AFRL-RI-RS-TR-2021-077

SPIDER: SUBSPACE PRIMITIVES THAT ARE INTERPRETABLE AND DIVERSE

UNIVERSITY OF MICHIGAN

APRIL 2021

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nations. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2021-077 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE CHIEF ENGINEER:

/ S /
MICHAEL J. MANNO
Work Unit Manager

/ S /
SCOTT D. PATRICK
Deputy Chief,
Intelligence Systems Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
APRIL 2021	FINAL TECHNICAL REPORT	APR 2017 – JUL 2020

4. TITLE AND SUBTITLE

SPIDER: SUBSPACE PRIMITIVES THAT ARE INTERPRETABLE AND DIVERSE

5a. CONTRACT NUMBER
FA8750-17-2-0125

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62702E

6. AUTHOR(S)

Jason Corso
Laura Balzano

5d. PROJECT NUMBER
D3ME

5e. TASK NUMBER
00

5f. WORK UNIT NUMBER
10

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
REGENTS OF THE UNIVERSITY OF MICHIGAN
OFFICE OF RESEARCH AND SPONSORED PROJECTS
503 THOMPSON ST
ANN ARBOR MI 48109-1340

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RIED
26 Electronic Pkwy
Rome, NY 13441

DARPA/I2O
675 North Randolph Street
Arlington, VA 22203-2114

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSOR/MONITOR'S REPORT NUMBER
AFRL-RI-RS-TR-2021-077

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Contribute three classes of primitives to the D3M program that are not only innovative machine learning methods but also provide a certain transparent mechanism by which the domain expert, data science-novice can naturally incorporate their expert knowledge to yield a better model. The three classes of primitives each attack the discovery of subspaces within the input data space with different tools: (1) multimodal embeddings leverage sparse models, (2) invariance discovery primitives, and (3) subspace clustering primitives.

15. SUBJECT TERMS

Multimodal Embeddings, Invariances, Subspace Clustering, Space Diverse

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			**MICHAEL J. MANNO**
U	U	U	UU	61	19b. TELEPHONE NUMBER (*Include area code*) **N/A**

Table of Contents

List of Figures

List of Tables

Summary

1.1 Basic Programmatic Data

SPIDER: Subspace Primitives that are Interpretable and Diverse

Award Number: FA8750-17-2-0125

Dates: 4/28/2017 - 7/31/2020

Organization: University of Michigan

Principal Investigator: Dr. Jason Corso

Co-Investigator: Dr. Laura Balzano

DARPA Program: Data-Driven Discovery of Models

1.2 Summary of Research Objectives

1.2.1 Problem Description

Subject experts often develop intuitions by working with experimental data or through extensive modeling and analysis of the domain. The current state-of-the-art in machine learning does not adequately provide a capability to incorporate these intuitions into data-driven models, instead relying on the belief that with more data and further expressivity of the model, modern machine learning methods such as deep neural network are sufficient for all problems. There are two fundamental problems with this current paradigm: 1) Large deep neural networks such as AlexNet use millions of parameters and as a consequence require a significant amount of data. However, it is a non-trivial and expensive task to collect good annotated data for all problems. 2) Despite the large data, there is enough evidence that explicitly accounting for invariances (well understood by subject matter experts) reduces the error in predictions, makes the learning problem well-behaved and generalizes well over novel inputs [3, 4].

1.2.2 Summary of Research Activities

We focused our work on subspace primitives that could be applied across the D3M problem space. These primitives were grouped into four types: Data Featurization Primitives, Multimodal Embeddings, Invariances, and Clustering and Dimensionality Reduction.

We delivered software that implemented innovations in these types of primitives; the software was stratified according to 8 different software packages and shared with the D3M performers. We produced 3 journal articles, 7 conference publications and 6 technical reports.

In the later sections, we talk about these activities in significantly more detail.

Introduction

2.1 Research Goals

The D3M program built a human-interactive automated machine learning system based on a discoverable collection of modeling and machine learning primitives. Our D3M effort was focused on developing several of those primitives so that data of any type and the related machine learning tasks can be handled by the D3M system. We developed data featurization primitives, supervised primitives for multimodal data, supervised primitives that exploit data invariants, unsupervised clustering primitives, and dimensionality reduction primitives.

Data Featurization When raw data are brought to the D3M system there must be an infrastructure in place to convert these data into forms expected by other modeling and learning primitives in the system. Specifically, for our own subspace modeling primitives, it is necessary to convert data into a vector space representation of some sort to leverage these subspace representations. We led the featurization effort for the program by providing a class-based API for implementing featurization as well as the function-based API spec for allowing TA2 teams to leverage various featurization primitives. These efforts started with our Michigan Platform for Activity Classification in Tensorflow (M-PACT), where we achieved near state-of-the-art in activity recognition. Much of our time over the course of the program was spent on developing associated primitives and updating their APIs to match D3M's specifications.

Multimodal Embeddings Multimodal methods allow for integration of various data sources, such as Twitter feeds, RGB video, audio, text, and analytical data to find commonalities and as a consequence enable inference by exploiting information across modalities. We focused on the generation of a specific sensory modality by establishing correspondences and learned common representation across modalities. We explored multimodal embeddings using the language of dictionary learning, neural networks, and multivariate function approximation.

Invariances Representation of pertinent invariances reduces the degrees of freedom in a model and yields consistent output that obey the structure of the problem brought to bear by the intuition of the expert domain scientist. For example, a video understanding scheme for cooking videos should capture the temporal rate invariances in chopping an onion between experts and novice chefs.

Clustering and dimensionality reduction Linear subspace models are commonly used for dimensionality reduction, but they can be limited in their inference power. During the course of SPIDER project we have added primitives to allow more flexible representation of the data: unions

(or mixtures) of subspace models that combine the low-dimensional representation with a clustering representation, and time-varying subspace models that can handle temporal data such as video. Both are important in large complex systems that are being monitored and modeled over time.

2.2 Research Accomplishments

Using the themes mentioned in Section 2.1 as building blocks we proposed novel algorithms to solve several problems in the general areas of these themes. Furthermore, we produced efficient implementations of these algorithms, which have appeared as D3M primitives over the course of the program. Looking at a finer-grained level the themes in Section 2.1 can be further subdivided into the following topics associated with the corresponding D3M primitive packages:

1. Featurization (`d3m.primitives.spider.featurization`)
2. Clustering (`d3m.primitives.spider.cluster`)
3. Distance metric learning (`d3m.primitives.spider.distance`)
4. Robust PCA (`d3m.primitives.spider.dimensionality_reduction`)
5. Multimodal analysis (`d3m.primitives.spider.featurization`, `d3m.primitives.spider.preprocessing`, `d3m.primitives.spider.supervised_learning`)
6. Deep network inference (`d3m.primitives.spider.featurization`, `d3m.primitives.spider.preprocessing`, `d3m.primitives.spider.supervised_learning`)
7. Regression (`d3m.primitives.spider.supervised_learning`)
8. Tensor recovery (`d3m.primitives.spider.unsupervised_learning`)

In addition to the above-mentioned primitives, we worked on several machine learning algorithms that would have been useful primitives if D3M had gone in a different direction. Finally, our work also resulted in 3 journal articles, 7 conference publications, and 6 technical reports that have been made public through ArXiv repository. We enumerate these publications in Section 3.3.

2.3 Impact

The subspace as a model primitive is ubiquitous in engineering and science, with examples ranging from array processing [5] and communications [6], to computer vision [7], environmental monitoring [8, 9], causal inference [10], and canonical correlation analysis [11]. Our specific focus on three subspace problems—dictionary learning, invariance subspaces in deep learning, and subspace clustering—touches on tools used by nearly every data scientist, research lab, and corporation that uses machine learning. Whereas contemporary automated statistics tools such as STATA

offer non-experts the ability to use statistics as a black box, these tools are limited and there is danger in using them blindly, e.g. p-hacking and overfitting. Systems resulting from SPIDER and the D3M program will bring more sophisticated tools to a greater user base while at the same time providing more transparency and adaptation with user-in-the-loop feedback and adaptive querying.

Methods, Assumptions and Procedures

3.1 Detailed Description of Technical Approach

The subspace primitives we have developed are each specifications of the following model:

$$\begin{bmatrix} X & Y \end{bmatrix} = \begin{bmatrix} U_1 W_1 & U_2 W_2 \end{bmatrix} \ ,$$

where X and Y may be data from different clusters, different pools in deep learning, or different modes in multimodal embeddings. The data are represented by a subspace U and weights W. For the multimodal embeddings, the weights are constrained to be sparse but common. For subspace clustering, the subspaces must have angular separation. For invariance, both the subspace and weights capture the constrained behavior. We use typical algebraic notation here but note that the subspaces could be some atypical vector space such as a Lie algebra and the operator on U and W could be non-linear. For example, in a deep network, the U represents a composition of sequential linear (*e.g.*, matrix multiplication) and non-linear (*e.g.*, max-pooling and sigmoid scaling) operations.

The technical challenges for subspace primitives range from fundamental theory to implementation robustness. Further theory for subspace clusters, multimodal embeddings, and invariance provide new understanding of the benefits and limitations of constraining a subspace model using domain knowledge or for purposes of interpretability. From the practical side, the learning process must be robust to noisy, poorly conditioned, and missing measurements as well as data that are unbalanced for class types or modal types. And as we have focused each of our primitives on different underlying machine learning technologies (dictionary learning, deep learning, and clustering), we pursued promising ideas for cross-pollination applicable to multiple primitives, *e.g.,* using sample grouping techniques for invariance to inform sample groupings in clustering and vice versa.

Multimodal Embeddings We solved the problem of representing the common information content in various modalities, as it also offers a window into understanding the novel information contained in each modality. We proposed a sparse representation as it is not only natural in typical high-dimensional data scenarios, such as team-interaction and bioinformatics samples, but it allows interpretability by the domain expert through the subspace defined by sparse set of activated bases. It has been successfully used in various learning tasks [12, 13]. Furthermore, there is increasing physiological evidence that humans use sparse coding in sensory representation [14, 15, 16]. The need for sparse coding is supported by the hypothesis of using the least energy possible in neuronal excitation to represent input sensory data.

Invariance We developed a class of D3M primitives that explicitly incorporate invariance into deep neural networks (DNNs). Our planned invariance primitives have been incorporated into the modern deep learning architectures as novel layers that require no problem-specific implementation but is rather selectable automatically as layers into the network.

We worked on explicit invariance layers in DNNs rather than conventional ways of introducing invariance, which include *data augmentation* [17, 18] and *network augmentation* [19, 20, 21]. These conventional methods expect the DNN to implicitly discover invariance by introducing variation in the training data or training variation-specific networks, respectively; both require significant data and present no theoretical guarantees on the invariances learned.

Subspace Clustering The union-of-subspaces (UoS) model is a flexible generalization of the low-rank subspace embedding. It includes single and multiple subspace approximations as well as manifold approximation models. There are several provably efficient algorithms to learn subspaces when data is observed fully [22, 23] as well as for the case when have missing entries in data [24]. We have implemented these primitives for the D3M program, extended them to novel data types such as mixed-type tabular data, developed and implemented versions of these primitives that are robust to outliers, and developed and implemented parameter tuning algorithms to distinguish which model is appropriate for data at hand (i.e., identify whether to use subspace or regular clustering model and identify the relevant parameters such as cluster number and subspace dimension).

3.2 Comparison with Current Technology

Subject experts often develop intuition by working with experimental data or through extensive modeling and analysis of the domain. The current state-of-the-art in machine learning and, specifically, subspace modeling, does not adequately provide a capability to incorporate these intuitions into data-driven models, instead relying on the belief that with more data and further expressivity of the model, modern machine learning methods such as deep neural networks are sufficient for all problems. There are two fundamental issues with this current paradigm: 1) Large deep neural networks use millions of parameters and as a consequence require significant amounts of labeled data. However, it is non-trivial and expensive to collect good labeled data for all problems. 2) Despite the large data, there is enough evidence that explicitly accounting for subspace structure, such as invariances, which are well understood by subject matter experts, reduces the error in predictions, makes the learning problem well-behaved and generalizes well over novel inputs [4, 25].

Multimodal Embeddings and Invariance Primitives The multimodal embeddings and invariance primitives developed through the course of project have helped relax these restrictive assumptions within the D3M system. The multimodal primitives have moved beyond the standard encoder-decoder structure, which still embodies these assumptions, and into representations that leveraged the information across modalities as weak labels. The invariance primitives explicitly learned to enforce certain subspaces present in the problem while not requiring the domain user to know the mathematical constructs.

Subspace Clustering Our technical approach differs from current technology in that it leverages the evidence accumulation clustering framework for subspace clustering algorithm development. This ensemble learning framework, analogous to boosting in classification, allows us to improve existing state-of-the-art to be more robust to missing and corrupted data and multiple data types. Additionally, it has led to algorithms with better computational and memory requirements. Existing provable algorithms [22] are computationally burdensome, requiring the solution of a large semi-definite program (SDP) for each data point in the dataset. Versions of these algorithm using alternating direction method of multipliers (ADMM) have also been developed but still require an optimization to be executed for every data point. In contrast, we have developed an algorithm using ensembles of the simple K-subspaces algorithm, iterating on two simple steps: first we estimate clusters using subspace projections, and then we re-estimate subspaces by finding the top principal components of each cluster. These operations have very efficient implementations and yet retain the same guarantees as current technology.

3.3 Deliverables

3.3.1 Deliverables Description

In addition to detailed technical reports on our work, our deliverables centered on software updates to the `spider` git repository (`https://gitlab.datadrivendiscovery.org/michigan/spider`), which integrates with the larger D3M universe and contains our primitives in a pip-installable Python package, along with unit tests and functional example pipelines using the D3M pipeline schema. We have also made available a public repository for our primitives at `https://github.com/dvdmjohnson/d3m_michigan_primitives`.

The following Michigan primitives are currently in full compliance with the latest API standards and available in the `spider` repository (listed here according to their global D3M entry points):

- `d3m.primitives.spider.cluster.EKSS`
- `d3m.primitives.spider.cluster.KSS`
- `d3m.primitives.spider.cluster.SSC_ADMM`
- `d3m.primitives.spider.cluster.SSC_CVX`
- `d3m.primitives.spider.cluster.SSC_OMP`
- `d3m.primitives.spider.supervised_learning.owl`
- `d3m.primitives.spider.unsupervised_learning.grasta`

We developed additional primitives that were part of the D3M universe at one point, but did not end up in our final contribution due to either changes in personnel, which compromised the continued reliability of these primitives, or complications from upgrading to meet D3M's dependencies:

- d3m.primitives.spider.dimensionality_reduction.go_dec
- d3m.primitives.spider.dimensionality_reduction.pcp_ialm
- d3m.primitives.spider.dimensionality_reduction.rpca_lbd
- d3m.primitives.spider.distance.rfd
- d3m.primitives.spider.featurization.audio_featurization
- d3m.primitives.spider.featurization.audio_slicer
- d3m.primitives.spider.featurization.i3d
- d3m.primitives.spider.featurization.logmelspectrogram
- d3m.primitives.spider.featurization.vgg16
- d3m.primitives.spider.preprocessing.trecs
- d3m.primitives.spider.supervised_learning.goturn
- d3m.primitives.spider.unsupervised_learning.grasta_masked
- d3m.primitives.spider.unsupervised_learning.grouse

Additionally, we produced the following technical reports on D3M-associated work:

- Grassmannian Optimization for Online Tensor Completion and Tracking in the t-SVD Algebra (https://arxiv.org/abs/2001.11419).
- HyperCon: Image-To-Video Model Transfer for Video-To-Video Translation Tasks (https://arxiv.org/abs/1912.04950).
- M-PACT: Michigan Platform for Activity Classification in Tensorflow (https://arxiv.org/pdf/1804.05879.pdf).
- T-RECS: Training for Rate-Invariant Embeddings by Controlling Speed for Action Recognition (https://arxiv.org/abs/1803.08094).
- Deep Unsupervised Clustering Using Mixture of Autoencoders (https://arxiv.org/abs/1712.07788).
- Deep Net Triage: Assessing the Criticality of Network Layers by Structural Compression (https://arxiv.org/abs/1801.04651).
- A Gene Filter for Comparative Analysis of Single-Cell RNA-Sequencing Trajectory Datasets (https://www.biorxiv.org/content/10.1101/637488v1).
- Optimally Weighted PCA for High-dimensional Heteroscedastic Data (https://arxiv.org/abs/1810.12862).

- Subspace Clustering Using Ensembles of K-Subspaces (https://arxiv.org/pdf/1709.04744.pdf).

Furthermore, we published the following papers:

- Memory-efficient Splitting Algorithms for Large-Scale Sparsity Regularized Optimization (https://arxiv.org/abs/1904.00423) Proceedings of the International Conference on Image Formation in X-Ray Computed Tomography (CT Meeting), 2018.
- A Robust Algorithm for Online Switched System Identification (https://www.sciencedirect.com/science/article/pii/S2405896318318111), Proceedings of the Symposium on System Identification (SYSID), 2018.
- Learning Dictionary-Based Unions of Subspaces for Image Denoising (https://ieeexplore.ieee.org/document/8553117), Proceedings of the European Signal Processing Conference (EUSIPCO), 2018.
- Streaming PCA and Subspace Tracking: The Missing Data Case (https://ieeexplore.ieee.org/document/8417980), Proceedings of the IEEE, 2018.
- Improving K-Subspaces via Coherence Pursuit (https://ieeexplore.ieee.org/document/8457298), IEEE Journal of Selected Topics in Signal Processing, 2018.
- End-to-End Dense Video Captioning with Masked Transformer (https://arxiv.org/abs/1804.00819), IEEE Conference on Computer Vision and Pattern Recognition, 2018.
- Learning to Share: Simultaneous Parameter Tying and Sparsification in Deep Learning (https://openreview.net/forum?id=rypT3fb0b&noteId=HkeRS68Ef), International Conference on Learning Representations, 2018.
- Streaming Principal Component Analysis from Incomplete Data (https://arxiv.org/abs/1612.00904), Journal for Machine Learning Research, 2019.
- Supervised Principal Component Analysis Via Manifold Optimization (https://ieeexplore.ieee.org/document/8755587), IEEE Data Science Workshop, 2019.
- Dynamic Graph Modules for Modeling Object-Object Interactions in Activity Recognition. (https://arxiv.org/abs/1812.05637), British Machine Vision Conference, 2019.
- A Weakly Supervised Multi-task Ranking Framework for Actor-Action Semantic Segmentation (https://link.springer.com/article/10.1007/s11263-019-01244-7), International Journal of Computer Vision, 2019.
- Panoramic Video Separation with Online Grassmannian Robust Subspace Estimation (https://openaccess.thecvf.com/content_ICCVW_2019/papers/RSL-CV/Gilman_Panoramic_Video_Separation_with_Online_Grassmannian_Robust_Subspace_Estimation_ICCVW_2019_paper.pdf), International Conference on Computer Vision (ICCV) Workshop, 2019.

- Rethinking Curriculum Learning With Incremental Labels and Adaptive Compensation (https://arxiv.org/pdf/2001.04529.pdf), British Machine Vision Conference, 2020.
- Online Tensor Completion and Free Submodule Tracking with the t-SVD, (https://ieeexplore.ieee.org/document/9053199), IEEE International Conference on Acoustics, Speech, and Signal Processing, 2020.
- A Temporally-Aware Interpolation Network for Video Frame Inpainting (https://ieeexplore.ieee.org/document/8892406), IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020.
- Unified Vision-Language Pre-Training for Image Captioning and VQA (https://arxiv.org/abs/1909.11059), AAAI Conference on Artificial Intelligence, 2020.
- Clustering Quality Metrics for Subspace Clustering (https://doi.org/10.1016/j.patcog.2020.107328) Pattern Recognition, 2020.

Results and Discussion

In this part of the report, we describe the technical contributions we made to the D3M program. Our team, as part of TA1, focused on creating data science and machine learning primitives that aligned with our overall research proposal, our team's strengths, and applications that related to those strengths. Many of those applications did not become a focus of the D3M program, such as video processing, highly unsupervised and large-scale learning problems, as well as streaming computation for large scale problems more generally.

4.1 Guiding Objectives

In our original proposal we identified seven distinct technical goals:

1. Multimodal subspace learning on large-scale data
2. Invariance layers within common feed-forward classification and regression networks
3. Subspace clustering with model selection
4. Connect different types of subspace primitives, and explore joint optimization and model fitting over combinations of them
5. Software development, including primitive implementations and integration other D3M teams
6. Testing of primitives on existing and novel datasets
7. Program participation, through PI meetings, calls and coordination with other D3M teams

Using these goals as a guide we made several technical breakthroughs throughout the span of this project. We elaborate on these technical accomplishments in Sections 4.2–4.13.

4.2 Featurization Primitives

4.2.1 Audio featurization primitive

We implemented and submitted a deep-learning-based primitive for high-level audio featurization under `spider.featurization.Audio` in our `spider` Python package. The primitive used

transfer learning on pre-trained deep networks and output fixed-length feature vectors for variable-length input audio. We eventually dropped support for this primitive due to changes in personnel during the program.

The core elements of this primitive that determine its performance are the underlying model and the data on which it was pre-trained. For the initial release, we used the Urban Sound dataset for both pre-training and model selection. Three types of models were tested: a deep, 1D convolutional CNN, a 2D convolutional CNN trained on spectrograms of the source signal, and a deep recurrent network consisting of stacked GRUs. Model performance was evaluated via the cross-entropy of a probability distribution over the possible classes (i.e., a softmax activation on the output of a fully-connected layer). Of these three models, the 1D convolutional CNN obtained the highest classification accuracy on a held-out partition of the Urban Sound dataset. This model incorporates 17 1D convolutional layers, each with a filter width of 3 and l2 regularization applied to the weights and biases. A ReLu activation and batch normalization follows each convolutional layer, and max pooling with a stride of 2 occurs every 4 layers.

Our internal experiments with audio featurization found that producing a strong deep-learning-based audio featurizer was difficult, due to the lack of a large-scale general audio dataset comparable to ImageNet [26] (the standard source for training general image featurization models). For this reason, we opted to replace the deep-learning-based implementation with a more traditional feature extractor. The resulting audio featurization primitive used a range of signal processing techniques including energy estimation, zero crossing rate, entropy, chroma vectors, mel-frequency cepstral coefficients, spectral centroid, and others to extract 34 features from each frame of audio. These features were then generally averaged over all frames in a given clip in order to produce the clip-level featurization result.

Audio data preprocessing

We employed a number of common data cleaning and preprocessing operations in testing and applying our audio featurization methods, and made two of these available as separate primitives. The first computed a log-scale spectrogram of the raw input time series. The spectrogram of an audio signal is computed as the squared magnitude of a discrete short-time fourier transform applied iteratively to overlapping windows of a signal. For a signal x and length m window w at time t, we represent the short-time Fourier transform as

$$\textbf{STFT}\{x[t]\}(m,\omega) = \sum_{n=-\infty}^{\infty} x[t]w[t-m]e^{-j\omega t} \quad . \tag{1}$$

Spectrograms are commonly used in audio pre-processing to separate time and frequency information onto separate axes. The spectrogram primitive served an additional use as a 2D visualization tool for audio. This primitive was formerly accessible at `spider.featurization.LogMelSpectrogram`, but was eventually dropped due to changes in personnel during the program.

The second primitive was a simple audio slicing utility that splits a single audio sample into multiple equal-length clips, with optional overlap between clips. This served as a featurization utility for interfacing with any primitives that require a fixed-length input. This was formerly available

at `spider.featurization.AudioSlicer`, but was eventually dropped due to changes in personnel during the program.

4.2.2 VGG-16 primitive

The `spider.featurization.vgg16` primitive implemented the VGG-16 artificial neural network structure as described in [27]. By default, this implementation downloaded weights pre-trained on ImageNet [26], providing a reasonable level of baseline performance without retraining. The primitive accepted as input either image file locations or images represented as NumPy arrays. These images would be interpolated as necessary to adhere to the model input size of (224,224,3). Additionally, one could select which layer to obtain inference features from, choose which weights (of appropriate structure) to load into the model, or choose to output ImageNet class predictions.

We eventually dropped support for this primitive due to complications arising from D3M upgrading its deep learning library dependencies.

4.3 Clustering primitives

Subspace clustering is a generalization of principal components analysis (PCA), in which high-dimensional data vectors are assumed to lie near one of several unknown low-dimensional subspaces. Subspace clustering algorithms rely on this generative model to cluster data according to its nearest subspace, with applications in computer vision and image processing. We implemented four subspace clustering algorithms (with varying computational complexity and clustering performance) as primitives.

We use the following notation. Consider a collection of points $\mathcal{X} = \{x_1, \ldots, x_N\}$ in $\mathbb{R}^D$ lying near a union of K subspaces $\mathcal{S}_1, \ldots, \mathcal{S}_K$ having dimensions $d_1, \ldots, d_K$. Let $X \in \mathbb{R}^{D \times N}$ denote the matrix whose columns are the elements of $\mathcal{X}$. As stated, the goal of subspace clustering is to label points in the unknown union of K subspaces according to their nearest subspace. Once the clusters have been obtained, the corresponding subspace bases can be recovered using principal components analysis (PCA).

Note that all of these primitives have been implemented with a similar object-oriented API under the `spider.cluster` package.

4.3.1 K-subspaces

K-subspaces (KSS) [28, 29, 30] is a k-means-like clustering algorithm that alternates between assigning points to clusters and estimating the subspace basis associated with each cluster. The algorithm attempts to solve the following optimization problem

$$\min_{\mathcal{C},\mathcal{U}} \sum_{k=1}^{K} \sum_{i:x_i \in c_k} \left\| x_i - U_k U_k^T x_i \right\|_2^2,$$

where $\mathcal{C} = \{c_1, \ldots, c_K\}$ denotes the set of estimated clusters and $\mathcal{U} = \{U_1, \ldots, U_K\}$ denotes the corresponding set of subspace bases, all assumed to have dimension d. The algorithm is computationally efficient and has been shown to converge to a local optimum. However, the performance of this algorithm on real datasets is not state-of-the-art, making it useful for data exploration. Like K-means, the performance of KSS depends heavily on the initialization, and in practice multiple random initializations are run, with the final clustering chosen as the one achieving the lowest cost.

The original release of this primitive can be accessed in the `spider` package as `spider.cluster.KSS`. The user may specify the model parameters K and d (all subspaces are assumed to have the same dimension), as well as the maximum number of iterations.

4.3.2 Ensemble K-subspaces

Ensemble K-subspaces (EKSS) [31] is an *evidence accumulation* clustering algorithm [32] that combines several instances of KSS with random initializations to obtain a robust clustering with theoretical guarantees. The algorithm computes B base clusterings of KSS and maintains an affinity matrix A, for which we have

$$A_{i,j} = \frac{1}{B} \left| \{ b : x_i, x_j \text{ are clustered together} \} \right| .$$

The affinity matrix is thresholded such that all but the top q entries in each row/column are set to zero, after which spectral clustering is performed to obtain the final clustering. EKSS is efficient, parallelizable in B (implementation in future release), and achieves strong performance on many real datasets.

The original release of this primitive can be accessed in the `spider` package as `spider.cluster.EKSS`. Beyond the model parameters for KSS, the user can specify the number of base clusterings and the threshold.

Ensemble k-subspaces theory

Our main theoretical result for the Ensemble k-subspaces clustering method guaranteed that when we use two 1-dimensional candidate subspaces to cluster the points in each random instance, and let the number of instances in the ensemble be asymptotic, our algorithm is able to exactly cluster the data points in a union of subspaces under conditions on the maximum affinity between subspaces and the number of points per subspace. This result matches state-of-the-art results, but the theory was limited in two ways: the requirement for an asymptotic number of instances, and the usage of 1-dimensional subspaces when typically an ensemble of higher-dimensional subspaces yields improved empirical results. During the project we have alleviated both of these weaknesses. Our theory is now for a finite number of instances, and shows that the number of instances required for the ensemble to converge is only $\mathbf{O}(\log N)$ where N is the number of total points to be clustered. It is a great benefit that this depends not on the problem dimensions but only the total number of data points. Our theory now also gives the same results when the candidate subspaces have arbitrary dimension. This has set us up to prove the correctness of the most sophisticated version of our algorithm, which achieves state-of-the-art empirical performance on several benchmark computer vision datasets.

Evidence accumulation ensemble k-subspaces

We developed a novel approach to the subspace clustering problem that leverages ensembles of the the k-subspaces (KSS) algorithm via the evidence accumulation clustering framework. Our algorithm forms a co-association matrix whose $(i,j)^{\text{th}}$ entry is the number of times points i and j are clustered together by several runs of KSS with random initializations. We analyzed the entries of this co-association matrix and showed that a naïve version of our algorithm can recover subspaces for points drawn from the same conditions as the Thresholded Subspace Clustering [33] algorithm. We also showed on synthetic data that our method performs well under subspaces with large intersection, subspaces with small principal angles, and noisy data. Finally, we produced a variant of our algorithm that achieves state-of-the-art performance across several benchmark datasets, including a resulting error for the COIL-20 database [34] that is less than half that achieved by prior algorithms.

4.3.3 Sparse subspace clustering

The Sparse Subspace Clustering (SSC) algorithm [22] takes advantage of the self-expressiveness property of the data, which states that each data point in a union of subspaces can be efficiently represented as a linear combination of other points; the key observation is that a sparse representation of a data point as a linear combination of other points will ideally use only a few points from its own subspace. Leveraging this fact, SSC solves a convex optimization program to form an affinity matrix, after which spectral clustering is used to obtain labels. We implemented two variants of SSC, one which uses the convex optimization package CVXPY, and another that relies on the Alternating Direction Method of Multipliers (ADMM) framework. The former is more exact but has high computational complexity, while the latter is efficient enough to work on real datasets.

Formally, the SSC algorithm seeks to solve

$$\begin{aligned} \min_{C,E,Z} \quad & \|C\|_1 + \lambda_e \|E\|_1 + \tfrac{\lambda_z}{2} \|Z\|_F^2 \\ \text{subject to} \quad & X = XC + E + Z, \ \text{diag}(C) = 0, \end{aligned}$$

where $\mathbf{C} \in \mathbb{R}^{n \times n}$ is a sparse coefficient matrix, $\mathbf{E} \in \mathbb{R}^{d \times n}$ is a matrix of sparse outlying entries, and $\mathbf{Z} \in \mathbb{R}^{d \times n}$ is a noise matrix. The affinity matrix is then computed as $|C| + |C|^T$, after which spectral clustering is applied.

SSC_CVX

The SSC_CVX primitive is an implementation of the SSC algorithm that uses the convex optimization package CVXPY. The original release of the SSC_CVX primitive can be accessed in the `spider` package as `spider.cluster.SSC_CVX`.

The user can specify several model parameters for this primitive, including whether or not to incorporate outliers, whether to allow the subspaces to be affine spaces instead of linear, and the number of subspaces. The two main methods of this primitive are `ssc_cvx.compute_sparse_coefficient_matrix()`, which returns the affinity matrix for the data, and the class method `ssc_cvx.fit_predict()`, which returns the labels for the data.

SSC_ADMM

The SSC_ADMM primitive is an implementation of the SSC algorithm that uses an Alternating Direction Method of Multipliers (ADMM) framework. The original release of the SSC_ADMM primitive can be accessed in the `spider` package as `spider.cluster.SSC_ADMM`.

The user can specify several model parameters for this primitive, including whether or not to incorporate outliers, whether to allow the subspaces to be affine spaces instead of linear, and the number of subspaces. The two main methods of this primitive are `ssc_admm.compute_sparse_coefficient_matrix()`, which returns the affinity matrix for the data, and the class method `ssc_admm.fit_predict()`, which returns the labels for the data.

SSC_OMP

Finally, we also implemented an Orthogonal Matching Pursuit (OMP) based solution for solving the SSC problem (SSC-OMP) [35]. The original release of the SSC_OMP primitive can be accessed in the `spider` package as `spider.cluster.SSC_OMP`.

4.3.4 Union of GLRM Clustering

In all the clustering primitives so far, we have assumed that our data are real-valued or at least that a standard norm objective function (ℓ_2 and Frobenius norm) are relevant. When one instead has a data table with a wide variety of data types, some of which are on different numerical scales while others are categorical, boolean, etc, this approach is no longer applicable. We began work with the Cornell group in D3M focused on clustering rows (or columns) of these data tables, and finding low rank approximations to the clusters. We had some nice initial results but the progress was not rapid enough for incorporation into D3M before our contract was ended.

4.4 Random forest distance primitive

The `spider.distance.metricl.rfd` primitive implemented the Random Forest Distance (RFD) metric learning technique [36]. Distance metric learning is an (usually supervised) approach to semantic modeling that seeks a function that takes as input a pair of instances and returns a semantically meaningful distance between them. RFD approaches this problem by first encoding instance-pairs into a vector that contains information on both the absolute ($\mathbf{v}$) and relative ($\mathbf{u}$) position of the two:

$$\phi(\mathbf{x}_i, \mathbf{x}_j) = \begin{bmatrix} \mathbf{u}(\mathbf{x_i}, \mathbf{x_j}) \\ \mathbf{v}(\mathbf{x_i}, \mathbf{x_j}) \end{bmatrix} = \begin{bmatrix} |\mathbf{x}_i - \mathbf{x}_j| \\ \frac{1}{2}(\mathbf{x}_i + \mathbf{x}_j) \end{bmatrix} \quad . \tag{2}$$

The resulting vectors are then fed into a random forest, which is trained on a number of sampled instance-pairs and then used to regress semantically meaningful nonlinear distance values at inference time. The resulting information can be used in a number of common machine learning tasks, such as semi-supervised clustering or k-nearest neighbor classification.

We eventually dropped support for this primitive due to changes in personnel during the program.

4.5 Robust PCA primitives

Standard principal component analysis attempts to estimate a low-rank subspace that lies near to a set of much higher dimensional data, which can then be used to represent the data in a form that is significantly easier to understand and process. This estimation, however, can be highly vulnerable to noise and outliers within the input data. Robust PCA attempts to address this problem by generalizing it: instead of attempting to capture all the variance using *only* a low-rank decomposition, RPCA breaks the input data matrix down into a low-rank component $\mathbf{L}$ and an additional sparse component $\mathbf{S}$, such that $\mathbf{D} = \mathbf{L} + \mathbf{S}$. The sparse component $\mathbf{S}$ allows the estimator to account and compensate for noise and outliers in a way that doesn't distort $\mathbf{L}$.

Specifically, RPCA seeks to solve:

$$\min_{\mathbf{L},\mathbf{S}} \|\mathbf{L}\|_* + \lambda \|\mathbf{S}\|_1 , \;\; s.t. \;\; \mathbf{D} = \mathbf{L} + \mathbf{S} \;\; , \tag{3}$$

where $\|\cdot\|_*$ is the nuclear norm (the sum of the matrix's singular values), λ is a positive weighting parameter and $\|\cdot\|_1$ is the sum of the absolute values of the matrix's components. This formulation has been shown [13] to be able to solve the matrix decomposition problem even with arbitrarily large errors in the input data, so long as those errors are sufficiently sparse.

We implemented three methods for solving the RPCA problem as primitives. The first (RPCA-IALM) solves the optimization problem via a relaxed, inexact variant of the augmented lagrange multipliers technique [37]. The second (RPCA-LBD) uses the low-rank and block-sparse matrix decomposition method [38] and the third (RPCA-GoDec) uses the go decomposition algorithm [39]. All of these primitives were once located in the `spider.dimensionality_reduction` submodule of our repository, but we eventually dropped support for them due to changes in personnel during the program.

4.5.1 GRASTA-based robust principal components analysis

The SPIDER team implemented several methods for solving the RPCA problem. Initially we had several of them as primitives in the `spider.dimensionality_reduction` submodule of our repository, but due to changes in personnel, we dropped support for most of them except for GRASTA. The first such method (RPCA-IALM) solves the optimization problem via a relaxed, inexact variant of the augmented lagrange multipliers technique [37]. The second (RPCA-LBD) uses the low-rank and block-sparse matrix decomposition method [38] and the third (RPCA-GoDec) uses the go decomposition algorithm [39]. All of these primitives were batch algorithms that required the entire dataset be accessible in memory and have higher than linear complexity in the dimension of the data vectors.

Next, we implemented the Grassmannian stochastic gradient descent algorithm for this problem, also known as GRASTA (Grassmannian Robust Adaptive Subspace Tracking Algorithm) [40]. Work on GRASTA in the past heavily depended on careful tuning of a step size parameter, which

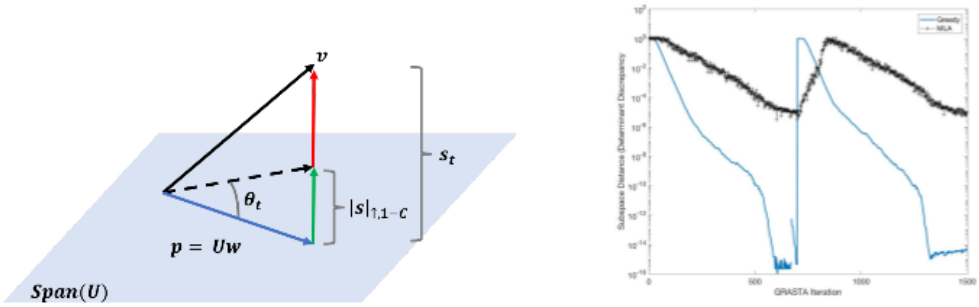


Figure 1: Left: illustration of greedy step size calculation. Right: Subspace determinant discrepancy of multi-level adaptive step size (state-of-the-art) and greedy step size for subspace learning with abrupt subspace change at 700 algorithm iterations. The greedy step size is also adaptive but far more aggressive at subspace recovery.

in the D3M system was impractical. To aid in automatic step size selection and improve the theoretical convergence guarantees of the algorithm, we developed and incorporated a new more easily automated choice of step size.

$$\min_{s.t. x - Uw - s = 0} \|s\|_1 \tag{4}$$

$$\begin{aligned} \theta_t &= \arctan(\|\,|s|_{\uparrow,1-C}\|/\|p\|) \\ p &= Uw \\ |s|_{\uparrow} &= sort(|s|, \text{ascend}). \end{aligned} \tag{5}$$

The Grassmannian Robust Adaptive Subspace Tracking Algorithm (GRASTA) minimizes the cost function given in (4.4) where an observed vector of data $x \in \mathbb{R}^m$ can be written as a low-rank r matrix $U \in \mathbb{R}^{m \times r}$ and weights vector $w \in \mathbb{R}^r$ plus an additive sparse vector $s \in \mathbb{R}^m$ of outliers corrupting the observed data.

The GRASTA algorithm initially developed to solve this problem relied on a complex, multilevel adaptive (MLA) step size during projected gradient descent onto the Grassmannian manifold that allows the algorithm to track a rapidly or slowly changing low-rank subspace. During the D3M program, we worked on simulating and evaluating a new greedy step size choice θ_t for GRASTA that approximates the projection residual, given in (4.5). This greedy step size approximates "closeness" of the subspace estimation by the angle between the projection of the data onto the subspace and its residual, where the projection of the data is approximated by sorting the estimated sparse residual vector s elements in ascending order and taking the first $1 - C$ elements corresponding to the non-sparsity percentage. Over the course of the D3M program we experimented with different versions of this step size, estimating the required sparsity parameter in different ways. GRASTA with a Lasso estimation step yields very good subspace recovery results agnostic to an *a priori* estimate of the data sparsity.

4.5.2 TV-GRASTA and RPCA

We developed a novel total variation (TV) and sparse regularized robust PCA (RPCA), algorithms specifically designed for computer vision applications in panoramic video foreground and background segmentation. This algorithm can robustly separate foreground from background even in the presence of heavy sparse noise that would normally degrade the separation in other RPCA methods. Our method is orders of magnitude faster than the state-of-the-art algorithm in [41] and has competitive performance. Like the Grassmannian Robust Adaptive Subspace Tracking Algorithm [42], our algorithm robustly estimates a low-rank subspace that spans data which may be corrupted with sparse errors and also have missing entries, and it can also operate in online mode, updating its estimates from single streaming data vectors. Our algorithm, which we call TV-GRASTA, also performs incremental gradient descent on the Grassmann manifold to learn the robust PCA estimate of the data. Such models can be used to separate foreground objects from their backgrounds in video. We show one example of our results on video with dynamic background corrupted with 20% shotgun noise from the DAVIS Challenge [43] in fig. 4.2.

TV-GRASTA optimizes the following nonconvex optimization problem:

$$\begin{aligned} \min_{s_t, e_t, U_t, w_t} \; & \beta_S \text{TV}(s_t) + \|s_t\|_1 + \beta_E \|e_t\|_1 \\ \text{s.t. } & x_{\Omega_t} = U_{\Omega_t} w_t + s_{\Omega_t} + e_{\Omega_t} \\ & U_t^T U_t = I \end{aligned} \tag{6}$$

where $e_t \in \mathbb{R}^m$ is a sparse vector to model sparse noise. The hyperparameters β_S and β_E balance the smoothness of the foreground signal with the sparsity of the noise. Here $\text{TV}(s_t) = \|Cs_t\|_1$ where

$$C \in \mathbb{R}^{2m \times m} = \begin{bmatrix} I_N \otimes D_M \\ D_N \otimes I_M \end{bmatrix} \tag{7}$$

and $D_a \in \mathbb{R}^{a \times a}$ is the first-order differences matrix and M, N are the dimensions of the registered frames from a video with dynamic background in a common reference using homography estimation and transformations. The vectorized video frames then have ambient dimension $m = MN$.

4.5.3 Noise-weighted PCA

In addition to our GRASTA-based robust PCA work, we also looked at novel methods of noise-resistant PCA. One of the key issues with PCA was that it was not robust to the heteroscedastic noise we might expect in modern data—specifically, it suffers from treating all samples as though they were equally informative. A natural fix was to weight the data before applying PCA to give noisier samples less influence. Through our work on this project we analyzed weighted PCA results and provide expressions for their asymptotic performance in terms of the weights. Surprisingly, we found that inverse noise variance weights, (i.e., weights that whiten the noise), were sub-optimal asymptotically [44].

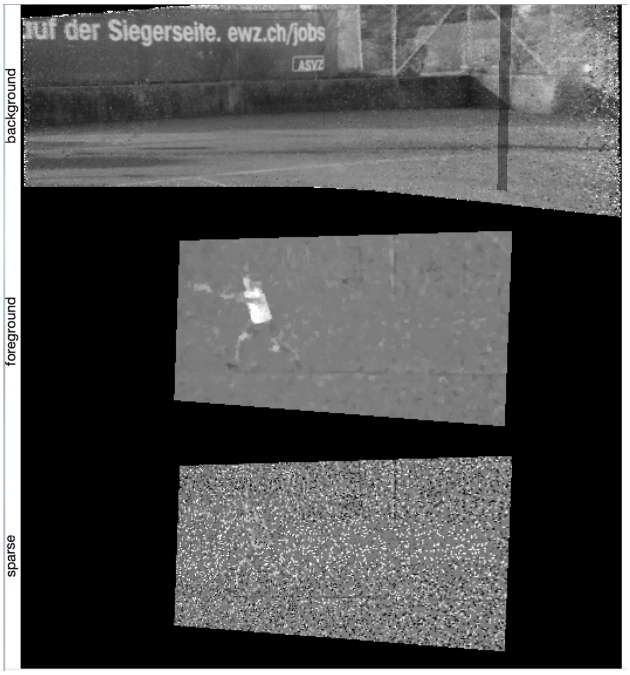


Figure 2: TV-GRASTA separation results of "Tennis" in the common reference perspective of the panorama. From top to bottom: recovered panorama background; recovered foreground; recovered sparse.

Following that work we analyzed weighted PCA to identify what weights would instead be optimal, according to an optimality criteria of maximizing the asymptotic recovery curve. In [45] we derived the optimal weights, and this analysis showed that inverse noise variance is not aggressive enough when data are highly noisy. This matches the intuition of the statistics community that PCA is not robust to outliers. The optimal weights are in between inverse noise variance and square inverse noise variance [45].

4.5.4 GROUSE primitive

GROUSE is an online algorithm adept at handling streaming single-vector data to incrementally update the principal component analysis (PCA) estimate of the data matrix $X \in \mathbb{R}^{m \times n}$ where each column of X is observed sequentially and may be incomplete where only a subset of the vector indices, $\Omega_t \subset \{1, \ldots, m\}$, is observed at time t. Specifically, the algorithm learns a rank-r subspace $U \in \mathbb{R}^{m \times r} \subset S^m$ that describes the maximal variance of the data from an m-dimensional vector space S^m where U has orthonormal columns, $U^T U = I$. GROUSE performs incremental gradient descent on the Grassman manifold, denoted $\mathcal{G}(m, r)$–the manifold of r-dimensional subspaces of ambient dimension m with orthonormal columns–to learn the subspace U_t at time t. GROUSE optimizes the following nonconvex optimization problem:

$$\min_{U_t, w_t} \frac{1}{2}\|U_{\Omega_t} w_t - x_{\Omega_t}\|_2^2 \quad \text{s.t. } U_t^T U_t = I. \tag{8}$$

Here, $U_{\Omega_t} \in \mathbb{R}^{|\Omega_t| \times r}$ and $x_{\Omega_t} \in \mathbb{R}^{|\Omega_t|}$ denote the subspace matrix and data vector at time t on the observed row indices respectively, and $w_t \in \mathbb{R}^r$ is the vector of principal components (or weights). The problem is nonconvex in U_t and w_t and because of the constraint for U_t to lie on the Grassmann manifold. GROUSE alternates optimization between the two variables, yielding a simple least squares of w_t and a U_t update from a geodesic step along the Grassmann manifold in the direction of the gradient of the objective function in Eq. 4.8 with respect to the components of U_t. GROUSE is orders of magnitude faster than other state-of-the-art PCA and low-rank matrix completion algorithms, due to its avoidance in computing singular value decompositions (SVDs), and can recover subspaces exactly from streaming incomplete data with theoretical guarantees.

GROUSE accurately estimates low-rank subspaces from streaming data with missing entries and do so with very few user-defined parameters. Our implementation of GROUSE in the SPIDER repository allowed the user to specify a constant step size of choice or revert automatically to the adaptive, greedy step-size choice. It also allowed a user to specify a "mask" matrix $M \in \{0,1\}^{m \times n}$ that defines the support indices of the observed entries. It could operate in a batch setting and an online setting with an internal function to update the subspace estimate from a newly observed data vector. We eventually dropped support for this primitive due to changes in personnel.

4.5.5 Supervised PCA as Manifold Optimization

Our approach to SPCA for D3M was to solve an optimization problem whose objective is a weighted sum of the PCA objective and an empirical risk associated with the prediction problem. In particular, we proposed to solve

$$\begin{aligned} &\underset{L,\beta}{\text{minimize}} \ \sum_{i=1}^{n} \ell(\boldsymbol{y}_i, L\boldsymbol{x}_i, \beta) + \lambda \|X - XL^T L\|_F^2 \\ &s.t. \ LL^T = I_k, \end{aligned} \tag{9}$$

where $\ell(\cdot)$ is a loss function in terms of the dependent variables and the dimension reduced data, n is the number of observations, k a user specified dimension of the subspace to be learned, and $\lambda > 0$ is a trade-off parameter. The constraint on L^T in (4.9) is over the Stiefel manifold, i.e., the set of all matrices with orthonormal columns.

The key feature of the proposed approach is that the predictor, parametrized by β, operates directly on the low-dimensional representation $L\boldsymbol{x}_i$. The variable L ties the two terms together and enables simultaneous DR and prediction, with λ affecting a trade-off between these two goals. We also remark that, unlike other approaches to high-dimensional prediction, there is no need to regularize β because the predictor acts on a low-dimensional space.

Regarding choice of loss function, we have limited our search to squared error loss in the regression case (LSPCA) and logistic loss in the classification case (LRPCA).

This work had extensive success with publication in IEEE data science workshop in 2019 [46] and a journal submission. This primitive was not implemented for D3M because of the timing of the end of our contract.

Next we show some of the compelling results achieved using our approach to supervised PCA.

Regression Experiments (LSPCA)

We compare performance on three real-world regression datasets, the Music dataset[1] ($n = 1059, p = 118$) of [47], the Residential dataset[1] ($n = 372, p = 105$) of [48], and the Parkinsons telemonitoring dataset[1] ($n = 5875, p = 20$) of [49].

Figure 3 shows average performance of our method and several competitors over 10 independent runs (each with random test/train splits) on each dataset. It is readily seen that LSPCA outperforms or is competitive with all competing methods on each dataset. This conclusion also applies to a collection of synthetic datasets we have explored as well. In each case, the variation explained by LSPCA approaches that of PCA.

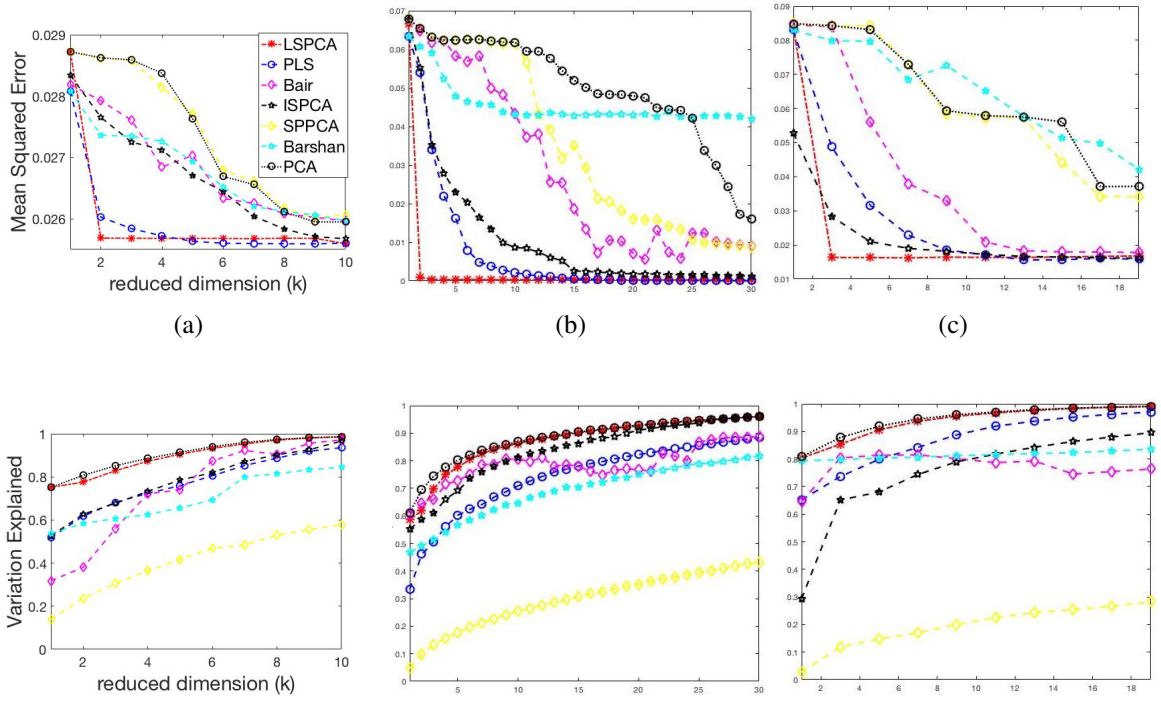


Figure 3: Mean squared error and variation explained vs. reduced dimension for each method on various datasets. (a) Parkinsons (b) Music (c) Residential

Classification Experiments (LRPCA)

We compare performance on four binary classification datasets used in previous work. Specifically the PCMAC [2] ($n = 1943, p = 3289$) and Arcene[2] ($n = 200, p = 10000$) datasets were utilized in

[1] https://archive.ics.uci.edu/ml/datasets.html
[2] http://featureselection.asu.edu/datasets.php

[50], while the Ionosphere[1] ($n = 351, p = 34$) and Sonar[1] ($n = 208, p = 60$) datasets were used in [51].

Classification results LRPCA performs best in terms of classification accuracy on three of the four datasets, and is competitive with other methods on the fourth dataset. LRPCA performs best in terms of variation explained in each case, after PCR, as expected. It is of note that SPPCA appears to be performing almost identically to PCR in these experiments. These results are shown in Tables 4.1 and 4.2. Interesting to note is that LRPCA can explain more variation than PCR in the test data for the PCMAC dataset, indicating that, in some situations, incorporation of label information can improve generalizability of variation explained.

Table 1: Classification accuracy on binary classification datasets for $k = 2$. Results are averaged over 5 independent runs. Standard deviation of classification accuracy is given in parentheses.

Dataset	LRPCA	PCA	Bair	Barshan	ISPCA	LDA
PCMAC	**0.893 (0.0143)**	0.520(0.075)	0.577(0.109)	**0.893 (0.0139)**	0.727(0.039)	0.690(0.045)
Arcene	0.693(0.037)	0.660(0.029)	0.693(0.077)	0.693(0.052)	0.700(0.082)	**0.823 (0.040)**
Ionosphere	**0.868 (0.036)**	0.630(0.051)	0.811(0.086)	0.856(0.067)	0.797(0.069)	0.865(0.024)
Sonar	**0.717 (0.024)**	0.552(0.090)	0.588(0.120)	0.681(0.034)	0.705(0.062)	0.693(0.065)

Table 2: Variation explained on binary classification datasets for $k = 2$. Results are averaged over 5 independent runs. Format: train variation explained / test variation explained.

Dataset	LRPCA	PCA	Bair	Barshan	ISPCA	LDA
PCMAC	0.229/**0.095**	**0.274**/0.092	0.199/0.067	0.071/0.052	0.103/0.039	0.008/0.015
Arcene	0.523/0.246	**0.568**/**0.269**	0.236/0.095	0.391/0.183	0.043/0.018	0.095/0.041
Ionosphere	0.586/0.582	**0.624**/**0.650**	0.452/0.458	0.395/0.390	0.477/0.486	0.177/0.177
Sonar	0.565/0.543	**0.628**/**0.609**	0.517/0.511	0.374/0.361	0.373/0.379	0.058/0.056

4.5.6 Time-varying Subspace Learning

Subspace models are useful in a plethora of applications, these include; video processing, medical imaging, communication systems, array signal processing, to name a few. In practice, due to variability in sampling conditions the subspace model changes over time in these applications [52, 40, 53, 54, 55, 56]. For an intuitive understanding of the time-varying phenomenon, consider video processing for instance. The best low-rank subspace representation for the frames in the video may change over time due to changes in the environment and(or) sampling modality (camera). Example scenarios include; *i)* change in subspace due to variability in the lighting conditions, *ii)* camera motion, and *iii)* objects entering and(or) exiting the scene over time. In such situations, the video processing outcomes can be improved by taking into account the temporal behavior of the representation model.

In the above-mentioned applications, the change in ground truth subspace is a well-documented phenomenon and is classically studied under the umbrella term of *subspace tracking*. Despite the rich history, the state-of-the-art solutions work under strict assumptions on the time-varying

behavior, e.g., changes suddenly while staying the same for a long interval of time, slowly rotating subspace, and change of only a few directions/principal vectors at the change point. In our work, we relaxed some of these assumptions by proposing a time-varying model where subspace changes over time by moving over a geodesic on a Grassmannian manifold.

Definition 4.5.1 (Geodesic on the Grassmannian)**.** *Let $A \in \mathbb{R}^{d\times k}$ and $B \in \mathbb{R}^{d\times k}$ be two orthonormal bases on the rank-k Grassmannian $\mathcal{G}(k,d)$. Let us observe m data points from a geodesic defined by A and B. Then for $i = 1,\dots,m$ and $t_i \in [0,\pi/2]$ the point on the geodesic for any i is given by*

$$U_i = \textit{span}\left(AZ\cos(\Theta t_i) + Y\sin(\Theta t_i)\right) \tag{10}$$

where $(I - AB^T)B_b(A^TB)^{-1} = YSZ^T$ is a thin SVD and $\Theta = \tan^{-1}(S)$. Here $Y \in \mathbb{R}^{d\times k}$, $S \in \mathbb{R}^{k\times k}$, and $Z \in \mathbb{R}^{k\times k}$.

Next, let $x_i \in \mathbb{R}^d$ for $i = 1,2,\dots,m$ be data derived from a time-varying low-rank model with noise:

$$x_i = U_i g_i + \eta_i \tag{11}$$

where $U_i \in \mathcal{G}(k,d)$ is as defined in (4.10), $g_i \in \mathbb{R}^k$ is a weight vector, and $\eta_i \in \mathbb{R}^d$ is an independent additive noise vector. Using the observations $\{x_i\}_{i=1}^m$, our objective is to estimate U_i, which can be posed as the following optimization problem:

$$\{U_i^*\}_{i=1}^m := \underset{U_i\in\mathbb{R}^{d\times k}, U_i^TU_i=I, i=1,\dots,m}{\operatorname{argmin}} \sum_{i=1}^{m} \left\|x_i - U_iU_i^Tx_i\right\|_F^2. \tag{12}$$

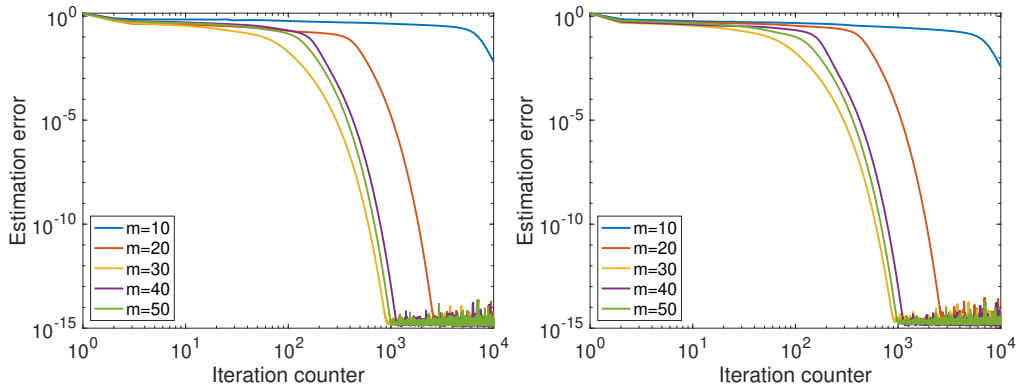
From (4.10), notice that in order to estimate all the subspaces $U_1,\dots,U_m$ we only need estimates of A, B, and Θ. Therefore, we propose an alternating minimization approach that estimates these quantities. Next, we provide some preliminary experiments that prove the efficacy of the proposed approach.

We provide the results of an experiment that investigates the convergence behavior of our proposed alternating minimization approach as a function of number of points, m, being sampled from the geodesic. This experiment is based on synthetic data being generated using the generative model in Definition 4 .5.1. This experiment serves two purposed: i) it serves as a proof of concept to show that the proposed algorithm works at least for a simple toy example and ii) a glimpse at convergence behavior of the proposed method as a function of number of samples m observed from the geodesic. The more points imply that we are traversing slowly over the geodesic, therefore it is not surprising to observe in Figure 4.4 that for larger values of m alternating minimization converges at a faster rate.

4.6 Multimodal primitives

4.6.1 Dense video captioning

Dense video captioning involves segmenting a long untrimmed video into semantically meaningful procedure segments that are each described with a natural language sentence. In prior literature,



(a) Error in estimate of the starting subspace $\mathbf{A}$.

(b) Error in estimate of the ending subspace $\mathbf{B}$.

Figure 4: Convergence behavior of alternating minimization to solve (4.12) for different number of points, m, generated from the geodesic.

dense video captioning is formulated as a multi-task learning problem (i.e. segment proposal and caption generation) by sharing the visual encoder between the proposal module and the captioning module. However, the connection from the proposal module to the captioning module is usually not differentiable, so the semantic information from language cannot have a direct impact on the segment proposals.

We proposed a novel approach that overcomes this limitation, bridging the two components via a differentiable visual mask. In contrast to the existing methods where the proposal module solves a class-agnostic binary classification problem regardless of the details in the video content, our model enforces consistency between the content in the proposed video segment and the corresponding language description. Our proposed method has been experimentally vetted on the YouCook2 and ActivityNet Captions datasets.

4.6.2 Weakly-supervised video language grounding

We also worked on another joint video and language understanding project focused on learning weakly-supervised language grounding in video. Given a video that is broken into temporal segments with each video segment accompanied by a sentence (e.g., recipe steps in a cooking video), our task of interest was to localize certain objects from that sentence for each frame in the video segment, if applicable. This problem was novel and interesting (and still is) since most prior methods tackled the grounding problem in a static image setting, while grounding has many potential applications in the video domain, such as autonomous driving and mobile robots. However, annotating video with dense object bounding boxes is expensive, so we tackled the problem with only weak supervision (i.e. the sentence description for each video segment) during training, which makes our problem particularly challenging.

To address this problem, we first collected data for our task, acquiring bounding-box annotations for our validation and test sets. We designed a pipeline using VATIC [57] on Amazon Mechanical Turk and used it to gather bounding box annotations for our previous YouCook2 data, resulting in a new YouCook2-BoundingBox dataset. All the annotators for this task were carefully appointed

and well-trained. We also manually verified the work after we collected all of the data to guarantee that the annotation accuracy was $> 95\%$.

Then, we experimented with several tentative solutions to this problem. Our experimental model consisted of two components, with the first one being a visual encoding module, wherein we encoded each video segment as a chunk of visual features. We experimented with two types of visual encoders, 3D-ResNet [58] over raw images and Spatio-temporal Graph ConvNets over supervoxels (ST-GCN) [59], with the latter generally proving relatively more effective. The second part of our architecture was an object grounding module, where we represented each target object mentioned in the caption individually and deployed a soft attention filter to localize the object in the video. Unfortunately, the proposed methods did not outperform the naive baseline for our problem (i.e., always predict the object at the center of the frame). We believe there were two reasons for this. First, when people record a video, they are prone to center the salient object in view, which biases the data strongly towards the naive center baseline. Second, our loss function did not force the model to consider the entire object in order to make the correct classification prediction. Hence, even when the model's attention was on only a small part of the target object, the training loss would still be low.

4.6.3 Visual Captioning with Grounded Objects

Furthermore, we pursued a project to unify the captioning framework for image and video. Basically, the goal of this research was to describe the contents of an input image/video with a natural language sentence. Our proposed model consisted of two major parts, a captioning module and a grounding module. The captioning module took in the image/video and region proposals and generated a sentence description. In the meantime, the grounding module evaluated how likely an object was present in the scene to provide visual evidence for the caption.

Some of these experiments were conducted on the Flickr30k Entities image captioning dataset. The same approach was evaluated on a video dataset called ActivityNet-BB. In order to be able to utilize this method and transfer grounding knowledge, we needed to transfer knowledge from a large-scale dataset (e.g. Visual Genome) to the target datasets. In our case, 78% of classes in Flickr30K and 64% of ActivityNet were also in VG, meaning that they could be directly transferred.

4.7 Temporally-Consistent Video-to-Video Translation Models

Video-to-video translation for super-resolution, inpainting, style transfer, etc. is more difficult than corresponding image-to-image translation tasks due to the temporal consistency problem that, if left unaddressed, results in distracting flickering effects. Although video models designed from scratch produce temporally consistent results, training them to match the vast visual knowledge captured by image models requires an intractable number of videos. To combine the benefits of image and video models, we proposed an image-to-video model transfer method called Hyperconsistency (HyperCon) that transforms any well-trained image model into a temporally consistent video model without fine-tuning. HyperCon works by translating a synthetic temporally interpolated video frame-wise and then aggregating over temporally localized windows on the interpolated

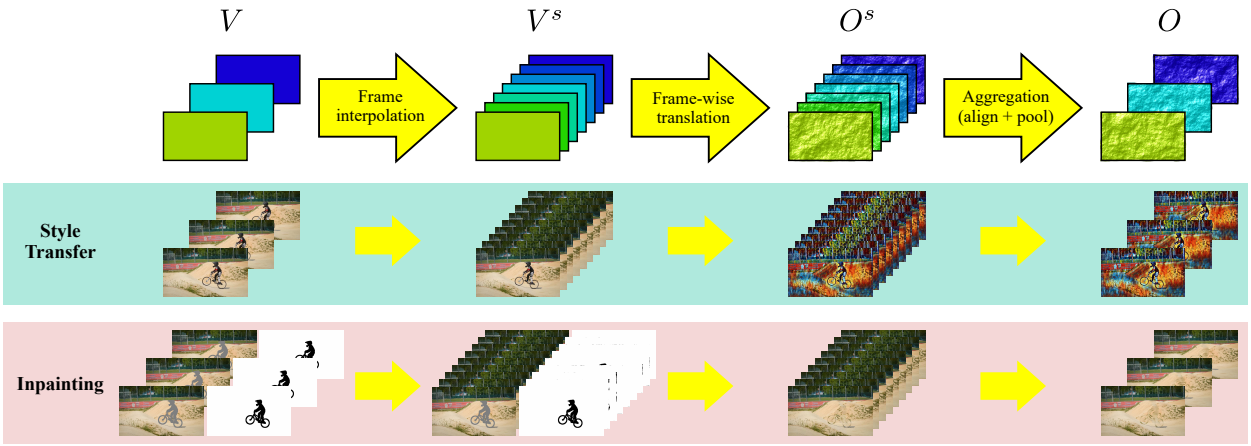


Figure 5: Our Hyperconsistency approach.

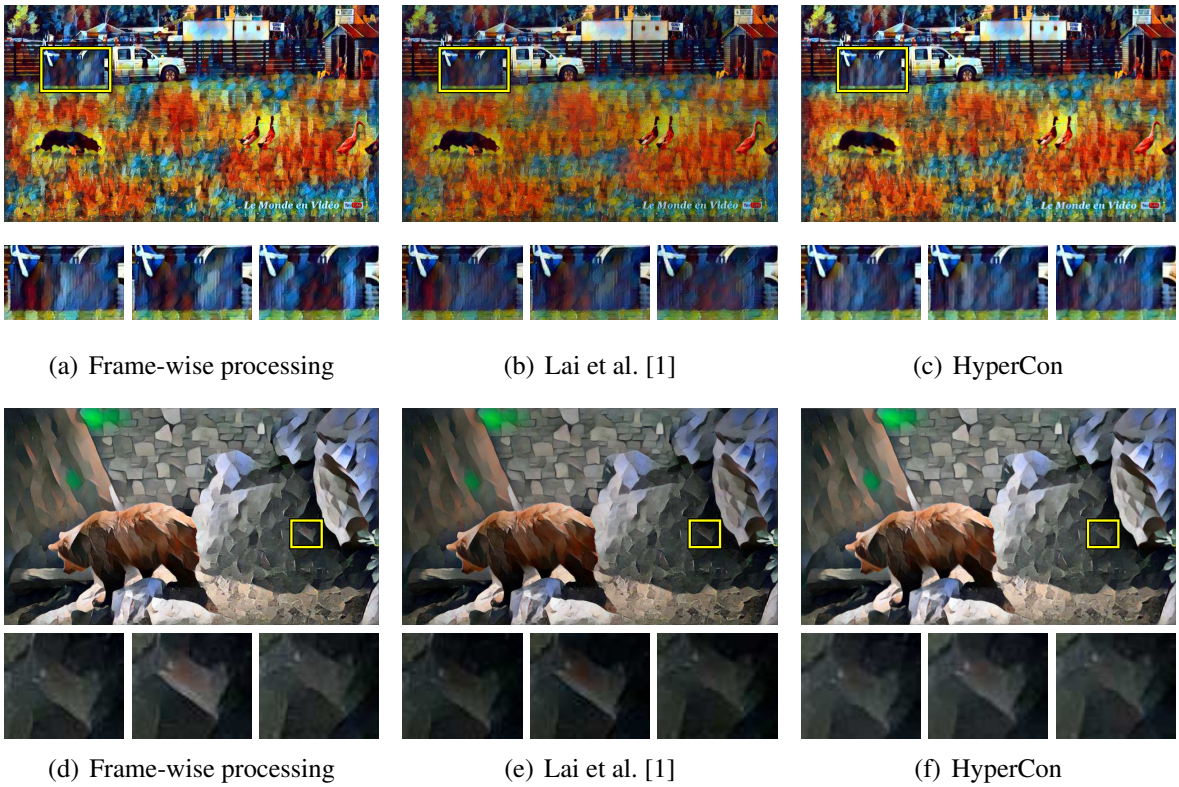
video. It handles both masked and unmasked inputs, enabling support for even more video-to-video tasks than prior image-to-video model transfer techniques.

Given an input video $V = \{v_1, \ldots, v_N\}$, our goal was to generate an output video $O = \{o_1, \ldots, o_N\}$ representing the N frames of V translated by some image-to-image model g. The frames of O should closely resemble the frames of V translated frame-wise by g; at the same time, O should be temporally consistent, i.e., exhibit as few flickering effects as possible.

With HyperCon, we generated O as follows. First, we inserted i frames between each pair of frames in V with a frame interpolation network. Let us denote this interpolated version of V as $V^s = \{v_1^s, \ldots, v_{N'}^s\}$, where N' is the number of frames in the interpolated video. Then, we independently translated each frame in V^s with g, yielding $O^s = \{o_1^s, \ldots, o_{N'}^s\}$. Finally, we aligned and pooled frames in O^s over a temporal sliding window with an appropriate stride to produce the frames of the final output video O.

We applied HyperCon to video style transfer and video inpainting and outperformed prior state-of-the-art models for the respective tasks. As shown in Figure 4.6 and Figure 4.7(a), our method modulated frame-wise results in a desired temporally-consistent manner, whereas prior work (Lai et al. [1]) produced inconsistencies. As shown in Figure 4.7(b), our method often generated more plausible inpainting predictions than prior state-of-the-art since it was better at replicating background textures.

We generated quantitative results that demonstrated the effectiveness of our approach. For style transfer, we quantified adherence to the intended style by measuring FID [61] between the set of frame-wise style-transferred frames and those produced by the temporal consistency method under evaluation. To measure temporal consistency, we used warping error [1] and patch-based PSNR and SSIM consistency [62]. Warping error estimates optical flow on the unprocessed video, and checks the extent to which this flow is followed in the stylized video. Patch-based consistency chooses a random patch in one frame, considers all patches in its spatial neighborhood in the next frame, and computes the maximum PSNR or SSIM between it and all candidate patches. We compared our method against the temporal consistency post-processing approach of Lai et al. [1] and greatly outperformed it in terms of both style coherence and temporal consistency, as shown in Table 4.3.



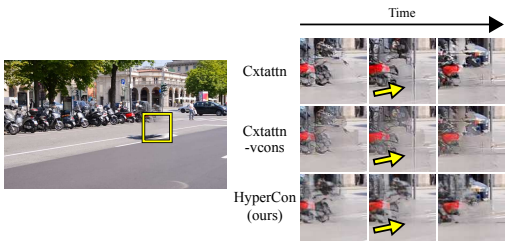
(a) Frame-wise processing (b) Lai et al. [1] (c) HyperCon

(d) Frame-wise processing (e) Lai et al. [1] (f) HyperCon

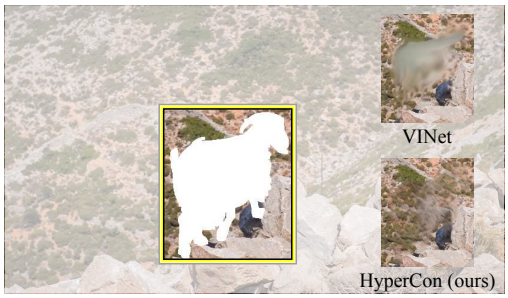
Figure 6: For video style transfer, our HyperCon method modulated colors to reduce flicker more effectively than prior work.

Furthermore, we asked Amazon Mechanical Turk (AMT) workers to compare videos from our method and the baseline by Lai et al. [1]. Specifically, t o m easure q u ality, w e s howed workers both videos side-by-side and asked which one they preferred; for style adherence, we showed them both videos alongside the frame-wise processed video, and asked which of the two videos was more similar to the frame-wise processed one. As shown by Table 4, workers selected our videos significantly m ore o ften t han t hose o f t he b a seline, i ndicating t hat o ur v ideos w ere more appealing and better adhered to the intended stylization than the videos from the baseline.

Finally, we compared two variants of our method—one using mean pooling to aggregate frame-wise processed frames, and another using median pooling—to various baselines for video inpainting, specifically f rame-wise i npainting [63] (C xtattn), f rame-wise i npainting w ith t emporal consistency post-processing via Lai et al. [1], and a video inpainting DNN VINet [60]. The metrics we used to quantify performance were LPIPS [64], a video equivalent VLPIPS, FID [61] between ground-truth and inpainted frames, a video equivalent VFID, and warping error. As shown in Table 5, our method variants generally outperformed the baselines, and always ranked in the top two across all evaluated metrics and datasets.



(a) Our HyperCon method produced fewer flickering artifacts than frame-wise processing (Cxtattn) and video consistency post-processing (Cxtattn-vcons) [1].



(b) Comparison of our method to the prior state-of-the-art video inpainting model VINet [60].

Figure 7: Video inpainting comparison.

Table 3: Quantitative comparison between baseline video consistency [1] (FST-vcons) and HyperCon (ours) across styles. $^{\uparrow}$ and $^{\downarrow}$ indicate that higher and lower is better, respectively; bold indicates where one score is better than the other. HyperCon obtained lower FID scores, indicating greater coherence to the intended style. It also obtained a lower warping error and higher patch-based consistency scores—indicating better temporal consistency—because it reduced flicker instead of replicating it like FST-vcons.

		mosaic				rain-princess			
Dataset	Method	FID$^{\downarrow}$	$E_{\text{warp}}{}^{\downarrow}$	$C_{PSNR}{}^{\uparrow}$	$C_{SSIM}{}^{\uparrow}$	FID$^{\downarrow}$	$E_{\text{warp}}{}^{\downarrow}$	$C_{PSNR}{}^{\uparrow}$	$C_{SSIM}{}^{\uparrow}$
DAVIS 2017	FST-vcons [1]	24.50	0.010194 ± 0.000500	20.51 ± 1.37	0.5830 ± 0.0140	10.87	0.007071 ± 0.000477	22.73 ± 1.35	0.6717 ± 0.0136
	HyperCon (ours)	**18.04**	**0.008810 ± 0.000493**	**21.04 ± 1.37**	**0.6662 ± 0.0157**	**10.53**	**0.006110 ± 0.000487**	**23.38 ± 1.35**	**0.7480 ± 0.0143**
DAVIS 2019	FST-vcons [1]	24.89	0.011999 ± 0.000672	18.79 ± 0.29	0.5451 ± 0.0169	14.77	0.008938 ± 0.000644	20.86 ± 0.38	0.6365 ± 0.0159
	HyperCon (ours)	**23.11**	**0.010677 ± 0.000710**	**19.20 ± 0.32**	**0.6105 ± 0.0192**	**13.59**	**0.008117 ± 0.000730**	**21.13 ± 0.43**	**0.6960 ± 0.0175**
ActivityNet	FST-vcons [1]	26.37	0.006727 ± 0.000311	22.58 ± 0.38	0.7075 ± 0.0115	9.07	0.003923 ± 0.000240	25.97 ± 0.42	0.8008 ± 0.0093
	HyperCon (ours)	**18.54**	**0.003233 ± 0.000149**	**25.13 ± 0.35**	**0.8514 ± 0.0079**	**7.62**	**0.002523 ± 0.000167**	**27.93 ± 0.75**	**0.8814 ± 0.0071**

4.8 Deep Net Primitives

4.8.1 T-RECS: Training for Rate-Invariant Embeddings by Controlling Speed for Action Recognition

An action should remain identifiable when modifying its speed: consider the contrast between an expert chef and a novice chef each chopping an onion. Here, we expect the novice chef to have a relatively measured and slow approach to chopping when compared to the expert. In general, the speed at which actions are performed, whether slower or faster than average, should not dictate how they are recognized. However, when tasked with classifying slowed down and sped up versions of action videos, existing action recognition models failed to perceive these distinct videos as the same action. For example, our tests found that when a ConvNet+LSTM model (where an LSTM is used to model the temporal change in features) attempted to classify a video from the HMDB51 dataset containing the action 'sword' sped up by a factor of 2.7, the model identified it as the action 'kiss'.

We explored the erratic behavior caused by varying input speeds on state-of-the-art deep network-based methods for action recognition. We measured the effect in terms of maximum performance

Table 4: Human evaluation of style transfer quality. We list how often subjects favorably selected each method, followed in parentheses by the p-value of a χ^2 test against the null hypothesis of random selection (0* indicates a p-value less than 1×10^{-10}). In all cases, subjects selected HyperCon (ours) significantly more often than FST-vcons [1].

	Method	Preferred	Style adherence
mosaic	FST-vcons	42.7%	20.9%
	HyperCon	**57.3%**	**79.1%**
	p-value	5.53e-7	0*
rain-princess	FST-vcons	44.5%	23.2%
	HyperCon	**55.5%**	**76.8%**
	p-value	1.54e-4	0*

Table 5: Comparison between HyperCon (ours) and the baselines on the simulated video inpainting task. $^\downarrow$ indicates that lower is better. Bold+underline and bold respectively indicate the 1st- and 2nd-place methods. Among the evaluated methods, the HyperCon models consistently placed in the top two across all performance measures.

	DAVIS 2017					DAVIS 2019					ActivityNet				
Method	$D_{\text{LPIPS}}^\downarrow$	$D_{\text{VLPIPS}}^\downarrow$	FID$^\downarrow$	VFID$^\downarrow$	$E_{\text{warp}}^\downarrow$	$D_{\text{LPIPS}}^\downarrow$	$D_{\text{VLPIPS}}^\downarrow$	FID$^\downarrow$	VFID$^\downarrow$	$E_{\text{warp}}^\downarrow$	$D_{\text{LPIPS}}^\downarrow$	$D_{\text{VLPIPS}}^\downarrow$	FID$^\downarrow$	VFID$^\downarrow$	$E_{\text{warp}}^\downarrow$
Cxtattn	0.0457	0.5838	20.94	1.435	0.002186	0.0442	0.5575	**15.55**	1.361	0.002539	**0.0432**	0.5981	**21.5173**	1.4417	0.000894
Cxtattn-vcons	0.0480	0.6076	23.23	1.502	0.001780	0.0478	0.5964	18.52	1.49	0.002166	0.0448	0.6067	23.1642	1.4383	0.000689
VINet	0.0616	0.6062	29.24	1.465	0.001882	0.0539	0.5455	18.22	**1.195**	0.002292	0.0608	0.6139	29.3806	1.3783	**0.000678**
HyperCon - mean	**0.0450**	**0.5272**	**18.49**	**1.073**	**0.001540**	**0.0437**	**0.5179**	16.07	1.274	**0.001847**	0.0454	**0.5728**	22.5111	**1.2251**	**0.000640**
HyperCon - median	**0.0424**	**0.5217**	**17.75**	**1.074**	**0.001614**	**0.0419**	**0.5089**	**15.24**	**1.254**	**0.001950**	**0.0441**	**0.5812**	**22.2705**	**1.2601**	0.000683

and stability in recognition accuracy across a range of input video speeds. By observing the trends in these metrics and summarizing them based on expected temporal behavior w.r.t. variations in input video speeds, we found two distinct types of network architectures. Type I models exhibited low stability as well as recognition performance, while Type II models exhibited inherent stability. We found that C3D [65] and I3D [66] behaved as Type I models while TSN [67] and ConvNet+LSTM [68] behaved as Type II models.

As a step toward solving the erratic behaviour caused by varying input video speeds, we proposed a pre-processing method named T-RECS, as a way to extend deep-network-based methods for action recognition to explicitly account for speed variability in the data. We did so by adaptively resampling the inputs to a given model. As our method is agnostic to the specific deep-network model; we applied it to four state-of-the-art action recognition architectures, C3D, I3D, TSN, and ConvNet+LSTM. On HMDB51 and UCF101, T-RECS-based I3D models showed a peak improvement of at least 2.9% in performance over the baseline while T-RECS-based C3D models achieved a maximum improvement in stability by 59% over the baseline, on the HMDB51 dataset. Also, we showed that T-RECS based models improve the stability of Type I models while maintaining the stability in Type II models and increasing their overall recognition performance.

The link to the repository is `https://github.com/MichiganCOG/T-RECS`. Additionally, we have a report detailing all of our findings. This report is available at `https://arxiv.org/pdf/1803.08094.pdf`.

4.8.2 Michigan Platform for Activity Classification in Tensorflow (M-PACT)

There are many hurdles that offer a large barrier to entry for new researchers in activity classification or prevent the replication of existing work, which hinders the development of new activity classification models. These hurdles include switching between multiple deep learning libraries and the development of boilerplate experimental pipelines to name a few. In M-PACT we overcame these existing issues by removing the need to develop boilerplate code and allowing users to quickly prototype action classification models while leveraging existing state-of-the-art (SOTA) models available in the platform. Key highlights of M-PACT include, modular development environments, a user friendly wrapper for layer definitions that offers the combined features of `tf.layers`, `tf.nn`, and other packages, multi-GPU support for efficient model training over a single node in addition to asynchronous and parallel data loading using `tfrecords`-based data containers, as well as a customized suite of pre-processing functions for videos that contains cropping, resizing, temporal re-sampling, oversampling, looping videos up to a desired number of frames, and much more. We achieved near state-of-the-art performance in the I3D model and have matched performance in TSN, C3D, and ResNet50+LSTM models to the original specifications. The link to the repository is `https://github.com/MichiganCOG/M-PACT`. Additionally, we have a report detailing the fundamental structure and all available functionality. This report is available at `https://arxiv.org/pdf/1804.05879.pdf`.

4.8.3 Deep Net Triage: Analyzing the Importance of Network Layers via Structural Compression

Deep network compression attempt to reduce the number of parameters in the network while maintaining a certain level of performance. Most compression methods eliminate non-contributing neurons and thus reduce model overhead. Deep network distillation seeks to train a smaller network that matches soft-max performance of a larger network. While both regimes have led to impressive performance for their respective goals, neither provided insight into the importance of a given layer in the original model, which is useful if we want to improve our understanding of the highly parameterized deep network models.

In this work, we studied the concept of deep net triage, which individually assessed small blocks of convolution layers to understand their collective contribution to the overall performance of a commonly used deep network, VGG16. We called this contribution to the health of the network "criticality". This method of assessment is named deep net triage because we assessed the "criticality" of layers within a network by comparing which, when compressed, is most influential to the overall health of the network. We proposed a suite of triage methods and compared them on problem spaces of varying complexity. We ultimately showed that, across these problem spaces, deep net triage is able to indicate the relative importance of different layers. We also showed that our local structural compression technique leads to an improvement in overall accuracy when the final model is fine-tuned globally.

In Deep Net Triage, we proposed five methods with which to modify a deep network: performing parameter updates only in the compressed layer with randomly initialized weights as the compressed layer (CL-RW); retraining the entire compressed model with randomly initialized weights

at the compressed layer (CM-RW); retraining only the compressed layer with weights initialized as a mean of the corresponding block's network weights (CL-MW); retraining the entire compressed model with compressed layer weights initialized as a mean of that block's parent network weights (CM-MW); and, creating a Student-Teacher network and training the compressed layer output tensors against the parent block's output tensors, before fine-tuning over the compressed layer weights (STN).

We showed through experimentation that structurally compressed and fine-tuned models can perform equivalent to, or better than, uncompressed parent models in a layer invariant manner. Additionally, we showed that parent-inspired initialization regimes applied only at the layer-level are unable to compete with fine-tuning over the entire global model. Lastly, we showed that Student-Teacher models evaluated at intermediate layers in the form of hints from uncompressed parent models promote faster convergence to maximal accuracy, despite being unable to outperform full-model training methods. We have a report detailing all our findings at `https://arxiv.org/pdf/1801.04651.pdf`.

4.8.4 Learning to Share: Simultaneous Parameter Tying and Sparsification in Deep Learning

Two common methods of limiting deep network complexity are sparsity-inducing regularization and parameter sharing or tying, in which certain sets of weights are forced to share a common value. Some forms of weight sharing are hard-wired to express certain invariances, with a notable example being the shift-invariance of convolutional layers. However, there may be other groups of weights that may be tied together during the learning process, thus further reducing the complexity of the network. In this work, we adopted a recently proposed sparsity-inducing regularizer, named GrOWL (group ordered weighted l_1) [69], which encourages sparsity and, simultaneously, learns which groups of parameters should share a common value. GrOWL has been proven effective in linear regression, being able to identify and cope with strongly correlated covariates. Unlike standard sparsity-inducing regularizers (e.g. l_1), GrOWL not only eliminates unimportant neurons by setting all the corresponding weights to zero, but also explicitly identifies strongly correlated neurons by tying the corresponding weights to a common value.

This ability of GrOWL motivates the following two-stage procedure: (i) use GrOWL regularization in the training process to simultaneously identify significant neurons and groups of parameter that should be tied together; (ii) retrain the network, enforcing the structure that was unveiled in the previous phase, i.e., keeping only the significant neurons and enforcing the learned tying structure. We evaluated the proposed approach on several benchmark datasets, showing that it can dramatically compress the network with slight or even no loss on generalization performance.

Our full work on this subject was accepted for presentation in ICLR 2018 (`https://openreview.net/forum?id=rypT3fb0b`).

4.8.5 Deep unsupervised clustering using mixture of autoencoders

Where subspace learning seeks to describe data in terms of low-dimensional linear subspaces, the field of manifold learning can be seen as a variant on this approach wherein the low-dimensional

components of the learned representation are nonlinear manifolds. Given this, we sought to explore the potential of manifold-based clustering, and to simultaneously leverage the power of deep learning in the unsupervised domain.

Specifically, we proposed a novel clustering method that uses an ensemble of autoencoders to learn a set of low-dimensional nonlinear manifolds. We combined this with a linked mixture assignment network, which takes the concatenated latent vectors from the autoencoders as input and infers the distribution over clusters. By jointly optimizing the two parts, we simultaneously assigned data to clusters and learn the underlying manifolds of each cluster. We tested this method on handwritten digit, document and time series datasets and achieved results similar to or better than state-of-the-art deep clustering methods, without the need for pretraining.

For more information, please see `https://arxiv.org/abs/1712.07788`.

4.8.6 Information Maximization Autoencoder

A central tenet for designing and learning a model for data is that the resulting representation should be compact yet informative. Therefore, the goal of learning can be formulated as finding informative representations about the data under proper constraints. In this work, we started with a stochastic encoder and aimed at maximizing the mutual information between the data x and the representations z. In this setting, a reconstruction or generating phase can be obtained as the variational inference of the true posterior $p_\theta(x|z)$. By explicitly targeting informative representations, the proposed model was capable of yielding better decoding quality. Moreover, we showed that the information maximization objective naturally induces a balance between the informativeness of each latent factor and the statistical independence between them, which gives a more principled way to learn semantically meaningful representations.

Another contribution of this work was a framework for simultaneously learning continuous and discrete representations, which yields a more flexible model for data that belong to different categories. Categorical data are ubiquitous in real-world tasks, and using a hybrid discrete and continuous representation to capture both categorical information and continuous variation in data is more consistent with the natural generation process. In this work, we focused on categorical data that are similar in nature, i.e., where different categories still share similar variations (features). We aimed to learn semantically meaningful discrete representations while maintaining disentanglement of the continuous latent factors that capture the variations shared over different categories.

4.9 Novel Regression Techniques

4.9.1 Ordered weighted least squares

The ordered weight ℓ_1 (OWL) norm family, which is independently proposed by Bogdan [70] and Zeng [71], is a state-of-the-art group sparsity inducing regularizer, and one of its specializations is an octagonal shrinkage and clustering algorithm for regression (OSCAR), has been shown to be an efficient supervised learning technique in [72] and [73].

The ordered weighted ℓ_1 (OWL) norm can be seen as a sorted and weighted variant of ℓ_1 norm. For $\boldsymbol{\beta} \in \mathbb{R}^p$, its OWL norm is defined as

$$\Omega_{\boldsymbol{w}}(\boldsymbol{\beta}) = \boldsymbol{w}^T |\boldsymbol{\beta}|_{\downarrow} = \sum_{i=1}^{p} w_i |\beta|_{[i]} \,,$$

where $\boldsymbol{w}$ is the nonnegative OWL weights which is required to be in the non-increasing order, and $|\beta|_{[i]}$ is the i-th largest element of $\boldsymbol{\beta}$ in magnitude.

Suppose we have n samples in the dataset. The OWL-regularized linear regression can be formulated as an unconstrained optimization:

$$\min_{\beta} \frac{1}{n} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \Omega_{\boldsymbol{w}}(\boldsymbol{\beta}) \tag{13}$$

where $\boldsymbol{y} \in \mathbb{R}^n$ is the targets, $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ is the feature matrix with samples as rows, and $\boldsymbol{\beta} \in \mathbb{R}^p$ is the coefficients. Figure 4.8 gives an example of $\boldsymbol{\beta} \in \mathbb{R}^2$ that illustrates the feature selection and grouping property of the OWL norm.

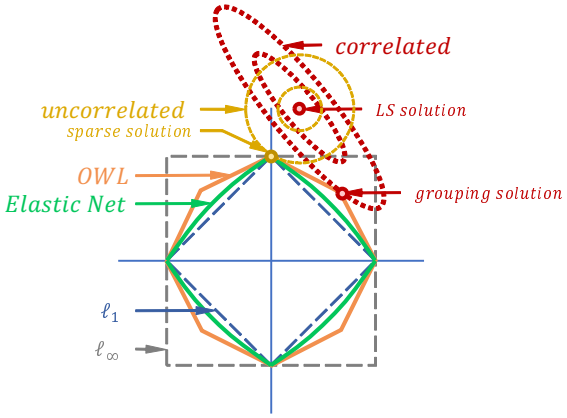


Figure 8: Feature selection and grouping property of OWL norm

4.9.2 OWL Regularized Matrix Completion

Matrix completion is a problem that aims to recover the missing entries of a matrix under some assumptions of the original matrix; one of the most commonly used assumption is the low rank structure of the matrix. The now "classical" convex relaxation of this problem uses a nuclear norm, which is the ℓ_1 norm of the singular values of a matrix. In this project, we instead tried using the OWL norm on the singular values. This would be a useful approach in settings where singular values are very similar, or there is little gap between them, which is a common scenario in practice but mostly ignored by the matrix completion literature. We had some preliminary success with this method but were not able to make significant gains over nuclear-norm regularized matrix completion.

4.10 Visual Tracking

4.10.1 Single Target Visual Tracking

In this research, we wanted to track a single target in a video by considering the motion of the target as well as identifying the target in each frame. Given a video sequence, the position of the object is initialized from the ground truth in the first frame. Using that information, the tracker is tasked with localizing the object in subsequent frames using bounding box coordinates. The area encompassed by the bounding box captures the image region corresponding to the object of interest. There were many existing methods that aimed to solve this problem, and even a yearly Visual Object Tracking challenge competition amongst various groups. Previous works used correlation filters, discriminative methods, convolutional features, and similarity learning. However, none of the methods used knowledge about motion to guide predictions in tracking. We used knowledge about how objects move in videos to make better informed predictions in frame to frame localization. One method of doing this was to use a recursive Bayesian filter such as a Kalman filter.

Recursive Bayesian filters are a class filters that can perform state estimation using an underlying probability distribution. These include Kalman filters, Extended Kalman filters, and Particle filters. Nevertheless, these can only be used for state-to-state transitions, meaning that the next estimate is based only on the current state, which is an unrealistic assumption when working with video data. As a result, a few works emerged that incorporate deep learning with state estimation to develop a learning based state estimation technique. One such work is titled LSTM-KF, which uses both LSTMs and Kalman filters to model the state transition, state covariance, and observation covariance. For our initial experiments, we used the GOTURN tracker as the baseline model. This is a Siamese network that takes as input two consecutive frames from a video and outputs bounding box coordinates. First we implemented the model in PyTorch and tested out the results on VOT challenge testing set. There seemed to be some small discrepancy in the performance of the implementation versus the original Caffe model. Then, we implemented the LSTM-KF model which performs Kalman filtering using LSTM cells. We tested the LSTM-KF method on synthetic motion data, three dimensions x,y,theta and observed that it performed sufficiently well on non-linear synthetic data.

Afterwards, we combined these ideas together into a new model. The bounding box coordinates are used as the state, and the LSTM-KF is used to filter/update the state using previous training data. We generated a synthetic moving boxes dataset that contained 130,000 frames to run the new model on it. The results from the synthetic data proved to be a slightly inconclusive, because the validation loss was lower than the training loss. This lead us to believe that the dataset may be too simple for the complex model. It would be easy to simply memorize the data, and the model would never overfit. So we transitioned to a ALOV dataset, which was used for training on the original GOTURN model, fine-tuning on that dataset, and then testing on the VOT challenge test set. The preliminary results showed that the bounding box does indeed "follow a better motion path" but its IOU with the ground truth was lower because it regresses to a smaller area in the image. This indicated that because we are operating strictly on the bounding box coordinates, the new model loses some notion of the content region itself. We could perform the state estimation on the feature vectors from the network, however this would be impractical because of the high dimensionality

of the feature vectors. This is the equivalent of doing high order matrix operations (during the Kalman filtering stage) resulting in extremely high memory usage and low operating speeds.

4.10.2 Visual Tracking with GoTurn primitive

We had added a new primitive to SPIDER for visual tracking with a known method called GoTurn (although we eventually stopped supporting it due to complications from D3M upgrading its deep learning library dependencies). It utilized a Siamese network, i.e. it took a pair of video frames as inputs, regressed to a set of bounding boxes for each frame in a video and applied an ℓ_1 loss between the ground truth labels and the predictions.

To train this primitive, a dataset with annotated coordinates was required. The primitive consisted of testing examples and pre-trained weights for utilization in TA2 systems. The primitive contained a fit and produce function, so it could be further fine-tuned with more data (e.g., the ALOV300 dataset that includes around 300 videos with annotated objects), or it could simply be used for testing.

4.11 LILAC

Conventional curriculum learning schemes organize data based on the samples' difficulty level. Over the course of the training phase, data of progressively increasing difficulty are introduced to neural network models. This setup forces neural networks to learn from a subset of training data early on. Further, data at such early stages can be imbalanced based on their difficulty ranking.

In LILAC, *Learning with Incremental Labels and Adaptive Compensation*, we proposed a new method that regulates labels instead of samples. This works in two phases, 1) *Incremental Label introduction*, and 2) *Adaptive Compensation*.

- Phase 1: Early on in LILAC, we focused on learning labels in fixed increments. Here we pit a small number of classes, Seen subset, against the entire remaining dataset, Unseen subset, for a fixed learning interval. Data in the Unseen subset use a designated pseudo-label while data in the Seen subset use their ground-truth labels. Within a short training period, of a few epochs, the neural network is trained with this fixed setup after which a small amount of data corresponding to a new set of labels is moved from Unseen to Seen. Then the entire training process is repeated. In recursively unmasking labels, the model has sufficient time to develop a strong understanding of each class by contrasting against a large and diverse set of negative examples. Further, when comparing the starting points when the model knows the entire dataset, optimization starts from a more well-structured point in the solution space in LILAC than in the case of random initialization.
- Phase 2: Once the network has had a chance to learn all the labels, we regularize the network to prevent over-fitting by providing a softer distribution as the target vector for previously misclassified samples. This is done to reduce the number of misclassifications as well as gently adapt the embedding space to incorporate them. Softening target vectors helps to make learning simpler.

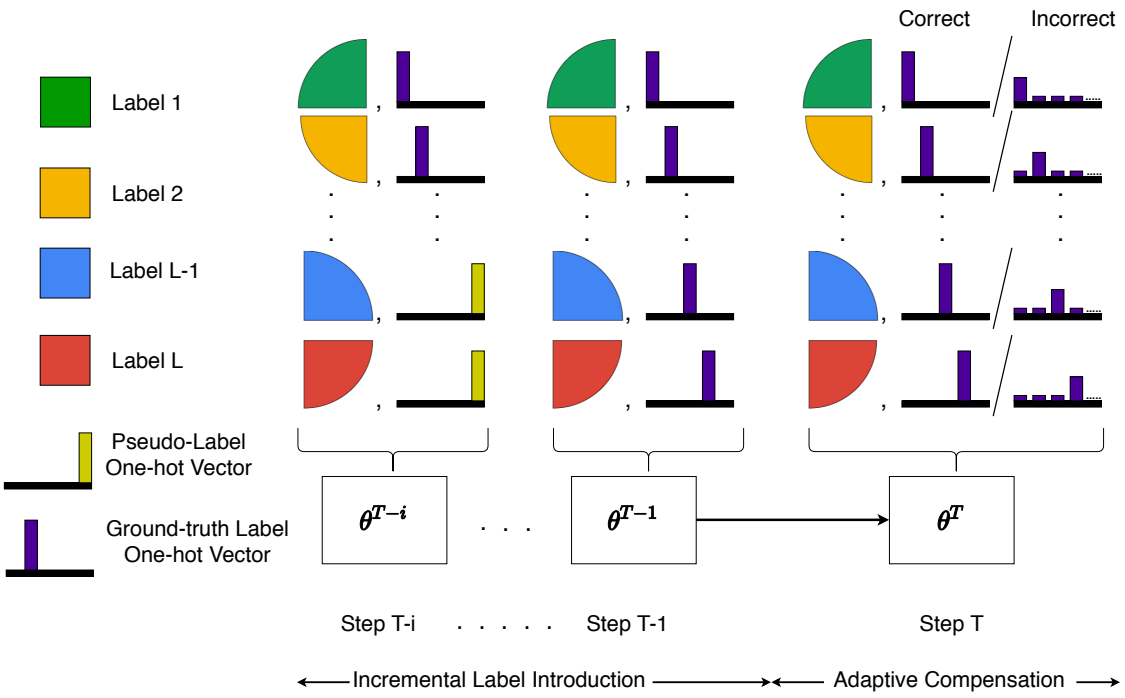


Figure 9: Illustration of the components of LILAC. The Incremental Learning (IL) phase introduces new labels at regular intervals while using the data corresponding to unknown labels as negative samples, using a pseudo-label. Once all labels have been introduced, the Adaptive Compensation (AC) phase of training begins wherein a prior copy of the network being trained is used to decide if samples are misclassified. If so, then a softer distribution is used as the ground-truth vector for those samples

Results

We clearly observed an improvement in performance with the addition of the IL phase and the AC phase, individually. Putting the them together, they complement each other to further boost performance and provide the best performing model across all the compared baselines (Table 6). Overall, LILAC consistently outperformed batch learning across all test datasets while existing comparable methods failed to do so. Among the top performing methods on CIFAR-10 that use standard preprocessing, random crop and flip, while using no dropouts and training from scratch, LILAC easily outperformed all of them (Table 7). Apart from a pure performance perspective, we analyzed the impact of varying certain key hyper-parameters in LILAC including, order of labels presented to the network, peak value of the softened alternate target vector (ϵ) and the number of additional labels unmasked in regular intervals (m).

Ordering of Labels Throughout the standard experiments, we assumed labels are used in the ascending order of value. When this is modified to a random ordering or in ascending order of difficulty, results from Table 8 suggest that there is no explicit benefit or pattern. Other than the extra impact of continually fluctuating label orders across trials, there isn't a large gap in performance.

Table 6: Under similar setups, LILAC consistently achieves higher accuracy and lower std. than batch learning, a property not shared by other baselines

Types	Training	Performance (%)		
		CIFAR 10	CIFAR 100	STL 10
	Batch Learning	95.19 ± 0.190	78.32 ± 0.175	72.88 ± 0.642
Standard	Fixed Curriculum	95.27 ± 0.112	77.89 ± 0.287	72.18 ± 0.601
	Label Smoothing	95.27 ± 0.111	79.06 ± 0.179	72.55 ± 0.877
Custom	Random Augmentation	95.27 ± 0.076	75.37 ± 0.480	73.67 ± 0.708
	Dynamic Batch Size	95.22 ± 0.131	78.73 ± 0.264	72.66 ± 1.081
Breakdown	Only IL (**ours**)	95.38 ± 0.135	78.73 ± 0.139	73.43 ± 0.903
	Only AC (**ours**)	95.38 ± 0.170	78.94 ± 0.179	72.94 ± 0.530
Overall	LILAC(**ours**)	$\mathbf{95.52 \pm 0.072}$	78.88 ± 0.201	$\mathbf{73.77 \pm 0.838}$
	LS + LILAC(**ours**)	95.34 ± 0.080	$\mathbf{79.08 \pm 0.307}$	73.59 ± 0.623

Table 7: Comparison of LILAC's performance against top performing algorithms on CIFAR-10 with *standard pre-processing (random crop + flip)*. Our method easily outperforms both the base shake-drop network ([2]) as well as previous methods.

Method	CIFAR-10
Wide Residual Networks [74]	96.11
Multilevel Residual Networks [75]	96.23
Fractional Max-pooling [76]	96.53
Densely Connected Convolutional Networks [77]	96.54
Drop-Activation [78]	96.55
Shake-Drop [2]	96.59
Shake-Drop + LILAC (ours)	**96.79**

Smoothness of Target Vector in Adaptive Compensation When extended to a variety of peak values for the alternate target vector, we observed that most variations of the peak performance still fall within the standard deviation range for each dataset. However, the peak average performance values usually occurred between 0.7 to 0.5.

Injection of Label Groups LILAC is designed to introduce as many or as few new labels as desired in the IL phase. We hypothesized that developing stronger representations can be facilitated by introducing a small number of new labels while contrasting it against a large variety of negative samples. Table 4.8 supports our hypothesis by illustrating the decrease in performance with an increase in the number of new labels introduced in each interval of the IL phase.

The link to the repository is `https://github.com/MichiganCOG/LILAC_v2`. Additionally, we have a report detailing all of our findings. This report is available at `https://arxiv.org/pdf/2001.04529.pdf`.

Table 8: (**Top**) Comparing random label ordering and difficulty-based label ordering against the ascending order assumption used throughout our experiments, we observe no preference to any ordering pattern. (**Middle**) The mid-range of ϵ values, 0.7-0.4, show an increase in performance while the edges show either too sharp or too flat a distribution leading to decreased performance. (**Bottom**) Only IL model results across all benchmarks illustrates the importance of introducing a small number of new labels in each interval of the IL phase. Values in brackets are for CIFAR-100.

Property	Performance (%)		
	CIFAR-10	CIFAR-100	STL-10
Label Order: Rnd.	95.30 ± 0.146	78.35 ± 0.280	73.10 ± 0.861
Label Order: Asc. Difficulty	95.25 ± 0.156	78.42 ± 0.115	73.69 ± 0.849
Label Order: Asc.	95.32 ± 0.156	78.73 ± 0.139	73.27 ± 0.220
$\epsilon = 0.9$	95.30 ± 0.072	78.48 ± 0.328	73.57 ± 0.980
$\epsilon = 0.8$	95.34 ± 0.141	78.52 ± 0.118	73.54 ± 0.984
$\epsilon = 0.7$	95.42 ± 0.189	78.72 ± 0.356	73.59 ± 0.872
$\epsilon = 0.6$	95.36 ± 0.096	78.75 ± 0.180	73.77 ± 0.838
$\epsilon = 0.5$	95.49 ± 0.207	78.88 ± 0.227	73.61 ± 0.810
$\epsilon = 0.4$	95.52 ± 0.072	78.88 ± 0.201	73.54 ± 0.959
$\epsilon = 0.3$	95.31 ± 0.125	78.66 ± 78.66	73.59 ± 0.955
$\epsilon = 0.2$	95.36 ± 0.095	78.47 ± 0.093	73.57 ± 0.963
m: 1	95.32 ± 0.156	$\mathbf{78.73 \pm 0.139}$	73.27 ± 0.220
m: 2 (4)	$\mathbf{95.38 \pm 0.135}$	78.34 ± 0.209	$\mathbf{73.43 \pm 0.903}$
m: 4 (8)	95.29 ± 0.069	78.37 ± 0.114	72.30 ± 0.543

4.12 Low-rank Tensor Recovery from Streaming and Highly-incomplete Data

Modern data is increasingly high-dimensional and multiway, increasing the storage and computational burden of signal processing algorithms. Many practical applications collect data over multiple modalities. Batch processing of large-scale tensor data quickly becomes computationally intractable, and even storing these tensors is problematic as the memory requirements grow rapidly with the number and size of the tensor modes. Additional challenges include large numbers of missing tensor entries and streaming multiway data that needs to be processed on the fly.

In this project we considered sampling and recovery of 3-way tensors using the algebraic framework of the t-SVD. Three-way tensors are treated as linear operators over the space of oriented matrices and group rings of fibers under the t-product multiplicative o perator. Using this framework, one obtains an SVD-like factorization referred to as the tensor-SVD (t-SVD) with a defined notion of rank referred to as the tubal-rank. A key property of the t-SVD is the optimality of the truncated t-SVD for data approximation under the Frobenius norm measure. The t-SVD has been well-studied in exact tensor recovery, image and video inpainting, hyperspectral data, and solving tensor robust PCA problems for video foreground/background separation.

Most existing t-SVD based methods are batch methods that require all of the data to be stored in memory at computation time and/or compute multiple SVDs. This is very time-consuming and

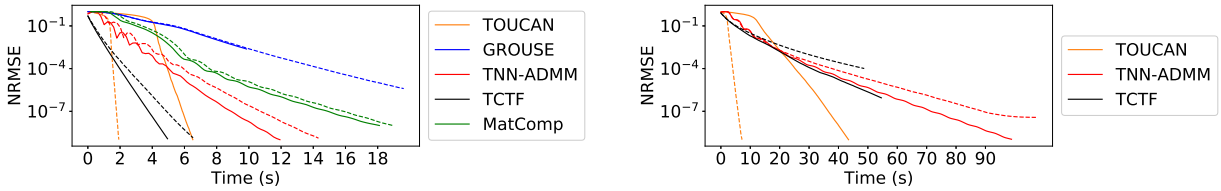
inefficient for large-scale data. Little work has been done to extend online matrix completion methods to the case of multiway tensor data using the t-SVD framework.

We proposed a new algorithm called TOUCAN (**T**ensor Rank-**O**ne **U**pdate on the **C**omplex Grassmanni**an**) to recover low-rank tensor data from streaming, highly-incomplete multiway data with incremental gradient descent on the product manifold of low-rank matrices in the Fourier domain using the recently proposed framework of the t-SVD. Our method is online by nature, avoids the SVD, maintains orthonormality on the tensor Grassmann manifold, and scales linearly in memory and computation. We show our method's ability to track dynamically time-varying low-rank FSMs from streaming two-way data in chemo-sensing and time-lapse video data. Additionally, our method can easily be extended to higher-order tensors.

4.12.1 Experiments for Tensor Completion with TOUCAN

We rigorously tested TOUCAN on both synthetic and real data. These results can be found in our paper [79]. For the sake of brevity we only provide our results for synthetic data in this report. We have compared our results to the state-of-the-art batch tensor completion and matrix completion algorithms.

Incremental Tensor Completion on Synthetic Data



(a) $n_1 = 200, n_2 = 500, n_3 = 20, r = 3$. 50% missing t-SVD generated data.

(b) $n_1 = 200, n_2 = 500, n_3 = 25, r = 5$. 80% missing t-SVD generated data.

Figure 10: t-SVD synthetic experiments. Solid lines are experiments with uniformly random samples, and dashed lines are experiments with uniformly sampled tubes.

We verified the validity and efficiency of TOUCAN in recovering large-scale missing tensor data synthetically generated from isotropic Gaussian distributions with low-tubal-rank. We computed the t-product of two low-tubal-rank tensors to yield a tensor of size $n_1 = 200, n_2 = 500, n_3 = 20$ and tubal-rank $r = 3$. We sampled 50% of tensor entries/tubes randomly according to a Bernoulli distribution. TOUCAN observes one lateral slice at each time instant, solves the inner CGD step to within a set tolerance (10^{-9}), and is allowed to process over the entire batch more than once until the desired termination tolerance. We compared against batch tensor completion algorithms (with improved computational efficiency using conjugate symmetry); one uses the tensor nuclear norm and performs ADMM in its optimization (TNN-ADMM) [80], and the other factorizes the tensor as the product of two low-rank tensors under the t-product (TCTF) [81]. We also compared to standard matrix PCA algorithms by matricizing the tensor and computing batch matrix completion

[82] and GROUSE [83] on each column of the matricized tensor. We plotted the normalized root-mean-squared error (NRMSE) of the recovered tensor to the true tensor by elapsed wall clock time in seconds in Fig. 4.10(a), terminating each algorithm if its completed tensor $\hat{\boldsymbol{\mathcal{X}}}$ satisfies $\|\hat{\boldsymbol{\mathcal{X}}} - \boldsymbol{\mathcal{X}}\|_F / \|\boldsymbol{\mathcal{X}}\|_F \leq 1e-9$. TOUCAN is competitive with state-of-the-art batch method Tensor Factorization (TCTF) [81] in the case of random entry sampling, and TOUCAN's efficient tubal-sampling implementation vastly outperforms the other algorithms while using only 0.6% of the memory compared to storing the entire tensor.

4.13 API development

Designing an effective API for TA1-TA2 interaction proved to be a significant technical challenge in itself, both because of the wide range of different primitive types that must be accommodated, and the fact that this API must be, essentially, "machine-readable." That is, the API was designed to be used not by human coders, but by TA2 systems attempting to automatically compose different primitives into an usable pipeline. For this reason, arriving at a workable API required many rounds of updates and interactions with TA2 teams, in particular the Berkeley team that spearheaded development of the new unified Python API.

Our team contributed significantly to this process, ultimately providing the base classes to be used for all distance computation and metric learning primitives, and collaborating with the Berkeley and SRI teams to produce the latest version of the clustering API, which specifies that clustering results be returned as a distinct results object, which can be fed into further primitives in order to extract specific information (e.g. label assignments, cluster centroids, internal clustering evaluation scores). This model of interaction allows for primitives to capture (in a machine-interpretable fashion) the ability of some algorithms to produce a variety of different types of information, and can be extended to support other types of primitives as well (e.g. to enable extracting feature importance values from a trained random forest).

Conclusions

5.1 Conclusions

The SPIDER project sought to contribute subspace modeling-based primitives to the D3M program. We delivered such primitives along either strata. The primitives were used by numerous teams but the rapidity of progress in the overall program limited the general utility of our primitives: the program was stuck in supervised methods whereas our methods become much more relevant in less supervised scenarios.

Bibliography

[1] W.-S. Lai, J.-B. Huang, O. Wang, E. Shechtman, E. Yumer, and M.-H. Yang, "Learning Blind Video Temporal Consistency," in *European Conference on Computer Vision*, aug 2018. [Online]. Available: https://arxiv.org/abs/1808.00449

[2] Y. Yamada, M. Iwamura, T. Akiba, and K. Kise, "Shakedrop regularization for deep residual learning," *arXiv preprint arXiv:1802.02375*, 2018.

[3] L. Balzano and S. J. Wright, "Local convergence of an algorithm for subspace identification from partial data," *Foundations of Computational Mathematics*, vol. 15, no. 5, pp. 1279–1314, 2015.

[4] A. Byravan and D. Fox, "Se3-nets: Learning rigid body motion using deep neural networks," *arXiv preprint arXiv:1606.02378*, 2016.

[5] H. Krim and M. Viberg, "Two decades of array signal processing research: the parametric approach," *IEEE signal processing magazine*, vol. 13, no. 4, pp. 67–94, 1996.

[6] X. Wang and H. V. Poor, "Blind multiuser detection: A subspace approach," *IEEE Transactions on Information Theory*, vol. 44, no. 2, pp. 677–690, 1998.

[7] R. Kennedy, L. Balzano, S. J. Wright, and C. J. Taylor, "Online algorithms for factorization-based structure from motion," *Computer Vision and Image Understanding*, vol. 150, pp. 139–152, 2016.

[8] R. L. Poirot, P. R. Wishinski, P. K. Hopke, and A. V. Polissar, "Comparative application of multiple receptor methods to identify aerosol sources in northern vermont," *Environmental science & technology*, vol. 35, no. 23, pp. 4622–4636, 2001.

[9] S. L. Miller, M. J. Anderson, E. P. Daly, and J. B. Milford, "Source apportionment of exposures to volatile organic compounds. i. evaluation of receptor models using simulated exposure data," *Atmospheric Environment*, vol. 36, no. 22, pp. 3629–3641, 2002.

[10] J. Pearl *et al.*, "Causal inference in statistics: An overview," *Statistics surveys*, vol. 3, pp. 96–146, 2009.

[11] H. Hotelling, "Relations between two sets of variates," *Biometrika*, vol. 28, no. 3/4, pp. 321–377, 1936.

[12] R. Grosse, R. Raina, H. Kwong, and A. Y. Ng, "Shift-invariance sparse coding for audio classification," *arXiv preprint arXiv:1206.5241*, 2012.

[13] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[14] H. B. Barlow, "Possible principles underlying the transformation of sensory messages," *Sensory communication*, pp. 217–234, 1961.

[15] M. S. Lewicki, "Efficient coding of natural sounds," *Nature neuroscience*, vol. 5, no. 4, pp. 356–363, 2002.

[16] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by v1?" *Vision research*, vol. 37, no. 23, pp. 3311–3325, 1997.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[18] X. Peng, B. Sun, K. Ali, and K. Saenko, "Exploring invariances in deep convolutional neural networks using synthetic images," *CoRR, abs/1412.7122*, vol. 2, no. 4, 2014.

[19] A. Kanazawa, A. Sharma, and D. Jacobs, "Locally scale-invariant convolutional neural networks," *arXiv preprint arXiv:1412.5104*, 2014.

[20] N. Neverova, C. Wolf, G. Taylor, and F. Nebout, "Moddrop: adaptive multi-modal gesture recognition," *TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 2014.

[21] D. Laptev, N. Savinov, J. M. Buhmann, and M. Pollefeys, "Ti-pooling: transformation-invariant pooling for feature learning in convolutional neural networks," *arXiv preprint arXiv:1604.06318*, 2016.

[22] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 35, pp. 2765–2781, Nov. 2013.

[23] M. Soltanolkotabi and E. J. Candes, "A Geometric Analysis of Subspace Clustering with Outliers," *The Annals of Statistics*, vol. 40, no. 4, pp. 2195–2238, 2012.

[24] L. Balzano, B. Eriksson, and R. Nowak, "High rank matrix completion and subspace clustering with missing data," in *Proceedings of the conference on Artificial Intelligence and Statistics (AIStats)*, 2012.

[25] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 2017–2025.

[26] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[28] P. S. Bradley and O. L. Mangasarian, "k-Plane clustering," *Journal of Global Optimization*, vol. 16, pp. 23–32, 2000.

[29] P. Tseng, "Nearest q-flat to m points," *Journal of Optimization Theory and Applications*, vol. 105, no. 1, pp. 249–252, 2000.

[30] P. K. Agarwal and N. H. Mustafa, "K-means projective clustering," in *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2004, pp. 155–165.

[31] J. Lipor, D. Hong, Y. Tan, and L. Balzano, "Subspace clustering using ensembles of k-subspaces," *accepted to IMA Journal on Information and Inference*, 2020.

[32] A. L. Fred and A. K. Jain, "Combining multiple clusterings using evidence accumulation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 6, pp. 835–850, 2005.

[33] R. Heckel and H. Bölcskei, "Robust subspace clustering via thresholding," *IEEE Transactions on Information Theory*, vol. 61, no. 11, pp. 6320–6342, 2015.

[34] S. A. Nene, S. K. Nayar, H. Murase *et al.*, "Columbia object image library (coil-20)," 1996.

[35] C. You, D. Robinson, and R. Vidal, "Scalable sparse subspace clustering by orthogonal matching pursuit," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3918–3927.

[36] C. Xiong, D. Johnson, R. Xu, and J. J. Corso, "Random forests for metric learning with implicit pairwise position dependence," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 958–966.

[37] Z. Lin, M. Chen, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," *arXiv preprint arXiv:1009.5055*, 2010.

[38] G. Tang and A. Nehorai, "Robust principal component analysis based on low-rank and block-sparse matrix decomposition," in *Information Sciences and Systems (CISS), 2011 45th Annual Conference on*. IEEE, 2011, pp. 1–5.

[39] T. Zhou and D. Tao, "Godec: Randomized low-rank & sparse matrix decomposition in noisy case," in *International conference on machine learning*. Omnipress, 2011.

[40] J. He, L. Balzano, and A. Szlam, "Incremental gradient on the grassmannian for online foreground and background separation in subsampled video," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[41] B. E. Moore, C. Gao, and R. R. Nadakuditi, "Panoramic robust pca for foreground-background separation on noisy, free-motion camera video," *CoRR*, vol. abs/1712.06229, 2017.

[42] J. He, L. Balzano, and J. C. S. Lui, "Online robust subspace tracking from partial information," *CoRR*, vol. abs/1109.3827, 2011. [Online]. Available: http://arxiv.org/abs/1109.3827

[43] J. Pont-Tuset, S. Caelles, F. Perazzi, A. Montes, K.-K. Maninis, Y. Chen, and L. Van Gool, "The 2018 davis challenge on video object segmentation," 03 2018.

[44] D. Hong, L. Balzano, and J. A. Fessler, "Asymptotic performance of pca for high-dimensional heteroscedastic data," *Journal of multivariate analysis*, vol. 167, pp. 435–452, 2018.

[45] D. Hong, J. A. Fessler, and L. Balzano, "Optimally weighted pca for high-dimensional heteroscedastic data," *arXiv preprint arXiv:1810.12862*, 2018.

[46] A. Ritchie, C. Scott, L. Balzano, D. Kessler, and C. S. Sripada, "Supervised principal component analysis via manifold optimization," in *2019 IEEE Data Science Workshop (DSW)*. IEEE, 2019, pp. 6–10.

[47] F. Zhou, Q. Claire, and R. D. King, "Predicting the geographical origin of music," in *2014 IEEE international conference on data mining (ICDM)*. IEEE, 2014, pp. 1115–1120.

[48] M. H. Rafiei and H. Adeli, "A novel machine learning model for estimation of sale prices of real estate units," *Journal of Construction Engineering and Management*, vol. 142, no. 2, p. 04015066, 2015.

[49] A. Tsanas, M. A. Little, P. E. McSharry, and L. O. Ramig, "Accurate telemonitoring of parkinson's disease progression by noninvasive speech tests," *IEEE transactions on Biomedical Engineering*, vol. 57, no. 4, pp. 884–893, 2010.

[50] J. Piironen and A. Vehtari, "Iterative supervised principal components," *arXiv preprint arXiv:1710.06229*, 2017.

[51] E. Barshan, A. Ghodsi, Z. Azimifar, and M. Z. Jahromi, "Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds," *Pattern Recognition*, vol. 44, no. 7, pp. 1357–1371, 2011.

[52] N. Vaswani and P. Narayanamurthy, "Fast robust subspace tracking via pca in sparse data-dependent noise," *arXiv preprint arXiv:2006.08030*, 2020.

[53] A. Bobu, E. Tzeng, J. Hoffman, and T. Darrell, "Adapting to continuously shifting domains," 2018.

[54] R. Otazo, E. Candes, and D. K. Sodickson, "Low-rank plus sparse matrix decomposition for accelerated dynamic mri with separation of background and dynamic components," *Magnetic Resonance in Medicine*, vol. 73, no. 3, pp. 1125–1136, 2015.

[55] K. Greenewald, S. Park, S. Zhou, and A. Giessing, "Time-dependent spatially varying graphical models, with application to brain fmri data analysis," in *Advances Neural Inform. Process. Syst.*, 2017, pp. 5832–5840.

[56] F. Liu, D. Choi, L. Xie, and K. Roeder, "Global spectral clustering in dynamic networks," *Proc. National Academy of Sci.*, vol. 115, no. 5, pp. 927–932, 2018.

[57] C. Vondrick, D. Patterson, and D. Ramanan, "Efficiently scaling up crowdsourced video annotation," *International Journal of Computer Vision*, vol. 101, no. 1, pp. 184–204, 2013.

[58] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[59] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," *arXiv preprint arXiv:1801.07455*, 2018.

[60] D. Kim, S. Woo, J.-Y. Lee, and I. S. Kweon, "Deep Video Inpainting," in *IEEE Conference on Computer Vision and Pattern Recognition*, may 2019. [Online]. Available: http://arxiv.org/abs/1905.01639

[61] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium," in *Advances in Neural Information Processing Systems*, 2017, pp. 6626–6637.

[62] A. Gupta, J. Johnson, A. Alahi, and L. Fei-Fei, "Characterizing and Improving Stability in Neural Style Transfer," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4067–4076.

[63] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative Image Inpainting with Contextual Attention," in *IEEE Conference on Computer Vision and Pattern Recognition*, jan 2018. [Online]. Available: http://arxiv.org/abs/1801.07892

[64] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, jun 2018, pp. 586–595. [Online]. Available: https://ieeexplore.ieee.org/document/8578166/

[65] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Computer Vision (ICCV), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4489–4497.

[66] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 4724–4733.

[67] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *European Conference on Computer Vision*. Springer, 2016, pp. 20–36.

[68] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2625–2634.

[69] U. Oswal, C. Cox, M. Lambon-Ralph, T. Rogers, and R. Nowak, "Representational similarity learning with application to brain networks," in *International Conference on Machine Learning*, 2016, pp. 1041–1049.

[70] M. Bogdan, E. Van Den Berg, C. Sabatti, W. Su, and E. J. Candès, "Slope—adaptive variable selection via convex optimization," *The annals of applied statistics*, vol. 9, no. 3, p. 1103, 2015.

[71] X. Zeng and M. A. Figueiredo, "The ordered weighted ℓ_1 norm: Atomic formulation, projections, and algorithms," *arXiv preprint arXiv:1409.4271*, 2014.

[72] H. D. Bondell and B. J. Reich, "Simultaneous regression shrinkage, variable selection, and supervised clustering of predictors with oscar," *Biometrics*, vol. 64, no. 1, pp. 115–123, 2008.

[73] L. W. Zhong and J. T. Kwok, "Efficient sparse modeling with automatic feature grouping," *IEEE transactions on neural networks and learning systems*, vol. 23, no. 9, pp. 1436–1447, 2012.

[74] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *British Machine Vision Conference 2016*. British Machine Vision Association, 2016.

[75] K. Zhang, M. Sun, T. X. Han, X. Yuan, L. Guo, and T. Liu, "Residual networks of residual networks: Multilevel residual networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 6, pp. 1303–1314, 2017.

[76] B. Graham, "Fractional max-pooling (2014)," *arXiv preprint arXiv:1412.6071*, 2014.

[77] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[78] S. Liang, Y. Kwoo, and H. Yang, "Drop-activation: Implicit parameter reduction and harmonic regularization," *arXiv preprint arXiv:1811.05850*, 2018.

[79] K. Gilman and L. Balzano, "Grassmannian optimization for online tensor completion and tracking in the t-svd algebra," *arXiv preprint arXiv:2001.11419*, 2020.

[80] Z. Zhang and S. Aeron, "Exact tensor completion using t-svd," *IEEE Trans. Signal Process.*, vol. 65, no. 6, pp. 1511–1526, 2016.

[81] P. Zhou, C. Lu, Z. Lin, and C. Zhang, "Tensor factorization for low-rank tensor completion," *IEEE Trans. Image Process.*, vol. 27, no. 3, pp. 1152–1163, 2017.

[82] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational mathematics*, vol. 9, no. 6, p. 717, 2009.

[83] L. Balzano, R. D. Nowak, and B. Recht, "Online identification and tracking of subspaces from highly incomplete information," *CoRR*, vol. abs/1006.4046, 2010. [Online]. Available: http://arxiv.org/abs/1006.4046

6.1 List of Acronyms

ADMM Alternating Direction Method of Multipliers

AMT Amazon Mechanical Turk

API Application Programmer Interface

CNN Convolutional Neural Network

COIL-20 Name of a computer object imaging laboratory database

CVXPY The name of a software library for convex optimization in Python

DAVIS Densely Annotated Video Segmentation

D3M Data-Driven Discovery of Models

DNN Deep Neural Network

EKSS Ensemble K-Subspaces

FID Frechet Inception Distance

GRASTA Grassmannian Robust Adaptive Subspace Tracking Algorithm

GROUSE Grassmanian Rank-One Update Subspace Estimation

GRU Gated Recurrent Unit

KSS K-Subspaces

LSPCA Least-Squares PCA

LPIPS Learned Perceptual Similarity

LRPCA Logistic-Reggression PCA

LSTM Long-Short-Term Memory Network

M-PACT Michigan Platform for Activity Classification in Tensorflow

MLA Multilevel, Adaptive

RGB Red-Green-Blue (or Color)

PCA Principal Components Analysis

PSNR Peak Signal-to-Noise Ratio

ReLu Rectified Linear Unit

RFD Random Forest Distance

RPCA Robust Principal Components Analysis

RPCA-IALM RPCA with Inexact Augmented Lagrange Multipliers

RPCA-LBD RPCA with Low-rank and Block-Sparse Matrix Decomposition

SDP Semi-Definite Program

SPIDER Subspace Primitives that are Interpretable and Diverse

SSC$_A DMM$ Subspace Clustering using ADMM Optimization

SSC$_C VX$ Subspace Clustering using Convex Optimization

SSC$_O MP$ Subspace Clustering using Orthogonal Matching Pursuit Optimization

SSIM Structural Similarity

STATA Abbreviated name of a statistics tool for black-box data analysis

STFT Short-Time Fourier Transform

SVD Singular Value Decomposition

TA1 Technical Area 1

TA2 Technical Area 2

TA3 Technical Area 3

T-RECS Training for Rate-Invariance Embeddings by Controlling Speed

TV Total Variation

UoS Union of Subspaces

VGG Visual Geometry Group (a DNN model is named after them)

VFID Video FID

VQA Visual Question Answering