JMLR: Workshop and Conference Proceedings: 44 (2015) p?-p? Submitted 11/1/2015; Published 12/11/2015

On Visual Feature Representations for Transition State Learning in Robotic Task Demonstrations

Animesh Garg* ANIMESH.GARG@BERKELEY.EDU
Sanjay Krishnan* SANJAYKRISHNAN@BERKELEY.EDU
Adithyavairavan Murali ADITHYA_MURALI@BERKELEY.EDU
Florian T. Pokorny FTPOKORNY@BERKELEY.EDU
Pieter Abbeel PABBEEL@CS.BERKELEY.EDU
Trevor Darrell TREVOR@BERKELEY.EDU
Ken Goldberg GOLDBERG@BERKELEY.EDU
** denotes equal contribution*
Departments of IEOR and EECS, University of California, Berkeley
Berkeley, CA 94720-1777, USA

Editor: Afshin Rostamizadeh

Keywords: Robot Motion Trajectory Segmentation, Change Point Identification, Multi-Modal Time Series Segmentation

Abstract

Robot learning from raw trajectory data is challenging due to temporal and spatial inconsistencies. A key problem is extracting conceptual task structure from repeated human demonstrations. In prior work, we proposed a Switched Linear Dynamical System (SLDS) characterization of the demonstrations; the key insight being that switching events induce a density over the state space. A mixture model characterization of this density, called Transition State Clustering, extracts the latent task structure. However, robotics is increasingly moving towards state spaces derived from vision, e.g., from Convolutional Neural Networks (CNNs). This workshop paper describes an extension called Transition State Clustering with Deep Learning (TSC-DL), where we explore augmenting kinematic and dynamic states with features from pre-trained Deep CNNs. We report results on two datasets comparing architectures (AlexNet and VGG), choices of convolutional layer for featurization, dimensionality reduction techniques, visual feature encoding. We find that TSC-DL matches manual annotations with up to 0.806 Normalized Mutual Information (NMI). We also found that use of both kinematics and visual data results in increases of up-to 0.215 NMI compared to using kinematics alone. Video results at: http://berkeleyautomation.github.io/tsc-dl/

1. Introduction

There are a number of techniques to use human demonstrations to facilitate robot learning such as imitation learning Kruger et al. (2010); Calinon et al. (2010b), inverse reinforcement learning Abbeel and Ng (2004), and skill-learning Konidaris and Barto (2009). However, even in a consistent environment, learning from raw trajectory data is challenging Krishnan et al. (2015). Tasks can be multi-step procedures that have complex interactions with the environment. It is, therefore, important to first extract the salient events common to a set of successful demonstrations. Such events can highlight inconsistencies, segment a complex task into simpler subtasks, and classify trajectories.

One approach for modeling such events is the Transition State model of Krishnan et al. (2015). Each demonstration is a realization of a switched linear dynamical system in the state space $\mathcal{X}$ with i.i.d zero-mean additive noise process $W(t)$:

$$\mathbf{x}(t+1) = A(t)\mathbf{x}(t) + W(t) : A(t) \in \{A_1, ..., A_k\}$$

The model further argues that switching events, i.e., when the transition law $A(t) \neq A(t+1)$, happen stochastically as a function of the current state. Thus, the observed transitions from repeated demonstrations induce a probability density f over the state space $\mathcal{X}$. The modes of the density, which intuitively represent a propensity of a state x to trigger a switch, are called *Transition States*. The inference is tractable for some model families of f, for example if it is a Gaussian Mixture Model (GMM), then these modes can be learned with Expectation Maximization.

The efficacy of the transition state model depends on the representation of the state space. To satisfy the model assumptions, trajectories in $\mathcal{X}$ must be locally linear and the density f must be from a model family for which parameter inference (or approximate parameter inference) is possible. For kinematic trajectories (e.g., $\mathcal{X} \subseteq SE(3)$), there is empirical intuition that these assumptions hold . However, the growing maturity of convolutional neural networks (CNNs) has facilitated an increasing use of visual features in robotics . Kinematic recordings from demonstrations are often accompanied by fixed camera video data. Furthermore, the availability of pre-trained models, through frameworks such as CAFFE, has allowed robotics to take advantage of the growing corpora of natural images to bootstrap robotic visual perception. This workshop paper presents an initial exploration and discussion applying the transition state model to multimodal trajectories of kinematics and fixed camera video featurized with pre-trained CNNs. We call this framework Transition State Clustering with Deep Learning.

There are a number of key feature representation questions regarding the use of visual features from CNNs and the transition state model. CNNs represent an immense increase in dimensionality (i.e., $> 100,000$) compared to kinematics/angular configuration spaces (typically < 100). Density estimation and parameter estimation are known to be difficult in sparse high-dimensional data. Consequently, we study whether there is a low-dimensional representation that is sufficiently rich to predict visually important transition events; experimentally comparing dimensionality reduction techniques such as Principal Component Analysis, Gaussian Random Projections, and Canonical Correlation Analysis. Next, we explore whether trajectories in the low dimensional space can be modeled as locally linear. Finally, we evaluate the impact of architecture (AlexNet vs. VGG) and spatiotemporal encoding (VLAD).

Our empirical results suggest that indeed the transition state model can apply to visual state spaces. The insight that trajectories of apparently very high dimensional CNN features lie on low dimensional manifolds is not new.However, these results surprisingly suggest the transferability of this property where convolutional layers from natural image classification CNNs are applied to videos from robotic demonstrations in novel lab environments. Next, we find that in some tasks the fidelity of the image trajectories is sufficient for transition state learning without kinematics data. Finally, we present a number of results describing the hyper-parameter trade-off space and empirical justification for selecting dimensionality reduction and feature encoding parameters.

2. Related Work and Background

1. Learning Switched Systems: Many models for learning switched state spaces either implicitly or explicitly assume that the dynamics are locally linear. It is important to note that locally linear

dynamics does not imply linear motions, as spiraling motions can be represented as linear systems. In Elhamifar and Vidal (2009), videos are modeled as transitions on a lower-dimensional linear subspace and segments are defined as changes in these subspaces. Willsky et al. (2009) proposed BP-AR-HMM. This model is explicitly linear by fitting a autoregressive model to time-series, where time $t+1$ is a linear function of times $t-k,\ldots,t$, to windows of data. The linear function switches according to an HMM with states parametrized by a Beta-Bernoulli model (i.e., Beta Process).

In fact, even the works that apply Gaussian Mixture Models for skill segmentation Calinon et al. (2010a); Lee et al.; Krüger et al. (2012), implicitly fit a locally linear dynamical model. Moldovan et al. (2015) proves that a Mixture of Gaussians model is equivalent to Bayesian Linear Regression; i.e., when applied to a time window it fits a linear transition between the states. Local linear models, including the one in this work, can be extended to locally non-linear models through kernelization or increasing time window sizes. Calinon et al. (2010b) uses state-space segmentation to teach a robot how to hit a moving ball. They use visual features through a visual trajectory tracking of a ball. The visual sensing model in Calinon et al. is tailored to the ball task, and in this paper, we use a set of general visual features for all tasks using CNNs.

2. Visual Gesture Recognition: A number of recent works, attempt to segment human motions from videos Hoai et al. (2011); Tang et al. (2012); Jones and Shao (2014); Wu and Shao (2014); Wu et al. (2015).Tang et al. and Hoai et al. proposed supervised models for human action segmentation from video. Building on the supervised models, there are a few unsuperivsed models for segmentation of human actions: Jones and Shao (2014); Yang et al. (2013); Wu and Shao (2014); Wu et al. (2015). Jones and Shao (2014) restricts their segmentation to learning from two views of the dataset (i.e., two demonstrations). Yang et al. (2013) and Wu et al. (2015) use k-means to learn a dictionary of primitive motions, Krishnan et al. (2015) found that transition state clustering outperforms a standard k-means segmentation approach. In fact, our model is complementary to these works and would be a robust drop-in-replacement for the k-means dictionary learning step. The approach taken by Di Wu et al. is to parametrize human actions using a skeleton model, and they learn the parameters to this skeleton model using a deep neural network. In this work, we explore using generic deep visual features for robotic segmentation without requiring task-specific optimization such as skeleton or action models using in human action recognition.

3. Deep Features in Robotics: Robotics is increasingly using deep features for visual sensing. For example, Lenz et al. uses pre-trained neural networks for object detection in grasping Lenz et al. (2015) and Levine et al. (2015) fine-tune pre-trained CNNs for policy learning. For this reason, we decide to explore methodologies for using deep features in transition state learning as well. We believe this is an important first step in a number of robot learning applications.

3. Transition State Clustering: The GMM Case

This section formalizes one variant of the transition state learning problem, when the noise process $W(t)$ is i.i.d zero-mean Gaussian, and the switching density is a Gaussian Mixture model.

3.1 Transition State Problem

Dynamical System Model: Let $\mathcal{D}=\{d_i\}$ be the set of demonstrations where each d_i is a trajectory $\mathbf{x}(t)$ of robot states and each state is a vector in the state-space $\mathcal{X}\subseteq\mathbb{R}^d$. There is a finite set of $d\times d$ matrices $\{A_1,...,A_k\}$, and an i.i.d zero-mean additive noise process $W(t)$ which accounts for noise

in the dynamical model:

$$\mathbf{x}(t+1) = A_i\mathbf{x}(t) + W(t) : A_i \in \{A_1, ..., A_k\}$$

Transitions between regimes are instantaneous where each time t is associated with exactly one dynamical system matrix $1, ..., k$

Transition States: Transition states are defined as the last states before a dynamical regime transition in *each* demonstration. Each demonstration d_i follows a switched linear dynamical system model, therefore there is a time series of regimes $A(t)$ associated with each demonstration.

Therefore, there will be times t at which $A(t) \neq A(t+1)$. Switching events are governed by a latent function of the current state $S : \mathcal{X} \mapsto \{0,1\}$, and we have noisy observations of switching events $\widehat{S}(x(t)) = S(x(t) + Q(t))$, where $Q(t)$ is a i.i.d noise process. Thus, across all demonstrations, the observed switching events induce a probability density f over the state space $\mathcal{X}$. The goal of transition state learning is to find a mixture model for f that approximately recovers the true latent function S.

3.2 Transition State Clustering: The Gaussian-GMM Case

Let us assume that $W(t)$ is an i.i.d Gaussian process, S is supported by only finitely many $x \in \mathcal{X}$, and $Q(t)$ is also an i.i.d Gaussian process. It follows that the density f is a Gaussian Mixture Model. Under this model, we overview a basic technique for parameter inference. It turns out a particularly efficient model for parameter inference is a reduction of this problem to hierarchical clustering by first identifying candidate transitions and then clustering over the candidate transitions.

Identifying Transitions: Suppose there was only one regime, then following from the Gaussian assumption, this would be a linear regression problem:

$$\arg\min_A \|AX_t - X_{t+1}\|$$

where X_t is a matrix where each column vector is $x(t)$, and X_{t+1} is a matrix where each column vector is the corresponding $x(t+1)$. Moldovan et al. Moldovan et al. (2015) proves that fitting a Jointly Gaussian model to $n(t) = \binom{\mathbf{x}(t+1)}{\mathbf{x}(t)}$ is equivalent to Bayesian Linear Regression.

Therefore, to fit a switched linear dynamical system model, we use a Mixture of Gaussians (GMM) to $n(t)$. GMMs define clusters based on their most likely mixture assignment. Each learned cluster denotes a different regime, while co-linear states are in the same cluster. To find transition states, we move along a trajectory from $t = 1, ..., t_f$, and find states at which $n(t)$ is in a different cluster than $n(t+1)$. These points mark a transition between clusters (i.e., transition regimes).

Pruning Inconsistency: We consider the problem of outlier transitions, ones that appear only in a few demonstrations. Each of these regimes will have constituent vectors where each $n(t)$ belongs to a demonstration d_i. Transition states that mark transitions to or from regimes whose constituent vectors come from fewer than a fraction ρ demonstrations are *pruned*. ρ should be set based on the expected rarity of outliers. In our experiments, we set the parameter ρ to 80% and show the results with and without this step.

Transition State Clustering: If we model the states at the transitions as drawn from a GMM model: $x(t) \sim N(\mu_i, \Sigma_i)$, Then, we can fit a GMM again to cluster the state vectors at the transition states. Each cluster defines an ellipsoidal region of the state-space space.

3.3 Multiple Sensing Modalities

A Gaussian model in a Euclidean space $\mathbb{R}^d$ assumes an L_2 metric. However, in a number of cases the state space contains data from multiple sensing modalities such as vision and kinematics the L_2 metric may not be sensible. We address this problem by adding expanding the GMM hierarchy, where we first fit a GMM with a subset of variables corresponding modality 1. Then, we partition the dataset by each transitions most likely mixture. Within each partition, we fit GMM corresponding to modality 2, and repeating this process until completion.

Modeling Temporal Effects
Time can be modeled as a separate sensing modality. Without temporal localization, the transitions may be ambiguous. For example, in a "Figure 8" trajectory, the robot may pass over a point twice in the same task. We define an augmented state space $\mathbf{x}(t) = \binom{k(t)}{t}$. Within a state cluster, we model the times which change points occur as drawn from a GMM: $t \sim N(\mu_i, \sigma_i)$, then we can apply a GMM to the set of times. This groups together events that happen at similar times during the demonstrations. The result is clusters of states and times. Thus, a transition state m_k is defined as tuple of an ellipsoidal region of the state-space and a time interval.

Visual Features
Similarly, visual features can be modeled with this technique. We define an augmented state space $\mathbf{x}(t) = \binom{k(t)}{z(t)}$, where $k(t) \in \mathbb{R}^k$ are the kinematic features and $z(t) \in \mathbb{R}^v$ are the visual features. Within each kinematics state cluster, we model the visual which change points occur as drawn from a GMM: $z \sim N(\mu_i, \sigma_i)$, then we can apply a GMM to the set of visual states.

3.4 Practical Considerations

Dirichlet Process GMM: One challenge with mixture models is hyper-parameter selection, such as the number of mixtures. Recent results in Bayesian statistics can mitigate some of these problems. We use the Dirichlet Process Gaussian Mixture Model at the multiple levels of the hierarchical clustering to set the number of mixtures using Variational EM.

Rolling Temporal Window: To better capture hysteresis and transitions that are not instantaneous, in this current paper, we use rolling window states where each state $\mathbf{x}_{(t)}$ is a concatenation of T historical states. We varied the length of temporal history T and evaluated performance of the TSC-DL algorithm for the suturing task using metric defined in Section 5.1 as shown in Figure 1. We empirically found a sliding window of size 3, i.e., $\mathbf{x}_{(t)} = \binom{\mathbf{k}(t)}{\mathbf{z}(t)}$, as the state representation led to improved segmentation accuracy while balancing computational effort.

Skill-Weighted Pruning Demonstrators may have varying skill levels leading to increased outliers, and so we extend our outlier pruning to include weights. Let, w_i be the weight for each demonstration $d_i \in \mathcal{D}$, such that $w_i \in [0,1]$ and $\hat{w}_i = \frac{w_i}{\sum w_i}$. Then a cluster $C_{kk'}$ is pruned if it does not contain change points $CP(n)$ from at least ρ fraction of demonstrations. This converts to:

$$\sum_{d_i} \hat{w}_i \mathbf{1}\Big(\sum_{n:N(n)\in d_i} \mathbf{1}(CP(n) \in C_{kk'}) \geq 1 \Big) \leq \rho$$

4. Transition State Clustering With Generalized Visual Features

We extend our prior work with states defined with generalized visual features from CNNs, and present the details of the TSC-DL in Algorithm 1. We define an augmented state space $\mathbf{x}(t) = \binom{k(t)}{z(t)}$, where $k(t) \in \mathbb{R}^k$ are the kinematic features and $z(t) \in \mathbb{R}^v$ are the visual features. We use layers

Algorithm 1: TSC-DL: Transition State Clustering with Deep Learning

Data: Set of demonstrations: $\mathcal{D}$
Parameters: pruning factor (ρ), time window (w), PCA dim (d_p), hyperparams ($\alpha_1, \alpha_2, \alpha_3, \alpha_4$)
Result: Set of Predicted Transitions $\mathcal{T}_i, \ \forall d_i \in \mathcal{D}$

foreach $d_i \in \mathcal{D}$ **do**
  $z_i \leftarrow$ VisualFeatures (d_i, w, d_p) and $k_i \leftarrow$ KinematicFeatures (d_i, w)
  $\mathbf{x}_i(t) \leftarrow \binom{\mathbf{k_i}(t)}{\mathbf{z_i}(t)} \ \forall t \in \{1, \ldots, T_i\}$
  $\bar{x}_i(t) \leftarrow \left[\mathbf{x}(t+1)^T, \ \mathbf{x}(t)^T, \ \mathbf{x}(t-1)^T\right]^T, \ \forall t$ and $N \leftarrow \left[N^T, \bar{x}_i(1)^T, \ldots, \bar{x}_i(T_i)^T\right]^T$
$\mathcal{C}^{CP} = \text{DPGMM}(N, \alpha_0)$ `// Cluster to get Change Points`
foreach $N(n) \in \mathcal{C}_i^{CP}, \ N(n+1) \in \mathcal{C}_j^{CP}, i \neq j$ **do** $CP \leftarrow CP \cup \{N(n)\}$
$\mathcal{C}_1 = \text{DPGMM}(CP, \alpha_1)$ `// Cluster over Visual Feature Subspace`
foreach $C_k \in \mathcal{C}_1$ **do**
  $CP(C_k) \leftarrow \{CP(n) \in C_k, \forall n \in \{1, \ldots, |CP|\}\}$
  $\mathcal{C}_{k2} \leftarrow \text{DPGMM}(CP(C_k), \alpha_2)$ `// Cluster over Kinematic Feature Subspace`
  foreach $C_{kk'} \in \mathcal{C}_{k2}$ **do**
    if $\sum_{d_i} \mathbf{1}\Big(\sum_{n:N(n)\in d_i} \mathbf{1}(CP(n) \in C_{kk'}) \geq 1\Big) \leq \rho|\mathcal{D}|$ **then** $\mathcal{C}_{k2} \leftarrow \mathcal{C}_{k2} \setminus \{C_{kk'}\}$ `// Pruning`
    $\forall \ d_i \in \mathcal{D}$ **do** $T_i \leftarrow T_i \cup \{CP(n) \in C_{kk'}, \forall n : N(n) \in d_i\}$ `// intra-cluster transitions ∀ d_i`
foreach $d_i \in \mathcal{D}$ **do**
  Repeat steps 1-13 for $\mathcal{D}' = \mathcal{D} \setminus d_i$
  $\mathtt{T}_j \leftarrow \mathtt{T}_j \cup T_j^{(i)}, \ \{\forall \ j : d_j \in \mathcal{D}'\}$ `// T_j^(i): ith iteration`
foreach $d_i \in \mathcal{D}$ **do** $\mathcal{T}_i \leftarrow DPGMM(\mathtt{T}_i, \alpha_4)$ `// Cluster over time to predict Transition Windows`
return $\mathcal{T}_i, \ \forall d_i \in \mathcal{D}$

from a pre-trained Convolutional Neural Network (CNNs) to derive the features frame-by-frame. We found that use of these features requires a number of pre-processing and post-processing steps; in addition to a number of design choices within the CNN such as which convolutional layer(s) to use for composing the visual featurization.

Pre-processing: CNNs are trained on static images for image classification, and as a result their features are optimized for identifying salient edges and colors. However, they do not capture temporal features and do differentiate the between robot and workspace features. Furthermore, since we aggregate across demonstrations, we need to ensure that these features are largely consistent. To reduce variance due to extraneous objects and lighting changes, we crop each video to capture the only the relevant workspace where robot manipulation occurs. Then, the videos are rescaled to 640x480 along with down-sampling to 10 frames per second for computational efficiency. All frames in the videos are normalized to a *zero* mean in all channels (RGB) individually Krizhevsky et al. (2012); Simonyan and Zisserman (2014). All of pre-processing were preformed with open source `ffmpeg` library.

Visual Featurization: Once the images were pre-processed, we applied the convolutional filters from the pre-trained neural networks. Yosinski et al. (2014) note that CNNs trained on natural images exhibit roughly the same Gabor filters and color blobs on the first layer. They established that earlier layers in the hierarchy give more general features while later layers give more specific ones. In our experiments, we explore the level of generality of features required for segmentation. In particular, we explore two architectures designed for image classification task on natural images:
(a) **AlexNet:** Krizhevsky et al. (2012) proposed multilayer (5 in all) a CNN architecture, and

(b) **VGG:** Simonyan and Zisserman (2014) proposed an alternative architecture termed VGG (acronym for Visual Geometry Group) which increased the number of convolutional layers significantly (16 in all).
We also compare these features to other visual featurization techniques such as SIFT and SURF for the purpose of task segmentation using TSC-DL.

Visual Feature Encoding: After constructing these features, the next step is encoding the results of the convolutional filter into a vector $z(t)$. We explore three encoding techniques: (1) Raw values, (2) Vector of Locally Aggregated Descriptors (VLAD) Arandjelovic and Zisserman (2013), and (3) Latent Concept Descriptors (LCD) Xu et al. (2014).

Visual Feature Dimensionality Reduction: After encoding, we feed the CNN features $z(t)$, often in more than 50K dimensions, through a dimensionality reduction process to boost computational efficiency. This also balances the visual feature space with a relatively small dimension of kinematic features (< 50). Moreover, GMM-based clustering algorithms usually converge to a local minima and very high dimensional feature spaces can lead to numerical instability or inconsistent behavior. We explore multiple dimensionality reduction techniques to find desirable properties of the dimensionality reduction that may improve segmentation performance. In particular, we analyze Gaussian Random Projections (GRP), Principal Component Analysis (PCA) and Canonical Correlation Analysis (CCA) in Table 1. GRP serves as a baseline, while PCA is used based on widely application in computer vision as in Xu et al. (2014). We also explore CCA as it finds a projection that maximizes the visual features correlation with the kinematics.

Robust Temporal Clustering: To reduce over-fitting and build a confidence interval as a measure of accuracy over the temporal localization of transitions, we use a *Jack-knife estimate*. It is calculated by aggregating the estimates of each $N-1$ estimate in the sample of size N. We iteratively hold out one out of the N demonstrations and apply TSC-DL to the remaining demonstrations. Then, over $N-1$ runs of TSC-DL, $N-1$ predictions are made $\forall d_i \in \mathcal{D}$. We temporally cluster the transitions across $N-1$ predictions, to estimate final transition time mean and variance $\forall d_i \in \mathcal{D}$. This step is illustrated in step 15-17 of Algorithm 1.

5. Experiments

5.1 Evaluation Metrics

It is important to note that TSC-DL is an unsupervised algorithm that does not use input labels. Therefore, we evaluate TSC-DL both intrinsically (without labels) and extrinsically (against human annotations).

Intrinsic metric: The goal of the intrinsic metric is compare the *relative* performance of different featurization techniques, encodings, and dimensionality reduction within TSC-DL without reference to external labels. The intrinsic metric we use measures the "tightness" of the transition state clusters. This metric is meaningful since we require that each transition state cluster contains transitions from a fraction of at least ρ of the demonstrations, the tightness of the clusters measures how well TSC-DL discovers regions of the state space where transitions are grouped together. This is measured with the mean *Silhouette Score* (denoted by $\mathtt{ss}$), which is defined as follows for each transition state i:

$$\mathtt{ss}(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}}, \quad \mathtt{ss}(i) \in [-1, 1]$$

if transition state i is in cluster C_j, $a(i)$ is defined the average dissimilarity of point i to all points in C_j, and $b(i)$ the dissimilarity with the closest cluster measured as the minimum mean dissimilarity of point i to cluster C_k, $k \neq j$. We use L_2-norm as the dissimilarity metric and re-scale $\mathtt{ss} \in [0,1]$ for ease of comparison.

Extrinsic metric: To calculate an absolute measure of similarity of TSC-DL predictions $\mathcal{T}$ with respect to manual annotations $\mathcal{L}$, we use *Normalized Mutual Information* (NMI) which measures the alignment between two label assignments. NMI is equal to the KL-divergence of the joint distribution with the product distribution of the marginals; intuitively, the distance from pairwise statistical independence. NMI lies in $[0,1]$, where 0 indicates independence while 1 is perfect matching. It is defined as,

$$NMI(\mathcal{T},\mathcal{L}) = \frac{I(\mathcal{T},\mathcal{L})}{\sqrt{H(\mathcal{T})H(\mathcal{L})}}, \quad NMI(\mathcal{T},\mathcal{L}) \in [0,1]$$

5.2 Evaluation of Visual Featurization

In our first experiment, we explore different visual featurization, encoding, and dimensionality reduction techniques. We applied TSC-DL to our suturing experimental dataset, and measured the silhouette score of the resulting transition state clusters. Table 1 describes the featurization techniques on the vertical axis and dimensionality reduction techniques on the horizontal axis. Our results suggest that on this dataset features extracted from the pre-trained CNNs resulted in tighter transition state clusters compared to SIFT features with a 3% lower ss than the worst CNN result. Next, we found that features extracted with the VGG architecture resulted in the highest ss with a 3% higher ss than the best AlexNet result. Qualitative results of TSNE plots of a subsequence is shown in Figure 2.

We also found that PCA for dimensionality reduction gave the best ss performance 7% higher than the best GRP result and 10% higher than best CCA result. Because CCA finds projections of high correlation between the kinematics and video, we believe that CCA discard features informative features resulting in reduced clustering performance. We note that neither of the encoding schemes, VLAD or LCD significantly improve the ss.

There are two hyper-parameters for TSC-DL which we set empirically: sliding window size (T = 3), and the number of PCA dimensions (k = 100). In Figure 1, we show a sensitivity plot with the ss as a function of the parameter. We calculated the ss using the same subset of the suturing dataset as above and with the VGG conv5_3 CNN. We found that T = 3 gave the best performance. We also found that PCA with k = 1000 dimensions was only marginally better than k = 100 yet required >30 mins to run. For computational reasons, we selected k = 100.

5.3 End-to-End Evaluation

For all subsequent experiments on real data, we use a pre-trained VGG CNN conv5_3 and encoded with PCA with 100 dimensions.

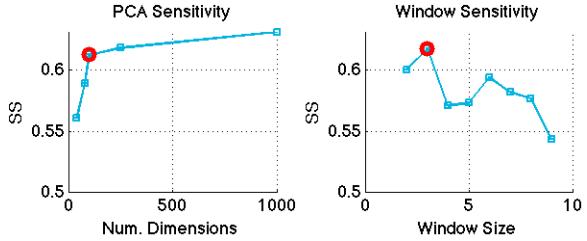

Figure 1: We evaluate the sensitivity of two hyper-parameters set in advance: number of PCA dimensions and sliding window size. The selected value is shown in red double circles.

	GRP	PCA	CCA
SIFT	-	0.443±0.008	-
AlexNet conv3	0.559±0.018	0.600±0.012	0.494±0.006
AlexNet conv4	0.568±0.007	0.607±0.004	0.488±0.005
AlexNet pool5	0.565±0.008	0.599±0.005	0.486±0.012
VGG conv5_3	0.571±0.005	**0.637±0.009**	0.494±0.013
VGG LCD-VLAD	0.506±0.001	0.534±0.011	0.523±0.010
AlexNet LCD-VLAD	0.517±0.001	0.469±0.027	0.534±0.018

Table 1: The silhouette score for each of the techniques and dimensionality reduction schemes on a subset of suturing demonstrations (5 expert examples). We found that PCA (100 dims) applied to VGG conv5_3 maximizes silhouette score

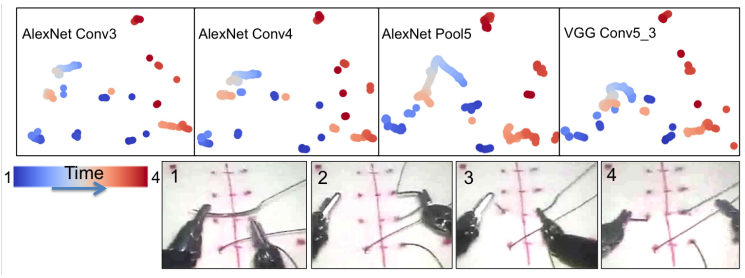

Figure 2: The figure illustrates TSNE Plots for various layers in Alex Net and VGG on a video sub-sequence of a Suturing Demonstration. We note that higher cluster compactness in Conv4 and Conv5_3 matches with higher ss scores above.

Suturing		K	Z	K+Z
Silhouette Score	E	0.630±0.014	0.576±0.018	0.654±0.065
	E+I	0.550±0.014	0.548±0.015	0.716±0.046
	E+I+N	0.518±0.008	0.515±0.021	**0.733±0.056**
NMI Score	E	0.516 ± 0.026	0.266 ± 0.025	0.597 ± 0.096
	E+I	0.427 ± 0.053	0.166 ± 0.057	**0.646 ± 0.039**
	E+I+N	0.307 ± 0.045	0.157 ± 0.022	0.625 ± 0.034

Table 2: Comparison of TSC-DL performance on Surgical Suturing Task. We compare the prediction performance by incrementally adding demonstrations from Experts (E), Intermediates (I), and Novices (N) respectively to the dataset.

1. Surgical Suturing: We apply our method to a subset of JIGSAWS dataset, from Gao et al. (2014), consisting of surgical task demonstrations under tele-operation using the da Vinci surgical system. The dataset was captured from 8 surgeons with 3 different skill levels, performing 5 repetitions each of suturing and needle passing. We use 39 demonstrations of a 4 throw suturing task (Figure 3) and we manually annotate these demonstrations for reference. We apply TSC-DL to kinematics and vision alone respectively and then the combination. With combined kinematics and vision, TSC-DL learns many of the important segments identified by manual annotation. After learning the segmentation, we apply it to a representative trajectory (Figure 3) and find that we accurately recover 10 out of 15 transitions aer similar to manual labeling.

Upon further investigation of the false positives, we found that they corresponded to crucial actions missed by manual labeling. For example, TSC-DL discovers that a crucial needle repositioning step where many demonstrators penetrate and push-through the needle in two different motions. We find segments that correspond to linear dynamical systems, and applies this criterion consistently. However, human annotators may miss subtle transitions such a quick two-step motion.

2. Toy Plane Assembly: In our next experiment, we explore segmenting a multi-step assembly of a toy *Plane* from the YCB dataset by Çalli et al. (2015). We collect 8 kinesthetic demos of the task on the PR2 robot. Figure 3 illustrates the segmentation for the plane assembly task. We find the plane assembly task using kinematics or vision alone results in a large number of segments. The combination can help remove spurious segments restricting our segments to those transitions that occur in most of the demonstrations–agreeing in similarity both kinematically and visually.

Human Demos: We extend the toy plane assembly experiment to collect 8 demonstrations each from two human users. These examples only have videos and no kinematic information. We note

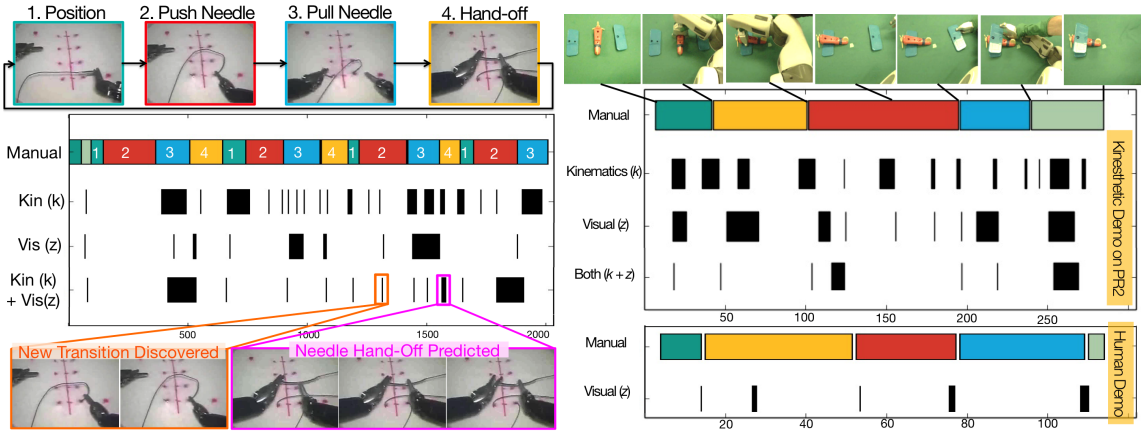

Figure 3: (a) The first row shows a manual segmentation of the suturing task in 4 steps: (1) Needle Positioning, (2) Needle Pushing, (3) Pulling Needle, (4) Hand-off. TSC-DL extracts many of the important transitions without labels and also discovers un-labled transition events. (b) We compare TSC-DL on for the Toy Plane Assembly task with 8 kinesthetic demos (top) and 8 human demos (bottom). No kinematics were available for the human demos. We illustrate the segmentation for an example demo in each case. Our manual annotation of the task has 5 steps and TSC-DL recovers this structure separately for both Kinesthetic demos on PR2 and Human demos with the only visual state.

that there was a difference between users in the grasping location of fuselage. We find that using both kinematics and visual data results in ss of 0.771 ± 0.067 (NMI:0.747 ± 0.016). While only visual data for human demo results in ss of 0.615 ± 0.018 (NMI: 0.766 ± 0.078).

6. Conclusion and Future Work

We model a set of robot task demos as linear dynamical system motions that transition as switching linear dynamic system. To learn Transition clusters, the proposed algorithm TSC-DL uses a hierarchical application of Dirichlet Process Gaussian Mixture Models (DP-GMM). TSC-DL levarages both kinematic data along with domain independent visual feature extraction from pre-trained CNNs. We apply TSC-DL to real data sets on (1) JIGSAWS surgical suturing, and (1) A toy plane assembly task. We also demonstrate that TSC-DL applies to human task demos in absence of kinematic information. On real datasets, we find that TSC-DL matches the manual annotation with up to 0.806 NMI. Our results also suggest that including kinematics and vision results in increases of up-to 0.215 NMI over kinematics alone. We demonstrated the benefits of an unsupervised approach with examples in which TSC-DL discovers inconsistencies such as segments not labeled by human annotators, and apply TSC-DL to learn across demonstrations with widely varying operator skill levels. We also validated surgical results in a different domain with demonstrations of assembly tasks with the PR2 and human-only demonstrations.

Future Work: Our results suggest a number of important directions for future work. First, we plan to apply the results from this paper to learn transition conditions for finite state machines for surgical subtask automation. The use of CNN features with fine tuning can allow for task structure learning directly from raw data (images) in cases of sufficient data availability, as opposed to using CNNs trained with datasets of different image statistics. Furthermore, recent advances in Recurrent Networks and LSTMs allow temporal information capture, however they also open questions on transferability of such features to new domains such as CNNs were used in this work.

Appendix A.

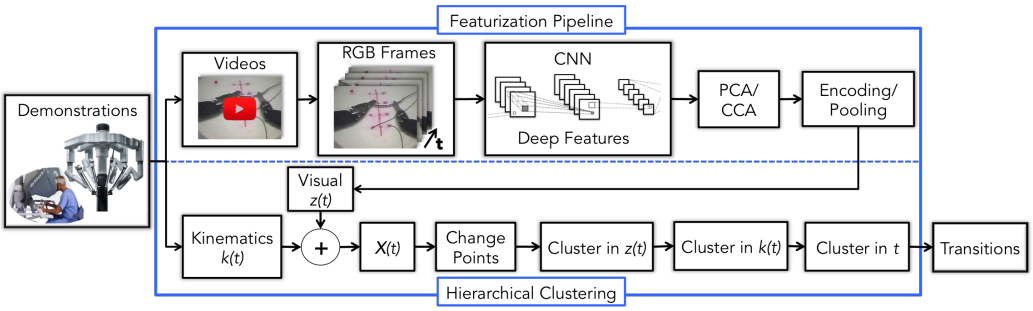

Figure 4: We use a visual processing pipeline with deep features to construct a trajectory of high-dimensional visual states $z(t)$. We concatenate encoded versions of these features with kinematics and apply hierarchical clustering to find segments.

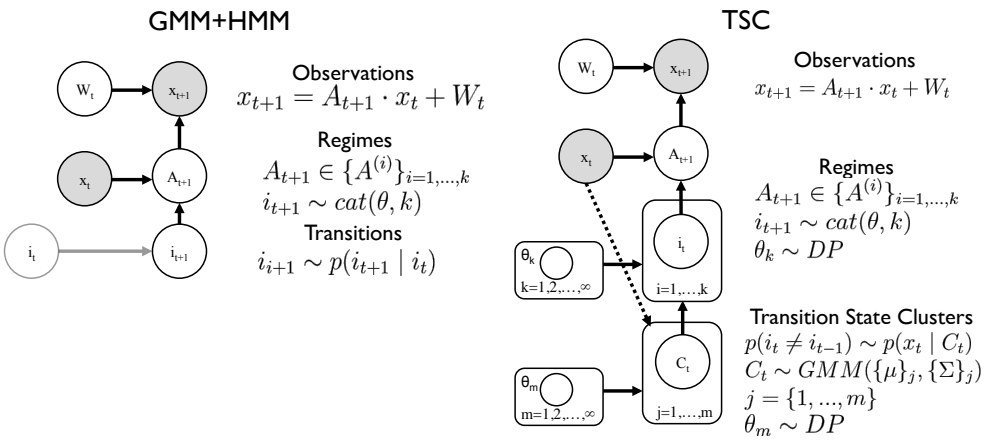

Figure 5: (1) A finite-state Hidden Markov Chain with Gaussian Mixture Emissions (GMM+HMM) , and (2) TSC-DL model. TSC-DL uses Dirichlet Process Priors and the concept of transition states to learn a robust segmentation.

We design TSC-DL to be robust to some types of variations in demonstrations. In Figure 5, we compare the graphical models of GMM+HMM, and TSC-DL. The TSC-DL model applies Dirichlet Process priors to automatically set the number of hidden states (regimes). The goal of the TSC-DL algorithm is to find spatially and temporally similar transition states across demonstrations. On the other hand, the typical GMM+HMM Baum-Welch model learns a $k \times k$ transition matrix. We empirically find that the TSC-DL model is robust to noise and temporal variation.

Acknowledgments

This research was supported in part by a seed grant from the UC Berkeley Center for Information Technology in the Interest of Society (CITRIS), by the U.S. National Science Foundation under Award IIS-1227536: Multilateral Manipulation by Human-Robot Collaborative Systems. This work has been supported in part by funding from Google and and Cisco.

References

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1. ACM, 2004.

Relja Arandjelovic and Andrew Zisserman. All about vlad. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1578–1585. IEEE, 2013.

Sylvain Calinon, Florent D'halluin, Eric L Sauser, Darwin G Caldwell, and Aude G Billard. Learning and reproduction of gestures by imitation. *Robotics & Automation Magazine, IEEE*, 17(2):44–54, 2010a.

Sylvain Calinon, Eric L Sauser, Aude G Billard, and Darwin G Caldwell. Evaluation of a probabilistic approach to learn and reproduce gestures by imitation. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2671–2676. IEEE, 2010b.

Berk Çalli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M. Dollar. Benchmarking in manipulation research: The YCB object and model set and benchmarking protocols. *CoRR, abs/1502.03143*, 2015.

Ehsan Elhamifar and René Vidal. Sparse subspace clustering. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2790–2797. IEEE, 2009.

Y. Gao, S. Vedula, C.E. Reiley, N. Ahmidi, B. Varadarajan, H.C. Lin, L. Tao, L. Zappella, B. Bejar, D.D. Yuh, C. Chen, R. Vidal, S. Khudanpur, and G.D. Hager. The jhu-isi gesture and skill assessment dataset (jigsaws): A surgical activity working set for human motion modeling. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2014.

Minh Hoai, Zhen-Zhong Lan, and Fernando De la Torre. Joint segmentation and classification of human actions in video. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011.

Simon Jones and Ling Shao. Unsupervised spectral dual assignment clustering of human actions in context. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 2014.

George Konidaris and Andrew G Barto. Efficient skill learning using abstraction selection. In *IJCAI*, volume 9, pages 1107–1112, 2009.

Sanjay Krishnan, Animesh Garg*, Sachin Patil, Colin Lea, Gregory Hager, Pieter Abbeel, and Ken Goldberg (*denotes equal contribution). Transition state clustering: Unsupervised surgical trajectory segmentation for robot learning. In *International Symposium of Robotics Research*. Springer STAR, 2015.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Volker Kruger, Dennis Herzog, Sanmohan Baby, Ales Ude, and Danica Kragic. Learning actions from observations. *Robotics & Automation Magazine, IEEE*, 17(2):30–43, 2010.

Volker Krüger, Vadim Tikhanoff, Lorenzo Natale, and Giulio Sandini. Imitation learning of non-linear point-to-point robot motions using dirichlet processes. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2029–2034. IEEE, 2012.

Sang Hyoung Lee, Il Hong Suh, Sylvain Calinon, and Rolf Johansson. Autonomous framework for segmenting robot trajectories of manipulation task. *Autonomous Robots*, 38(2):107–141.

Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *The International Journal of Robotics Research*, 2015.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *arXiv preprint arXiv:1504.00702*, 2015.

T. Moldovan, S. Levine, M.I. Jordan, and P. Abbeel. Optimism-driven exploration for nonlinear systems. In *Int. Conf. on Robotics and Automation (ICRA)*, 2015.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.

Kevin Tang, Li Fei-Fei, and Daphne Koller. Learning latent temporal structure for complex event detection. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012.

Alan S Willsky, Erik B Sudderth, Michael I Jordan, and Emily B Fox. Sharing features among dynamical systems with beta processes. In *Advances in Neural Information Processing Systems*, pages 549–557, 2009.

Chenxia Wu, Jiemi Zhang, Silvio Savarese, and Ashutosh Saxena. Watch-n-patch: Unsupervised understanding of actions and relations. In *Computer Vision and Pattern Recognition, IEEE Conference on*, 2015.

Di Wu and Ling Shao. Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition. In *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, 2014.

Zhongwen Xu, Yi Yang, and Alexander G Hauptmann. A discriminative cnn video representation for event detection. *arXiv:1411.4006*, 2014.

Yang Yang, Imran Saleemi, and Mubarak Shah. Discovering motion primitives for unsupervised grouping and one-shot learning of human actions, gestures, and expressions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(7):1635–1648, 2013.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, 2014.