown in Equation 4.13 below. Here, only the latent coefficients $\boldsymbol{f}_l^{[T]}(\boldsymbol{x}^{(i)})$ from the last layer are shown in the loss function, but other intermediate outputs $j \neq l$ could also be included.

$$\underset{\{\mathbf{B}_j, \boldsymbol{b}_j\}}{\arg\min} \sum_{i=1}^{n} L\big(\boldsymbol{f}_l^{[T]}(\boldsymbol{x}^{(i)}),\, \boldsymbol{y}^{(i)}\big) \tag{4.13}$$

From an agnostic perspective, an ADNN can thus be seen as an end-to-end deep network architecture with a particular sequence of linear and nonlinear transformations and tied weights. More iterations ($T > 1$) result in networks with greater effective depth, potentially allowing for the representation of more complex nonlinearities. However, because the network architecture was derived from an algorithm for inference in our DeepCA model instead of arbitrary compositions of parameterized transformations, the greater depth requires no additional parameters and serves the very specific purpose of satisfying constraints on the latent variables while enforcing consistency with the model parameters.

4.1.5 Sparse Measurements as Constraints for Depth Completion

DeepCA also allows for other constraints that would be impossible to effectively enforce with a single feed-forward pass through a network. As an example,

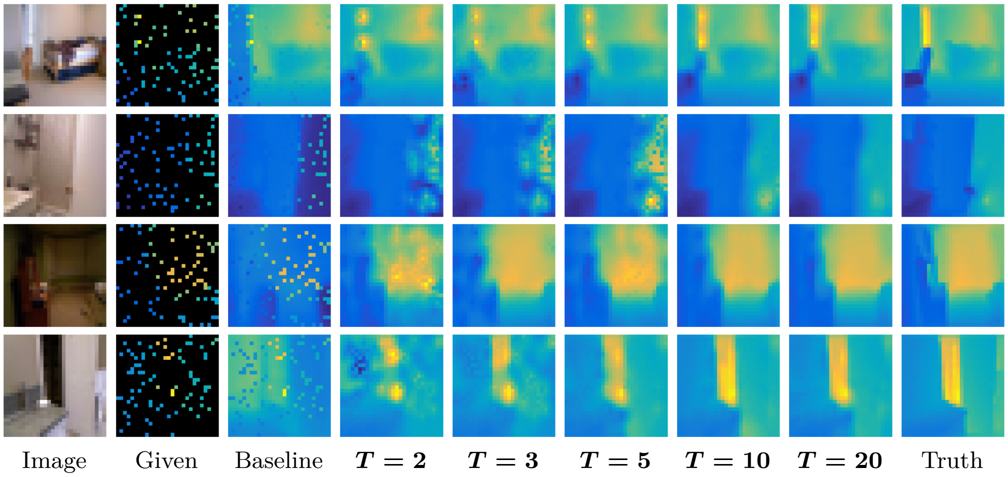
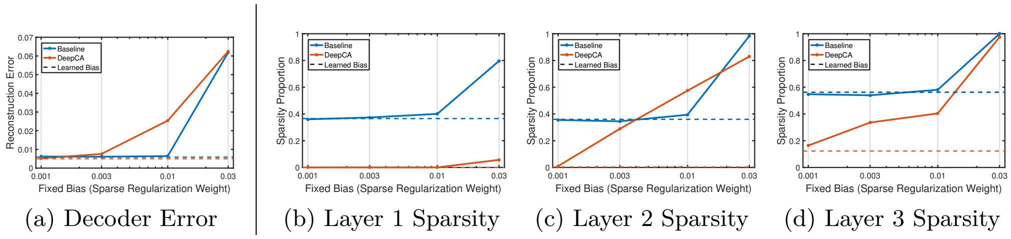


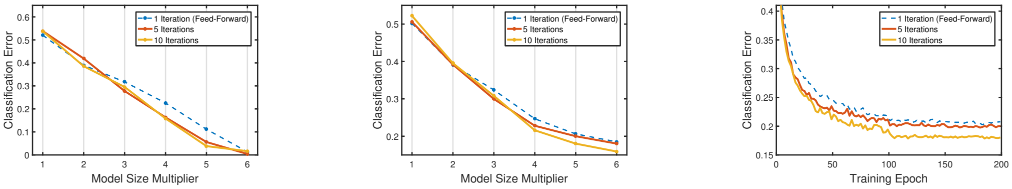
Figure 4.3: A demonstration of DeepCA applied to single-image depth prediction using images concatenated with sparse sets of known depth values as input. Baseline feed-forward networks are not guaranteed to produce outputs that are consistent with the given depth values. In comparison, ADNNs with an increasing number of iterations ($T > 1$) learn to satisfy the sparse output constraints, resolving ambiguities for more accurate predictions without unrealistic discontinuities.

we consider the task of single-image depth prediction, a difficult problem due to the absence of three-dimensional information such as scale and perspective. In many practical scenarios, however, sparse sets of known depth outputs are available for resolving these ambiguities to improve accuracy. This prior knowledge can come from additional sensor modalities like LIDAR or from other 3D reconstruction algorithms that provide sparse depths around textured image regions. Feed-forward networks have been proposed for this problem by concatenating known depth values as an additional input channel [93]. However, while this provides useful context, predictions are not guaranteed to be consistent with the given outputs leading to unrealistic discontinuities. In comparison, DeepCA enforces the constraints by treating predictions as unknown latent variables. Some examples of how this behavior can resolve ambiguities are shown in Figure 4.3 where ADNNs with additional iterations learn to propagate information from the given depth values to produce more accurate predictions.



(a) Decoder Error (b) Layer 1 Sparsity (c) Layer 2 Sparsity (d) Layer 3 Sparsity

Figure 4.4: A demonstration of the effects of fixed (solid lines) and learnable (dotted lines) bias parameters on the reconstruction error (a) and activation sparsity (b-d) comparing feed forward networks (blue) with DeepCA (red). All models consist of three layers each with 512 components. Due to the conditional dependence provided by recurrent feedback, DeepCA learns to better control the sparsity level in order improve reconstruction error. As ℓ_1 regularization weights, the biases converge towards zero resulting in denser activations and higher network capacity for reconstruction.

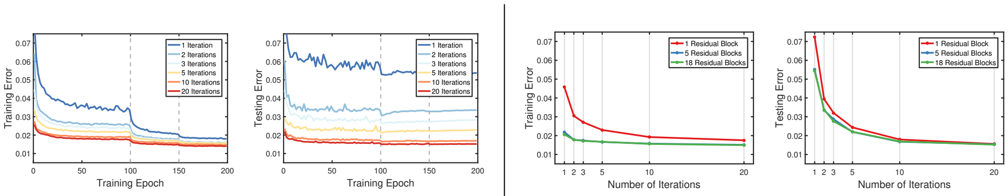


(a) Training Error (b) Testing Error (c) Optimization

Figure 4.5: The effect of increasing model size on training (a) and testing (b) classification error, demonstrating consistently improved performance of ADNNs over feed-forward networks, especially in larger models. The base model consists of two 3 × 3, 2-strided convolutional layers followed by one fully-connected layer with 4, 8, and 16 components respectively. Also shown are is the classification error throughout training (c).

4.1.6 Experimental Results

In this section, we demonstrate some practical advantages of more accurate inference approximations in our DeepCA model using recurrent ADNNs over feed-forward networks. Even without additional prior knowledge, standard convolutional networks with ReLU activation functions still benefit from additional recurrent iterations as demonstrated by consistent improvements in both supervised and unsupervised tasks on the CIFAR-10 dataset [74]. Specifically, for an



(a) Training (b) Testing (c) Train Error (d) Test Error

Figure 4.6: Quantitative results on reduced-size images from the NYU-Depth V2 dataset. The training (a) and testing (b) reconstruction errors throughout optimization show that more iterations ($T > 1$) reduce convergence time and give much lower error on held-out test data. With a sufficiently large number of iterations, even lower-capacity models with encoders consisting of fewer residual blocks all achieve nearly the same level of performance with small discrepancies between training (c) and testing (d) errors.

unsupervised autoencoder with an ℓ_2 reconstruction loss, Figure 4.4 shows that the additional iterations of ADNNs allow for better sparsity control, resulting in higher network capacity through denser activations and lower reconstruction error. This suggests that recurrent feedback allows ADNNs to learn richer representation spaces by explicitly penalizing activation sparsity. For supervised classification with a cross-entropy loss, ADNNs also see improved accuracy as shown in Figure 4.5, particularly for larger models with more parameters per layer. Because we treat layer biases as learned hyperparameters that modulate the relative weight of ℓ_1 activation penalties, this improvement could again be attributed to this adaptive sparsity encouraging more discriminative representations across semantic categories.

While these experiments emphasize the importance of sparsity in deep networks and justify our DeepCA model formulation, the effectiveness of feed-forward soft thresholding as an approximation of explicit ℓ_1 regularization limits the amount of additional capacity that can be achieved with more iterations. As such, ADNNs provide much greater performance gains when prior knowledge is available in the form of constraints that *cannot* be effectively approximated by feed-forward nonlinearities. This is exemplified by our application of output-constrained single-image depth prediction where simple feed-forward correction of the known depth values results in inconsistent discontinuities. We demonstrate

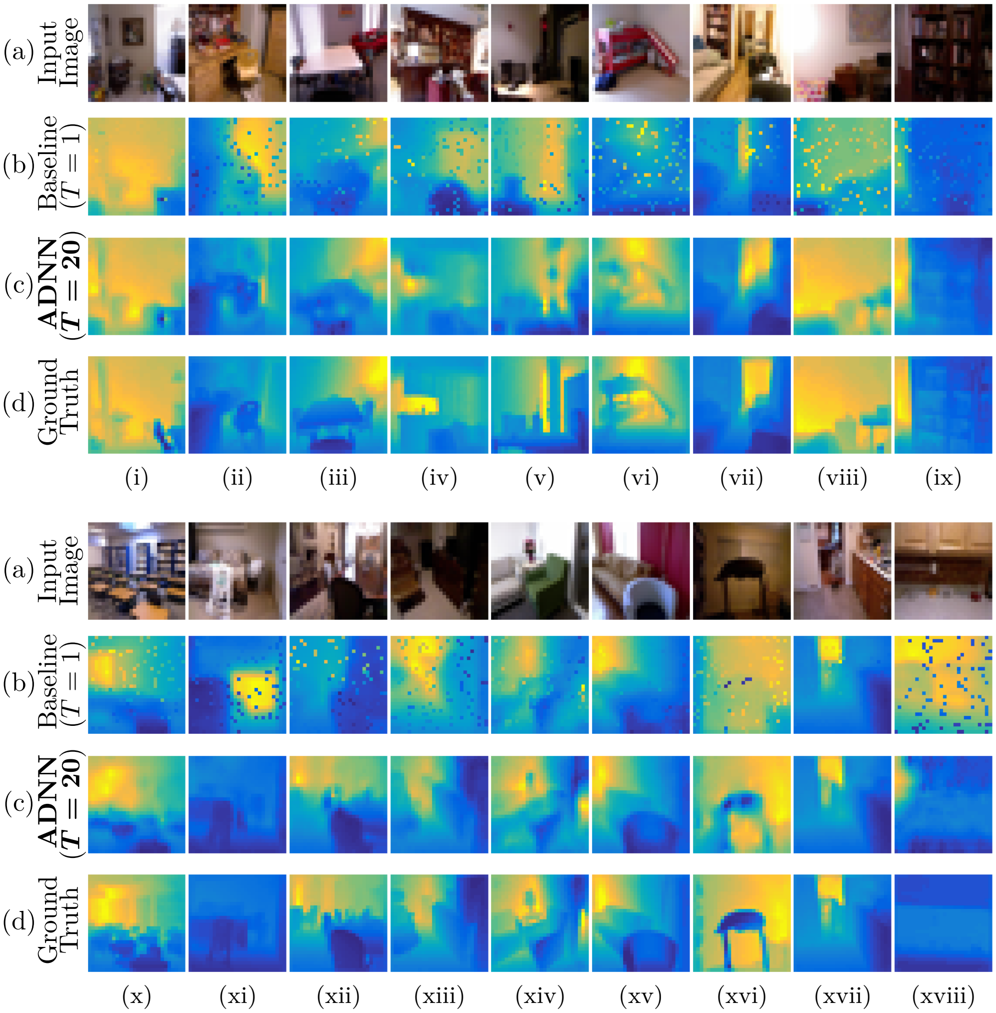


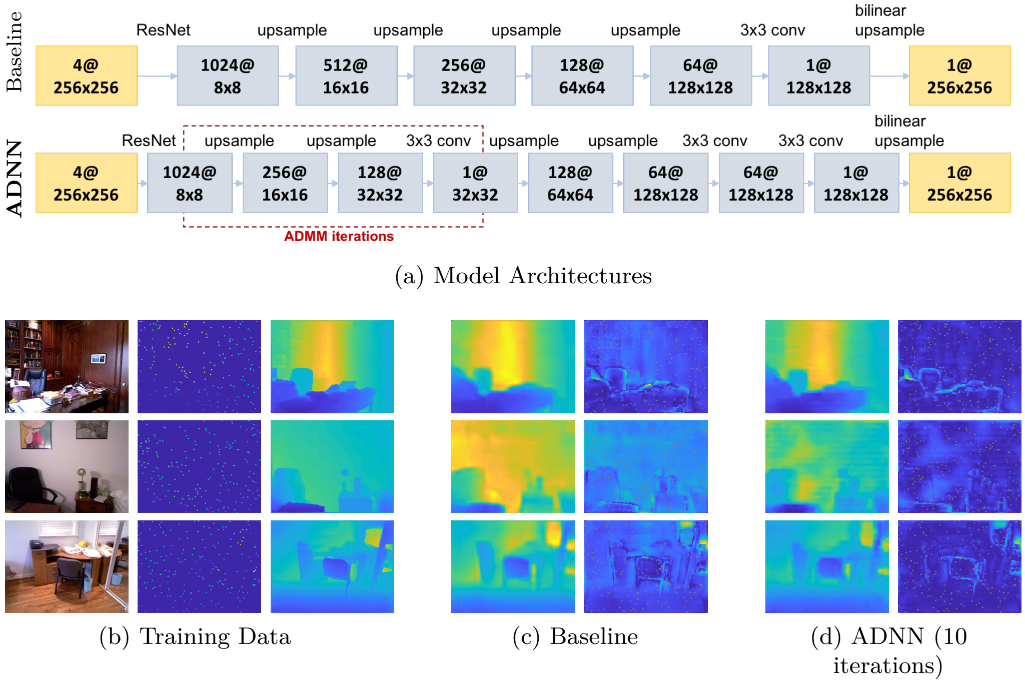
Figure 4.7: Qualitative depth prediction results given a single image (a) and a sparse set of known depth values as input. Outputs of the baseline feed-forward model (b) are inconsistent with the constraints as evidenced by unrealistic discontinuities. An ADNN with $T = 20$ iterations (c) learns to enforce the constraints, resolving ambiguities for more detailed predictions that better agree with ground truth depth maps (d). Depending on the difficulty, additional iterations may have little effect on the output (xvii) or be insufficient to consistently integrate the known constraint values (xviii).

this with the NYU-Depth V2 dataset [106], from which we sample 60k training images and 500 testing images from held-out scenes. To enable clearer visualization, we resize the images to 28 × 28 and then randomly sample 10% of the ground truth depth values to simulate known measurements. Following [93], our model architecture uses a ResNet encoder for feature extraction of the image concatenated with the known depth values as an additional input channel. This is followed by an ADNN decoder composed of three transposed convolution upsampling layers with biased ReLU nonlinearites in the first two layers and a constraint correction proximal operator in the last layer. Figure 4.6 shows the mean absolute prediction errors of this model with increasing numbers of iterations and different encoder sizes. While all models have similar prediction error on training data, ADNNs with more iterations achieve significantly improved generalization performance, reducing the test error of the feed-forward baseline by over 72% from 0.054 to 0.015 with 20 iterations even with low-capacity encoders. Qualitative visualizations in Figure 4.7 show that these improvements result from consistent constraint satisfaction that serves to resolve depth ambiguities.

In Figure 4.8, we also show qualitative and quantitative results on the full-sized images, an easier problem due to reduced ambiguities provided by higher-resolution details. Quantitative metrics in Table 4.1 (following [93]) demonstrate the effect of changing the ResNet encoder size on prediction performance. Despite having far fewer learnable parameters, ADNNs perform comparably to a state-of-the-art feed-forward model due to explicit enforcement of the sparse output constraints. While feed-forward models have achieved good performance given sufficient model capacity [93], they generalize poorly due to globally-biased prediction errors causing disagreement with the known measurements. By explicitly enforcing agreement with the sparse output constraints, ADNNs reduce outliers and give improved test performance that is comparable with feed-forward networks requiring significantly more learnable parameters.

4.1.7 Conclusion

DeepCA is a novel deep model formulation that extends shallow component analysis techniques to increase representational capacity. Unlike feed-forward



(a) Model Architectures

(b) Training Data

(c) Baseline

(d) ADNN (10 iterations)

Figure 4.8: Results on full-sized images from the NYU-Depth V2 dataset, comparing a feed-forward baseline and an ADNN architecture with 10 iterations (a). Given input images, constraints, and ground truth depth maps (b) for both baseline (c) and ADNN (d) architectures, example predictions and absolute error maps are visualized.

Table 4.1: Quantitative ADNN results on the full-sized NYU dataset.

Method	ResNet	# Params	RMSE	Rel	δ_1	δ_2	δ_3
Baseline	18	1.5×10^7	0.54	0.16	79.2	94.7	99.4
ADNN	18	1.2×10^7	0.28	0.06	95.5	99.4	99.9
Baseline	10	8.8×10^6	0.56	0.16	79.8	94.6	99.4
ADNN	10	$\mathbf{6.5 \times 10^6}$	0.24	0.05	**97.3**	**99.6**	**99.9**
[93]	50	3.4×10^7	**0.23**	**0.04**	97.1	99.4	99.8

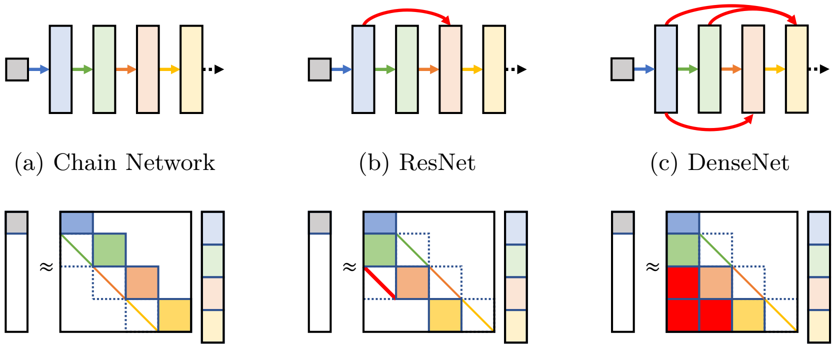
networks, intermediate network activations are interpreted as latent variables to be inferred using an iterative constrained optimization algorithm implemented as a recurrent ADNN. This allows for learning with arbitrary loss functions and provides a tool for consistently integrating prior knowledge in the form of constraints or regularization penalties. Due to its close relationship to feed-forward networks, which are equivalent to one iteration of this algorithm with proximal operators replacing nonlinear activation functions, DeepCA also provides a novel perspective from which to interpret deep learning, suggesting possible new directions for the analysis and design of network architectures from the perspective of sparse approximation theory.

4.2 Model Selection with the Deep Frame Potential

We propose to interpret feed-forward deep networks as a method for approximate inference in related sparse coding problems. These problems aim to optimally reconstruct zero-padded input images as sparse, nonnegative linear combinations of atoms from architecture-dependent dictionaries, as shown in Figure 4.9. We propose to indirectly analyze practical deep network architectures with complicated skip connections, like residual networks (ResNets) [54] and densely connected convolutional networks (DenseNets) [62], simply through the dictionary structures that they induce.

To accomplish this, we introduce the deep frame potential for summarizing the parameters of feed-forward deep networks. As a lower bound on mutual coherence–the maximum magnitude of the normalized inner products between all pairs of dictionary atoms [39]–it is theoretically tied to generalization properties of the related sparse coding problems. However, its minimizers depend only on the dictionary structures induced by the corresponding network architectures. This enables dataless model comparison by jointly quantifying contributions of depth, width, and connectivity.

Our approach is motivated by sparse approximation theory [43], a field that encompasses properties like uniqueness and robustness of shallow, overcomplete representations. In sparse coding, capacity is controlled by the number of dictionary atoms used in sparse data reconstructions. While more parameters allow



(a) Chain Network

(b) ResNet

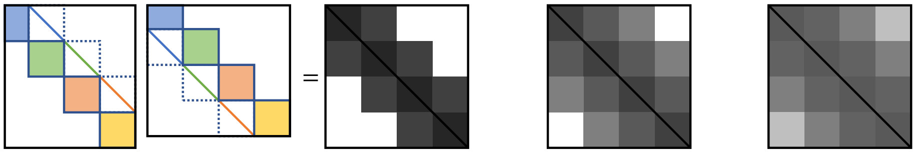
(c) DenseNet

(d) Induced Dictionary Structures for Sparse Approximation

Figure 4.9: Why are some deep neural network architectures better than others? In comparison to (a) standard chain connections, skip connections like those in (b) ResNets [54] and (c) DenseNets [62] have demonstrated significant improvements in training effectiveness, parameter efficiency, and generalization performance. (d) We provide one possible explanation for this phenomenon by approximating network activations as solutions to sparse approximation problems with different induced dictionary structures. To summarize these architecture-dependent differences, we propose the deep frame potential–a measure of coherence that is related to representation stability–as a criterion for dataless model selection.

for more accurate representations, they may also increase input sensitivity for worse generalization performance. Conceptually, this is comparable to overfitting in nearest-neighbor classification, where representations are sparse, one-hot indicator vectors corresponding to nearest training examples. As the number of training data increases, the distance between them decreases, so they are more likely to be confused with one another. Similarly, nearby dictionary atoms may introduce instability that causes representations of similar data points to become very far apart leading to poor generalization performance. Thus, the robustness of shallow representations is fundamentally limited by the proximity of dictionary atoms through similarity measures like mutual coherence.

However, deep representations have not shown this same correlation between model size and sensitivity [158]. While adding more layers to a deep neural network increases its capacity, it also simultaneously introduces implicit regularization to reduce overfitting. This can be explained through the proposed



(a) Chain Network Gram Matrix (b) ResNet (c) DenseNet

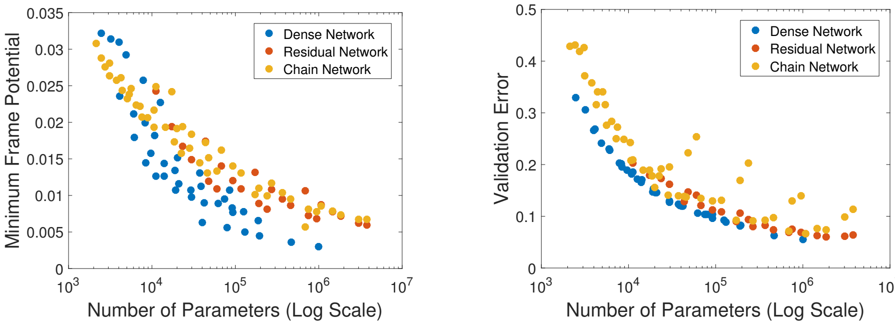
Figure 4.10: In comparison to (a) chain networks, skip connections in (b) residual networks and (c) densely connected networks produce Gram matrix structures with more nonzero elements.

connection to sparse coding, where additional layers increase both capacity and effective input dimensionality. In a higher-dimensional space, dictionary atoms can be spaced further apart for more robust representations. Furthermore, we argue in Figure 4.10 that architectures with denser skip connections induce dictionary structures with more nonzero elements, which provides additional freedom to reduce mutual coherence with fewer parameters. In Figure 4.11, we show that this correlates with improved generalization performance.

We propose to use the minimum deep frame potential as a cue for model selection. Instead of requiring expensive validation on a specific dataset to approximate generalization performance, architectures are chosen based on how efficiently they can reduce the minimum achievable mutual coherence with respect to the number of model parameters. In this paper, we provide an efficient frame potential minimization method for a general class of convolutional networks with skip connections, of which ResNets and DenseNets are shown to be special cases. Furthermore, we derive an analytic expression for the minimum value in the case of fully-connected chain networks. Experimentally, we demonstrate correlation with validation error across a variety of network architectures.

4.2.1 Architecture-Induced Dictionary Structure

While deep representations can be analyzed by accumulating the effects of approximating individual layers in a chain network as shallow sparse coding problems [117], this strategy cannot be easily adapted to account for more complicated interactions between layers. Instead, we adapt the framework of Deep Compo-



(a) Minimum Deep Frame Potential

(b) Validation Error

Figure 4.11: Parameter count is not a good indicator of generalization performance for deep networks. Instead, we compare different network architectures via the minimum deep frame potential, the average nonzero magnitude of inner products between atoms of architecture-induced dictionaries. In comparison to chain networks, skip connections in residual networks and densely connected networks produce Gram matrix structures with more nonzero elements allowing for (a) lower deep frame potentials across network sizes. This correlates with improved parameter efficiency giving (b) lower validation error with fewer parameters.

nent Analysis [99], which jointly represents all layers in a neural network as a single sparse coding problem. In Equation 4.14, the activations $\boldsymbol{w}_j \in \mathbb{R}^{k_j}$ of a feed-forward chain network approximate the solutions to a joint optimization problem where $\boldsymbol{w}_0 = \boldsymbol{x} \in \mathbb{R}^{k_0}$ and the regularization functions Φ_j are nonnegative sparsity-inducing penalties as defined in Equation 2.4.

$$\begin{aligned} \boldsymbol{w}_j &:= \phi_j(\mathbf{B}_j^\mathsf{T} \boldsymbol{w}_{j-1}) \quad \forall j = 1, \dots, l \\ &\approx \underset{\{\boldsymbol{w}_j\}}{\arg\min} \sum_{j=1}^{l} \|\mathbf{B}_j \boldsymbol{w}_j - \boldsymbol{w}_{j-1}\|_2^2 + \Phi_j(\boldsymbol{w}_j) \end{aligned} \tag{4.14}$$

The compositional constraints between adjacent layers are relaxed and replaced by reconstruction error penalty terms, resulting in a convex, nonnegative sparse coding problem.

By combining the terms in the summation of Equation 4.14 together into a single system, this problem can be equivalently represented as shown in Equation 4.15, where the latent variables $\boldsymbol{w}_j$ are stacked together in the vector $\boldsymbol{w}$,

$\Phi(\boldsymbol{w}) = \sum_j \Phi_j(\boldsymbol{w}_j)$, and the input $\boldsymbol{x}$ is augmented with zeros.

$$\arg\min_{\boldsymbol{w}} \left\| \overbrace{\begin{bmatrix} \mathbf{B}_1 & \mathbf{0} & & \\ -\mathbf{I} & \mathbf{B}_2 & \ddots & \\ & \ddots & \ddots & \mathbf{0} \\ & & -\mathbf{I} & \mathbf{B}_l \end{bmatrix}}^{\mathbf{B}} \overbrace{\begin{bmatrix} \boldsymbol{w}_1 \\ \boldsymbol{w}_2 \\ \vdots \\ \boldsymbol{w}_l \end{bmatrix}}^{\boldsymbol{w}} - \begin{bmatrix} \boldsymbol{x} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} \right\|_2^2 + \Phi(\boldsymbol{w}) \tag{4.15}$$

The layer parameters $\mathbf{B}_j \in \mathbb{R}^{k_{j-1} \times k_j}$ are blocks in the induced dictionary $\mathbf{B}$, which has $\sum_j k_{j-1}$ rows and $\sum_j k_j$ columns. It has a lower block-triangular structure of nonzero elements that summarizes the corresponding feed-forward deep network architecture wherein the off-diagonal identity matrices connect adjacent layers.

Model capacity can be increased both by adding additional parameters to a layer or by adding layers, which implicitly pads the input data $\boldsymbol{x}$ with more zeros. This can actually reduce mutual coherence by increasing the system's dimensionality. Depth allows model complexity to scale jointly alongside effective input dimensionality so that the induced dictionary structures still have the capacity for low mutual coherence and improved capabilities for memorization and generalization.

We extend this model formulation to incorporate more complicated network architectures. Because mutual coherence is dependent on normalized dictionary atoms, we observe that the magnitudes of their elements and their inner products can both be reduced by increasing the number of nonzeros. In Equation 4.16, we replace the identity connections of Equation 4.15 with blocks of nonzero parameters to allow for lower mutual coherence.

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{0} & & \\ \mathbf{B}_{21}^{\mathsf{T}} & \mathbf{B}_{22} & \ddots & \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{B}_{l1}^{\mathsf{T}} & \cdots & \mathbf{B}_{l(l-1)}^{\mathsf{T}} & \mathbf{B}_{ll} \end{bmatrix} \tag{4.16}$$

This lower block triangular structure is induced by the feed-forward activations in Equation 4.17, which again approximate the solutions to a nonnegative sparse coding problem.

$$\begin{aligned} \boldsymbol{w}_j &\coloneqq \phi_j\Big(-\mathbf{B}_{jj}^{\mathsf{T}}\sum_{k=1}^{j-1}\mathbf{B}_{jk}^{\mathsf{T}}\boldsymbol{w}_k\Big) \quad \forall j = 1,\dots,l \qquad (4.17)\\ &\approx \underset{\{\boldsymbol{w}_j\}}{\arg\min}\sum_{j=1}^{l}\Big\|\mathbf{B}_{jj}\boldsymbol{w}_j + \sum_{k=1}^{j-1}\mathbf{B}_{jk}^{\mathsf{T}}\boldsymbol{w}_j\Big\|_F^2 + \Phi_j(\boldsymbol{w}_j)\end{aligned}$$

In comparison to Equation 4.14, additional parameters introduce skip connections between layers so that the activations $\boldsymbol{w}_j$ of layer j now depend on those of all previous layers $k < j$.

These connections are similar to the identity mappings in residual networks [54], which introduce dependence between the activations of pairs of layers for even $j \in [1, l-1]$:

$$\boldsymbol{w}_j \coloneqq \phi_j(\mathbf{B}_j^{\mathsf{T}}\boldsymbol{w}_{j-1}),\ \boldsymbol{w}_{j+1} \coloneqq \phi_{j+1}(\boldsymbol{w}_{j-1}{+}\mathbf{B}_{j+1}^{\mathsf{T}}\boldsymbol{w}_j) \qquad (4.18)$$

In comparison to chain networks, no additional parameters are required; the only difference is the addition of $\boldsymbol{w}_{j-1}$ in the argument of ϕ_{j+1}. As a special case of our more general framework, we interpret the activations in Equation 4.18 as approximate solutions to the optimization problem in Equation 4.19:

$$\begin{aligned} &\underset{\{\boldsymbol{w}_j\}}{\arg\min}\ \|\boldsymbol{x} - \mathbf{B}_1\boldsymbol{w}_1\|_2^2 + \sum_{j=1}^{l}\Phi_j(\boldsymbol{w}_j) \qquad (4.19)\\ &+ \sum_{\text{even}\, j}\Big\|\boldsymbol{w}_j - \mathbf{B}_j^{\mathsf{T}}\boldsymbol{w}_{j-1}\Big\|_2^2 + \Big\|\boldsymbol{w}_{j+1} - \boldsymbol{w}_{j-1} - \mathbf{B}_{j+1}^{\mathsf{T}}\boldsymbol{w}_j\Big\|_2^2\end{aligned}$$

This results in the induced dictionary structure of Equation 4.16 with $\mathbf{B}_{jj} = \mathbf{I}$ for $j > 1$, $\mathbf{B}_{jk} = \mathbf{0}$ for $j > k+1$, $\mathbf{B}_{jk} = \mathbf{0}$ for $j > k$ with odd k, and $\mathbf{B}_{jk} = \mathbf{I}$ for $j > k$ with even k.

Building upon the empirical successes of residual networks, densely connected convolution networks [62] incorporate skip connections between earlier layers as well. This is shown in Equation 4.20 where the transformation $\mathbf{B}_j$ of concatenated variables $\boldsymbol{w}_k$ for $k = 1,\dots,j{-}1$ is equivalently written as the summation of smaller transformations $\mathbf{B}_{jk}$.

$$\boldsymbol{w}_j \coloneqq \phi_j\Big(\mathbf{B}_j^{\mathsf{T}}\left[\boldsymbol{w}_k\right]_{k=1}^{j-1}\Big) = \phi_j\Big(\sum_{k=1}^{j-1}\mathbf{B}_{jk}^{\mathsf{T}}\boldsymbol{w}_k\Big) \qquad (4.20)$$

These activations again provide approximate solutions to the problem in Equation 4.17 with the induced dictionary structure of Equation 4.16 where $\mathbf{B}_{jj} = \mathbf{I}$ for $j > 1$ and the lower blocks $\mathbf{B}_{jk}$ for $j > k$ are filled with learned parameters.

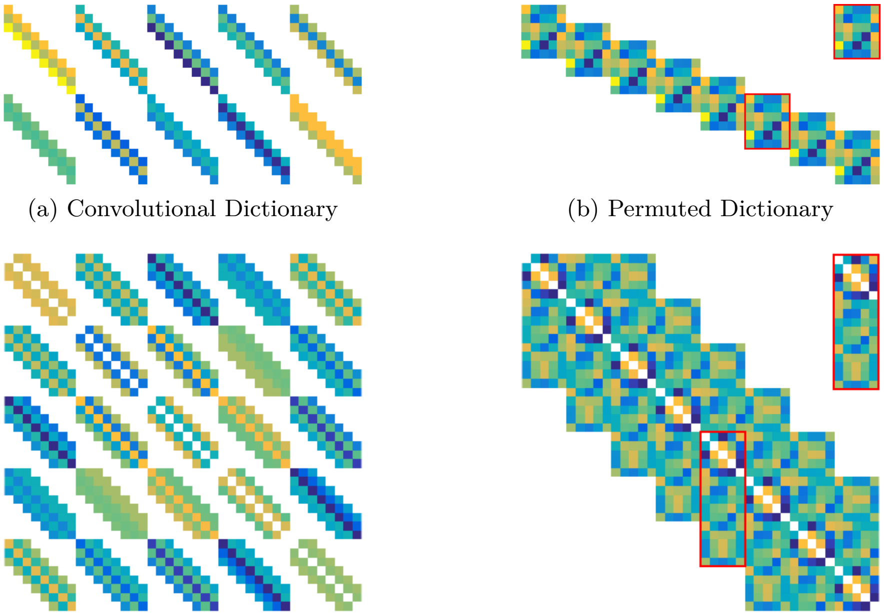
Skip connections enable effective learning in much deeper networks than chain-structured alternatives. While originally motivated from the perspective of making optimization easier [54], adding more connections between layers also improves generalization performance and parameter efficiency [62]. As compared in Figure 4.11, denser skip connections induce dictionary structures with denser Gram matrices. This suggests that architectures can be quantified and compared based on their capacities for inducing dictionaries with low mutual coherence.

4.2.2 The Deep Frame Potential

We propose to use a lower bound on the mutual coherence of the induced structured dictionary as a means for data-independent comparison of architecture capacities. However, directly optimizing mutual coherence from Equation 2.5 can be difficult in practice due to its piecewise structure. A tight lower bound can be found by replacing the maximum off-diagonal Gram matrix element of the Gram matrix with the mean, as shown in Equation 4.21 where $N(\mathbf{G})$ is the number of nonzero off-diagonal elements in the Gram matrix $\mathbf{G}$ and $\operatorname{Tr}\mathbf{G}$ equals the number of dictionary atoms.

$$\mu^2(\mathbf{B}) \geq F^2(\mathbf{B}) = \frac{\|\mathbf{G}\|_F^2 - \operatorname{Tr}\mathbf{G}}{N(\mathbf{G})} \tag{4.21}$$

Equality is met in the case of equiangular tight frames when the normalized inner products between all dictionary atoms are equivalent [71]. In practice, we employ the averaged frame potential $F^2(\mathbf{B})$ as a strongly-convex objective function because of its superior optimization properties [10]. Due to the block-sparse structure of the induced dictionary matrices from Equation 4.16, we evaluate the frame potential in terms of local blocks $\mathbf{G}_{jj'} \in \mathbb{R}^{k_j \times k_{j'}}$ that are nonzero only if layer j is connected to layer j'. In the case of convolutional layers with localized spatial support, there is also a repeated implicit structure of nonzero elements as visualized in Figure 4.12.



(a) Convolutional Dictionary

(b) Permuted Dictionary

(c) Convolutional Gram Matrix

(d) Permuted Gram Matrix

Figure 4.12: A visualization of a one-dimensional convolutional dictionary with two input channels, five output channels, and a filter size of three. (a) The filters are repeated over eight spatial dimensions resulting in a block-Toeplitz structure that is revealed through (b) row and column permutations. (c) The corresponding gram matrix can be efficiently computed by considering (d) repeated local filter interactions. This structure allows for lower coherence than an equivalent fully-connected layer with the same number of parameters.

To compute the Gram matrix, we first need to normalize the global induced dictionary $\mathbf{B}$ from Equation 4.16. By stacking the column magnitudes of layer j as the elements in the diagonal matrix $\mathbf{C}_j = \operatorname{diag}(\boldsymbol{c}_j) \in \mathbb{R}^{k_j \times k_j}$, the normalized parameters can be represented as $\tilde{\mathbf{B}}_{ij} = \mathbf{B}_{ij}\mathbf{C}_j^{-1}$. Similarly, the squared norms of the full set of columns in the global dictionary $\mathbf{B}$ are $\mathbf{N}_j^2 = \sum_{i=j}^{l} \mathbf{C}_{ij}^2$. The full normalized dictionary can then be found as $\tilde{\mathbf{B}} = \mathbf{B}\mathbf{N}^{-1}$ where $\mathbf{N}$ is a block diagonal matrix with $\mathbf{N}_j$ as its blocks. The blocks of the Gram matrix $\mathbf{G} = \tilde{\mathbf{B}}^\mathsf{T}\tilde{\mathbf{B}}$ are then given as:

$$\mathbf{G}_{jj'} = \sum_{i=j'}^{l} \mathbf{N}_j^{-1}\mathbf{B}_{ij}^\mathsf{T}\mathbf{B}_{ij'}\mathbf{N}_{j'}^{-1} \tag{4.22}$$

For chain networks, $\mathbf{G}_{jj'} \neq \mathbf{0}$ only when $j' = j+1$, representing the connection between adjacent layers. In this case, the blocks can be simplified as:

$$\mathbf{G}_{jj} = (\mathbf{C}_j^2 + \mathbf{I})^{-\frac{1}{2}}(\mathbf{B}_j^\mathsf{T}\mathbf{B}_j + \mathbf{I})(\mathbf{C}_j^2 + \mathbf{I})^{-\frac{1}{2}} \tag{4.23}$$

$$\mathbf{G}_{j(j+1)} = -(\mathbf{C}_j^2 + \mathbf{I})^{-\frac{1}{2}}\mathbf{B}_{j+1}(\mathbf{C}_{j+1}^2 + \mathbf{I})^{-\frac{1}{2}} \tag{4.24}$$

$$\mathbf{G}_{ll} = \mathbf{B}_l^\mathsf{T}\mathbf{B}_l \tag{4.25}$$

Because the diagonal is removed in the deep frame potential computation, the contribution of $\mathbf{G}_{jj}$ is simply a rescaled version of the local frame potential of layer j. The contribution of $\mathbf{G}_{j(j+1)}$, on the other hand, can essentially be interpreted as rescaled ℓ_2 weight decay where rows are weighted more heavily if the corresponding columns of the previous layer parameters have higher magnitudes. Furthermore, since the global frame potential is averaged over the total number of nonzero elements in $\mathbf{G}$, if a layer has more parameters, then it will be given more weight in this computation. For more general networks with skip connections, however, the summation from Equation 4.22 introduces more complicated interactions; it cannot be determined from local properties of individual layers.

Essentially, the deep frame potential summarizes the structural properties of global dictionary $\mathbf{B}$ induced by the deep network architecture by balancing interactions within each individual layer through local coherence properties and between connecting layers.

While the deep frame potential is a function of parameter values, it's minimum value is determined only by the dictionary structure induced by the deep network architecture. Furthermore, we know that it must be lower bounded by a nonzero constant for overcomplete dictionaries. In this section, we theoretically derive this lower bound for the special case of chain networks and provide intuition for why skip connections increase the capacity for low mutual coherence.

First, observe that a lower bound for the norm of $\mathbf{G}_{j(j+1)}$ from Equation 4.24 cannot be readily attained because the rows and columns are rescaled independently. This means that a lower bound for the norm of $\mathbf{G}$ must be found by jointly considering the entire architecture-induced matrix structure, not simply through summation of its components. To accomplish this, we instead consider the matrix $\mathbf{H} = \tilde{\mathbf{B}}\tilde{\mathbf{B}}^{\mathsf{T}}$, which is full rank and has the same norm as $\mathbf{G}$:

$$\|\mathbf{G}\|_F^2 = \|\mathbf{H}\|_F^2 = \sum_{j=1}^{l} \|\mathbf{H}_{jj}\|_F^2 + 2\sum_{j=1}^{l-1} \left\|\mathbf{H}_{j(j+1)}\right\|_F^2 \tag{4.26}$$

We can then express the individual blocks of $\mathbf{H}$ as:

$$\mathbf{H}_{11} = \mathbf{B}_1(\mathbf{C}_1^2 + \mathbf{I}_{k_1})^{-1}\mathbf{B}_1^{\mathsf{T}} \tag{4.27}$$

$$\mathbf{H}_{jj} = \mathbf{B}_j(\mathbf{C}_j^2 + \mathbf{I})^{-1}\mathbf{B}_j^{\mathsf{T}} + (\mathbf{C}_{j-1}^2 + \mathbf{I})^{-1} \tag{4.28}$$

$$\mathbf{H}_{j(j+1)} = -\mathbf{B}_j(\mathbf{C}_j^2 + \mathbf{I})^{-1} \tag{4.29}$$

In contrast to $\mathbf{G}_{j(j+1)}$ in Equation 4.24, only the columns of $\mathbf{H}_{j(j+1)}$ in Equation 4.29 are rescaled. Since $\tilde{\mathbf{B}}_j$ has normalized columns, its norm can be exactly expressed as:

$$\left\|\mathbf{H}_{j(j+1)}\right\|_F^2 = \sum_{n=1}^{k_j} \left(\frac{c_{jn}}{c_{jn}^2 + 1}\right)^2 \tag{4.30}$$

For the other blocks, we find lower bounds for their norms through the same technique used in deriving the Welch bound, which expresses the minimum mutual coherence for unstructured shallow dictionaries [152]. Specifically, we apply the Cauchy-Schwarz inequality giving $\|\mathbf{A}\|_F^2 \geq r^{-1}(\operatorname{Tr}\mathbf{A})^2$ for positive-semidefinite matrices $\mathbf{A}$ with rank r. Since the rank of $\mathbf{H}_{jj}$ is at most k_{j-1}, we can lower bound the norms of the individual blocks as:

$$\|\mathbf{H}_{11}\|_F^2 \geq \frac{1}{k_0}\left(\sum_{n=1}^{k_1} \frac{c_{1n}^2}{c_{1n}^2 + 1}\right)^2 \tag{4.31}$$

$$\|\mathbf{H}_{jj}\|_F^2 \geq \frac{1}{k_{j-1}}\left(\sum_{n=1}^{k_j} \frac{c_{jn}^2}{c_{jn}^2 + 1} + \sum_{p=1}^{k_{j-1}} \frac{1}{c_{(j-1)p}^2 + 1}\right)^2$$

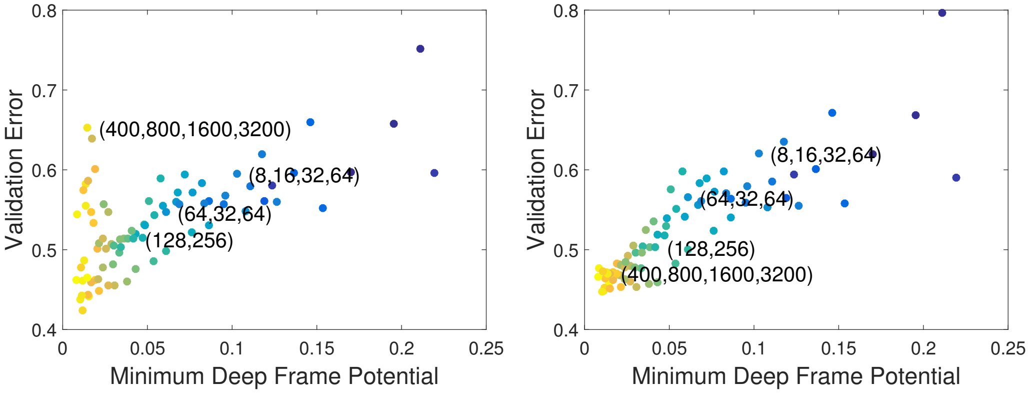
In this case of dense shallow dictionaries, the Welch bound depends only on the data dimensionality and the number of dictionary atoms. However, due to the structure of the architecture-induced dictionaries, the lower bound of the deep frame potential depends on the data dimensionality, the number of layers, the number of units in each layer, the connectivity between layers, and the relative magnitudes between layers. Skip connections increase the number of nonzero elements in the Gram matrix over which to average and also enable off-diagonal blocks to have lower norms.

For more general architectures that lack a simple theoretical lower bound, we instead propose bounding the mutual coherence of the architecture-induced dictionary through empirical minimization of the deep frame potential $F^2(\mathbf{B})$ from Equation 4.21. Frame potential minimization has been used previously to construct finite normalized tight frames due to the lack of suboptimal local minima, allowing for effective optimization using gradient descent [10]. We propose using the minimum deep frame potential of an architecture–which is independent of data and individual parameter instantiations–as a means to compare different architectures. In practice, model selection is performed by choosing the candidate architecture with the lowest minimum frame potential subject to desired modeling constraints such as limiting the total number of parameters.

4.2.3 Experimental Results

In this section, we demonstrate correlation between the minimum deep frame potential and validation error on the CIFAR-10 dataset [74] across a wide variety of fully-connected, convolutional, chain, residual, and densely connected network architectures. Furthermore, we show that networks with skip connections can have lower deep frame potentials with fewer learnable parameters, which is predictive of the parameter efficiency of trained networks.

In Figure 4.13, we visualize a scatter plot of trained fully-connected networks with between three and five layers and between 16 and 4096 units in each layer. The corresponding architectures are shown as a list of units per layer for a few representative examples. The minimum frame potential of each architecture is compared against its validation error after training, and the total parameter count is indicated by color. In Figure 4.13a, some networks with many parameters–indicated by warmer colors–have unusually high error due to the difficulty in training very large networks. In Figure 4.13b, the addition of a deep frame potential regularization term overcomes some of these optimization difficulties for improved parameter efficiency. This results in high correlation between minimum frame potential and validation error. Furthermore, it emphasizes the diminishing



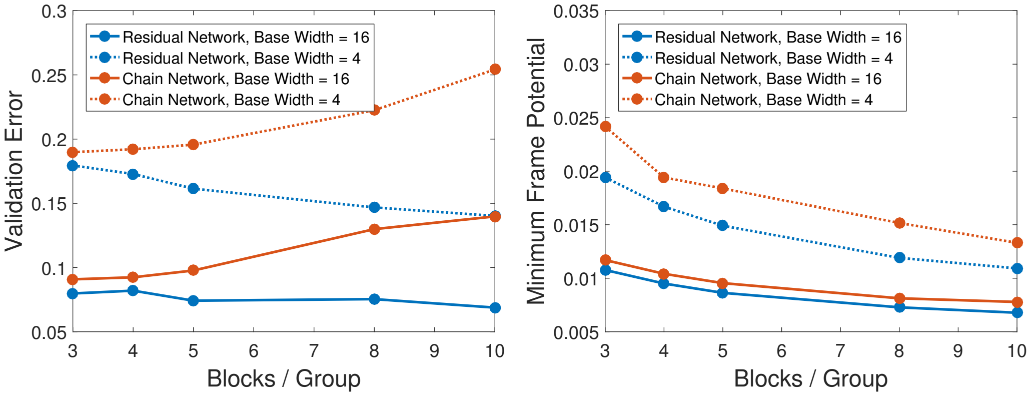
(a) No Regularization

(b) Frame Potential Regularization

Figure 4.13: A large-scale comparison of fully connected deep network architectures with varying depths and widths, where warmer colors indicate more total parameters. (a) Some very large networks cannot be trained effectively resulting in unusually high validation errors. (b) This can be remedied through frame potential regularization, resulting in high correlation between minimum frame potential and validation error.

returns of increasing the size of fully-connected chain networks; after a certain point, adding more parameters does little to reduce both validation error and minimum frame potential.

To evaluate the effects of residual connections [54], we adapt the simplified CIFAR-10 ResNet architecture from [153], which consists of a single convolutional layer followed by three groups of residual blocks with activations as in Equation 4.18. Before the second and third groups, the number of filters is increased by a factor of two and the spatial resolution is decreased by half through average pooling. To compare networks with different sizes, we modify their depths by changing the number of residual blocks in each group from between 2 and 10 and their widths by changing the base number of filters from between 4 and 32. For our experiments with densely connected skip connections [62], we adapt the simplified CIFAR-10 DenseNet architecture from [85]. Like the residual network, it consists of a convolutional layer followed by three groups of of activations as in Equation 4.20 with decreasing spatial resolutions and increasing numbers of filters. Within each group, a dense block is the concatenation of smaller convolu-



(a) Validation Error

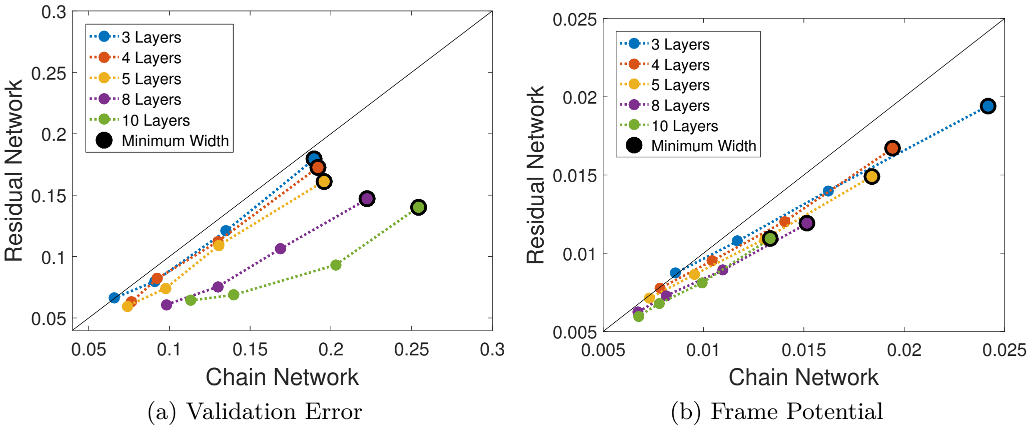
(b) Minimum Frame Potential

Figure 4.14: A visualization of the effect of increasing depth in chain networks and residual networks. Validation error is compared against layer count for two different network widths. (a) In comparison to chain networks, even very deep residual networks can be trained effectively resulting in decreasing validation error. (b) Despite having the same number of total parameters, residual connections also allow for lower minimum frame potentials.

tions that take all previous outputs as inputs with filter numbers equal to a fixed growth rate. Network depth and width are modified by respectively increasing the number of layers per group and the base growth rate from between 2 and 12. Batch normalization [64] was also used in all experiments.

In Figure 4.14, we compare the validation errors and minimum frame potentials of residual networks and comparable chain networks with residual connections removed. In Figure 4.14a, the validation error of chain networks increases for deeper networks while that of residual networks is lower and consistently decreases. This emphasizes the difficulty in training very deep chain networks. In Figure 4.14b, we show that residual connections enable lower minimum frame potentials following a similar trend with respect to increasing model size, again demonstrating correlation between validation error and minimum frame potential.

In Figure 4.15, we compare chain networks and residual networks with exactly the same number of parameters, where color indicates the number of residual blocks per group and connected data points vary from a minimum width of



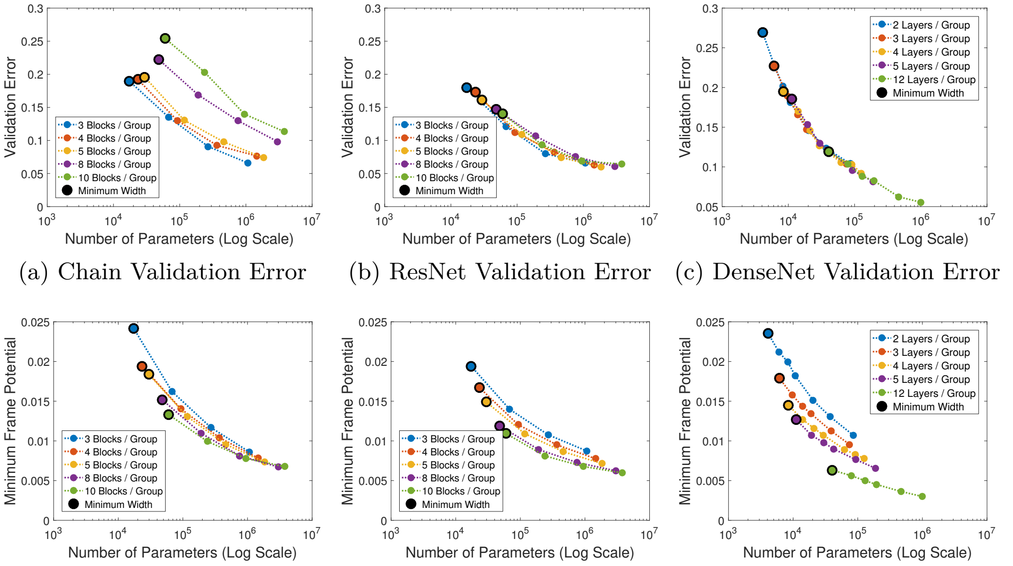
(a) Validation Error

(b) Frame Potential

Figure 4.15: A comparison of (a) validation error and (b) minimum frame potential between residual networks and chain networks. Colors indicate different depths and datapoints are connected in order of increasing widths. The addition of skip connections results in reduced error correlating with frame potential with dense networks showing superior efficiency with increasing depth.

4 base filters to a maximum of 32. The addition of skip connections reduces both validation error and minimum frame potential, as visualized by consistent placement below the diagonal line indicating lower values for residual networks than comparable chain networks. This effect becomes even more pronounced with increasing depths and widths.

In Figure 4.16, we compare the parameter efficiency of chain networks, residual networks, and densely connected networks of different depths and widths. We visualize both validation error and minimum frame potential as functions of the number of parameters, demonstrating the improved scalability of networks with skip connections. While chain networks demonstrate increasingly poor parameter efficiency with depth in Figure 4.16a, the skip connections of ResNets and DenseNets allow for further reducing error with larger network sizes in Figures 4.16b,c. Considering all network families together as in Figure 4.11a, we see that denser connections also allow for lower validation error with comparable numbers of parameters. This trend is mirrored in the minimum frame potentials of Figures 4.16d,e,f which are shown together in Figure 4.11b. Despite some fine variations in behavior across different families of architectures, minimum frame



(a) Chain Validation Error (b) ResNet Validation Error (c) DenseNet Validation Error

(d) Chain Frame Potential (e) ResNet Frame Potential (f) DenseNet Frame Potential

Figure 4.16: A demonstration of the improved scalability of networks with skip connections, where line colors indicate different depths and data points are connected showing increasing widths. (a) Chain networks with greater depths have increasingly worse parameter efficiency in comparison to (b) the corresponding networks with residual connections and (c) densely connected networks with similar size, of which performance scales efficiently with parameter count. This could potentially be attributed to correlated efficiency in reducing frame potential with fewer parameters, which saturates much faster with (d) chain networks than (e) residual networks or (f) densely connected networks.

potential is generally correlated with validation error across network sizes and effectively predicts the increased generalization capacity provided by skip connections.

4.2.4 Conclusion

In this paper, we proposed a technique for comparing deep network architectures by approximately quantifying their implicit capacity for effective data representations, allowing for model selection without requiring a validation dataset. Based upon recent theoretical connections between sparse approximation and deep neural networks, we demonstrated how architectural hyper-parameters such as convolution, depth, width, and skip connections induce different structural properties of the dictionaries in corresponding sparse coding problems. We compared these dictionary structures through lower bounds on their mutual coherence, which is theoretically tied to their capacity for uniquely and robustly representing data via sparse approximation. A theoretical lower bound was derived for chain networks and the deep frame potential was proposed as an empirical optimization objective for constructing bounds for more complicated architectures with skip connections.

Experimentally, we observed a correlation between minimum deep frame potential and validation error across different families of modern architectures with skip connections, including residual networks and densely connected convolutional networks. This suggests a promising direction for future research towards the theoretical analysis and practical construction of deep network architectures derived from connections between deep learning and constrained data decomposition.

Bibliography

[1] Jose Alvarez and Mathieu Salzmann. Learning the number of neurons in deep networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[2] Pablo Arbeláez, Bharath Hariharan, Chunhui Gu, Saurabh Gupta, Lubomir Bourdev, and Jitendra Malik. Semantic segmentation using regions and parts. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[3] Devansh Arpit, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks. In *International Conference on Machine Learning (ICML)*, pages 233–242, 2017.

[4] Mukund Balasubramanian and Eric L Schwartz. The Isomap algorithm and topological stability. *Science*, 295(5552):7–7, 2002.

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

[6] Chenglong Bao, Hui Ji, Yuhui Quan, and Zuowei Shen. Dictionary learning for sparse coding: Algorithms and convergence analysis. *Pattern Analysis and Machine Intelligence (PAMI)*, 38(7):1356–1369, 2016.

[7] Ronen Basri and David W Jacobs. Lambertian reflectance and linear subspaces. *Pattern Analysis and Machine Intelligence (PAMI)*, 25(2):218–233, 2003.

[8] Vasileios Belagiannis and Andrew Zisserman. Recurrent human pose estimation. In *International Conference on Automatic Face & Gesture Recognition (FG)*, 2017.

[9] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

[10] John Benedetto and Matthew Fickus. Finite normalized tight frames. *Advances in Computational Mathematics*, 18(2-4), 2003.

[11] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence (PAMI)*, 35(8):1798–1828, 2013.

[12] Yoshua Bengio and Olivier Delalleau. On the expressive power of deep architectures. In *Algorithmic Learning Theory*. Springer, 2011.

[13] Yoshua Bengio, Jean-François Paiement, Pascal Vincent, Olivier Delalleau, Nicolas Le Roux, and Marie Ouimet. Out-of-sample extensions for LLE, Isomap, MDS, Eigenmaps, and spectral clustering. In *Advances in Neural Information Processing Systems (NIPS)*, 2004.

[14] Michael W Berry, Murray Browne, Amy N Langville, V Paul Pauca, and Robert J Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *Computational Statistics & Data Analysis*, 52(1):155–173, 2007.

[15] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1), 2011.

[16] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. Dsac-differentiable ransac for camera localization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[17] Matthew Brand. Charting a manifold. In *Advances in Neural Information Processing Systems (NIPS)*, 2002.

[18] Myron L Braunstein and George J Andersen. A counterexample to the rigidity assumption in the visual perception of structure from motion. *Perception*, 13(2):213–217, 1984.

[19] Forrest Briggs, Xiaoli Z. Fern, and Raviv Raich. Rank-loss support instance machines for miml instance annotation. In *Knowledge Discovery and Data Mining (KDD), ACM SIGKDD International Conference on*, pages 534–542, 2012.

[20] Andreas Buja, Trevor Hastie, and Robert Tibshirani. Linear smoothers and additive models. *The Annals of Statistics*, pages 453–510, 1989.

[21] R. Cabral, F. De La Torre, J. Costeira, and A. Bernardino. Matrix completion for weakly-supervised multi-label image classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PP(99):1–1, 2014.

[22] Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[23] Joao Carreira, Rui Caseiro, Jorge Batista, and Cristian Sminchisescu. Semantic segmentation with second-order pooling. In *European Conference on Computer Vision (ECCV)*, 2012.

[24] Miguel A Carreira-Perpiñán and Zhengdong Lu. Dimensionality reduction by unsupervised regression. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[25] Miguel A Carreira-Perpiñán and Weiran Wang. Distributed optimization of deeply nested systems. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2014.

[26] Peter G Casazza and Gitta Kutyniok. *Finite frames: Theory and applications.* Springer, 2012.

[27] Caihua Chen, Min Li, Xin Liu, and Yinyu Ye. Extended admm and bcd for nonseparable convex minimization models with quadratic coupling terms: convergence analysis and insights. *Mathematical Programming*, 2017.

[28] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. DeepLab: Semantic image segmentation with deep

convolutional nets, atrous convolution, and fully connected CRFs. *Pattern Analysis and Machine Intelligence (PAMI)*, PP(99), 2017.

[29] Liang-Chieh Chen, Alexander Schwing, Alan Yuille, and Raquel Urtasun. Learning deep structured models. In *International Conference on Machine Learning (ICML)*, 2015.

[30] Ramazan Gokberk Cinbis, Jakob Verbeek, Cordelia Schmid, et al. Multifold mil training for weakly supervised object localization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[31] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3), 1995.

[32] Peter Craven and Grace Wahba. Smoothing noisy data with spline functions. *Numerische Mathematik*, 31(4):377–403, 1978.

[33] Carl De Boor. *A practical guide to splines*, volume 27. Springer-Verlag New York, 1978.

[34] Fernando De la Torre. A least-squares framework for component analysis. *IEEE Transactions Pattern Analysis and Machine Intelligence (PAMI)*, 34(6):1041–1055, 2012.

[35] F. De la Torre and M. J. Black. Robust parameterized component analysis: Theory and applications to 2d facial appearance models. *Computer Vision and Image Understanding*, 91:53–71, 2003.

[36] Misha Denil, Babak Shakibi, Laurent Dinh, Marc'Aurelio Ranzato, and Nando De Freitas. Predicting parameters in deep learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

[37] Steven Diamond, Vincent Sitzmann, Felix Heide, and Gordon Wetzstein. Unrolled optimization with deep priors. *arXiv preprint arXiv:1705.08041*, 2017.

[38] David L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4), 2006.

[39] David L Donoho and Michael Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via 1 minimization. *Proceedings of the National Academy of Sciences*, 100(5):2197–2202, 2003.

[40] David L. Donoho, Michael Elad, and Vladimir N. Temlyakov. Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Transactions on Information Theory*, 52(1), 2005.

[41] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *International Conference on Computer Vision (ICCV)*, 2015.

[42] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[43] Michael Elad. *Sparse and redundant representations: from theory to applications in signal and image processing.* Springer Science & Business Media, 2010.

[44] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *Journal of Machine Learning Research (JMLR)*, 20(55), 2019.

[45] Roland W Fleming, Ron O Dror, and Edward H Adelson. Real-world illumination and the perception of surface reflectance properties. *Journal of vision*, 3(5):3–3, 2003.

[46] Brendan J. Frey and Nebojsa Jojic. Transformed component analysis: Joint estimation of spatial transformations and image components. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 1999.

[47] Ankit Gandhi, Karteek Alahari, and C.V. Jawahar. Decomposing bag of words histograms. In *International Conference on Computer Vision (ICCV)*, pages 305–312. IEEE, 2013.

[48] Nicolas Gillis. Sparse and unique nonnegative matrix factorization through data preprocessing. *Journal of Machine Learning Research (JMLR)*, 13(November):3349–3386, 2012.

[49] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[50] Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *International Conference on Machine Learning (ICML)*, 2010.

[51] Benjamin Haeffele, Eric Young, and Rene Vidal. Structured low-rank matrix factorization: Optimality, algorithm, and applications to image processing. In *International Conference on Machine Learning (ICML)*, 2014.

[52] Trevor Hastie and Werner Stuetzle. Principal curves. *Journal of the American Statistical Association*, 84(406):502–516, 1989.

[53] Soren Hauberg, Aasa Feragen, and Michael J. Black. Grassmann averages for scalable robust pca. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[54] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[55] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision (ECCV)*, 2016.

[56] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[57] Marc Henniges, Richard E. Turner, Maneesh Sahani, Julian Eggert, and Jörg Lücke. Efficient occlusive components analysis. *Journal of Machine Learning Research*, 15:2689–2722, 2014.

[58] Minh Hoai, Lorenzo Torresani, Fernando De la Torre, and Carsten Rother. Learning discriminative localization from weakly labeled data. *Pattern Recognition*, 47(3):1523–1534, 2014.

[59] Ian P Howard. *Seeing in depth, Vol. 1: Basic mechanisms.* University of Toronto Press, 2002.

[60] Patrik O. Hoyer. Non-negative matrix factorization with sparseness constraints. *The Journal of Machine Learning Research*, 5:1457–1469, 2004.

[61] Peiyun Hu and Deva Ramanan. Bottom-up and top-down reasoning with hierarchical rectified gaussians. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[62] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[63] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[64] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, pages 448–456, 2015.

[65] Adrian Ion, João Carreira, and Cristian Sminchisescu. Probabilistic joint image segmentation and labeling by figure-ground composition. *International Journal of Computer Vision (IJCV)*, 2014.

[66] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*, pages 675–678. ACM, 2014.

[67] Nebojsa Jojic and Brendan J. Frey. Learning flexible sprites in video layers. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[68] Armand Joulin, Francis Bach, and Jean Ponce. Discriminative clustering for image co-segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[69] Christian Jutten and Jeanny Herault. Blind separation of sources, part i: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, 24(1):1–10, 1991.

[70] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *International Conference on Learning Representations (ICLR)*, 2014.

[71] Jelena Kovačević, Amina Chebira, et al. An introduction to frames. *Foundations and Trends in Signal Processing*, 2(1), 2008.

[72] Kenneth Kreutz-Delgado, Joseph F Murray, Bhaskar D Rao, Kjersti Engan, Tai Sing Lee, and Terrence J Sejnowski. Dictionary learning algorithms for sparse representation. *Neural Computation*, 15(2):349–396, 2003.

[73] Alex Krizhevsky et al. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[74] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[75] Roland Kwitt and Peter Meerwald. Salzburg texture image database (STex). http://wavelab.at/sources/STex/.

[76] James T. Kwok and Ivor W. Tsang. The pre-image problem in kernel methods. In *International Conference on Machine Learning (ICML)*, pages 408–415, 2003.

[77] Neil Lawrence. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 2005.

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

[79] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

[80] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755), 1999.

[81] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *International Conference on Machine Learning (ICML)*, 2009.

[82] Kuang-Chih Lee, Jeffrey Ho, and David J Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *Pattern Analysis and Machine Intelligence (PAMI)*, 27(5):684–698, 2005.

[83] Ke Li, Bharath Hariharan, and Jitendra Malik. Iterative instance segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[84] Stan Z Li, XinWen Hou, HongJiang Zhang, and QianSheng Cheng. Learning spatially localized, parts-based representation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[85] Yixuan Li. Tensorflow densenet. https://github.com/YixuanLi/densenet-tensorflow, 2018.

[86] Chen-Hsuan Lin and Simon Lucey. Inverse compositional spatial transformer networks. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[87] Ce Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing via label transfer. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(12), Dec 2011.

[88] Tongliang Liu, Dacheng Tao, and Dong Xu. Dimensionality-dependent generalization bounds for k-dimensional coding schemes. *Neural computation*, 2016.

[89] Yang Liu, Jing Liu, Zechao Li, Jinhui Tang, and Hanqing Lu. Weakly-supervised dual clustering for image semantic segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[90] David Lopez-Paz, Suvrit Sra, Alex Smola, Zoubin Ghahramani, and Bernhard Schölkopf. Randomized nonlinear component analysis. In *International Conference on Machine Learning (ICML)*, 2014.

[91] David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[92] Tom Lyche and Knut Mørken. *Spline methods*. Department of Mathematics, University of Oslo, 2008.

[93] Fangchang Ma and Sertac Karaman. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *International Conference on Robotics and Automation (ICRA)*, 2018.

[94] David Marr and Herbert Keith Nishihara. Representation and recognition of the spatial organization of three-dimensional shapes. *Proceedings of the Royal Society of London B*, 200(1140):269–294, 1978.

[95] Martin Marsden and I. J. Schoenberg. *On Variation Diminishing Spline Approximation Methods*, pages 247–268. Birkhäuser Boston, Boston, MA, 1988.

[96] Andre Martins and Ramon Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International Conference on Machine Learning (ICML)*, 2016.

[97] Sebastian Mika, Bernhard Schölkopf, Alexander J Smola, Klaus-Robert Müller, Matthias Scholz, and Gunnar Rätsch. Kernel pca and de-noising in feature spaces. In *Advances in Neural Information Processing Systems (NIPS)*, 1999.

[98] Cisse Moustapha, Bojanowski Piotr, Grave Edouard, Dauphin Yann, and Usunier Nicolas. Parseval networks: Improving robustness to adversarial examples. *arXiv preprint arXiv:1704.08847*, 2017.

[99] Calvin Murdock, MingFang Chang, and Simon Lucey. Deep component analysis via alternating direction neural networks. In *European Conference on Computer Vision (ECCV)*, 2018.

[100] Calvin Murdock, Ming-Fang Chang, and Simon Lucey. Deep component analysis via alternating direction neural networks. In *European Conference on Computer Vision (ECCV)*, 2018.

[101] Calvin Murdock and Fernando De la Torre. Semantic component analysis. In *International Conference on Computer Vision (ICCV)*, 2015.

[102] Calvin Murdock and Fernando De la Torre. Additive component analysis. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[103] Calvin Murdock and Fernando De la Torre. Approximate grassmannian intersections: Subspace-valued subspace learning. In *International Conference on Computer Vision (ICCV)*, 2017.

[104] Calvin Murdock and Simon Lucey. Dataless model selection with the deep frame potential. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[105] Hariharan Narayanan and Sanjoy Mitter. Sample complexity of testing the manifold hypothesis. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

[106] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision (ECCV)*, 2012.

[107] Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library (COIL-20). Technical report, Technical Report CUCS-005-96, 1996.

[108] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[109] M. H. Nguyen, L. Torresani, F. De la Torre, and C. Rother. Weakly supervised discriminative localization and classification: a joint learning process. In *International Conference on Computer Vision (ICCV)*, 2009.

[110] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *International Conference on Computer Vision (ICCV)*, 2015.

[111] Roman Novak et al. Sensitivity and generalization in neural networks: an empirical study. In *International Conference on Learning Representations (ICLR)*, 2018.

[112] Bruno A Olshausen et al. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

[113] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Is object localization for free? - weakly-supervised learning with convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[114] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Is object localization for free? – weakly-supervised learning with convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[115] Umut Ozertem and Deniz Erdogmus. Locally defined principal curves and surfaces. *Journal of Machine Learning Research (JMLR)*, 12:1249–1286, 2011.

[116] George Papandreou, Liang-Chieh Chen, Kevin Murphy, and Alan L Yuille. Weakly-and semi-supervised learning of a dcnn for semantic image segmentation. *arXiv preprint arXiv:1502.02734*, 2015.

[117] Vardan Papyan, Yaniv Romano, and Michael Elad. Convolutional neural networks analyzed via convolutional sparse coding. *Journal of Machine Learning Research (JMLR)*, 18(83), 2017.

[118] Neal Parikh, Stephen Boyd, et al. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3), 2014.

[119] Ankit B Patel, Minh Tan Nguyen, and Richard Baraniuk. A probabilistic framework for deep learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[120] Deepak Pathak, Philipp Krahenbuhl, and Trevor Darrell. Constrained convolutional neural networks for weakly supervised segmentation. In *International Conference on Computer Vision (ICCV)*, 2015.

[121] Pedro O Pinheiro and Ronan Collobert. From image-level to pixel-level labeling with convolutional networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[122] Tomaso Poggio, Vincent Torre, and Christof Koch. Computational vision and regularization theory. In *Readings in Computer Vision*, pages 638–643. Elsevier, 1987.

[123] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

[124] Yaniv Romano, Aviad Aberdam, Jeremias Sulam, and Michael Elad. Adversarial noise attacks of deep learning architectures-stability analysis via sparse modeled signals. *Journal of Mathematical Imaging and Vision*, 2018.

[125] Carsten Rother, Vladimir Kolmogorov, Tom Minka, and Andrew Blake. Cosegmentation of image pairs by histogram matching – incorporating a global constraint into MRFs. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[126] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[127] Sam T Roweis, Lawrence K Saul, and Geoffrey E Hinton. Global coordination of local linear models. *Advances in Neural Information Processing Systems (NIPS)*, 2002.

[128] Olga Russakovsky, Amy L. Bearman, Vittorio Ferrari, and Li Fei-Fei. What's the point: Semantic segmentation with point supervision. *ArXiv preprint arXiv:1506.02106*, 2015.

[129] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[130] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *European Conference on Computer Vision (ECCV)*, 2006.

[131] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[132] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

[133] Josef Sivic, Bryan C. Russell, Alexei A. Efros, Andrew Zisserman, and William T. Freeman. Discovering objects and their location in images. In *International Conference on Computer Vision (ICCV)*, 2005.

[134] Jeremias Sulam, Aviad Aberdam, Amir Beck, and Michael Elad. On multi-layer basis pursuit, efficient algorithms and convolutional neural networks. *Pattern Analysis and Machine Intelligence (PAMI)*, 2019.

[135] Jeremias Sulam, Vardan Papyan, Yaniv Romano, and Michael Elad. Multi-layer convolutional sparse modeling: Pursuit and dictionary learning. *arXiv preprint arXiv:1708.08705*, 2017.

[136] Jian Sun, Huibin Li, Zongben Xu, et al. Deep admm-net for compressive sensing mri. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[137] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014.

[138] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

[139] Gavin Taylor, Ryan Burmeister, Zheng Xu, Bharat Singh, Ankit Patel, and Tom Goldstein. Training neural networks without gradients: A scalable admm approach. In *International Conference on Machine Learning (ICML)*, 2016.

[140] Joshua B. Tenenbaum, Vin De Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[141] Joseph Tighe and Svetlana Lazebnik. Superparsing: Scalable nonparametric image parsing with superpixels. In *European Conference on Computer Vision (ECCV)*, 2010.

[142] James T Todd and Francene D Reichel. Ordinal structure in the visual perception and cognition of smoothly curved surfaces. *Psychological Review*, 96(4):643, 1989.

[143] Shubham Tulsiani, Tinghui Zhou, Alexei A Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[144] Matthew A. Turk and Alex P. Pentland. Face recognition using eigenfaces. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 1991.

[145] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85, 2008.

[146] Vladimir N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and Its Applications*, XVI(2), 1971.

[147] A. Vezhnevets, V. Ferrari, and J.M. Buhmann. Weakly supervised semantic segmentation with a multi-image model. In *International Conference on Computer Vision (ICCV)*, 2011.

[148] A. Vezhnevets, V. Ferrari, and J.M. Buhmann. Weakly supervised structured output learning for semantic segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[149] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research (JMLR)*, 11:3371–3408, 2010.

[150] Xiaolong Wang, David Fouhey, and Abhinav Gupta. Designing deep networks for surface normal estimation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[151] Larry Wasserman. *All of nonparametric statistics.* Springer Science & Business Media, 2006.

[152] Lloyd Welch. Lower bounds on the maximum cross correlation of signals. *IEEE Transactions on Information theory*, 20(3), 1974.

[153] Yuxin Wu et al. Tensorpack. https://github.com/ppwwyyxx/tensorpack/blob/master/examples/ResNet/cifar10-resnet.py, 2016.

[154] Jia Xu, Alexander G. Schwing, and Raquel Urtasun. Tell me what you see and i will show you where it is. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[155] Yangyang Xu and Wotao Yin. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on imaging sciences*, 6(3):1758–1789, 2013.

[156] Amir R Zamir, Te-Lin Wu, Lin Sun, William Shen, Jitendra Malik, and Silvio Savarese. Feedback networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[157] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations (ICLR)*, 2017.

[158] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations (ICLR)*, 2017.

[159] Wei Zhang, Sheng Zeng, Dequan Wang, and Xiangyang Xue. Weakly supervised semantic segmentation for social images. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.